MODELLING CHANGES, STAKEHOLDERS AND THEIR RELATIONS IN SEMANTIC 3D CITY MODELS

Son H. Nguyen[1,*], Thomas H. Kolbe[1]

[1] Chair of Geoinformatics, Department of Aerospace and Geodesy, Technical University of Munich (TUM), Germany - (son.nguyen, thomas.kolbe)@tum.de

KEY WORDS: Change Detection, Change Interpretation, Digital Twins, Semantic 3D City Models, CityGML, Stakeholders.

ABSTRACT:

Urban digital twins have been increasingly adopted by cities worldwide. Digital twins, especially semantic 3D city models as key components, have quickly become a crucial platform for urban monitoring, planning, analyses and visualization. However, as the massive influx of data collected from cities accumulates quickly over time, one major problem arises as how to handle different temporal versions of a virtual city model. Many current city modelling deployments lack the capability for automatic and efficient change detection and often replace older city models completely with newer ones. Another crucial task is then to make sense of the detected changes to provide a deep understanding of the progresses made in the cities. Therefore, this research aims to provide a conceptual framework to better assist change detection and interpretation in virtual city models. Firstly, a detailed hierarchical model of all potential changes in semantic 3D city models is proposed. This includes appearance, semantic, geometric, topological, structural, Level of Detail (LoD), auxiliary and scoped changes. In addition, a conceptual approach to modelling most relevant stakeholders in smart cities is presented. Then, a model - reality graph is used to represent both the different groups of stakeholders and types of changes based on their relative interest and relevance. Finally, the study introduces two mathematical methods to represent the relevance relations between stakeholders and changes, namely the relevance graph and the relevance matrix.

1. INTRODUCTION

Urban modelling and digitalization is gaining traction globally. The number of transformative urban digital twin and city modelling deployments is expected to grow from a handful of early implementations in 2019 to exceed 500 by 2025 (ABI Research, 2019). In the context of smart cities, an urban digital twin can be thought of as a virtual model and a replica of a city in the physical world. Digital twins combine a vast amount of information collected from numerous different sources ranging from direct, closely-linked measurements (typically IoT in-situ sensors) to remote sensing (with all types of devices and platforms) as well as other manually updated information. Figure 1 shows an overview of such urban digital twins. As a result, digital twins, especially their virtual city models, have quickly become a crucial platform not only for storing, visualizing and monitoring urban objects, but also for urban interpretation, simulation and analyses in general. This involves many types of transformation on the virtual city models, such as refinement, generalization, derivation and enrichment (see Figure 1b). Not only can these changes be reflected back to the original physical counterpart (e.g. for automated manufacturing, manual construction or destruction, control of components or systems, etc.), they can also be analysed to provide an insight into changes that occurred in the real world (see Figure 1a).

However, as a city may have multiple digital twins or modified copies of a digital twin (i.e. digital triplets), one major problem arises as how to handle changes that occurred in the real-world objects and their digital representations. Many current city modelling deployments often replace the older city models completely with newer ones, which not only wastes time and resources, but also does not reflect any meaningful progress made in the physical and digital city objects during the given time

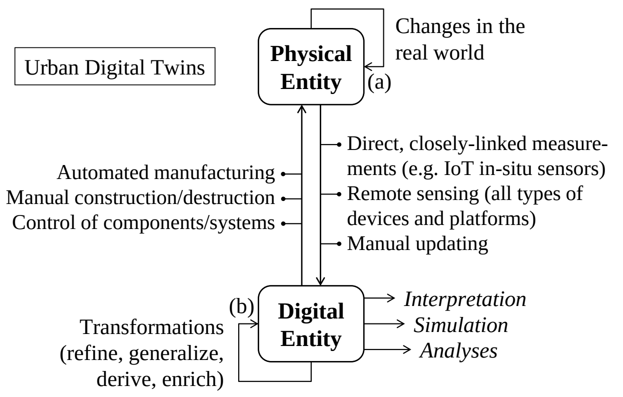
Urban Digital Twins
Physical Entity
Changes in the real world
(a)
Automated manufacturing
Manual construction/destruction
Control of components/systems
Direct, closely-linked measurements (e.g. IoT in-situ sensors)
Remote sensing (all types of devices and platforms)
Manual updating
(b)
Digital Entity
Transformations (refine, generalize, derive, enrich)
Interpretation
Simulation
Analyses

Figure 1. An overview of urban digital twins.

period. Thus, to help enable and maintain the continuous data flow between the real-world cities and their virtual city models, automatic and efficient change detection is required.

In the context of smart cities, virtual semantic 3D city models can be employed as a digital representation of a physical city in the real world. Most virtual semantic 3D city models nowadays are encoded in CityGML, which is an international open standard issued by the Open Geospatial Consortium (OGC) for the storage and exchange of semantic 3D city and landscape models (Gröger et al., 2012). It is an application schema of the Geography Markup Language 3 (GML3), which is an XML application for expressing spatial and geographical data. CityGML defines a common information model and data exchange format for most urban and rural objects such as buildings, bridges, tunnels, transportation, vegetation, etc. In contrast to virtual reality (VR) models that focus exclusively on the geometrical and graphical aspects, CityGML represents city objects in classes and relations with respect to their semantic, geometric, topolo-

[*] Corresponding author

gical and appearance properties (Kolbe and Donaubauer, 2021). Moreover, CityGML allows modelling objects in five different Level of Details (LoD 0 - 4). Finally, CityGML allows multiple syntactic representations of the same object including reusing already defined objects. For example, a shared wall between two adjacent rooms can be defined only once as a surface in the solid representing the first room and then referenced as a hyperlink (i.e. XLink) by the other solid representing the second room. This wall surface can also further be defined as a single polygon, or a composite surface that contains multiple smaller surfaces, etc. These characteristics of CityGML provide users more flexibility to define 3D city objects with highly complex spatial and semantic contents, but they at the same time also pose a great challenge for change detection between CityGML datasets in general.

As a result, only a few studies on change detection in virtual semantic 3D city models have been published so far, some of the earliest include (Bakillah et al., 2009) and (Redweik and Becker, 2015). Recently, (Nguyen et al., 2017) was one of the first to introduce graphs as a means to represent, store and compare CityGML datasets. The study provided detailed concepts on the mapping and matching process as well as an open-source implementation using a graph database (such as Neo4j).

However, finding changes is only one aspect of the problem, another crucial task is to make sense of them: what causes changes, why and when changes occur, what impact they could have on other objects, etc. Therefore, the seemingly simple question as what can be considered as a "change" in semantic 3D city models is not trivial. In a recent study, (Nguyen and Kolbe, 2020) proposed a multi-perspective approach to interpreting such changes by dividing them into different categories. These were then evaluated with respect to several groups of stakeholders. The proposed model of changes and stakeholders showed great potential but could however be further refined. Moreover, like city models, stakeholders' interests in different types of changes may also change over time. This requires a more flexible representation of the relevance relations between changes and stakeholders.

Therefore, this research aims to provide a conceptual framework to better interpret changes in semantic 3D city models encoded in CityGML. This includes a more refined model of changes and stakeholders as well as mathematical methods to represent the relevance relations between different groups of stakeholders and different types of changes. Section 2 reviews the related work and literature that are relevant to the research. Sections 3 and 4 explain how changes and stakeholders in the context of semantic 3D city models can be modelled respectively. Section 5 then proposes how the relevance relations between different changes and stakeholders can be mathematically represented. Finally, Section 6 concludes the paper with some discussions and future work.

2. RELATED WORK

The difference algorithms (used for *diff*) proposed by Hunt–Szymanski (Hunt and Szymanski, 1977) and Myers (Myers, 1986) are well-known solutions for the longest common subsequence problem in text files. These are however not applicable to markup languages such as the Extensible Markup Language (XML) in general and CityGML in particular, since they contain complex hierarchically structured information that cannot simply be considered as plain texts.

Relational databases have been a dominating workhorse in the data world and already employed in various application fields. Performing change detection in XML documents using their relational representation has been proposed by many studies, such as *DiffXML* (Chen et al., 2004) and *XANADUE* (Leonardi and Bhowmick, 2007). However, as explained by (Keller, 1997), (Golobisky and Vecchietti, 2005) and (Agoub et al., 2016), representing XML and especially CityGML objects using a relational data model has some known limitations. Firstly, the conceptual schema of CityGML specifies how real-world objects are classified in a class hierarchy as well as of which components they should be composed (Gröger et al., 2012). As a result, to avoid information loss, converting CityGML objects to corresponding relational entities may require a large number of additional tables and `JOIN` relations. Secondly, CityGML elements are defined in a complex hierarchical structure and often contain multi-level deep relationships to other objects. This causes an exponential growth in memory and runtime required to perform `JOIN` operations in a relational database.

Thus, an alternative approach is to perform change detection on the tree representation of XML documents. Published algorithms in this field include *LaDiff* (Chawathe et al., 1996) and *X-Diff* (Wang et al., 2003). Based on the latter algorithm, (Redweik and Becker, 2015) proposed concepts to detect changes between CityGML documents by considering both the geometric and semantic information stored in the tree representation of city models. However, since CityGML allows reusing and linking already defined objects, cycles or nodes having more than one incoming edge may exist. Thus, CityGML documents cannot generally be limited to a tree representation.

Graph representation of CityGML documents has gained attention in recent years. Using the *TGraph* technology, (Falkowski and Ebert, 2009) presented "a graph-based schema for integrated models of urban data". The research demonstrated how the semantic, geometric, topological and appearance information available in CityGML can be processed using compliant graph-based models. (Agoub et al., 2016) addressed the problems of using relational databases to store complex well-defined objects, attributes and relations. The authors then suggested employing a graph database (such as Neo4j and ArangoDB) to solve these problems. These studies however only focused on mapping and storing CityGML objects as graphs. They did not address how changes between such graphs can be detected.

(Nguyen et al., 2017) proposed an approach to mapping and matching CityGML documents using their graph representations in a graph database. It was one of the first to provide a working open-source implementation on change detection in semantic 3D city models. The mapping methods were capable of capturing semantic, geometric, topological and appearance information as well as complex hierarchical relations between objects, thus preventing any significant information loss. The matching methods were designed to fully utilize the mapped graph representations and their spatial properties to improve efficiency on finding changes. The research however left some fundamental questions open as what meaning such changes have, why they occur, what impact they may have on the dataset, etc. and how to handle changes that exist only in the datasets but do not reflect any real changes in the physical world. These problems cannot be solved based on the detected changes alone; different groups of involved stakeholders must also be considered, since they play a crucial role in determining the relevance, reasoning and meaning of all changes.

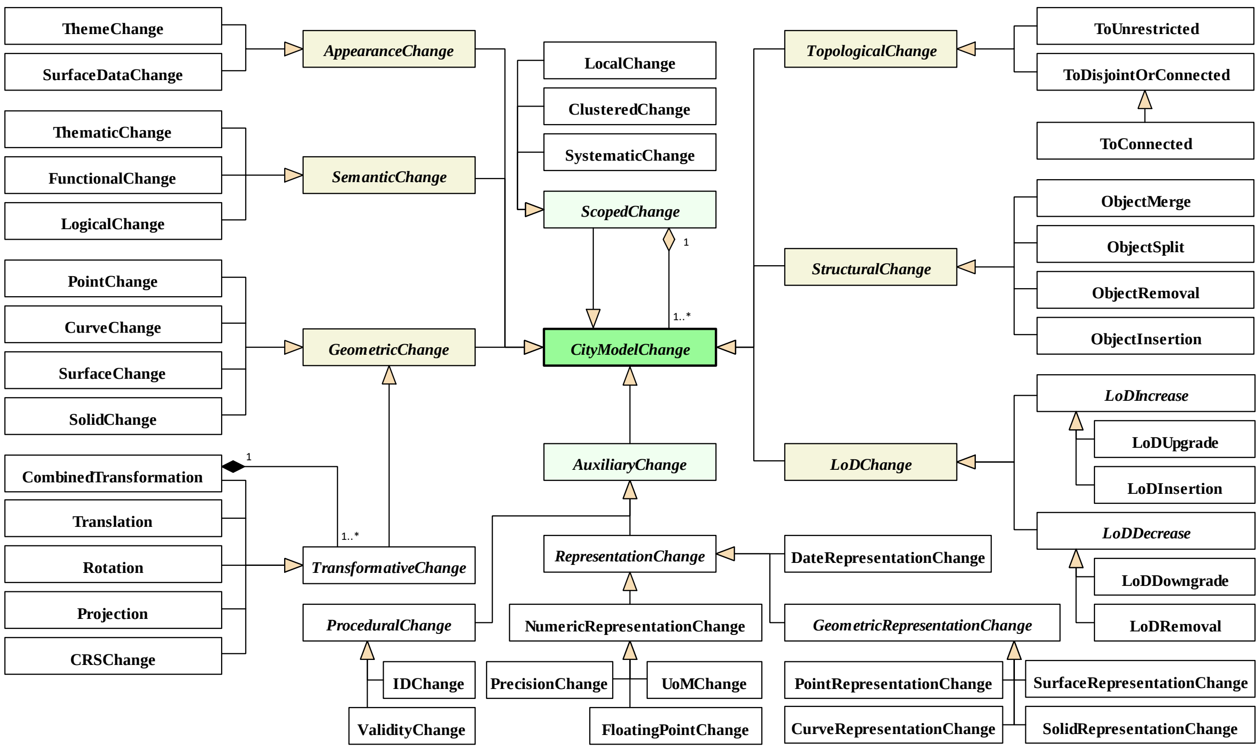

Figure 2. A UML class diagram of changes in semantic 3D city models.

In this context, (Nguyen and Kolbe, 2020) proposed a multi-perspective approach to interpreting detected changes by dividing them and stakeholders into several categories. The relevance relations between different types of changes and groups of stakeholders were then evaluated. The research was one of the first attempts to address how changes in the 3D models should be interpreted with respect to the real world. The suggested classification of changes and stakeholders could however be further refined. Therefore, this research aims to propose a more refined conceptual model of changes and stakeholders as well as quantifiable methods to represent their relevance relations.

3. CHANGES IN SEMANTIC CITY MODELS

Semantic 3D city models encoded in CityGML can represent most common urban objects and relations with respect to their semantic, geometric, topological and appearance information. Changes can therefore occur in many forms and places, each of which can have a different meaning and effect on the dataset. In order to provide a better understanding of the progresses made in the cities, the first crucial step is to systematically identify such changes. In this section, a conceptual model of all potential changes in semantic 3D city models is proposed. They are classified as appearance, semantic, geometric, topological, structural, LoD as well as auxiliary and scoped changes, each of which shall be explained in more details in the following sections. Figure 2 shows a UML class diagram of these changes.

3.1 Appearance Changes

The class ***AppearanceChange*** represents changes in the surface-based observable properties of city objects. These are not limited to visual data only and can be assigned to an arbitrary theme (such as solar potential, infrared radiation, pollution, etc.). In each Level of Detail (LoD), different appearances can be defined for different themes. Themes are thus used as identifiers to group appearances thematically. Changes in such themes may include a name change of a specific theme or re-assignment of themes among different LoDs, etc. These are represented by the class ***ThemeChange***.

On the other hand, changes can also occur in the appearances within a theme themselves. In CityGML, appearances are represented in surface data, which allow the modelling of simple surface properties with constant light reflection as materials and other coordinate-based surface properties as textures. Thus, changes in these objects are represented by the class ***SurfaceDataChange***. Note that the surface data are referenced to the corresponding geometric objects using their identifiers. This allows linking the closely related surface data and surfaces while being able to preserve the original geometric contents. Thus, this section only covers changes that occurred on the surface data side; changes of the surface itself are considered as geometric changes explained in Section 3.3.

3.2 Semantic Changes

Semantic information is a defining feature of semantic 3D city models and describes the thematic, functional and logical aspects of city objects. Thematically, an object can contain information available specifically for its type, and objects of the same type can be grouped together. This allows identifying city objects of different types (such as buildings, bridges, tunnels, etc.) with respect to their hierarchy (e.g. an office is a building, which is a city object). Functionally, an object can be a part of a collection or a collection of other parts. This is typically observed in the composition and aggregation relations. Finally, objects are structured according to their logical criteria and interrelationships (Kolbe and Donaubauer, 2021). City objects are represented using a set of predefined classes such as *Building*, *Bridge*, *Tunnel*, etc. Complex objects can be recursively subdivided into smaller parts. For instance, a building can be subdivided into several building parts, installations and rooms, which can then be further defined using boundary surfaces such as roof, wall and ground surfaces. Thus, changes in the above-mentioned aspects shall be represented by the classes ***ThematicChange***, ***FunctionalChange*** and ***LogicalChange*** respectively.

Note that some logical changes in a semantic 3D city models may indicate changes in the class structure and relationships in the specification of the encoding itself (such as buildings cannot be subdivided into building parts any more, etc.). These refer to the level M1 (i.e. model level) and higher according to the Model Driven Architecture (MDA) in UML in general and CityGML in particular (Kleppe et al., 2003, Kutzner, 2016). This means that such changes mostly occur between different versions of the encoding, such as between CityGML versions 1.0, 2.0 and 3.0. A comparison between these versions is out of the scope of this research. In contrast, the structural changes introduced in Section 3.5 refer to the level M0 only (i.e. instance level) according to the MDA and are classified using the modelling rules allowed by the same version of CityGML.

3.3 Geometric Changes

CityGML uses the GML3 representation of 3D geometries based on the ISO 19107 model (Herring, 2001). This includes a set of geometric primitives for each dimension from 0D up to 3D, namely: *Point*, *_Curve*, *_Surface* and *_Solid*. A 3D solid is bounded by 2D surfaces, while a 2D surface is bounded by 1D curves, and a 1D curve is confined by 0D control points. In CityGML, only the subclass *LineString* of *_Curve* and *Polygon* of *_Surface* are used for 1D and 2D respectively. These geometric primitives can then be combined to form more complex geometries, such as complexes, composites and aggregates. Thus, based on the dimensionality and the geometric primitives, geometric changes in semantic 3D city models are represented by the class ***GeometricChange*** and its subclasses ***PointChange***, ***CurveChange***, ***SurfaceChange*** and ***SolidChange***.

In addition, the class ***TransformativeChange*** is a specialization of the class *GeometricChange*. It represents most common affine transformations of geometric objects, such as translations, rotations and projections. Moreover, since each geometry is associated with a 3D coordinate reference system (CRS), a change in these spatial reference systems leads to a transformation of all point coordinates within the affected geometries. Therefore, such changes can be represented by the classes ***Translation***, ***Rotation***, ***Projection*** and ***CRSChange***. These can be arbitrarily combined to form more complex transformations represented by the class ***CombinedTransformation***. Compared to other geometric changes, the transformative changes are more difficult to detect but can provide a much deeper insight into the changes applied to the city models. For example, combined with the scopes introduced in Section 3.8, transformative changes can help detect some of the most common systematic changes, such as a height offset or changed spatial reference systems between entire datasets, etc.

Note that this section considers geometric changes as actual modifications to the numeric and geometric contents of objects. In case of changes that only affect the representations of the same geometric contents, please refer to the auxiliary changes in Section 3.7. In addition, modifications to the complex, composite and aggregate geometries often not only cause changes to the geometric but also the topological contents. Such topological changes shall be handled separately in Section 3.4.

3.4 Topological Changes

Like appearance, semantics and geometry, topology is another key information aspect available in semantic 3D city models. The geometrical-topological model of CityGML allows storing geometric objects with implicit topological relations. This is the case for complex objects formed from geometric primitives, namely aggregates, complexes and composites. **Aggregates** allow unrestricted spatial relations between their members, i.e. they can be disjoint, touching, overlapping, etc. In GML3, aggregates are defined as *MultiPoint*, *MultiCurve*, *MultiSurface* and *MultiSolid* depending on the dimension. On the other hand, a **complex** is a collection of geometric primitives, which topologically must either be disjoint or at most touching at their boundaries. **Composites** are a specialization of complexes and only allow geometric objects of the same dimension. Additionally, their components must connect and meet at their boundaries. Thus, composites are defined for geometries in 1D or higher dimensions and represented by the classes *CompositeCurve*, *CompositeSurface* and *CompositeSolid*. An example of the topological relations in 2D implied by geometric aggregates, complexes and composites are shown in Figure 3.

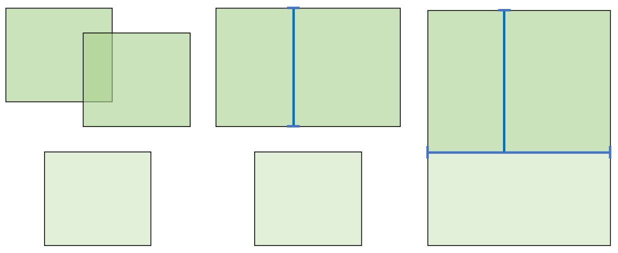
(a) *Multi-Surface* (b) *Geometric-Complex* (c) *Composite-Surface*

Figure 3. The topological differences in combined 2D geometries. Adapted from (Gröger et al., 2012).

In addition, CityGML also allows the explicit modelling of the topological adjacency relations between objects. A shared boundary (i.e. a curve boundary between touching surfaces, or a surface boundary between solids) may be defined only once and can then be referenced by other adjacent features and geometries. For example, a wall surface shared by two adjacent buildings can be defined only once in the solid representing the first building and then referenced by the second solid representing the other building. Such adjacency relations are realized using the shared geometries' identifier as hyperlinks (or XLinks). This helps not only reduce redundancy but also maintain the explicit topological relations between objects. Therefore, the classes ***ToUnrestricted***, ***ToDisjointOrConnected*** and ***ToConnected*** are proposed to represent changes that result in unrestricted, disjoint or connected, and explicit connected topological relations between city objects respectively.

3.5 Structural Changes

In CityGML, top-level features (such as buildings, tunnels, bridges, etc.) can be further divided into smaller components similarly to how they are physically structured in the real world. For instance, a building can consist of an arbitrary number of building parts, rooms and installations. They are further bounded by boundary surfaces, such as roof, wall and ground surfaces. Modifications to the structure of such objects are represented by the class ***StructuralChange***. Depending on the methods, such changes can be further divided into ***ObjectMerge***, ***ObjectSplit***, ***ObjectRemoval*** and ***ObjectInsertion***.

3.6 LoD Changes

CityGML supports multi-scale modelling with five different Levels of Details (LoD 0 - 4). A feature can have a single or multiple LoDs simultaneously. Thus, changes regarding the LoDs can occur in two directions: an increase and a decrease in the level of details, which are represented by the classes ***LoDIncrease*** and ***LoDDecrease*** respectively. An increase in the level of details of an object can be thought of as an increase in its highest available LoD. This includes an upgrade of a single available LoD to a higher one, or an additional higher LoD to the existing ones. Such are covered by the subclasses ***LoDUpgrade*** and ***LodInsertion*** respectively. For instance, consider the case, where an object was defined in LoD 1 and 2, and is now available in LoD 2, 3 and 4. The existence of the new highest LoD 4 indicates an increase in the level of details. This change can be interpreted as a combination of one upgrade from LoD 1 to 3 and one new added LoD 4. Changes within the LoD 2 may still persist but they do not reflect changes in the LoDs themselves and hence shall be covered by other types of changes instead (such as geometric, appearance changes, etc.). Similarly, a decrease in the level of details of an object can be thought of as a decrease in its highest available LoD. This again includes a downgrade of a single available LoD to a lower one, or the removal of the highest LoD from the existing ones. These are represented by the subclasses ***LoDDowngrade*** and ***LoDRemoval***.

3.7 Auxiliary Changes

Most of the changes discussed so far are closely related to the reality. Some changes however do not reflect this relation and are mostly relevant at the document level only. This can be caused simply by an automatic maintenance process on the dataset, but also by a change in the syntactic representations of objects allowed by the encoding standard. Such are identified by the class ***AuxiliaryChange*** and its two subclasses ***ProceduralChange*** and ***RepresentationChange***.

Most virtual semantic 3D city models nowadays are maintained and managed by a number of software solutions and automated processes, each of which has their own styles of formatting and storing city models based on the encoding rules allowed by CityGML. A change to a different program or a different version of the same system often leads to discrepancies in the produced documents. This can also occur when an automated maintenance procedure is performed on the datasets on a regular basis. Such are called procedural changes. The majority of procedural changes can be found in object attributes, such as changed identifiers and indicators (e.g. as timestamps, booleans, etc.) for the validity period or availability of the object in the dataset or database. These are represented by the subclasses ***IDChange*** and ***ValidityChange*** respectively.

CityGML allows different syntactic ways to define the same object. As a result, an object can have multiple representations, all of which are valid and describe the same information in the real world. Changes between such representations of the same objects are therefore considered auxiliary and covered by the subclass ***RepresentationChange***. Such representational changes can occur in almost every element and object that contains structured information. This is observed e.g. in floating-point numbers, which can be represented by combinations of significands, bases and exponents. For example, the numbers 1.23, 12.3×10^{-1} and 123×10^{-2} are different floating-point representations of the same value. Such are represented by the class ***FloatingPointChange***. In addition, measurements from real-world objects often have instrument and rounding errors that can be tolerated up to a certain threshold. A smaller tolerance threshold requires a higher precision in the measurements. In practice, values that vary within a small enough threshold of error tolerance are often not differentiated and thus can be considered as acceptable numeric representations of the same value. For example, both 1.234 and 1.235 represent the same value within an error tolerance of 10^{-3}. This is classified as a ***PrecisionChange***. Moreover, measured values are always assigned with a predefined unit of measurements (UoM), such as centimetre, metre, etc. Similarly, the above-mentioned error tolerance can also be defined with a UoM. This must be considered when comparing two measurements. For instance, with an error tolerance of $1\,\mathrm{mm}$, two measurements $1.001\,\mathrm{m}$ and $100\,\mathrm{cm}$ are considered equivalent. This is classified as a ***UoMChange*** combined with a precision change.

Date or time values are another example that can be represented differently while preserving the same contents. The date and time representation allows many variations, namely: in the calender used (Gregorian, lunar calender, etc.), the order and number of digits of days, months and years (such as in the form of day-month-year, month-day-year or year-month-day), characters used to separate the date and time components, whether 24-hour or 12-hour clock is used, etc. Such variants are covered by the class ***DateRepresentationChange***.

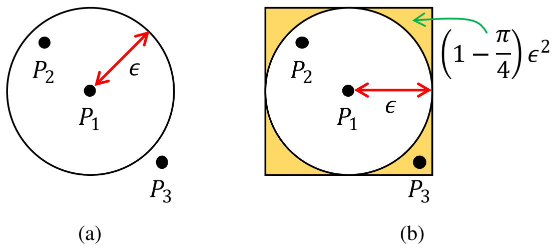

Figure 4. An illustration of a point's neighbourhood in 2D confined by an error tolerance ϵ based on the Euclidean distances (left) and individual point coordinates (right).

A large part of representational changes are observed in geometric objects. Two points are considered geometrically equivalent if they are located in the neighbourhood of one another confined by the error tolerance ϵ. Using the Euclidean distances, this neighbourhood is a circle in 2D or a sphere in 3D space with the reference point as centre and ϵ as radius. As shown in Figure 4a, the point P_2 is located within the neighbourhood of P_1. Hence, they are considered to be the same point. Alternatively to the Euclidean distances, the individual point coordinates can also be compared with each other. Since these are numbers, the same rules explained for the class *NumericRepresentationChange* also apply. Their neighbourhood now becomes a square in 2D and a cube in 3D space with the reference point as centre and 2ϵ as side length. As shown in Figure 4b, the point P_3 is now also located inside the new enlarged neighbourhood of P_1. They are thus considered equivalent. Compared to the Euclidean distances, which require expensive arithmetic operations such as multiplications and square roots, matching point coordinates only requires subtractions, which are significantly faster. The new neighbourhood shown in Figure 4b is larger than that of Figure 4a by $(4-\pi)\,\epsilon^2$ in 2D or $(8-4\pi/3)\,\epsilon^3$ in 3D space. With a very small value of ϵ, this difference is negligible in most cases. Geometrically equivalent points, whose coordinates have changed slightly, are covered by the class ***PointRepresentationChange***.

LineString objects in CityGML are defined by a set of control points. Since these can be collinear, matching two LineStrings should not be based on the number of control points alone. Figure 5 illustrates two different LineString representations of the same curve. The first LineString (shown in black) contains five control points, where P_1, P_2, P_3 and P_3, P_4, P_5 are collinear respectively. The control points Q_1, Q_2 and Q_3 of the second LineString (shown in blue) are located within the neighbourhood of points P_1, P_3 and P_5 respectively (as explained for the class *PointRepresentationChange*). Thus, before matching, existing collinear points (or points, whose distance to a common line satisfies the error tolerance ϵ) should be excluded with the exception of the two outer points (e.g. the points P_2 and P_4 of the first LineString shall be excluded while maintaining the outer points P_1, P_3 and P_5). Then, two LineStrings are considered geometrically equivalent, if all their remaining corresponding control points are also equivalent as explained in the class *PointRepresentationChange*. Such LineStrings are represented by the class ***CurveRepresentationChange***.

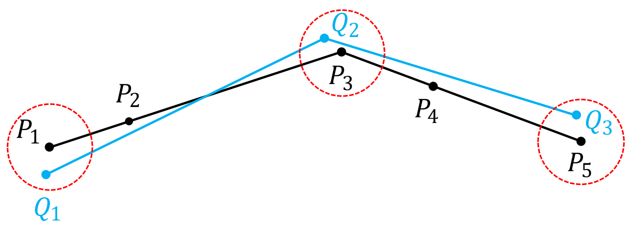

Figure 5. An illustration of two different representations of the same LineString object.

CityGML uses the special class *Polygon* to represent 2D surfaces. A polygon is defined by at most one exterior and an arbitrary number of interiors (i.e. holes). These boundaries are closed 1D curves and must be coplanar. Therefore, two coplanar polygons are considered geometrically equivalent, if most of their inner area confined by the exteriors and interiors overlaps. An ideal overlapping ratio per inner area should thus be close to 1. Note that, despite being 2D geometries, polygons can be given in 3D space, meaning they can have 3D spatial alignments and orientations. This must be considered while matching polygons. A representational change of the same polygon is classified as a ***SurfaceRepresentationChange***.

Similarly to polygons, a 3D solid is defined by at most one exterior and an arbitrary number of interiors. These boundaries are connected 2D surfaces and are allowed to touch at most at their sharing edges. Two solids are considered geometrically equivalent, if most of their inner volume overlaps. An ideal overlapping ratio per inner volume should thus be close to 1. This is however computationally expensive, since all calculations must be done in 3D space. Instead, a simpler alternative is to compute either the overlapping volume of the solids' 3D minimum bounding boxes, or the overlapping area of the solids' 2D footprints. This however does not ensure the geometric equivalence between solids (since different solids can have the same 3D minimum bounding box or 2D footprint), but can be utilized to help search for matching candidates first, before further 3D calculations can be applied. Different representations of the same solid are considered as a ***SolidRepresentationChange***.

3.8 Scoped Changes

A change can have an impact and an effect on a number of elements in semantic 3D city models. This is observed by analysing the scope of changes. A scoped change is an aggregation of other types of changes (e.g. semantic, geometric, auxiliary changes, etc.) proposed previously. Based on this collection size, scoped changes are further divided into three subclasses: ***LocalChange***, ***ClusteredChange*** and ***SystematicChange***. A local change only affects a specific attribute, element or object. For example, a changed generic attribute of a building's energy consumption has a limited scope and is considered local. On the other hand, a clustered change occurs over a number of objects that are spatially or semantically related. For instance, a newly constructed building may have an impact on the surrounding areas: new plants, new pedestrian paths, etc. Finally, a change is considered systematic when it is applied to all objects of the same type in the entire city model. For example, a global height offset between two city models leads to a systematic change in the height coordinates of all geometries (i.e. points, curves, surfaces and solids) in the entire datasets. The complexity to determine such scopes increases with their coverage. An indicator counting the affected elements (both semantically and spatially) can be stored for each change. Their scope can then be analysed by comparing the indicator's value with a set of predefined thresholds relative to the number of objects within a top-level feature as well as to the number of top-level features of the same type within the city model.

4. STAKEHOLDERS OF SEMANTIC CITY MODELS

Section 3 proposed a conceptual model covering different types of changes in semantic 3D city models. A change is however not simply a deviation in the data, it also has a meaning, a reason and a purpose in the real world. This is predominantly determined by the human factor. Thus, to provide a realistic understanding of progresses made in the cities, involved stakeholders must also be considered. Stakeholder analysis is however a dynamic field of study. Like cities, stakeholders and their interests in changes also vary over time. Hence, the objective of this research is not to provide a static model for all purposes and use cases, but rather to propose a flexible conceptual framework for identifying stakeholders and their interests in changes in semantic 3D city models. This focuses on the components that do not change frequently (i.e. the existence of the stakeholders themselves) and the aspects that do vary over time (e.g. motivations and interests in specific types of changes).

Figure 6 shows an example of how different groups of stakeholders can be identified in a class hierarchy. They can be divided into the following groups: public (government agencies, non-governmental organisations and individual citizens), constructive (constructors and architects), data (data producers, administrators and publishers), innovative (scientists and developers) and application stakeholders (urban planners, visualizers and analysts). Depending on the specific use cases, this may be further adjusted. On the other hand, stakeholders can also be identified with their motivations and interests in specific types of changes. These can however vary over time and be shared among many stakeholders, causing potential multiple inheritance in the class hierarchy that needs to be adjusted frequently. Thus, this research proposes representing the most commonly shared groups of motivations and interests as separate interfaces, such as *RealityMotivated* and *ModelMotivated* in the context of semantic 3D city models as shown in Figure 6, which can then be assigned to any classes of stakeholders. Such relations are flexible and can be reassigned without causing significant adjustments to the structure of the class hierarchy. The same can also be applied to all types of changes using the

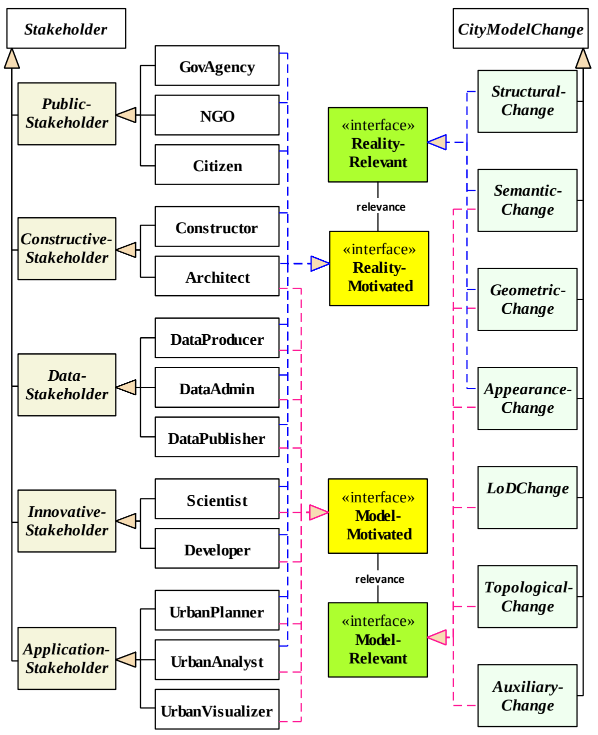

Figure 6. An example of a UML class diagram of stakeholders and their relations to changes in semantic 3D city models.

interface *RealityRelevant* and *ModelRelevant*. The stakeholders' interests in changes can then be represented by the *relevance* relations connecting the corresponding pairs of such interfaces. Moreover, it is important to distinguish the following three cases: (1) interests only in changes in the real world, (2) interests only in changes in the city models, and (3) interests in changes in the city models in order to conclude changes in the real world. In this context, Figure 7 describes an example of the relative interests of stakeholders as well as the relevance of different types of changes in a model - reality graph, where each aspect is represented by an axis. Close proximity between stakeholders and changes shown in this graph indicates greater level of interest and relevance between them.

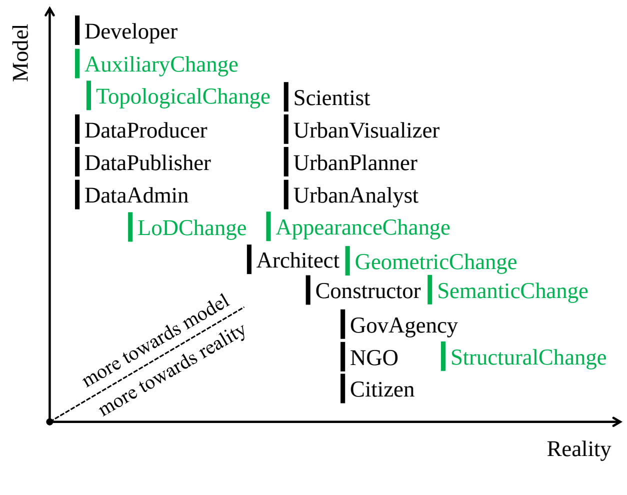

Figure 7. An example of the relative interests of stakeholders (black) and the relevance of changes (green) with respect to the real world and the city models.

5. RELEVANCE RELATIONS BETWEEN STAKEHOLDERS AND CHANGES

With both different types of changes and groups of stakeholders identified and conceptualized, the objective of this section is to define a quantifiable approach to describe and handle the relevance relations between stakeholders and changes. Firstly, for a stakeholder $s_i \in S = \{s_1, s_2, \ldots, s_m\}$ and a change $c_j \in C = \{c_1, c_2, \ldots, c_n\}$, where S and C are the set of all groups of stakeholders and types of changes respectively, their *relevance value* (as in how relevant c_j is to s_i) can be defined using the following *relevance function*:

$$\begin{aligned} f_{rel} : S \times C &\to [0,1] \subset \mathbb{R} \\ (s_i, c_j) &\mapsto f_{rel}(s_i, c_j) \end{aligned} \tag{1}$$

By applying the fuzzy logic, the relevance value $f_{rel}(s_i, c_j)$ is defined as a real value in $[0, 1]$, where 0 means c_j is not relevant to s_i by any means and 1 means c_j is absolutely relevant to s_i. Such can be assessed based on real-world data surveyed for changes and stakeholders. The relevance values of all combinations of s_i in S and c_j in C can be represented using either a *relevance graph* or a *relevance matrix*. Depending on the use cases, one approach may be preferable to the other.

5.1 Relevance Graph

The relevance graph $G_{rel} = (V, E)$ is a bipartite graph consisting of a set of nodes $V = V_S \cup V_C$, where V_S and V_C are a set of nodes representing all groups of stakeholders and types of changes respectively, and a set of (undirected) edges $E \subseteq \{e_{i,j} = (s_i, c_j) \mid s_i \in V_S, c_j \in V_C\}$. Each edge $e_{i,j}$ connecting stakeholder node s_i and change node c_j represents their relevance relation and has a weight $w_{i,j}$ equal to the relevance value $f_{rel}(s_i, c_j)$ as defined in Equation (1). Figure 8 illustrates an example of such relevance graph.

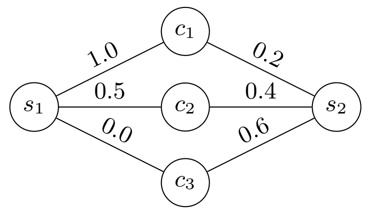

Figure 8. An example of a relevance graph for two groups of stakeholders and three types of changes.

5.2 Relevance Matrix

The relevance matrix $R_{rel} \in [0, 1]^{m \times n}$ is defined as follows:

$$R_{rel} = \begin{pmatrix} r_{1,1} & r_{1,2} & \cdots & r_{1,n} \\ r_{2,1} & r_{2,2} & \cdots & r_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ r_{m,1} & r_{m,2} & \cdots & r_{m,n} \end{pmatrix} = (r_{i,j}) \tag{2}$$

where $r_{i,j} = f_{rel}(s_i, c_j)$, $m = |S|$ and $n = |C|$ as defined in Equation (1). Thus, the relevance matrix describes the adjacency properties between the two partitions V_S and V_C of the graph G_{rel} defined in Section 5.1. An example of such matrix corresponding to the relevance graph illustrated in Figure 8 is shown as follows:

$$R_{rel} = \begin{pmatrix} w_{1,1} & w_{1,2} & w_{1,3} \\ w_{2,1} & w_{2,2} & w_{2,3} \end{pmatrix} = \begin{pmatrix} 1.0 & 0.5 & 0.0 \\ 0.2 & 0.4 & 0.6 \end{pmatrix} \tag{3}$$

6. CONCLUSION AND FUTURE WORK

This study is part of an ongoing research, whose ultimate goal is to enable automatic detection and meaningful interpretation of changes in the context of semantic 3D city models. This paper proposed a conceptual model for identifying and classifying different types of potential changes that may occur in CityGML documents. These are appearance, semantic, geometric, topological, structural, LoD, auxiliary and scoped changes.

A conceptual framework for modelling different groups of stakeholders in semantic 3D city models was also presented. The model focuses on the aspects that do not change frequently (i.e. the existence of stakeholders) and those that vary over time, namely their interests in different types of changes. A model - reality graph was then introduced to represent both stakeholders and changes based on their relative interest and relevance. This serves as one of the first attempts to identify, model and link changes with stakeholders based on their relevance relations.

Finally, two mathematical methods were proposed to represent the relevance relations connecting different groups of stakeholders and different types of changes in one place, namely the relevance graph and the relevance matrix. The relevance graph is preferable in use cases, where complex relations between stakeholders and changes exist, graphs are used to represent CityGML documents during the change detection process, or when high flexibility in the models is desired. On the other hand, the relevance matrix is preferable when many calculations in vector and matrix-space across stakeholders and changes are required, or when stability in the models is of importance.

In future studies, the proposed models and concepts can be employed and extended for further applications, such as complex change detection and interpretation, automatic updating and versioning. The relevance graph and matrix can be applied to current implementations for change detection in semantic 3D city models to assess their interoperability and usability.

ACKNOWLEDGEMENTS

We acknowledge the company CADFEM and the Leonhard Obermeyer Center (LOC) of the Technical University of Munich (TUM) for supporting this work.

REFERENCES

ABI Research, 2019. Digital Twins, Smart Cities, and Urban Modeling. Technical Report AN-5239, Allied Business Intelligence, Inc.

Agoub, A., Kunde, F., Kada, M., 2016. Potential of Graph Databases in Representing and Enriching Standardized Geodata. *Dreiländertagung der DGPF, der OVG und der SGPF*, 36.

Bakillah, M., Bédard, Y., Mostafavi, M. A., Brodeur, J., 2009. SIM-NET: A View-Based Semantic Similarity Model for Ad Hoc Networks of Geospatial Databases. *T. GIS*, 13(5-6).

Chawathe, S. S., Rajaraman, A., Garcia-Molina, H., Widom, J., 1996. Change Detection in Hierarchically Structured Information. *ACM SIGMOD Record*, 25(2), 493–504.

Chen, Y., Madria, S., Bhowmick, S., 2004. DiffXML: Change Detection in XML Data. *Database Systems for Advanced Applications*, Lecture Notes in Computer Science, Springer, Berlin, Heidelberg, 289–301.

Falkowski, K., Ebert, J., 2009. Graph-based Urban Object Model Processing. *City Models, Roads and Traffic (CMRT'09): Object Extraction for 3D City Models, Road Databases and Traffic Monitoring-Concepts, Algorithms and Evaluation, Paris, France*, 9.

Golobisky, M., Vecchietti, A., 2005. Mapping UML Class Diagrams into Object-Relational Schemas. *Proceedings of ASSE*, 1, 65–79.

Gröger, G., Kolbe, T. H., Nagel, C., Häfele, K.-H., 2012. OpenGIS(R) City Geography Markup Language (CityGML) Encoding Standard, Version 2.0. OGC.

Herring, J. R., 2001. The OpenGIS Abstract Specification, Topic 1: Feature Geometry (ISO 19107 Spatial Schema), Version 5. OGC Document Number 01-101. OGC.

Hunt, J. W., Szymanski, T. G., 1977. A Fast Algorithm for Computing Longest Common Subsequences. *Commun. ACM*, 20(5), 350–353.

Keller, W., 1997. Mapping Objects to Tables - A Pattern Language. *Proc. Of European Conference on Pattern Languages of Programming Conference (EuroPLOP)'97*.

Kleppe, A. G., Warmer, J., Bast, W., 2003. *MDA Explained: The Model Driven Architecture: Practice and Promise*. Addison-Wesley Longman Publishing Co., Inc., USA.

Kolbe, T. H., Donaubauer, A., 2021. Semantic 3D City Modeling and BIM. W. Shi, M. F. Goodchild, M. Batty, M.-P. Kwan, A. Zhang (eds), *Urban Informatics*, The Urban Book Series, Springer, Singapore, 609–636.

Kutzner, T., 2016. Geospatial Data Modelling and Model-driven Transformation of Geospatial Data based on UML Profiles. Dissertation, Technical University of Munich, Germany.

Leonardi, E., Bhowmick, S. S., 2007. XANADUE: A System for Detecting Changes to XML Data in Tree-unaware Relational Databases. *Proceedings of the 2007 ACM SIGMOD International Conference on Management of Data*, SIGMOD '07, Association for Computing Machinery, New York, NY, USA.

Myers, E. W., 1986. An O(ND) Difference Algorithm and Its Variations. *Algorithmica*, 1(1-4), 251–266.

Nguyen, S. H., Kolbe, T. H., 2020. A Multi-Perspective Approach to Interpreting Spatio-Semantic Changes of Large 3D City Models in CityGML Using a Graph Database. *Proceedings of the 15th International 3D GeoInfo Conference 2020*, ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences, VI-4-W1-2020, ISPRS, London, United Kingdom, 143–150.

Nguyen, S. H., Yao, Z., Kolbe, T. H., 2017. Spatio-semantic Comparison of Large 3D City Models in CityGML Using a Graph Database. *Proceedings of the 12th International 3D GeoInfo Conference 2017*, ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences, IV-4/W5, ISPRS, Melbourne, Australia, 99–106.

Redweik, R., Becker, T., 2015. Change Detection in CityGML Documents. *3D Geoinformation Science: The Selected Papers of the 3D GeoInfo 2014*, Springer, 107–121.

Wang, Y., DeWitt, D. J., Cai, J. Y., 2003. X-Diff: An Effective Change Detection Algorithm for XML Documents. *Proceedings of the 19th International Conference on Data Engineering 2003*, 519–530.