Comparison of versioning methods to improve the information flow in the planning and building processes

Helen Eriksson1,2 | Jing Sun3 | Väino Tarandi3 | Lars Harrie2

1Lantmäteriet, Gävle, Sweden
2Department of Physical Geography and Ecosystem Science, Lund University, Lund, Sweden
3Department of Real Estate and Construction Management, KTH Royal Institute of Technology, Stockholm, Sweden

Abstract

Version management is a prerequisite for digital information flow between phases in the planning and building processes. Information evolves over time and many parties retrieve information from the various phases. The aim of this article is to evaluate versioning methods, focusing on geodata buildings in the 3D cadastre process. The main attention in the evaluation is on the comprehensive ISO standard Product Lifecycle Support (PLCS). PLCS is evaluated against two simpler versioning methods, the versioning in CityGML 3.0 and a modified Git versioning method implemented in CityJSON. CityGML 3.0 fulfills all but one requirement and PLCS meets all. The methods vary in complexity; the Git proposal is a simple solution, easy to implement and maintain, while PLCS includes all functionalities and is complex to implement. There is a trade-off between number of functionalities and complexity, it is therefore important that the intended purpose determines the choice of versioning method.

1 | INTRODUCTION

Standardized, digital and effective planning and building processes have become increasingly important during the last decade, for example as a solution to shortage of housing in many urban areas. One deficiency for such processes today is the handling of digital information. Information is not always shared between the actors involved...
and between the different phases (e.g., planning, design, construction and facility management) (Halfawy, Froese, Vanier, & Kyle, 2004; Hallberg & Tarandi, 2011). If information is shared, the handover between the phases is often in paper format or in inappropriate digital formats that are not machine readable, such as PDF files. This lack of appropriate sharing of digital information leads to duplication of work and possible time delays. There are several reasons for these shortcomings in information sharing, including technical, legal and business reasons.

To facilitate information sharing, the actors must have a common view of the planning and building processes. There is also a desire that an object (e.g., a building) not only includes information needed in the current phase, but also information that can be used in other phases of a process (Kiviniemi & Codinhoto, 2014). In this article we concentrate on the latter (i.e., how the information within the planning and building processes should be modelled). The time dimension is important here as real-world objects evolve over time. The core of this study is how we should deal with the time dimension to facilitate information sharing between phases in the planning and building processes.

Building information is a vital part of the planning and building processes and is the focus of this article. To describe the details of a building, building information modelling (BIM; Borrmann, König, Koch, & Beetz, 2018) can be used. However, BIM is not suitable to use for application at the city level, as a BIM model usually gives a detailed description of a single building. For that purpose, 3D city models (3D geodata) that present buildings over a larger area are more appropriate. Today, most BIM data and 3D city model data are temporal snapshots that describe a state at a certain time. If there are no links between the snapshots in time (through common object identifiers and versions), it is not possible to describe a process using the data. Even though the snapshots of data would be linked, the snapshot representation still has shortcomings, as it is not possible to get information about what happened in between the snapshots. To facilitate process modelling we could use for example an object lifeline representation (based on modelling changes over time for objects, attributes and relations; Worboys & Duckham, 2004), an event-based representation (based on process calculus where everything is seen as events; Worboys, 2005) or a lifecycle data approach (as used in product lifecycle management [PLM] systems; Stark, 2015).

The major outcome of this article is the comparison of version management methods focusing on 3D city models, both to get a better understanding of their functionalities and because this knowledge can contribute to the standardization work for 3D city models. The specific aim is to compare and evaluate three version management methods for 3D city models. The main focus in this evaluation is on an implementation of the Product Lifecycle Support (PLCS) standard (ISO, 2012a). PLCS is a comprehensive lifecycle standard originally developed for the process industry, but it has also been used in the BIM field (Tarandi, 2011, 2015). This standard is examined and its applicability for versioning of 3D geodata city models is evaluated. The PLCS standard is exhaustive and probably capable of solving all requirements of versioning of 3D city models. The question is whether this really is needed or if it is sufficient to use a simpler versioning method. Therefore two other versioning methods for 3D city models are included in the evaluation: the new versioning module and lifecycle representations in the proposed version 3.0 of CityGML (Kutzner & Kolbe, 2018) and the modified Git versioning control technique implemented in CityJSON (Vitalis, Labetski, Ohori, LeDoux, & Stoter, 2019).

The 3D cadastre process is used in the study to exemplify the need for version management in 3D city models and in BIM models. The cadastre process has several requirements for version management (see Section 3) and out of these, three requirements were chosen for the evaluation of versioning methods: (a) retrieve information from a specific moment in time (transaction time and valid time); (b) have different simultaneous alternative descriptions of a 3D city model; and (c) keep track of the synchronization of building information between BIM and geodata models. These requirements are generic (i.e., applicable for many other parts in the planning and building processes).

The article is organized as follows. Section 2 explains why it is important to have a continuous digital representation of buildings in the planning and building processes. In Section 3 we state the requirements of temporal modelling to appropriately describe a process using 3D cadastre (see Section 2.3) as our test case. Sections 4–6 are devoted to the three temporal modelling and versioning techniques used in this study. A general description
of the techniques, as well as an evaluation of the methods against the stated requirements for our test case, are provided. The article concludes with a discussion and conclusions.

2 | THE NEED FOR DIGITAL REPRESENTATION OF THE BUILDINGS IN THE PLANNING AND BUILDING PROCESSES

2.1 | The evolution of a building model

The digital representation of a building is evolving in parallel with the physical real-world building. During the different phases the architecture, engineering and construction (AEC) companies will continuously develop and update a BIM model; from an initially simple architectural model that gradually becomes an advanced BIM model. In several phases, this model is passed to other actors, and some actors—such as municipalities—are more interested in a comprehensive geodata description of the building. This means that throughout the process there will be two parallel descriptions of the building, a BIM description and a geodata description. However, to achieve a process-oriented digital information flow, it is important that the buildings are treated as the same real-world objects throughout their whole lifecycle, with information that evolves over time and that can be described with different attributes and geometries either as BIM or geodata (Figure 1).

Figure 2 provides a simplified description of the digital information flow between the phases in the planning and building processes. It should be noted that this figure does not describe the current situation but rather is a model for how information could be exchanged in the future (an example of this is described further in Section 3.1). The figure illustrates that 3D buildings can be represented as both BIM and geodata, with different geometric and semantic content where the level of detail will increase over time. That is, the information content will constantly change due to both changes in the work processes and updates from the surveying units (normally part of a municipality) that collect geodata during the whole process, both before and after a building is constructed.

To accurately describe the whole process, we need unique identifiers and appropriate temporal information. Besides this, the versioning of BIM and geodata models should also be synchronized. This is especially important

**FIGURE 1** The lifecycle of a building (building figures from Biljecki, Ledoux, & Stoter, 2014 with LOD descriptions according to CityGML 2.0)
at stages where legal decisions are made on the data (e.g., if a building permit decision is performed on both BIM and geodata—for an in-depth description of this, see Donkers, Ledoux, Zhao, & Stoter, 2016; Noardo et al., 2019; Olsson, Axelsson, Hooper, & Harrie, 2018). This calls for methods to integrate BIM data and geodata which has been extensively studied during the last decade (see e.g., de Laat & van Berlo, 2011; Deng, Cheng, & Anumba, 2016; El-Mekawy, Östman, & Hijazi, 2012; Isikdag & Zlatanova, 2009; Liu et al., 2017; Stouffs, Tauscher, & Biljecki, 2018; Sun, Olsson, Eriksson, & Harrie, 2019).

City officials will need information about both the current situation and certain previous moments in time to be able to perform their work. This could, for example, be information about a building from the date that the building permit was approved or from when the property formation application was approved. To include such a temporal aspect of the information flow in the process is a prerequisite for this to be a digital process from which actors in all phases can retrieve the relevant information.

2.2 | Version management

A version management system is required to facilitate the information flow in the planning and building processes. Manmade feature types such as buildings, roads and bridges need more sophisticated versioning methods than natural objects like rivers, lakes and land use (that might not need versioning at all). One important design issue is to determine which timestamps are required. Is time for updates of objects in the database, so-called transaction time, required? Or is it time for real-world events, so-called valid time, that is required? Or are bi-temporal time representations required, that is when both transaction time and valid time are included (Figure 3).

Often, both BIM and geodata information about manmade city objects is used in the planning and building processes. This information should be versioned, and the versioning of BIM and geodata objects should be separated, especially if it concerns the lifecycle of events that trigger new versions for real-world objects. Such events...
are often legal decisions—for geodata this could be when the building permit is accepted or the building is completed and inspected; and for BIM when the "As Built" model or other signed agreements between actors from different sectors are accepted. There are also events that should trigger a new version for both the BIM and the geodata model so that the models become synchronized (Figure 4). This could, for example, be when a property formation application is accepted and the resulting BIM model is generalized and transformed to a geodata model.

Once it is decided that versioning is needed, one must decide what should trigger a new version; that is should all changes result in a new major version, or should there be different levels of the versions, so-called semantic versioning (e.g., 1, 1.1, 1.1.1, 1.2). In that case, what event should trigger a major or minor version to change? Also, should the geometry of an object and the attribute information about the object be included in the same versioning or should they be versioned separately?

Another aspect to consider is if it should be possible to have several simultaneous alternative descriptions of objects in a 3D city model. Reasons for having this could, for example, be to represent different alternatives for a
new residential area; or to show alternative ways of describing 3D property units when exploiting an underground area containing tunnels and sewer systems (Figure 5).

This study focuses on the information flow in the 3D cadastre process, and mainly on building and 3D property unit information. A question here is if the versioning of the building and 3D property units should be separated or not, and if the geometry should be included in this or also versioned separately.

## 2.3 The 3D Cadastre Process

As an example of the need for version management in 3D city models and BIM models, we use the 3D cadastre process. The number of 3D property unit registrations has increased as a result of the high density of many cities and the need for better utilization of spaces, both within buildings and below ground (Paulsson & Paasch, 2013). Even though there has been much research in legislation, registration, integration, modelling, visualization and management of the 3D cadastre process (Aien, Kalantari, Rajabifard, Williamson, & Wallace, 2013; Atazadeh, Kalantari, Rajabifard, Ho, & Ngo, 2017; Cemellini, van Oosterom, Thompson, & de Vries, 2020; Donkers et al., 2016; Hjelmblom, Paasch, Paulsson, Edlund, & Bökman, 2019) there are still many issues to be resolved. In the perspective of this study we are interested in one of the open issues: digital information sharing in the 3D cadastre process, and especially how this information sharing sets requirements on version management of the digital data.

Today, much of the cadastral information sharing between actors—such as AEC companies, cadastral surveying units and city surveying units—is in non-machine-readable formats, such as PDF files. Furthermore, the 3D cadastre is still defined in 2D drawing and textual descriptions, and the cadastral index maps are still in most cases 2D maps (El-Mekawy, Paasch, & Paulsson, 2015). Considering the increasing numbers of 3D property unit registrations, this makes it quite challenging to manage the digital representation of the buildings efficiently.

There have been several studies in using BIM models and 3D city models in the 3D cadastre process. Atazadeh et al. (2017) implemented three BIM-based models (purely legal, purely physical and integrated models) of multi-storey buildings to investigate their performance. The results illustrated that the integrated models could provide a more visual communication of the location of legal boundaries. Góźdź, Pachelski, Van Oosterom, and Coors (2014) have implemented and extended a CityGML–LADM ADE model in order to address both physical aspects and legal counterparts in 3D cadastre. The study results visualized the legal spaces on the LOD1 level and demonstrated the benefits of providing relations between spatial building objects.
From a technical aspect, both BIM and 3D city models can be utilized as physical models to be integrated with original cadastral information or legal models. A BIM could be used for visualization of the extent of the 3D property units, but for an overview of 3D property units in larger areas, a 3D city model is required (Sun, Mi, Olsson, Paulsson, & Harrie, 2019). If the 3D city model has links to other legal registers, it could also be used as a 3D cadastral index map. Sun, Mi, et al. (2019) integrated models conforming to the standard Industry Foundation Classes (IFC) (ISO, 2013) and CityGML models with cadastral information stored in a legal model conforming to the Land Administration Domain Model (LADM) (ISO, 2012b) at the conceptual level and data level. The reason was that IFC is better suited for registration of the 3D property units; while CityGML can provide a general visualization of 3D cadastral, that could be used in a 3D cadastral index map for the area.

3 | REQUIREMENTS OF VERSION MANAGEMENT

This section starts with a future scenario of the 3D cadastre process, and based on this scenario requirements of the version management of the 3D building geodata are formulated. Even though we use the 3D cadastre process as a base for defining the requirements, we can anticipate that similar requirements would have been found if we had exemplified it with another process such as the building permit process.

3.1 | A future scenario of the 3D cadastre process

Figure 6 illustrates a simplified version of a future vision of the 3D cadastre process. Its aim is to motivate the version handling and not to describe all the details in the process. The process starts with the applicants preparing an application based on a BIM model (around the “As Design” stage). The application is submitted to an official cadastre unit that evaluates the application. The BIM model (including the 3D property unit) is converted to geodata and a

**FIGURE 6** Overview of the 3D property formation process and how information can be used for visualization
3D geodata building object (A in the figure) is created as part of a 3D city model. The information is also used for a preliminary update of the 3D cadastral index map (B in the figure); this index map is merely a subset of the 3D city model complemented with property unit information. The preliminary index map is then used in the evaluation of the application. This example anticipates that the application is approved (i.e., there is a positive decision about the 3D property formation). This entails that the 3D cadastre register is updated and that later the 3D cadastral index map can be updated (C in the figure). At a later stage the building is finalized and a BIM model (in the "As Built" stage) is created and delivered to the municipality. The new BIM model is used to update both the 3D geodata building (D in the figure) and the cadastral index map (E in the figure). Later on, as part of a regular survey, the building is measured by terrestrial geodetic measurements, laser scanning and/or aerial photographs. These new observations are used to update the 3D geodata building (F in the figure) as well as the 3D cadastral map (G in the figure).

The process in Figure 6 sets requirements on the version management of the BIM models, the 3D geodata building models and the property units, and by that also the 3D cadastral index map. In this study, the version management of the BIM models is excluded. There are three versions of the 3D geodata building (A, D, F in the figure) and four versions of the 3D cadastral index map (B, C, E, G in the figure).

In the test case we evaluate whether it is possible to version the 3D geodata buildings and the 3D cadastral index map. We also want to evaluate if the geometry of the building and its attribute information can be versioned separately from each other in any of the tested versioning methods (which could be desirable when the geometry is changed after a photogrammetric survey). This is illustrated in Figure 7, where we distinguish between building and cadastral index map information and where also the geometry of buildings is treated separately. That is, the change of versions for steps A–G in Figure 6 is described in Figure 7.

### 3.2 | General version management requirements in the 3D cadastre process

Based on the descriptions in Sections 2 and 3.1, we identify five requirements on version management in the 3D cadastre process.

A 3D property unit is defined as a volume within a physical building, and the physical building and the 3D property units should be versioned separately as these two object types can change independently. This leads us to the first requirement: there must be a link between the 3D property unit and the 3D building, as 3D property units are often delimited by physical parts of the building, such as walls, floors and ceilings. This requirement is true for both BIM and geodata information.

The second requirement is that it should be possible for a city official at the cadastral unit to retrieve information about an old version of a building or a cadastral index map as this information could be required for another property formation application.

| A | B | C | D | E | F | G |
|---|---|---|---|---|---|---|
| **Building** | 1 | 1 | 1 | 2 | 2 | 2 |
| **Building geometry** | 1 | 1 | 1 | 2 | 2 | 3 | 3 |
| **Property unit** | - | 1 | 2 | 2 | 3 | 3 | 4 |

**Figure 7** Versioning of buildings, property units and the separate versioning of the geometry of building in the 3D property formation process. The numbers represent the number of the version.
Sometimes there is a need to visualize many different solutions of a 3D city model or a 3D cadastral index map. Therefore, the third requirement is that it should be possible to describe many different simultaneous alternative descriptions of 3D city models and 3D cadastral index maps. The requirement also includes the possibility of merging alternatives.

The fourth requirement is that versioning of the geometry of the building should be separated from the versioning of the remainder of the information (e.g., function, storeys above and below ground and roof type). That is, there is a difference between the physical building and an observation of a physical building (e.g., of the geometry). This requirement holds for both BIM and geodata information.

When a BIM model is simplified and transformed to a geodata model, the versioning of these two models should be synchronized, and to be able to trace the source, there should be a link from the geodata model to the BIM model. This is the fifth and final requirement.

### 3.3 Test requirements in the case study

The version management requirements in Section 3.2 are quite general and in order to get more precise requirements for our test case, we narrow the requirements to how versioning must be handled on the 3D geodata models. The following requirements are chosen to be included in our evaluation. It should be possible to:

1. Retrieve information from a specific moment in time about
   - buildings, building geometries and 3D property units in a 3D city model (i.e., transaction time for the objects);
   - real-world buildings and 3D property units (i.e., valid time for the objects).
2. Have different simultaneous alternative descriptions of a 3D city model (that is, versions can fork and merge).
3. Keep track of the synchronization of building information between BIM and geodata models.

### 3.4 Comparison of version management methods

In this study we evaluate three versioning methods based on the version management requirements described above. Our main focus in the evaluation is on PLCS, a comprehensive and well-established ISO standard originally developed for the process and manufacturing industry. The PLCS standard has also been used to get a more process-oriented approach to the information flow for objects in BIM models (Tarandi, 2011, 2015), and in this study we examine and evaluate if PLCS can also be used for 3D geodata city models. The PLCS standard is implemented in a database in a testbed and tests are performed where both 3D geodata and BIM datasets are inserted and updated in the database. The other two versioning methods that are evaluated are both designed for versioning of 3D city models: the versioning module in CityGML 3.0 (Chaturvedi, Smyth, Gesquière, Kutzner, & Kolbe, 2016), where bi-temporal timestamps, versions and links between the versions and transactions can be described; and a modified Git versioning method implemented in CityJSON (Vitalis et al., 2019), which includes transaction date and versions. There are no implementations of the new versioning module in CityGML 3.0 or of the modified Git versioning method. Therefore, the evaluation of these methods is performed on the information models, but the same versioning information as in the evaluation of the PLCS standard is used.

The reasons for selecting these three methods are that it is interesting to compare an internationally accepted standard that originates in the manufacturing industry and also has been used in the BIM field with two methods that are designed for versioning of 3D city models. These two versioning methods are also interesting to compare as the CityGML 3.0 versioning module is more complex and strives to overcome the shortcomings in the version management in CityGML 2.0, while the main purpose of the modified Git versioning method is to have a simple and pragmatic method for the versioning of 3D city models (as most 3D city models are currently not versioned at all).
4 | LIFECYCLE MANAGEMENT METHODS IN A PLM SYSTEM

4.1 | Background

The term "lifecycle" can be used for planning, collecting, storing and publishing data. Another usage of the term comes from the product management domain and is referred to as product lifecycle (PLC). It describes data for supporting the production process so that data are transferable between the design, construction and maintenance phases of a product. This way of modelling lifecycle data has become increasingly common in the BIM domain also, which affects geodata in those cases where BIM data and geodata are integrated.

4.1.1 | PLM in the manufacturing sector

In the manufacturing sector, PLM has been used for many years to provide the correct information about a product at the correct time and in the appropriate context. An issue when using PLM is that it primarily focuses on the design and manufacturing phases; almost no product information is transferred to the distribution, use, support, recycling or disposal phases (Terzi, Bouras, Garetti, & Kiritsis, 2010), which is also a prerequisite for a sustainable product lifecycle management (Vadoudi, Allais, Reyes, & Troussier, 2014). Manufacturing processes often include both product data management and enterprise resource planning systems, and these systems describe products semantically differently. This can cause interoperability issues in PLM, and make it difficult to create a global description of the whole product development process (Paviot, Cheutet, & Lamouri, 2011). The authors see the PLCS standard as a possible means to overcome these interoperability issues.

4.1.2 | Lifecycle management in the planning, building and construction sector

Owen (2009) and Owen et al. (2013) describes a holistic vision to overcome the ineffectiveness in terms of time and cost in the construction lifecycle. This integrated design and delivery solution (IDDS) contains four key topics: (1) collaborative processes; (2) integrated information and automation systems; (3) enhanced skills; and (4) knowledge management. Tarandi (2015) refers to the integrated information topic of IDDS when describing the BIM Collaboration Hub, a prototype platform for creating BIM repositories that makes it possible to store BIM information created using different software and in different phases of the construction process in the same place. It is based on the PLCS standard, and BIM data conforming to the PLCS standard is mapped to this. Hallberg and Tarandi (2011) also use BIM together with IFC and PLCS, which makes the information accessible and readable by anyone. This can facilitate the creation of a detailed long-term plan for maintenance, repair and rehabilitation as all information is more easily available for facility management (FM) applications. To achieve this, actors in earlier phases of the process need to create a BIM that is also suitable for later stages. This is not always the case. Kiviniemi and Codinhoto (2014) describe that even though BIM is successfully used in the design and construction processes, BIM is rarely used in FM activities. This is both because the BIM does not include all the information needed in FM, and due to organizational issues such as cultural barriers and lack of legal frameworks.

4.1.3 | Lifecycle management methods in PLCS

The PLCS standard was originally developed for the process and manufacturing industry, for example to support complex products such as planes and ships. It includes information required for through-life configuration and change management of a product and it supports a seamless flow from the design and manufacturing phases to the phases for product support and feedback of usage and change. Product structures, assemblies and breakdowns;
product through life; specification and planning of activities; activity history of a product; and product history can, for example, all be represented by the PLCS standard.

The PLCS standard has also been used to get a more process-oriented approach to the information flow for objects in BIM models. For example, Tarandi (2015) shows that this is possible by developing the BIM Collaboration Hub, where the PLCS standard is implemented in a data collaboration environment called ShareAspace (https://www.eurostep.com/products/shareaspace/), an environment used in many production industry projects. Within a Swedish Testbed project (https://www.smartbuilt.se/projekt/innovationer-och-nya-tillaempningar/testbaedd/#) that is part of the Smart Built Environment programme, ShareAspace was extended to handle BIM data (IFC) and 3D city model data (CityGML) and also to handle links between BIM data and 3D city model data (Figure 8).

Figure 9 gives a high-level overview of the PLCS standard. PLCS supports generic objects of a limited number of entities. The basic entity is the product which can be sub-classed into parts, requirements, documents, slots, interfaces, systems and breakdowns (e.g., physical and functional).

The entities are specialized by external classification and can support any kind of objects. For the classification, an international or national classification system like the Swedish CoClass (CoClass, 2016) should be used.

The design versions of a building or building element are represented by a physical breakdown element and the planned production versions and realized product versions are represented by individual planned_product versions and realized_product versions.

All versions of an object have a start date when stored in the platform. The objects are linked to their parents with a link that has effectivity with start date set but an empty end date. The end date will be set when a new version is created, and this will end the old version in the structure (see Figure 14 later). The old versions will be stored in the platform for future access.

4.2 | Evaluation of requirements
4.2.1 | Materials and methods

To evaluate the versioning in a PLM system, we made practical implementations. The extended implementation of ShareAspace from the above-mentioned Testbed project was used here.
The study area was the neighbourhood around a new building, Multihuset, in Malmö, Sweden (Figure 10). Multihuset is divided into two property units: Bryggan 1 containing housing and Bryggan 2 containing grocery store, health centre, parking, kindergarten, offices and pharmacy.

The following datasets were used:

- BIM models of Multihuset from three stages in the construction process (the last is an almost complete relational handling BIM model), delivered in IFC 2.3 format from the construction company NCC.
- A cadastral dossier of the 3D registered property in Multihuset provided by the cadastre unit at Malmö municipality. Included in the dossier were construction maps of the storeys of Multihuset that specified the extent of the 3D properties. The 3D property units were digitized on the fifth floor and stored as IFC elements IfcSpace and IfcZone (see Sun, Mi, et al., 2019 for details).
- A 3D city model in SketchUp format for the area around Multihuset provided by the surveying unit at Malmö municipality. This model was converted to CityGML 3.0 using the ETL tool FME (https://www.safe.com/).
- CityGML files with the level of detail 2 (LOD2) translated from the IFC models of Multihuset using an FME script (an extended version of the script described in Olsson, 2018).

4.2.2 | Use cases in the Testbed project

Results from two of the use cases from the Testbed project were used in this test case. The first use case describes the lifecycle handling of 3D geodata buildings during the planning and building process at a municipality (e.g., building permits and property formation). Here the geodata building Multihuset is mapped to the IFC format and...
then imported into ShareAspace as spatial breakdown objects (Figure 11). The design version BIM model in IFC format is also imported into ShareAspace.

The second use case illustrates how 3D property units can be versioned over time through the design, construction, planning, construction and operation phases. Here, a BIM model of Multihuset from early in the design phase is stored in ShareAspace by the designer. The city official exports the BIM model from ShareAspace and links 3D property units to the individual building elements and spaces in the BIM model to get the connecting globally unique identifiers (GUIDs) (Figure 12). The 3D property unit is then imported and linked to the building elements and spaces via the retrieved GUIDs.

A virtual building storey is created (see 3D prop storey in Figure 8) to link the 3D property to the relevant building elements and spaces from one or more buildings on one or more 2D cadastre units. The 3D property unit is linked to the Municipality element in this test case (Figure 8). This structure is also shown in Figure 13 for the phase as_planned. In the same way, the 3D property unit can link to the relevant versions of building elements and spaces of the BIM model in the as_designed and as_realized phases.

4.2.3 Retrieving information from a specific moment in time

In PLCS it is possible to retrieve information about buildings and 3D property units both from when objects were modified in the database and from when something happened to the real-world object.

The BIM and geodata models are imported to ShareAspace and the elements are linked with a relation that has effectivity with start and end dates. The start date is set to dates according to Figure 14. The structure for the Multihuset building in ShareAspace is described in Figure 15. The versioned objects are stored in the platform with the date for the import (i.e., the transaction date).

The building Multihuset in both geodata and BIM formats is connected to the abstract building Multihuset (ABS). Information from a specific moment in time for a building can be retrieved by navigating in ShareAspace to the point of interest in time, in this case to 2019-12-10 (Figure 16). The valid time is found in the relationships between objects in the structure, from effectivity start. The transaction time is found on the object itself, from upload date. The point in time for navigation is set as "actual time" in ShareAspace and the user interface displays the information structure for this moment in time. In Figure 16 the objects from the geodata building and the BIM building in the as_designed phase are shown with the valid versions on each object for the time chosen. These dates correspond to the real-world object events for any lifecycle phase, but have different meaning depending
on which phase they represent. For example, an as_design event and an as_built event have different connections to the real-world objects.

The models from a selected point in time can be merged and visualized. This is done by exporting them in IFC format and then converting them to kmz format. The merged model can then be automatically placed with its real-world coordinates (using a geodetic reference system) in a visualization tool. An example of this is shown in Figure 17, where OpenCities Planner is used to visualize the 3D city model of Malmö.

It is also possible to retrieve information from a specific moment in time for 3D property units. An abstract storey, named Floor 14-15-16-17, represents a 3D property unit in Multihuset, as described in Figure 13. Version 1 of this object is the connection holder to the as_planned building elements and spaces of the BIM-model as_planned.
This version can be retrieved by navigating in ShareAspace to when the as_planned version was valid, in this case to the actual date 2020-11-24 (see Figure 18). In this test only three objects were linked.

The objects can be exported using the ShareAspace RestAPI that is based on the IFC and PLCS standards. An example of this is shown in Figure 19, where room 106 (Bryggan_Plan 15-106) is visualized in the software Solibri Model Checker (https://www.solibri.com/how-it-works). This is a tool for showing individual buildings in detail, while OpenCities Planner can be used for an overview of the city.
In the PLCS standard it is also possible to separate the versioning of the geometry of a building or its parts from the remainder of the information, as the objects with identity and properties are separated from the geometry. The geometry thus has its own versioning.

4.2.4 | Having different simultaneous alternative descriptions of a 3D city model

PLCS supports multiple simultaneous alternative descriptions for structures in the as_designed and as_planned phases. Such alternative descriptions are called variants in PLCS. Structures that can be included in a variant are buildings, and
FIGURE 15  Different parallel versions of Multihuset in geodata and BIM formats can be stored in ShareAspace

FIGURE 16  The structure of the Multihuset (ABS) and the valid linked elements from 2019-12-10

FIGURE 17  Merged models from the selected date visualized in OpenCities Planner
also roads, bridges, parks and other structures that have lifecycle phases. The structures can then be versioned on each individual element until one of the variants is selected to be merged with an existing as_designed or as_planned structure.

4.2.5 | Keeping track of the synchronization between BIM and geodata models

As the BIM and geodata models share the same topological structure in ShareAspace, there can be links between the BIM model and the geodata model both for buildings and property units.

5 | VERSION MANAGEMENT IN CITYGML 3.0

5.1 | Background

The most comprehensive standard today for the exchange of 3D geodata city models is the CityGML standard by the Open Geospatial Consortium (OGC) (Gröger, Kolbe, Nagel, & Häfele, 2012; Liu et al., 2017). The main objective
of CityGML is to represent the geometric and semantic aspects of features in a city. The current official version, CityGML 2.0, is now being revised both to increase the usage of the standard in different areas and to improve the interoperability with other relevant standards. One of the new features of the proposed version 3.0 of CityGML is a versioning module.

In CityGML 2.0, the attributes creationDate and terminationDate are included to describe transactions on objects in the database, and it also includes the attributes yearOfConstruction and yearOfDemolition of a building. In CityGML 3.0, this has been extended to also include the attribute dateOfRenovation and the data type is changed from Year to Date format for dateOfConstruction, and dateOfDemolition. The attributes validFrom and validTo that refer to the lifespan of real-world objects are also added. This time representation is influenced by the lifecycle information in the geographical themes of the INSPIRE directive (2007/2/EC; European Parliament, 2007).

In the new CityGML 3.0 versioning module, the feature types Version, VersionTransition and Transaction make it possible to have different versions of 3D city models and to define which objects belong to a certain version.

![FIGURE 20 Selected parts from CityGML 3.0, the versioning module in purple](image-url)
Links between different versions of the 3D city model are created using VersionTransition, where the reason for change can be described. The feature type Transaction describes the transactions (insert, delete or replace) included in a certain VersionTransition. Versions are also allowed to fork in order to, for example, represent alternative plans of a city.

CityGML 3.0 has two identifiers, the identifier property and the gml:id attribute. Chaturvedi et al. (2016) propose that the identifier property should be used as a persistent identifier for real-world objects and the gml:id attribute should be constructed as a combination of the identifier property and the creationDate attribute to be unique within a version. That is, when a new version is created, the identifier value will be the same, and the gml:id will change, having the same identifier value, but a new creationDate value.

5.2 | Evaluation of requirements

Below is a description of how the versioning module in CityGML 3.0 can meet the test case requirements defined in Section 3.2.

5.2.1 | Retrieving building information from a specific moment in time

In CityGML 3.0 it is possible to retrieve information about objects from when something was modified in the database and from when something happened to the real-world object. This is true for both physical objects, such as buildings (Building), and for logical objects, such as 3D property units (BuildingUnit). Figure 21 gives an example of this for a building; attribute validFrom (2019-12-10) describes that something changed on the building itself, and creationDate (2019-12-17) that a building object changed in the database.

Another requirement is that the versioning of the geometry of a building should be separated from the versioning of the remainder of the information about a building. This is not possible in CityGML 3.0 as the geometry objects neither inherit date attributes from AbstractFeatureWithLifespan nor have any relation to the versioning module (Figure 20).

5.2.2 | Having different simultaneous alternative descriptions of a 3D city model

It is possible to have different simultaneous alternative descriptions of both buildings and 3D property units in CityGML 3.0. This is done by creating a fork version of an existing version. Versions can also be merged into a new version. An example of how this can be implemented in CityGML 3.0 is shown in the object diagram in Figure 22. Here, features from the Building and Versioning UML application schemas are instantiated to describe two alternative scenarios for a new residential area (versions V-02 and V-03). Version V-03 is chosen, so version V-01 and V-03 are merged into version V-04.
5.2.3 | Keeping track of the synchronization between BIM and geodata models

ExternalReference is a data type that all city objects in CityGML have a relation to, and enables all city objects to link to external sources (Figure 23). Provided that the BIM model has a unique and persistent identifier, that the BIM model is versioned and that the CityGML 3.0 versioning is used, ExternalReference can be used to keep track of the synchronization of versions between BIM and geodata models.

6 | VERSIONING USING THE MODIFIED GIT VERSIONING METHOD

6.1 | Background

Vitalis et al. (2019) propose a new approach for versioning of 3D city models and implement it in CityJSON, which is a JSON encoding of the major part of the CityGML 2.0 data model (Ledoux et al., 2019; https://www.cityjson.
CityJSON is relatively new and is not an official OGC standard, but it is under consideration for being approved as a community standard.

The data structure of the versioning proposal by Vitalis et al. (2019) is similar to the DAG (directed acyclic graph) structure in Git, a popular version control system mainly used for computer programming. Git consists of the DAG data structure, metadata that describes the versions and a software to support this. The DAG consists of: (1) nodes that represent objects or data; (2) directed edges from one node to another representing a relationship between them, i.e., an arrow from the new revision to the revision from which it was derived; (3) a root node (i.e., a node without parents); and (4) leaf nodes (i.e., one or more nodes without children) (Figure 24). All nodes know their parent so the graph can be traversed from the leaves to the root.

The reason for not using the Git repository as-is for the versioning of 3D city models is, according to Vitalis et al. (2019), that the way Git tracks changes is not optimal for 3D city models. For example, the order of elements is important in Git but not in 3D city models, so this would cause unwanted version changes.

In the implementation of the modified Git versioning proposal, all versions of all CityJSON objects are stored in a repository in a CityJSON file. Here all city objects are listed under the CityObjects property (Figure 25a) and all versions are listed under the versioning property (Figure 25b). This versioning proposal has only one date, the date property that is included in the versioning property and defines the date of a version in a 3D city model.

In the versioning proposal, all versions of the city objects (e.g., buildings) must have a different id that does not have any semantic or functional meaning. That is, the identifiers are not persistent over time, for example building1 and building1-renovated in Figure 25a.

It must be noted that this versioning proposal only focuses on the versioning of city objects. The proposal deliberately omits to track changes of real-world objects and why these changes occurred, as this is often application dependent and would also result in a complex solution that could be difficult to implement. Vitalis et al. (2019) propose that this type of lifecycle management should be implemented in separate software instead.

6.2 | Evaluation of requirements

The sections below describe how the modified Git versioning proposal can meet the test case requirements defined in Section 3.2.

6.2.1 | Retrieving information from a specific moment in time

The modified Git versioning proposal does not include lifecycle management of real-world objects. It is only possible to retrieve information about objects from when an object changed in the 3D city model, that is transaction

![Figure 24](image-url) Example of a directed acyclic graph structure, from Vitalis et al. (2019)
time. This is accomplished by creating a version to which an object, for example a building, belongs (Figure 26). In this example, version v1 includes the building BU-01 and has creation date 2019-12-17.

CityJSON does not include any logical objects such as 3D property units (BuildingUnit) and therefore the versioning of such objects is not applicable.

The requirement about versioning the geometry of a building separately from the versioning of the remainder of the information is not possible in CityJSON as the geometry objects do not include the necessary information.

### 6.2.2 Having different simultaneous alternative descriptions of a 3D city model

It should be possible to describe alternative versions in the modified Git versioning proposal as it is based on the Git versioning control system where fork, merge and compare functionalities are included. This has not yet been implemented in CityJSON, but Vitalis et al. (2019) mention that this should be possible and describe it as future work.

### 6.2.3 Keeping track of the synchronization between BIM and geodata models

CityJSON does not include the data type ExternalReference from CityGML, so this option cannot be used to include references to external sources. An option that does exist is to describe this in the metadata, as metadata according to the ISO standard 19115 is included in CityJSON. Here the metadata class md:Lineage can be used to
keep track of the synchronization of versions between BIM and geodata models, as it describes metadata about the data source (`md:source`). Figure 27 gives an example of how this can be done.

7 | DISCUSSION

7.1 | Applications requirements for version management

There are many issues that need to be resolved to achieve a more efficient flow of digital information in the planning and building processes. Besides all legal, business-oriented and technical barriers, the information should be standardized and it should be possible to add and modify information in the different phases of the processes, which in turn requires version management of the information. The choice of versioning method for 3D city models should be based on requirements from applications in the various phases within the processes, and depending on the type of information, different versioning methods might be needed. Examples of applications that are dependent on working version management are:

- The **building permit process**, where decisions are made on both BIM and geodata models. Both models must be versioned and the models must also be synchronized at certain stages.
- When a decision is made on the BIM model in the **3D property formation process**, it can be generalized and transformed to a geodata model. Such an integrated BIM–geodata model could improve visual communication of the location of legal boundaries (Atazadeh et al., 2017), and the transformed geodata model could be part of a 3D city model that in turn can act as a **3D cadastral index map** provided that it also has links to other legal registers (Sun, Mi, et al., 2019). All this requires frequent updates of the information included, that is of the

---

**FIGURE 26** Versioning of a building using the modified Git versioning proposal

```
"CityObjects": {
...
"BU-01": {
"type": "Building",
"attributes": {
"storeysAboveGround": 1,
"storeysBelowGround": 0,
"function": "residential building"
}
}
...

"versioning": {
"versions": {
"v1": { 
"author": "City Official"
"date": "2019-12-17"
"objects": [
"BU-01"
]
}
}
}
```
BIM models, the 3D city model and the 3D cadastral index map, and also that the BIM and geodata models are synchronized.

- Applications for different types of analyses (e.g., for noise reduction, solar panel planning and flooding) need several simultaneous alternative descriptions of 3D city objects to be able to evaluate various scenarios. Such alternative descriptions can also be used to visualize different alternative solutions for new residential areas in a city or alternatives for 3D property units in a complex underground environment. This requires that it is possible to keep simultaneous alternative descriptions of the 3D city model and preferable that one solution can be chosen and merged into the model.

- Last but not least, city officials need information about certain previous moments in time, mainly legal dates, to perform their work. For example, the date when a building permit was accepted or when a building was completed and accepted.

### 7.2 Advantages and disadvantages with the evaluated versioning methods

The evaluated versioning methods range from being only a version management of 3D city models to a fully fledged versioning and lifecycle management system. This is also reflected in the results of the evaluations, summarized in Table 1.

As described in Section 7.1 above, it is important to choose the best method for the intended application, and to be able to judge this one should be aware of the advantages and disadvantages of the methods. For example, the modified Git versioning proposal does not fulfill many of the requirements in Table 1, but Vitalis et al. (2019) made a conscious decision when keeping their versioning proposal simple. They believe that having bi-temporal timestamps and mixing fork, merge, planned and realized as reasons for a version transition (as is the case in the CityGML 3.0 versioning proposal) will conflate versioning with lifecycle modelling. According to Vitalis et al. (2019) there should not be one lifecycle management solution that includes everything, instead a variety of methods should be used depending on what the 3D city model will be used for, as trying to include all aspects in one solution will make the solution complex and difficult to implement. In contrast, having a combination of many different versioning methods can also be an issue as the information must be transferred between them.

The PLCS standard has the opposite intention compared to Vitalis et al. (2019). PLCS is a very comprehensive standard that includes “all” lifecycle requirements a product or object needs during its lifetime, and it also fulfills all requirements stated in Table 1. Examples of functionalities in PLCS are: management of documents, activities, changes, maintenance, requirements and breakdowns; as well as linking of multiple identifications, classifications, observations, statuses, persons, organizations and representations to the objects. Advantages with having all
functionalities of the version and lifecycle management included in the same standard are that the information flow will be smoother as no conversion between different systems is needed, a disadvantage is that it is a complex system to implement and maintain.

The new versioning module in CityGML 3.0 is somewhere between the PLCS standard and the modified Git versioning. It fulfils all but one requirement in Table 1 (only lacking the possibility to have versioning on the geometries) and could therefore be seen as a possible solution for 3D city models that require more than pure versioning of the models. But as described by Vitalis et al. (2019), this can also make it more difficult to implement. As no implementations exist yet, it is also difficult to foresee if and how the combination of the date attributes (dateOfConstruction, dateOfRenovation and dateOfDemolition) and the valid time interval (validFrom and validTo) will work.

### 7.3 Possible future developments of the evaluated versioning methods

Both the versioning module in CityGML 3.0 and the modified Git versioning are proposals that most likely will change and evolve in the future. This would, for example, make it possible to add additional versioning and lifecycle functionalities to these methods. For example, in CityGML 3.0, all geometry objects could inherit the feature type AbstractFeatureWithLifespan and have a relation to the Version feature type. By this it would be possible to version the geometries separately. In the same way, attributes from the PLCS standard could be added. Also, the modified Git versioning proposal could be extended, for example with the attributes dateOfConstruction, dateOfRenovation, dateOfDemolition, validFrom and validTo from CityGML 3.0. This would make the methods slightly more complex, but would also fulfil more requirements.

When it comes to a comprehensive standard like PLCS, a ready-made implementation of the standard for lifecycle management of geodata and BIM models should be available for it to be used, as the standard is complex and could be interpreted in several ways. There may not be many occasions when this standard is needed in a geodata–BIM context, but when it is, it is invaluable that such a standard exists.

### 8 CONCLUSIONS

Version management is one of the prerequisites for a digital information flow in the planning and building process, as information will evolve and be used for multiple purposes and by different actors during its lifecycle.
There are many version management aspects and requirements to take into consideration when different actors should be able to share, retrieve and modify information throughout the planning and building processes. Out of these requirements, we have chosen three that are necessary in the 3D cadastre process: (a) retrieve information from a specific moment in time about objects in a 3D city model (i.e., transaction time) and about real-world objects (i.e., valid time); (b) have different simultaneous alternative descriptions of a 3D city model; and (c) keep track of the synchronization of building information between BIM and geodata models.

In this study we compare three versioning methods by evaluating them against the three requirements described above. The main focus in the evaluation is on the PLCS standard (ISO, 2012a), a comprehensive lifecycle standard originally developed for the process industry but currently also used for BIM models. In a test case we evaluate if PLCS can also be used for versioning of 3D city models. Two other less complex versioning methods for the versioning of 3D city models were also evaluated: the versioning module in CityGML 3.0 (Chaturvedi et al., 2016), where bi-temporal timestamps, versions and links between the versions and transactions can be described; and a modified Git versioning method implemented in CityJSON (Vitalis et al., 2019), which includes transition dates and versions. The reason for also including these was to examine if such methods have sufficient functionalities to fulfil the stated versioning requirements.

The results of the evaluation are summarized in Table 1. It can be concluded that all three methods can be used to keep track of the synchronization between BIM and geodata models, which is a requirement for many applications, such as the processes for building permits and 3D property formation and also for management of 3D cadastral index maps. All methods can also store transaction dates for buildings, which makes it possible to version 3D city models. To have simultaneous alternative descriptions of 3D city models is possible in both PLCS and CityGML 3.0, and it would also be possible to achieve in the modified Git proposal as it is a functionality in standard Git. This is a requirement for applications that perform analyses and evaluate various scenarios, and also for applications that visualize different alternative solutions in 3D city models. PLCS is the only method that can version a geometry of a building separated from the other building information, and it also includes numerous other lifecycle features not evaluated in this study.

It must be noted that even though CityGML and the Git versioning proposal in CityJSON do not include all functionalities that were evaluated, it would be possible to include additional functionalities in these proposals and thereby fulfil more requirements (see Section 7.3).

To conclude, the evaluated methods vary in complexity from the modified Git proposal that was deliberately a simple versioning method for 3D city models, to the CityGML 3.0 versioning module that is a bit more complex, to PLCS that is a very comprehensive lifecycle standard. The question is whether a complete solution that covers everything within the lifecycle of an object, or many solutions that cover different aspects of the version management, is the best choice. The version management requirements are complex, and it will always be a trade-off between meeting all the requirements and how difficult it is to implement and use the chosen solution. Careful preparatory work and an understanding of how, to what and by whom the information will be used during the whole lifecycle of the object (e.g., a building or a 3D property unit) is therefore essential before choosing a version management method for information in planning and building processes.

ACKNOWLEDGMENTS
We would like to thank participants in the Swedish testbed project, Testbädd: Smarta plan-, bygg- och förvaltningssprocesser över hela livscyklens, for exchanging ideas and technical work, especially Örjan Falk, KTH; Jonas Rosén and Torbjörn Holm, Eurostep for help with ShareAspace; and Perola Olsson and Tim Johansson for the conversion of the test data. We are also grateful for the implementation of ShareAspace in the testbed made by Eurostep, the provision of test data from Malmö stad and the building company NCC. Finally, we would like to thank our anonymous reviewers for their constructive comments.

CONFLICT OF INTEREST
The authors declare no conflict of interest.
REFERENCES

Aien, A., Kalantari, M., Rajabifard, A., Williamson, I., & Wallace, J. (2013). Towards integration of 3D legal and physical objects in cadastral data models. *Land Use Policy*, 35, 140–154.

Atazadeh, B., Kalantari, M., Rajabifard, A., Ho, S., & Ngo, T. (2017). Building information modelling for high-rise land administration. *Transactions in GIS*, 21, 91–113.

Biljecki, F., Ledoux, H., & Stoter, J. (2014). Redefining the level of detail for 3D models. *GIM International*, 28, 21–23.

Borrmann, A., König, M., Koch, C., & Beetz, J. (2018). Building information modeling: Why? What? How? In A. Borrmann, M. König, C. Koch, & J. Beetz (Eds.), *Building information modeling: Technology foundations and industry practice* (pp. 1–24). Cham, Switzerland: Springer.

Cemellini, B., van Oosterom, P., Thompson, R., & de Vries, M. (2020). Design, development and usability testing of an LADM compliant 3D Cadastral prototype system. *Land Use Policy*, https://doi.org/10.1016/j.landusepol.2019.104418

Chaturvedi, K., Smyth, C. S., Gesquiére, G., Kutzner, T., & Kolbe, T. H. (2016). Managing versions and history within semantic 3D city models for the next generation of CityGML. In S. Geertman, W. Reinhardt, & F. Toppen (Eds.), *Advancing geoinformation science for a changing world* (Lecture Notes in Geoinformation and Cartography, pp. 191–206). Berlin, Germany: Springer.

CoClass. (2016). *Swedish digital construction classification system for built environment*. Retrieved from https://byggtjanst.se/tjanster/coclass/

de Laat, R., & van Berlo, L. (2011). Integration of BIM and GIS: The development of the CityGML GeoBIM extension. In T. Kolbe, G. König, & C. Nagel (Eds.), *Advances in 3D geo-information sciences* (Lecture Notes in Geoinformation and Cartography, pp. 211–225). Berlin, Germany: Springer.

Deng, Y., Cheng, J. C. P., & Anumba, C. (2016). Mapping between BIM and 3D GIS in different levels of detail using schema mediation and instance comparison. *Automation in Construction*, 67, 1–21.

Donkers, S., Ledoux, H., Zhao, J., & Stoter, J. (2016). Automatic conversion of IFC datasets to geometrically and semantically correct CityGML LOD3 buildings. *Transactions in GIS*, 20, 547–569.

Ekholm, A., Blom, H., Eckerberg, K., Löwnertz, K., & Tarandi, V. (2013). *BIM–Standardiseringsebehov*. Retrieved from https://www.bimalliance.se/library/3174/130620_bim_rapport_standardiseringsebehov.pdf

El-Mekawy, M., Östman, A., & Hijazi, I. (2012). A unified building model for 3D urban GIS. *ISPRS International Journal of Geo-Information*, 1, 120–145.

El-Mekawy, M. S. A., Paasch, J. M., & Paulsson, J. (2015). Integration of legal aspects in 3D cadastral systems. *International Journal of E-Planning Research*, 4, 47–71.

Eriksson, H., Johansson, T., Olsson, P. O., Andersson, M., Engvall, J., Hast, I., & Harrie, L. (2020). Requirements, development, and evaluation of a national building standard: A Swedish case study. *ISPRS International Journal of Geo-Information*, 9, 78.

European Parliament. (2007). Directive 2007/2/EC of the European Parliament and of the Council establishing an Infrastructure for Spatial Information in the European Community (INSPIRE) (Official Journal, L 108, 25/04/2007: 1–14). Retrieved from http://inspire.ec.europa.eu/documents/directive20072european-parliament-and-council-14-march-2007-establishing

Góźdź, K., Pachelski, W., Van Oosterom, P., & Coors, V. (2014). The possibilities of using CityGML for 3D representation of buildings in the cadastre. In *Proceedings of the Fourth International FIG Workshop on 3D Cadastres Conference*, Dubai, United Arab Emirates (pp. 339–352). Copenhagen, Denmark: FIG.

Gröger, G., Kolbe, T. H., Nagel, C., & Häfele, K.-H. (2012). OGC city geography markup language (CityGML) encoding standard. Retrieved from http://www.opengeospatial.org/standards/citygml

Halfawy, M. M. R., Froese, T. M., Vanier, D., & Kyle, B. (2004). An integration approach for developing AEC/FM total project systems. In *Proceedings of the 2004 CIB Triennial World Building Congress*, Toronto, Canada (pp. 1–11). Delft, the Netherlands: CIB.

Hallberg, D., & Tarandi, V. (2011). On the use of open BIM and 4D visualisation in a predictive life cycle management system for construction works. *Journal of Information Technology & Construction*, 16, 445–466.

Hjelmblom, M., Paasch, J. M., Paulsson, J., Edlund, M., & Bökman, F. (2019). Towards automation of the Swedish property formation process: A structural and logical analysis of property subdivision. *Nordic Journal of Surveying & Real Estate Research*, 14(1), 29–63.
Worboys, M. (2005). Event-oriented approaches to geographic phenomena. *International Journal of Geographical Information Science*, 19, 1–28.

Worboys, M. F., & Duckham, M. (2004). GIS: A computing perspective (2nd ed.). Boca Raton, FL: CRC Press.

How to cite this article: Eriksson H, Sun J, Tarandi V, Harrie L. Comparison of versioning methods to improve the information flow in the planning and building processes. *Transactions in GIS*. 2021;25:134–163. https://doi.org/10.1111/tgis.12672