Life Cycle Rules for 3D Building Data

[1. Introduction 2](#_Toc101803800)

[**1.1. What are life cycle rules?** 2](#_Toc101803801)

[**1.2. Identifiers** 2](#_Toc101803802)

[**1.3. Version Information** 2](#_Toc101803803)

[2. Basic Workflows 3](#_Toc101803804)

[**2.1. Creation and Integration** 3](#_Toc101803805)

[**2.2. Deletion** 3](#_Toc101803806)

[**2.3. Modification and update** 3](#_Toc101803807)

[3. Overlapping rules 4](#_Toc101803808)

[**3.1. Intersecting** 4](#_Toc101803809)

[**3.2. Overlapping** 4](#_Toc101803810)

[**3.3. Identical** 5](#_Toc101803811)

Published 25.04.2022

# Introduction

## **What are life cycle rules?**

This paper defines general life cycle rules for 3D building data. They determine the situations in which features retain or change their identities. For example, in which cases building is completely replaced or when it just gets a new version? By following these rules, it is possible to maintain history information of buildings, and utilize it later. On the contrary, the end user can utilize these rules to check whether version information is accurate.

The building is considered as non-existent or archived, if it does not contain any other parts except its foundation. For example, if real-life building burn down and only its foundations are left, it does not exist anymore and should be archived in the database. If someone builds a new building on the same foundations, it is considered a new building and it gets new ID.

All rules are connected to the geometry and the ID number of the buildings. The two most common formats are the CityGML and the CityJSON. The rules can be applied similarly to both formats, but the terms *ID* and *version information* refer to different concepts.

## **Identifiers**

In the CityJSON, applying the life cycle rules is simple because all parts of building have a same ID number (*fid*). Thus, only one ID changes.

The CityGML is more complex. Each building part (like *Roofsurface* and *WallSurface*) has unique ID (*gml:id*), which is hierarchically connected to building body with *parent\_id* attribute. For that reason, all parts that are higher in hierarchy should be updated too. When one part of building is updated or replaced, then also its parent feature must be updated. That will create a new version for the building part, for its parent object, and furthermore for their aggregated object in parental hierarchy.

## **Version Information**

In addition of identifiers, both formats contain multiple mandatory attributes related to the life cycle and version information of the data, which should also be up to date.

In the CityGML 2.0, attributes *creationDate* and *terminationDate* represent transaction time in a database, and attributes *yearOfConstruction* and *yearOfDemolition* represent building’s life cycle in the real-life. The attribute *versionID* describe a current version of the data.

The CityGML 3.0 standard contains more parameters related to building lifetime, like v*alidFrom, validTo, dateOfRenovation.* Most of these attributes are also recommended to use in the CityJSON specification. If the data does not contain these attributes, they should be added to ensure that data comply with its specifications and standards.

Shortly, whenever buildings or their parts are created, updated or integrated, also their temporal and version information (e.g. *versionID* and *dateOfRenovation*) should be updated when appropriate. Mandatory attributes depends on the used data format (see [CityGML standard](https://docs.ogc.org/guides/20-066.html#ug-model-versioning-section) and [CityJSON specification](https://www.cityjson.org/specs/1.1.1/)).

# Basic Workflows

## **Creation and Integration**

When new features are **created** or **integrated** to the database, there are four options:

1. No matching ID and no overlapping geometry
   * New feature with unique ID and version information
2. Matching ID and identical geometry
   * New geometry and version information to the updated feature
3. Matching ID, but no overlapping geometry.
   * If the feature represents different building, there are errors because all IDs should be unique. In that case, change ID with careful consideration. Ensure that ID is unique and logical.
4. No matching ID, but geometry is overlapping
   * See **Overlapping rules**
5. Temporal information do not match
   * Usually newer is better, so replace and archive the older one. Remember to update version information. If that causes overlapping, see **Overlapping rules**

## **Deletion**

When features are **deleted**, there are two options:

1. Feature contain errors that affect to other features, like intersections (See **Overlapping rules**)
   * Remove error messages from both features because they don’t intersect after deletion
   * Archive the feature and its ID
2. Feature has no errors that affect to other features
   * Archive the feature and its ID

## **Modification and update**

When features are **modified** or **updated**, there are three options

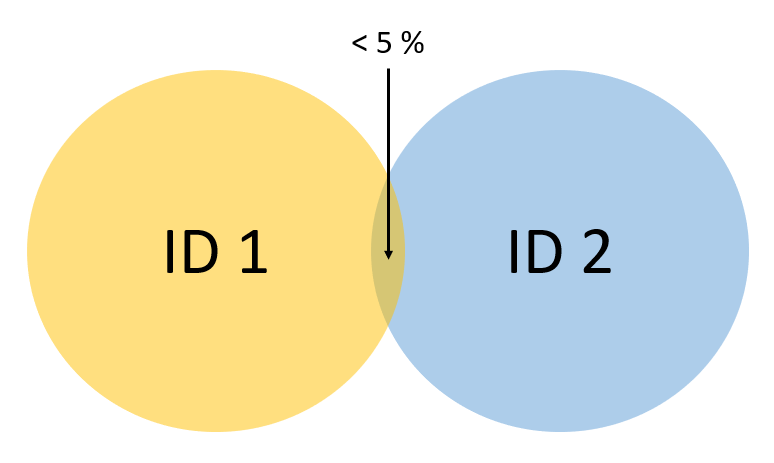
1. No changes to ID or to geometry
   * Update version and other information for the same ID
2. Same ID, changes to geometry
   * If new geometries intersect or overlap with another features, see **Overlapping rules**
   * Otherwise, update geometry and version information for the same ID
3. Changes to ID
   * Create new feature with unique ID. Version information resets.
   * Replace and archive the old feature and its ID

# Overlapping rules

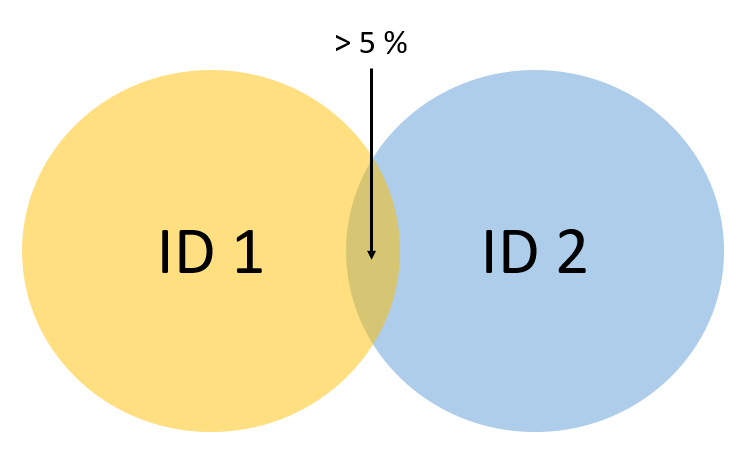
The following rules are based on the lowest parts of 3D buildings. Usually they are extracted GroundSurfaces or calculated 2D-forced footprints of the buildings. If the features do not contain area (like lines and points), a buffer with 1.5-meter radius should be generated around them. If the building consists of several parts, their areas should be dissolved and used together.

The actions are always performed based on the highest possible overlapping level (**1. intersecting -> 2. overlapping -> 3. identical**)

## **Intersecting**

1. Features are **intersecting** if they touch or overlap each other, but not more than 5 % of their total area. (Error margin)
   * Do **not** modify or replace the old feature
   * ****Add new feature and write error message in metadata/attributes of both intersecting parts

## **Overlapping**

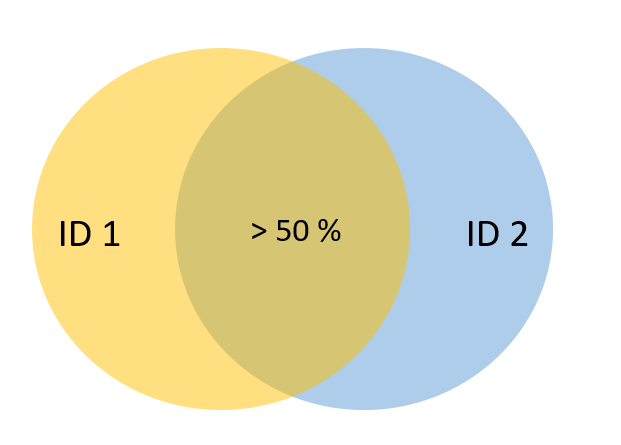
1. Features are **overlapping** if they touch each other more than 5 %, but less than 50 % of their total area.
   * Remove and replace the old feature
   * New feature replaces the old feature with new unique ID

**NOTE**: It is important to check all features before any deletion

## **Identical**

1. Features are **identical** if they overlap more than 50 % of their total area. In that case, the area of the new feature should be 0.5 – 2 times as large as the area of the old feature.

**NOTE**: It is important to check all features before any deletion

1. Matching ID
   * Update the old feature with newer version
2. No matching ID
   * Create a new feature with unique ID
   * Archive and replace the old feature