CityGML Data Quality Application Domain Extension

1. Introduction

This document presents an Application Domain Extension (ADE) to the CityGML v2.0 open data model. The ADE has been prepared to support the storing and managing of data quality information about the various objects and attributes that are relevant to city models and are represented in CityGML V2.0. The Data Quality ADE schema is compatible with JSON format, and as such is a CityJSON extension containing new objects and attributes.

2. Data quality covered by this ADE

2.1 Prefrace

Data Quality is a data model that quantitatively describes the degree of closeness between the digital model/representation ('digital twin') and the physical reality. Data Quality allows us to determine how suitable the digital model is for a specific use, i.e., "fitness for use". Therefore, different parts of the Data Quality model represented in this ADE will be relevant depending on the use and need of the digital model.

In this Data Quality model we focus on the core of the CityGML model, which is designed to describe the physical reality as is. Other extensions that include additional information that is outside the scope of the physical reality are not covered in this ADE.

The entailed use cases this ADE is designed to serve are visualization, navigation, data fusion, georeferencing, automatic positioning - to name a few. Use cases, such as city planning and infrastructure management, may require additional data quality categories, and thus require extensions to this ADE.

2.2 Introduction

The International Organization for Standardization (ISO) defines geographic information quality as the totality of characteristics of a product that bear on its ability to satisfy stated and implied needs. ISO/TC 2117 (Technical Committee) defines a set of standards that define the measures of geographic information quality (standard 19138). We build upon these definitions, and begin by separating the Data Quality model to seven categories (metrics) handled in this ADE:

I. **Positional and geometric reliability** – accuracy of entity position in space and the accuracy of the entity geometric shape. Positional accuracy can be indicated by absolute (external) accuracy and relative (internal) accuracy.

II. **Semantic reliability** – accuracy of categorical data that is not spatial-oriented. Semantic accuracy (also referred to as thematic accuracy) can be indicated by classification correctness and non-quantitative attribute correctness, which - although not spatial-oriented - can be numerically estimated.

III. **Visual quality** – the quality of the virtual image (e.g., photorealism) that can be rendered based on the geometric model and imagery.

IV. **Completeness** – absence or excess of data (errors of commission). Completeness describes the relationship between the represented objects and their conceptualizations.

V. **Temporal reliability** – the accuracy of data that is related to time. Temporal reliability can be indicated by the correctness of the object's temporal references and the validity of data regarding time.

VI. **Logical consistency** – the validity of the data and its properties. Logical consistency is the coherence in the data structure of the digitized objects, manifested in errors like topological consistency.

VII. **Semantic consistency** – consistency and validity of the categorical data, that is not spatial-oriented.

For each category we define in this ADE metric-sets that can be attached to the existing CityGML data according to their JSON schema.

It should be noted that although the last two categories, namely Logical consistency and Semantic consistency, are closely associated with data quality, they are not characterized with attributed quality data. These two categories represent the validity of the data in terms of its internal topological, geometric and semantic structure (see GML3 and Chapter 8 in CityGML). Accordingly, these two categories are explained later in this document, although they do not have metric-sets and accordingly no JSON schema, and should be checked and validated independently before insertion via systematic rules.

3. Cascading mechanism

Data Quality attachments in this ADE can be done according to five levels - cascading from coarse to fine: Zone, subZone, CityObject, subCityObject and geometric primitives (i.e., Polygon, Line and Vertex). The cascading mechanism is summarized in Table 1 and depicted in Figure 1.

A ***Zone*** is defined by a two-dimensional polygon. The quality metrics attached to the zone define the least-quality for all the objects that fall inside its demarcation. I.e., objects that exist inside the zone can show better quality metrics - but not worse. Each zone refers to the data quality metrics of a specific city object family (e.g., buildings, roads, vegetation), such that several zone polygons can cover the same area. It should be noted that three-dimensional quality metrics can be attached to the zone (e.g., height), although the zone is represented by a two-dimensional polygon.

A Zone can contain several ***subZones***, whereas the quality metrics that are defined for each subZone describe the quality of the objects inside its demarcation, while replacing the value for a specific metric in the parent Zone.

CityObject is the equivalent to a CityGML object. If a metric is attached to a CityObject it will determine the quality of that object, replacing the metrics that are defined by the Zone(s) that include this CityObject. As before, the metrics cannot be of worse quality than the metrics of the Zone.

CityObject can have ***subCityObjects***, I.e., WallSurface is a subCityObject of a Building. Similarly, a metric can be attached to the wall of the building, covering the metric for the building object. A metric can also be attached to a specific object's categorical property.

A single CityGML object can have elements with different qualities. The geometry representation of an object in CityGML is built from geometric primitives: ***Polygon***, ***Line***, ***Vertex***. A metric can be attached to a geometry primitive, i.e, Vertex.

In summary, according to the cascading mechanism, to retrieve a specific metric, we start from the interest level and progress to the upper levels until we find the desired metric. Table 1 lists all levels, with the schematics depicted in Figure 1.

Level	**Entity**	**Description**
1	Zone	2D boundary
2	subZone	2D boundary that is included in another zone
3	CityObject	Building, Road, Tree …
4	subCityobject	Window, Door …
5	Polygon or Line or Point	Geometry primitive

Table 1. The cascading mechanism.

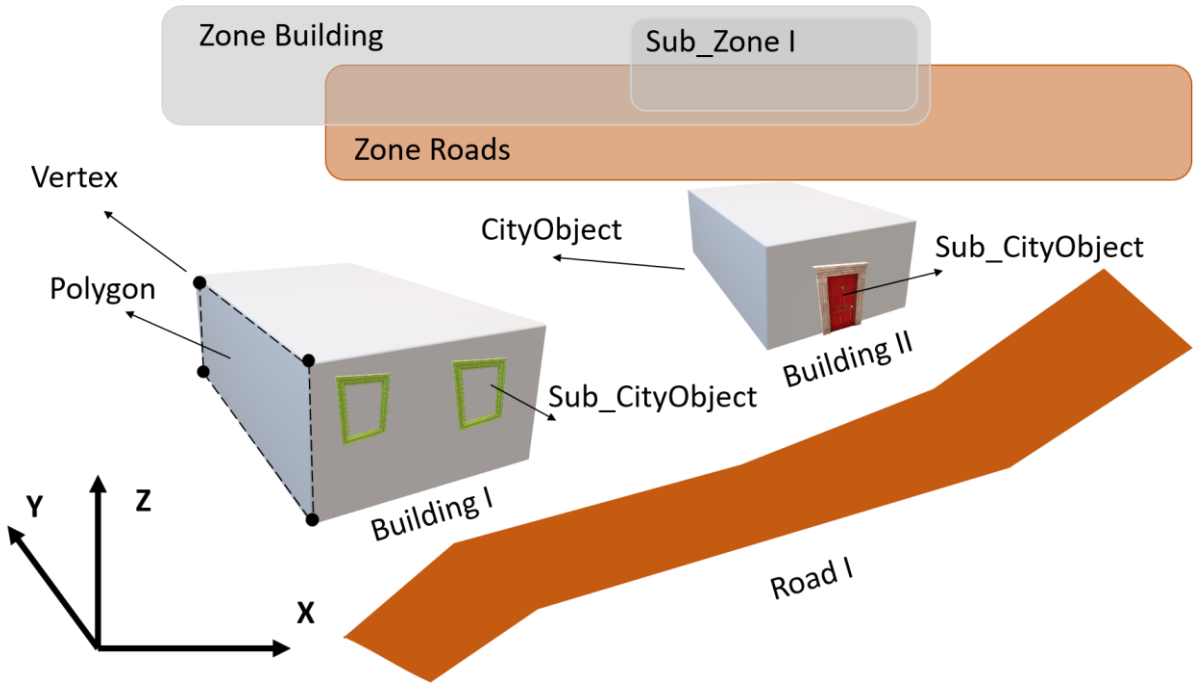

Figure 1. Schematics of the cascading mechanism.

4. Data quality metrics

4.0 Units of measurements

Since some metrics in this ADE are related to quantitative measures, such as distance and orientation (angle), we use attributes related to the units of measurements. Thus, allowing the user to choose and define the corresponding unit from predefined Enumerations. The schema defines - where viable - default units of measure of meters (m) (and square meter - m2) and radians (rad), including value limit where relevant (e.g., decimal degree between 0 and 360).

For distance related attributes we use:

Unit type (string)	Enum
Meters	m
Feet	feet

For angle related attributes we use:

Unit type (string)	Enum
Radians	rad
Decimal Degrees	dd

For squared values:

Unit type (string)	Enum
Squared meter	m2
Squared feet	feet2

For example, for a resolution value of 5 meters we will define (see chapter 5 for more comprehensive JSON examples):

"uom": { "type": "string", "enum": ["m", "feet"] }

"+resolution-value": {

"value": 5.0,

"uom": "m"

4.1 Positional and geometric reliability

This category refers to “accuracy of position” and “accuracy of geometric shape” of an object. Additionally, accuracy of the geometric derivatives, such as length, area and volume, can also be calculated based on the defined metrics.

We define four metric sets, listed in Table 2, and any subgroup of these sets can be used as a viable description.

The metric sets are:

	Set name	**Can be attached to**
1	Two-dimensional metrics	Zone,subZone, CityObject, subCityObject, Polygon
2	Three-dimensional metrics	All
3	Polygon level metrics	subCityObject, Polygon
4	Vertex level metrics	Vertex

Table 2. Metric sets for positional and geometric reliability.

4.1.1 Set #1 - two-dimensional metrics

The two-dimensional metric set consists of two metrics: CE90 and LE90. The metric CE90 defines the horizontal position accuracy of the target, and LE90 defines the accuracy of any two-dimensional length that can be derived from the target geometry (e.g., target width). Both CE90 and LE90 are coupled with uom (either meters or feet), and represent the confidence interval of 90%.[1]

This set is composed of the types:

{ CE90, LE90 } +uom

This metric set is useful for attaching positional and geometric accuracy to two-dimensional (2D) geometries in LOD0 like building footprints, and can be used when only the two-dimensional positional accuracy is known.

4.1.2 Set #2 - three-dimensional metrics

Similarly, a simple three-dimensional metric set consists of two metrics SE90 and LE90. The metric SE90 defines the three-dimensional position accuracy of the target, and LE90 defines the accuracy of any three-dimensional length that can be derived from the target geometry. Both SE90 and LE90 are coupled with uom (either meters or feet), and represent the confidence interval of 90%.[2]

This set is composed of the types:

{ SE90, LE90 } +uom

This metric set is useful for attaching positional and geometric accuracy to three-dimensional (3D) geometries in LOD1 - and above.

[1] CE90 is the circular error at the 90th percentile. This means that a minimum of 90 percent of the points measured has a horizontal error less than the stated CE90 value. LE90 is the 90th percentile linear error, meaning that a minimum of 90 percent of vertical errors fall within the stated LE90 value.

[2] SE90 is the spherical error at the 90th percentile that combines CE90 and LE90.

4.1.3 Set #3 - polygon level metrics

This set is attached to planar polygon level objects, such as wall, door and window. This metric set can describe a combination of the positional accuracy, the orientation accuracy, the elevation accuracy and the texture accuracy, coupled with uom. For example, Surface level attachment is suitable for a Textured-Mesh.

This set is composed of these types, which are detailed below and depicted in Figure 2.

{ position, azimuth, elevation, texture_CE90 } +uom

Metric name	Description	Units
position	Refers to the positional accuracy in space of the surface centroid, represented by six parameters (values) derived from the 3x3 variance matrix	Square meters (m2) or Square feet (feet2)
azimuth	The accuracy of the azimuth (Az) of the surface's normal vector (n) (see figure below)	Radians (rad) or Decimal Degrees (dd)
elevation	The accuracy of the height of the surface's normal vector (h) (see figure below)	Radians (rad) or Decimal Degrees (dd)
texture_CE90	Refers to the polygon surface's positional accuracy of a point that is determined according to the texture that is attached to the surface. This metric indicates how good the mapping (registration) of the texture picture and the polygon geometry is. This value relates to the radius of the error circle	Meters (m) or Feet (feet)

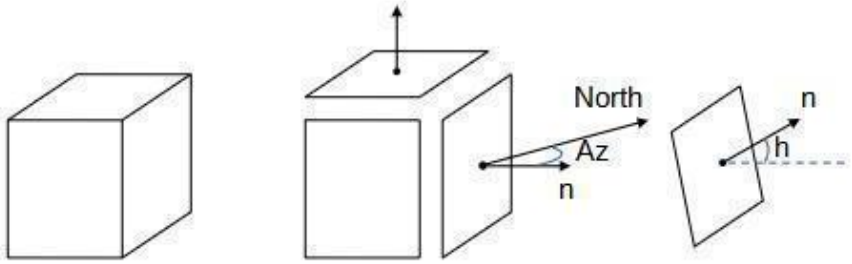

Figure 2. An object depicting all metrics used in positional and geometric reliability Set #3.

The metrics from this polygon set describe the position accuracy of a planar polygon, and can be used to calculate sampled points existing on the polygon's plane.

4.1.4 Set #4 - Vertex level metric set

This set is attached at a vertex-level. Vertex level attachment allows to describe positional accuracy for individual points via variance-covariance matrix.

This set is composed of the type:

{ position } +uom

See position in the polygon level metrics (Set #3) in 4.1.3.

4.1.5 Derived measurements

Geometric properties like object perimeter, area, and length can be calculated based on the object geometry. In this case, the accuracy of that property is calculated based on the given object's geometric accuracy (law of error propagation). On the other hand, if such a value is provided independently of the object's geometry, it will be stored as an additional object attribute, and the accuracy of the calculated property will be stored as attribute accuracy.

4.2 Semantic reliability

Here, we define a single metric that describes the correctness probability of the object's given semantic category. The metric refers to a specific property of the object, and can be attached to all levels.

This set is composed of the type:

{ reliability }

Metric	Description	Units
reliability	Percentage value that indicates the probability of correct semantics	Percentage (0-1)

On the zone level, semantic reliability will refer to all layers (objects) that fall inside the zone boundary. Type reliability is a number between 0 and 1, indicating the probability for an object in a given layer to show the correct semantics.

4.3 Visual quality

The quality of the virtual image that can be rendered based on the accumulated information (i.e., 3D model, texture imagery,...).

4.3.1 Set #1 – general metric set

General metrics refer to the basic properties of the visual component - texture and imagery quality. This metric set can be attached to Zone, subZone, CityObject, subCityObject and Polygon.

This set is composed of the types:

{ textureType, resolution } +uom

Metric	Description	Units
textureType	Type of texture attached	Enum (see below)
resolution	A value that refers to the imagery resolution	Meters (m) or Feet (feet)

The possible texture types are (textureType Enum above):

Texture type	Description
none	No texture
typeColor	Features are colored according to their type
typical	The texture is based on a typical image
realistic	Realistic texture is based on the real image of the surface

4.3.2 Set #2 – photogrammetric metric set

Photogrammetric metrics refer to a realistic texture with properties relevant to images acquired via a photogrammetric process.

This set is composed of the types:

{ propriety, isOccludded, distanceToCamera, verticality, resolution} +uom

These metrics refer to the realistic texture, depicted in Figure 3:

Metric	Description	Units
propriety	A qualitative metric indicating the general quality of the texture, i.e., "How close the texture is to the real world image"	Enum (see below)
isOccludded	A qualitative metric indicating the existence of texture occlusions	0 (False) / 1 (True)
distanceToCamera	The distance from the camera center to the surface centroid	Meters (m) or Feet (feet)
verticality	The angle between the surface centroid and the camera center direction, to the surface normal direction	Radians (rad) or Decimal Degrees (dd)
resolution	The nominal pixel size in meters. This metric is relevant only for realistic texture (photorealism).	Meters (m) or Feet (feet)

The possible propriety types are qualitative, and can be: Bad, Poor, Fair, Good, and Excellent.

This metric set is attached to a surface-plane, allowing to estimate the quality of point detection algorithms, e.g., SIFT.

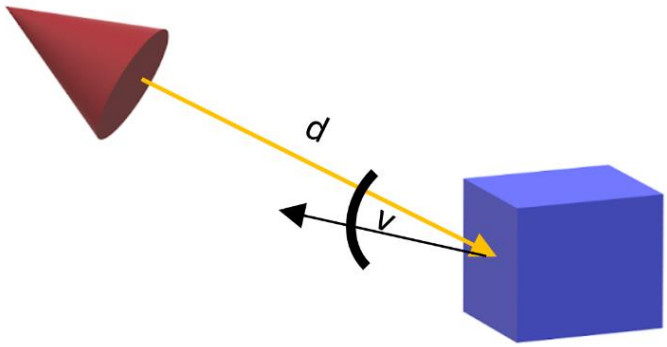

Figure 3. An object depicting metrics used in Visual Quality Set #2.

4.4 Completeness

This metric set represents the absence or excess of data for a given (object) dataset (layer). This metric set can be attached to Zone, subZone, CityObject, and subCityObject.

This set is composed of the types:

{ isExist, completenessAbsence, completnessExcess, lodx.y, isMesh}

These metrics refer to the values of object completeness in a specific dataset (layer), composed of:

Metric	Description	Units
isExist	Value 'True' indicates that there exists measurements (ground truth) for the current layer in the zone boundary. Value 'False' indicates that there exists no measurements, and therefore there is a possibility that objects from the current layer exist in the real world but are not stored (mapped) in the dataset	0 (False) / 1 (True)
completeness Absence	Percentage value that indicates the absence completeness value (ratio) of a dataset (number of objects) relative to the real world. For example, a value of 0.9 indicates that the dataset does not represent 10% of the objects existing in the real world (in the zone boundary)	Percentage (0-1)
completeness Excess	Percentage value that indicates the excess completeness value (ratio) of a dataset (number of objects) relative to the real word. For example, a value of 1.2 indicates that ~16% of the objects in the dataset may not appear in the real world (in the zone boundary)	Percentage (>1)
lodx.y	Indicating the specific LOD within the zone boundary (see Figure 4 below) in which the above values refer to.**This metric is relevant to the zone level only.** Relevant lod values can be: x ∈(0,1,2,3,4) y ∈(0,1,2,3) This metric is equivalent to presentLoDs that can be placed in the metadata according to the CityJSON Spec. The presentLoDs reports what LODs present in the given CityJSON file, or in other words - it describes the data set.	Percentage (0-1)

isMesh	Indicates the existence of a mesh model for a specific Sub-CityObject. **This metric is relevant to the subCityObject level only.**	0 (False) / 1 (True)

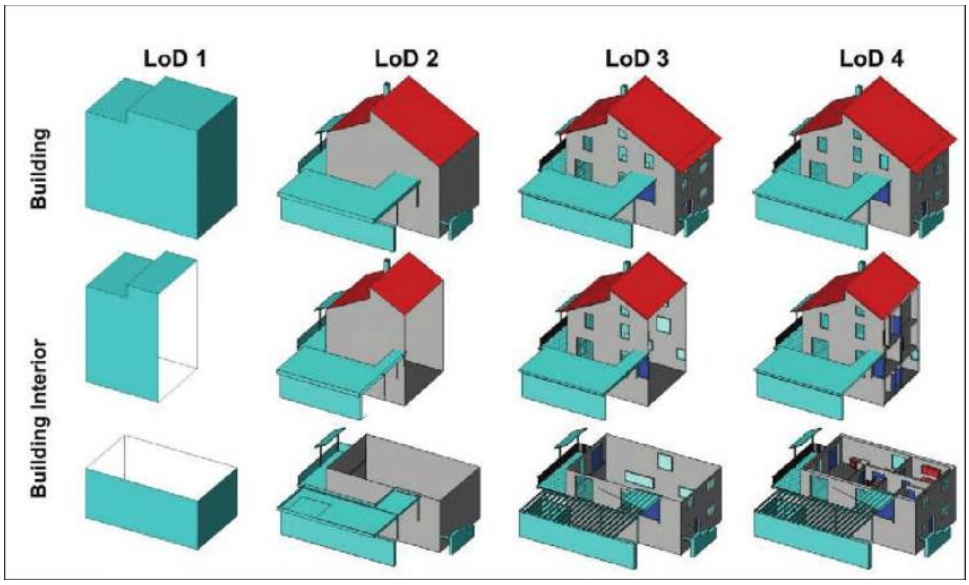

Figure 4. CityGML's Levels of Detail (source: Karlsruhe Institute of Technology).

4.5 Temporal reliability

This metric set represents the timely relevance (updatedness) of the data related to time. This metric set can be attached to all levels.

This set is composed of the types:

{ measureDate, measureTime, transience }

Metric	Description	Units/Format
measureDate	The date in which the data was observed (acquired)	Date (e.g., 2001-10-26)
measureTime	The time of the day in which the data was observed (acquired)	Time (e.g., 21:32:52+02:00)
transience	The number of days in which the layer or object is not expected to change	Float (number of days)

Although CityJson does have fields associated with temporal metadata (e.g., "datasetReferenceDate", "temporalExtent"), these serve for different purposes since they do not have a cascading mechanism (they cannot be attached to a specific primitive, for example), and they also refer to the files and not to the actual materials (measurements) on which the different models are built upon.

4.6 Logical consistency

As stated in 4.2, Logical consistency and Semantic consistency do not have metric sets and should be checked and validated independently before insertion according to a fit-for-purpose formulation design. The data and property validation rules are:

4.6.1 Geometry validity

I. Polygon is planar - verifying that all points (polygon vertices) forming the surface (polygon) exist on the same plane (e.g., using the plane formula).
II. Polygon ring is closed - verifying that all points (polygon vertices) forming the surface (polygon) are sequentially connected via lines (e.g., using topological rules).
III. Polygon rings are not self-intersecting - polygon rings may not self-intersect nor touch or cross themselves.
IV. Polygon ring is clockwise, inner holes are counterclockwise.
V. Polygon rings are not intersecting with each other.
VI. Polygon vertices are different - validation that no duplicates exist for all polygon vertices' coordinate values, i.e., at least one ordinate value differs.

4.6.2 Surface validity

In case a surface is a collection of surfaces that join in pairs on common boundary curves and considered as a whole:

VII. All polygon normals are pointed outside the volume.

4.6.3 Solid validity

VIII. All edges are part of two planar-polygons.

4.7 Semantic consistency

Semantic consistency is an indispensable precondition for building a correct ontology of the various objects in CityGML. For example, a window cannot be a part of a tree and a building wall is defined as a (CityGML's) RoofSurface.

To achieve geometric-semantical consistent models, it is necessary to:

1. specify allowed alternatives with formal rules, and

2. validate the geometry.

5. Metric definition JSON-schema

5.1 Data Quality ADE guide

This document describes the CityJSON data quality ADE, which defines the objects and fields used to represent the various data quality elements (metrics) of the city model.

Figure 5 depicts a simple example of CityJSON file structure with the quality ADE.

```
{
  "type": "CityJSON",
  "version": "1.0",
  "extensions": {
    "Quality": {
      "url": "http://someurl.com/quality.json",
      "version": "1.0"
    }
  },
  "metadata": {
  },
  "CityObjects": {
  }
}
```

Figure 5. A CityJSON file structure with the Data Quality ADE (in green).

This is a standard CityJSON file, whereas the Data Quality ADE (“Quality”) is specified inside the ***extensions*** object. The url should point to a downloadable JSON file that contains the Data Quality ADE schema. It will be used during validation of the file "quality.json".

Since the CityJSON schema allows adding properties and objects, a **CityJSON file with extension data is a valid CityJSON file.** In this case, the"quality.json" file is required to validate the Data Quality ADE according to the specified schema.

The (added) Data Quality can appear in two forms, both according to CityGML:

1. As a CityObject (e.g., Zone and subZone)
2. As a property (attribute) of the CityObject

5.2 CityObject - type Zone

Currently, the Data Quality ADE has only one CityObject named "***+Zone***". The Zone object holds general quality information in the defined Zone according to the various quality categories (see Figure 1).

Figure 6 depicts an example of a Zone object with Data Quality properties of Completeness, Temporal Reliability, and Visual Quality (set #1). Zone is marked as type is "+Zone" in the example, since according to CityJSON all extension data is marked with a plus sign. In this example, several Data Quality modulus (metrics) exist, although basically only one module suffices to validate the Quality extension.

```
"UUID_20h2fhf0-93hf-22ed-w11w-00e3e303j0jf": {
    "type": "+Zone",
    "geometry": [
        {
            "type": "MultiLineString",
```

```
            "boundaries": [
                [
                    [
                        10,
                        11,
                        12,
                        13
                    ]
                ]
            ],
            "lod": 0
        }
    ],
    "module": "Buildings",
    "completenessSet": {
        "isExist": true,
        "completenessAbsence": 0.9,
        "completenessExcess": 1.1,
        "lod0.0": 0,
        "lod1.0": 0,
        "lod2.0": 0.9,
        "lod3.0": 0,
        "lod4.0": 0,
    },
    "temporalReliability": {
        "measureDate": "2010-11-01",
        "measureTime": "16:06:00",
        "transience": 112
    },
    "positionalQuality": {
        "set1": {
            "CE90": {"value": 3.5, "uom": "m"},
            "LE90": {"value": 1.5, "uom": "m"}
        },
        "set2": {
            "SE90": {"value": 4.2, "uom": "m"},
            "LE90": {"value": 2.2, "uom": "m"}
        }
    },
    "visualQuality": {
         "set1": {
            "textureType": none,
            "resolution": {"value": 0.15, "uom": "m"}
        }
    },
}
```

Figure 6. An example of a Zone object with Data Quality properties.

Geometry (required)

The geometry of the Zone should be a 2D polygon. Still, 2D polygons are not supported in CityJSON, since all polygon vertices must have three ordinates (3D), such that the elevation/height ordinate of all polygon vertices should be set to zero (0).

Module (optional)

This refers to CityGML's module name. Module "Buildings", in this example, means that the Zone refers to all the CityObjects that are in the "Buildings" module (layer). If the module property is not used in the JSON file, it means that the Zone data refers to all CityObjects that exist inside the Zone's boundary.

CompletenessSet (optional)

This property (metric) contains data about the completeness status inside the zone. Note that this data can refer to different objects according to the module property described above. In this example, it refers only to the Buildings ("module": "Buildings"). See 4.4 for more details.

```
"completenessSet": {
    "isExist": true,
    "completenessAbsence": 0.9,
    "completenessExcess": 1.1,
    "lod0.0": 0,
    "lod1.0": 0,
    "lod2.0": 0.9,
    "lod3.0": 0,
    "lod4.0": 0,
},
```

In this example, "isExist" is true, meaning that there exists (reference/real world) measurements for the Building layer inside the zone boundary. "completenessAbsence" equals 0.9, indicating that the dataset does not represent 10% of the objects existing in the real world, and "completenessExcess" equals 1.1, indicating that ~13% of the objects in the dataset may not appear in the real world. "lod2.0" means that the values are valid only for LOD 2.0.

temporalReliability (optional)

This property (metric) contains date and time (temporal) metadata concerning the validity of the data. See 4.5 for more details.

```
"temporalReliability": {
    "measureDate": "2010-11-01",
    "measureTime": "16:06:00",
    "transience": 112
}
```

In this example, "measureDate" and "measureTime" means that the data was acquired on November 1st, 2010, at 16:06:00. "transience" equals 112, meaning that the data in the layer is not expected to change for 112 days after the value in "measureDate".

positionalQuality (optional)

This property (metric) describes the spatial accuracy associated with the layer geometries. See 4.1 for more details.

```
"positionalQuality": {
    "set1": {
      "CE90": {"value": 3.5, "uom": "m"},
      "LE90": {"value": 1.5, "uom": "m"}
    },
    "set2": {
      "SE90": {"value": 4.2, "uom": "m"},
      "LE90": {"value": 2.2, "uom": "m"}
    }
}
```

In this example, two sets are used for representing the position quality of all geometries associated with the layer (Buildings, in this example) inside the zone. Set 1 represents the 2D quality, where “CE90” equals 3.5 means that the horizontal position accuracy is 3.5 meters (“uom” equals “m”), and “LE90” equals to 1.5 means that this is the accuracy of 2D values, such as horizontal/vertical length, derived from the geometry is 1.5 meters. Set 2 represents the 3D quality, where “SE90” equals 4.2 means that the spatial position accuracy is 4.2 meters, and “LE90” equals to 2.2 means that this is the accuracy of 3D values, such as spatial length, derived from the geometry is 2.2 meters.

visualQuality (optional)

This property (metric) describes the quality of the virtual image that can be rendered. See 4.3 for more details.

```
    “visualQuality”: {
         “set1”: {
            "textureType": none,
            "resolution": {"value": 0.15, "uom": "m"}
        }
    },
```

In this example, one set is used for representing the visual quality. Here, “textureType” equals none, meaning that no texture was used, whereas “resolution” equals 0.15 meaning that the associated imagery resolution (pixel size) is 0.15 meters (“uom” equals “m”).

5.3 Additional properties to existing CityObjects

The additional properties - in this case Data Quality metrics - appear inside CityObjects, and begin with +quality- prefix (according to the CityJSON format), depicted in Figure 7.

```
{
      "type": "CityJSON",
      "version": "1.0",
      "CityObjects": {
            "UUID_f498ba14-6b8d-11eb-91c3-00155d01d30d": { ... },
            "UUID_f498ba14-6a8d-11eb-91c3-00155d01d30e": {
                  "type": "Building",
                  "geometry": [ ... ],
                  "+quality-semanticReliability": 0.99,
                  "+quality-visualQuality": {
                       "set1": {
                          "textureType": false,
                          "resolution": {"value": 0.45, "uom": "m"}
                       },
                  },
                  "+quality-visualQuality": {
                        "set2": {
                              "targetSurface": [
                                    0,
                                    0,
                                    1
                              ],
                              "propriety": false,
```

```
                        "isOccludded": false,
                        "distanceToCamera":
                              {"value": 2500, "uom": "m"},
                        "verticality": {"value": 0.5, "uom": "dd"},
                        "resolution": {"value": 0.5, "uom": "m"}
                  }
            },
            "+quality-temporalReliability": {
                  "measureDate": "2017-10-22",
                  "measureTime": "10:33:00",
                  "transience": 100
            },
            "+quality-completeness": {
                  "isExist": true,
                  "completnessExcess": 1.2,
            }
            "+quality-positionalQuality": {
                  "set1": [
                        {
                              "CE90": {"value": 2.45, "uom": "m"},
                              "LE90": {"value": 2.51, "uom": "m"}
                        },
                        {
                              "target": 0,
                              "CE90": {"value": 1.52, "uom": "m"},
                              "LE90": {"value": 1.51, "uom": "m"}
                        },
                        {
                              "target": 1,
                              "CE90": {"value": 2.45, "uom": "m"}
                              "LE90": {"value": 1.88, "uom": "m"}
                        }
                  ],
                  "set2": [
                        {
                              "SE90": {"value": 2.45, "uom": "m"},
                              "LE90": {"value": 2.51, "uom": "m"}
                        },
                        {
                              "target": 0,
                              "SE90": {"value": 1.52, "uom": "m"},
                              "LE90": {"value": 1.51, "uom": "m"}
                  },
                        {
                              "targetGeometry": [
                                    0,
                                    0,
                                    13
                              ],
                              "SE90": {"value": 1.00, "uom": "m"}
                        },
                        {
                              "targetGeometry": [
```

```
                                          0,
                                          0,
                                          15
                                    ],
                                    "SE90": {"value": 1.5, "uom": "m"}
                              }
                        ],
                        "set3": {
                              "targetSurface": [
                                    0,
                                    0,
                                    1
                              ],
                              "position": [
                                    1.5,
                                    1.5,
                                    1.5,
                                    0.1,
                                    0.1,
                                    0.01
                              ],
                              "azimuth": {"value": 1.1, "uom": "dd"},
                              "elevation": {"value": 0.8, "uom": "dd"},
                              "texture_CE90": {"value": 0.2, "uom": "m"}
                        }
                  }
            }
      },
      "vertices": [ ... ]
}
```

Figure 7. An example of a CityObject with Data Quality properties.

+quality-semanticReliability

```
"+quality-reliability": 0.99
```

A percent number indicating the probability for an object in a given layer to show the correct semantics, in this case 99%.

+quality-visualQuality

```
"+quality-visualQuality": {
  "set1": {
      "textureType": realistic,
      "resolution": {"value": 0.45, "uom": "m"}
  }
}
"+quality-visualQuality": {
  "set2": {
      "targetSurface" : [0,0,1]
      "propriety": false,
      "isOccluded": false,
      "distanceToCamera": {"value": 2500, "uom": "m"},
```

```
      "verticality": {"value": 0.5, "uom": "dd"},
      "resolution": {"value": 0.5, "uom": "m"}
  }
}
```

Two sets are given in this example: the general metric (#1), where in this example the texture type is realistic and the resolution is 0.45 meters; and the photogrammetric set (#2), where in this example the propriety is unknown, there is no texture occlusion, the distance to the camera center is 2500 meters, the vertical angle is 0.5 decimal degrees ("uom" equals "dd") and the nominal pixel size is 0.5 meters.

+quality-temporalReliability

```
"+quality-temporalReliability": {
  "measureDate": "2017-10-22",
  "measureTime": "10:33:00",
  "transience": 100
}
```

In this example, the data was acquired on October 22nd, 2017, at 10:33:00, and the data in the layer is not expected to change for 100 days after the value in "measureDate"

+quality-completeness

A percent value representing the completeness of the current cityobject

```
"+quality-completeness": {
   "isExist": true,
   "completnessExcess": 1.2,
}
```

In this example, "isExist" is true, meaning that there exists measurements for the objects, whereas "completenessExcess" equals 1.2 indicating that ~25% of the objects may not appear in the real world.

+quality-positionalQuality

In this example, all four sets for positional quality are used. Not all must appear, and any subset can be present in a real dataset.

Set1 - two-dimensional metrics

```
"+quality-positionalQuality": {
"set1": [
    {
      "CE90": {"value": 2.45, "uom": "m"},
      "LE90": {"value": 2.51, "uom": "m"}
    },
    {
        "target": 0,
        "CE90": {"value": 1.52, "uom": "m"},
        "LE90": {"value": 1.51, "uom": "m"}
    },
    {
        "target": 1,
        "CE90": {"value": 2.45, "uom": "m"}
        "LE90": {"value": 1.88, "uom": "m"}
```

```
    }
]
}
```

In Set1 (2D quality) the horizontal position accuracy (“CE90”) equals 2.45 meters, and derived 2D values (e.g., length) (“LE90”) equal 2.51 meters. These values are attached to all *n* city object parts, whereas parts 0 and 1 have unique values that replace the general (upper level) ones, which are attributed according to the “target”. The table below depicts the values according to the different parts as they appear in the JSON file, i.e., for “target” 0 and 1, with theoretical values for parts 2, 3, ...n.

Semantic part	LE90	CE90
0	1.51	1.52
1	1.88	2.45
2	2.51	2.45
3	1.70	2.55
...		
N	2.51	2.45

Set2 - three-dimensional metrics

```
"+quality-positionalQuality": {
    "set2": [
        {
          "SE90": {"value": 2.45, "uom": "m"},
          "LE90": {"value": 2.51, "uom": "m"}
        },
        {
          "target": 0,
            "SE90": {"value": 1.52, "uom": "m"},
            "LE90": {"value": 1.51, "uom": "m"}
        },
        {
        "targetGeometry": [0, 0, 13],
            "SE90": {"value": 1.00, "uom": "m"}
        },
        {
        "targetGeometry": [0, 0, 15],
            "SE90": {"value": 1.5, "uom": "m"}
        }
    ]
}
```

Set2 behaves similarly to Set1, whereas SE90 replaces CE90 and LE90 refers to, e.g., spatial length. Additionally, a specific element geometry can be targeted using the targetGeometry field that represents an array of indices that describe the location of an elementary geometry (i.e., " Surface", "Line" or "Point"). The type of the elementary geometry can be determined by the geometry type. “targetGeometry” refers to the relevant geometry, as it appears in the JSON file (see below).

Geometries built from Surfaces	Geometries built from Lines	Geometries built from Points
Solid, MultiSolid, CompositeSolid, MultiSurface, CompositeSurface	MultiLineString	MultiPoint

The example here refers to Solid geometry, therefore the targetGeometry refers to the planar-polygon in the geometry field of the current CityJSON object. For example, the target [0, 0, 13] refers to geometry[0]['boundaries'][0][13]. The number of indices can be different for different geometry types, because the polygon is located in different array depth.

In the example above of Set2, the actual values will be:

Planar polygon index	Semantic part	LE90	SE90
0,0,12	0	1.51	1.52
0,0,13	0	1.51	1.00
0,0,14	1	2.51	2.45
0,0,15	1	2.51	1.50
0,0,17	2	2.51	2.45

Note that when a target geometry is Point, LE90 is redundant.

Set3 - polygon level metrics

```
"set3": {
    "targetSurface": [
        0,
        0,
        1
    ],
    "position": [
        1.5,
        1.5,
        1.5,
        0.1,
        0.1,
        0.01
    ],
    "azimuth": {"value": 1.1, "uom": "dd"},
    "elevation": {"value": 0.8, "uom": "dd"},
    "texture_CE90": {"value": 0.2, "uom": "m"}
}
```

Set3 describes the accuracy of a specific "Surface" (planar polygon). targetSurface contains the index of the "Surface" in the geometry array, similarly as described in Set2. Similarly to Set1 and Set2, if the targetSurface is omitted, then the data refers to all surfaces of the current city object. Set3 can also contain "target" field, whereas in that case the data refers to all surfaces in that target. Note that Set3 can be used only in geometries that consist of surfaces.

The position contains six values from which the covariance matrix (positional accuracy) can be constructed, as follows:

#1	#4	#6
#4	#2	#5
#6	#5	#3

In the case of the above example, the matrix values will be:

1.50	0.10	0.01
0.10	1.50	0.10
0.01	0.10	1.50

Note that these values are given in meters, and refer to ECEF coordinate system.