The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences, Volume L-4/W2-2026
21st International 3D GeoInfo Conference 2026, 28–29 September 2026, Sofia, Bulgaria

Improving the usability and discoverability of CityJSON Extensions

Hugo Ledoux

Delft University of Technology, the Netherlands — h.ledoux@tudelft.nl

Keywords: CityJSON, data modelling, extension, CityGML, ADE

Abstract

The data model CityGML supports domain-specific extensions via *Application Domain Extensions* (ADEs); users can add new attributes, classes, and even new geometries for specific domains. A 2018 survey found only 10 of 44 ADEs had publicly available schemas, and I show in this paper that a follow-up survey of 16 post-2018 ADEs shows similar problems, with only 7 providing schemas and 5 providing example datasets (both very useful for reuse by others). During its development, the encoding CityJSON addressed some of the limitations of ADEs through its *Extensions* mechanism: Extensions are simpler and more restricted, for instance no new geometries. However, creating and hosting Extensions remains cumbersome, limiting adoption in practice. This paper presents two contributions to lower that barrier. First, a web application (hosted on the CityJSON website) lets practitioners build Extension files interactively, without needing to write JSON Schema by hand. Second, a public registry on GitHub collects, hosts, and versions all known Extensions at stable, predictable URLs; 9 Extensions are currently included. The registry improves discoverability, provides worked examples, and resolves hosting difficulties such as CORS restrictions.

1. Introduction

The CityGML standard was developed to store and exchange 3D semantic city models, and its core data model represents a consensus set of classes and attributes suited to most practical applications (Gröger and Plümer, 2012). For domain-specific use cases, however, practitioners often need to extend this core model. CityGML supports this through *Application Domain Extensions* (ADEs): additional classes, geometric types, and attributes specific to a given domain can be added in a structured and documented way, and can inherit from the CityGML core classes (Biljecki et al., 2018). An example is the *Noise ADE*, where specific noise-related attributes are defined for buildings (e.g. decibel levels during periods of the day) and where new classes are defined (e.g. `NoiseRoadSegment` where a part of a road can be modelled with specific noise characteristics).

Because of their conceptual appeal, several academic publications describe and present ADEs for a wide range of applications: Biljecki et al. (2018) presented in 2018 a survey with 44 ADEs, and I present in Section 2.1 16 more that have been either proposed or significantly improved since then. These cover a wide range of applications, including food waste (Braun et al., 2021), energy modelling (Agugiaro and Padsala, 2025), spatial planning (Akahoshi et al., 2020), and cultural heritage (Yang et al., 2024).

As highlighted by Biljecki et al. (2018), ADEs are often developed as conceptual exercises (typically within a research project) and rarely reach the stage of a maintained and publicly accessible schema with example data. As further explained in Section 2.1, only 10/44 ADEs before 2018 and 7/16 ADEs since then have publicly released their XML schema definition file (XSD), and even fewer have released demo datasets. The XSD file documents the extensions to the core model and makes it possible for others to create, use, understand, and validate a CityGML dataset; they are essential to implement support for the ADE in a given software package. The majority of publications only contain UML diagrams and/or a description of the extensions, and only some publish schema snippets embedded as figures in PDF documents, rendering them effectively unusable by others.

The situation is exacerbated by the latest version of CityGML (v3.0, released in 2021), which substantially changed both the data model and the ADE mechanism (Kutzner et al., 2020). This prevents existing ADEs from being easily ported to CityGML v3.0, requiring a complete rewrite.

When developing CityJSON (Ledoux et al., 2019), its *Extensions* mechanism (the equivalent of ADEs) was deliberately simplified to avoid the complexity that, I believe, partly led to the disappointing adoption of CityGML ADEs by practitioners. As further explained in Section 2.3, Extensions are more restrictive and constrained: new geometric primitives are not permitted, and only four cases are allowed: a new city object, a new attribute, a new semantic surface, and a new root-level property. These restrictions ensure that standard CityJSON software can process extended files without modification (which is not the case with CityGML ADEs). However, despite this simplicity and the enthusiasm of the community for application-specific classes/attributes, the adoption of Extensions in practice has also not been widespread: creating Extension files is still perceived as cumbersome, and only a limited number of Extensions have been developed and used by cities and organisations.

I present in this paper two contributions that, I believe, will lower the barrier to adopting CityJSON Extensions:

1. a web application that allows practitioners to build Extension files (`*.ext.json`) interactively, without writing JSON schemas by hand (Section 3); and
2. a public registry that collects, hosts, and exposes all known Extensions, making them discoverable and immediately reusable (Section 4).

It should be noted that the need for such a registry has been expressed by practitioners in the past. In their OGC Discussion Paper, Japan's PLATEAU project (Sogawa et al., 2025) states

that "there is no system for broadly sharing ADEs and related technologies" and calls for "a platform that aggregates knowledge and resources related to ADE development from various countries and cities [to] enable the reuse of these resources in next-generation development, and facilitate the spread of the standard."

2. Related work

2.1 ADEs for CityGML v2.0 and earlier

ADEs were already possible with CityGML v1.0 (released in 2008) through GML substitution groups and generic application properties, and a few early extensions were developed by research groups and SIG3D members. However, the concept was formalised and better documented with CityGML v2.0 (Gröger and Plümer, 2012), and almost all named ADEs—including the 44 reviewed by Biljecki et al. (2018)—are specified against v2.0. When v2.0 was released, earlier v1.0-era ADEs were generally migrated and re-published as v2.0 extensions.

ADEs are defined as extensions to the CityGML data model, are implemented as XML Schema Definition (XSD) schemas, and define custom data types and constraints. Practitioners have to define their own XSD schemas, and those should be made available for understanding and validating a dataset. Van den Brink et al. (2013) illustrate how complex developing an ADE can be in practice, requiring expertise in UML modelling, XSD design, and the CityGML data model itself. It should be mentioned that the CityGML data model prescribes a finite set of attributes for its main classes, and that if extra attributes are required they must be defined in an ADE.

In Table 1, I extend the survey of Biljecki et al. (2018) with ADEs that have been substantially improved or newly released since 2018. First, observe that of the 16 ADEs listed, only 7 provide publicly available XSD schemas; in the original survey, only 10/44 did. Second, observe that a few ADEs from the earlier survey have since been significantly improved (notably the Energy ADE of Agugiaro and Padsala (2025) and the Utility Network ADE), but most remain proposals or partial implementations (i.e. without a publicly available XSD schema).

2.2 ADEs for CityGML v3.0: at the UML level

CityGML v3.0 (OGC, 2021) substantially changed both the data model and the ADE mechanism (Kutzner et al., 2020). ADEs can no longer be defined simply as XSD hooks; classes must first be modelled in UML with proper inheritance, and then translated to XSD for a CityGML-XML encoding (OGC, 2023).

Bachert et al. (2024) mention that porting an ADE from v2.0 to v3.0 requires a complete rewrite of UML classes and schemas—an effort that has so far deterred adoption. Agugiaro and Padsala (2025) explain the details of this effort for the Energy ADE, and Yao et al. (2025) describe the substantial changes required to migrate the 3D City Database to support CityGML v3.0. Uggla et al. (2023) describe how the Swedish 3CIM specification, a national extension of CityGML v2.0, required significant ETL processes even for the earlier version, and the migration to v3.0 has not yet been attempted.

The difficulties of upgrading to CityGML v3.0 in practice are reflected in Table 1: currently 9 ADEs target CityGML v3.0, and only 4 have XSD schemas available.

```
{
  "type": "CityJSONExtension",
  "name": "noise",
  "description": "Extension to model the noise",
  "url": "https://cityjson.github.io/extensions/noise/2.0.0/noise.ext.json",
  "version": "2.0.0",
  "versionCityJSON": "2.0",
  "extraAttributes": {...},
  "extraCityObjects": {
    "+NoiseBuilding": {
      "allOf": [
        { "$ref": "cityobjects.json#/_AbstractBuilding" },
        {
          "properties": {
            "type": { "enum": ["+NoiseBuilding"] },
            "attributes": {
              "properties": {
                "buildingLDenMin": {"type": "number"}
              }
            }
          },
          "required": ["type"]
        }
      ]
    }
  },
  "extraRootProperties": {},
  "extraSemanticSurfaces": {}
}
```

Figure 1. Example of a CityJSON Extension file: `noise.ext.json`

2.3 CityJSON Extensions

A CityJSON Extension is defined as a JSON file (`*.ext.json`) that documents how the core data model may be extended and is used during the validation of CityJSON files. As shown in Figure 1, each Extension file must declare its name, URI, version, and the JSON Schema snippets for the new objects and attributes it introduces. These snippets can reuse and refer to the definitions and geometric primitives defined in the core CityJSON schemas. In this specific example (more details are available at `https://www.cityjson.org/specs/#extensions`), a new city object would be defined (called `+NoiseBuilding`, reusing the `Building` geometries and properties) and a new attribute `buildingLDenMin` would be added. New city objects must contain a `"type"` and a `"geometry"` property (omitted here because `_AbstractBuilding` defines it), and their names must begin with a `+` character to avoid conflicts with core types (because JSON Schemas cannot have namespaces, this prefix convention serves as a lightweight alternative). This Extension file would be publicly available and usable at `https://cityjson.github.io/extensions/noise/2.0.0/noise.ext.json`.

It should be noted that CityJSON, being based on JSON, inherently allows arbitrary attributes to be attached to any city object without requiring an Extension (unlike CityGML which requires an Extension to define custom attributes, see above). This "schema-less" flexibility means that practitioners can add custom attributes and document them in prose (e.g. a report) without creating a formal Extension file. In practice, many users take this approach, which is sufficient for internal use but does not enable machine-readable validation or interoperabil-

Table 1. Overview of CityGML ADEs proposed or substantially developed after the review of Biljecki et al. (2018).

	ADE	**Purpose**	**CityGML version**	**Reference(s)**	**Schema/examples public?**
1	Energy ADE	Urban energy modelling	2.0 3.0 (in-progress)	Agugiaro and Padsala (2025)	Yes (schema + data)
2	EnvPlan ADE	Integration of BIM and environmental planning	2.0	Wilhelm et al. (2021)	Yes (schema + data)
3	FWE ADE	Food/Water/Energy modelling at building and land-use level	2.0	Padsala et al. (2021) URL-1	Yes (schema)
4	FEW ADE (Montréal)	Food–Energy–Water nexus for a Montréal case study	—	Braun et al. (2021)	No
5	GRextADE	Greek national extension for 3D city models and cadastre management	2.0	Liamis and Mimis (2022)	No
6	Grotto ADE	Cultural heritage: semantic modelling of Buddhist niches and grotto components	3.0	Yang et al. (2024)	No
7	IfcADE	Retaining IFC semantics when converting BIM to CityGML	3.0	Biljecki et al. (2021)	No
8	Pedestrian ADE	3D modeling of pedestrian mobility space	3.0	Saeidian et al. (2026) URL-3	Yes (schema + data)
9	Property Valuation ADE	3D indoor/outdoor variables for property valuation	3.0	El Yamani et al. (2024)	No
10	Public Safety Features ADE	Indoor public safety (firefighter pre-planning) in OGC pilot	2.0	OGC (2020)	No
11	Strata ADE (SmartKADASTER)	3D strata parcels and legal units in Malaysia	2.0	Siew et al. (2021)	No
12	Sve-Test ADE	National building standard for Sweden, linking BIM and national registers	3.0	Eriksson et al. (2020)	No
13	Utility Network ADE	Multi-utility networks	3.0 (in-progress)	URL-2	Yes (schema)
14	Urban Planning ADE (i-UR)	Urban planning in Japan (PLATEAU)	2.0	Akahoshi et al. (2020)	Yes (schema + data)
15	Vegetation ADE	Detailed and harmonised vegetation modelling	3.0	Petrova-Antonova et al. (2024)	No
16	VicULA ADE	Underground legal/physical spaces; 3D underground parcels and rights	3.0	Saeidian et al. (2023) URL-4	Yes (schema + data)

URL-1: `https://transfer.hft-stuttgart.de/gitlab/in-source/fwe-ade`
URL-2: `https://github.com/tum-gis/citygml3-utility-network-ade`
URL-3: `https://github.com/Bahram-Saeidian/Pedestrian_ADE_CityGML3`
URL-4: `https://github.com/Bahram-Saeidian/VicULA_ADE_CityGML3`

ity with third-party software. Extensions provide documentation and validation that ad hoc attributes cannot, but this is less compelling when the alternative—simply adding attributes in a standard CityJSON file—requires no effort at all. This tension between JSON's permissive nature and the desire for structured, validated extensions is, I believe, one of the reasons for the limited adoption of Extensions.

The CityJSON Extensions mechanism for extending the core model is deliberately more constrained than that of CityGML ADEs: new geometric primitives are not permitted, and extension points are limited to four cases (also seen in Figure 1):

1. adding new city objects;
2. adding new attributes to existing city objects;
3. adding new semantic surfaces; and
4. adding new root-level properties.

These restrictions ensure that standard CityJSON software can read and process files containing Extensions without any modification to the parsing code.

Despite the enthusiasm expressed in the scientific literature, the adoption of Extensions in practice has not been widespread. Table 2 lists the known CityJSON Extensions, both those in the public registry and those described in the literature. To the best of my knowledge, many of the Extensions presented in the scientific literature have their Extension files publicly hosted (Lei et al., 2024; Nys et al., 2021; Tufan et al., 2022; Vitalis et al., 2019). However, some Extensions described in papers do not provide a usable schema file: Vaienti et al. (2022) describe a Historical CityJSON Extension but do not publish a standalone Extension file, and their follow-up work (Vaienti et al., 2023) uses the extension within a specific pipeline without making it reusable. Similarly, Boumhidi et al. (2024) and Jeddoub (2026) both tackle the Dynamizer concept for integrating dynamic data; the former is not structured as a standard Extension, while the latter has been formalised and added to the registry (Table 2). Guler (2023) develops a CityJSON Extension for 3D spatial planning based on the Land Administration Domain Model, and Guler et al. (2026) extend this work to two countries, demonstrating the cross-national applicability of the Extension mechanism. Lim et al. (2024) investigate the integration of movement data with 3D city models using the Dynamizer module, complementing the work of Boumhidi et al. (2024).

3. An automatic builder of Extension files

Creating a CityJSON Extension currently requires writing a JSON file by hand, which demands familiarity with both the CityJSON data model and JSON Schema syntax[1], the latter being somewhat complex for non-developers. The official website of CityJSON provides a tutorial for some aspects of the Noise ADE of CityGML, see `https://www.cityjson.org/tutorials/extension/`. Furthermore, since there are not many Extension files developed and publicly available, it is difficult for practitioners to learn how to build an Extension JSON file by browsing existing ones.

[1] `https://json-schema.org/`

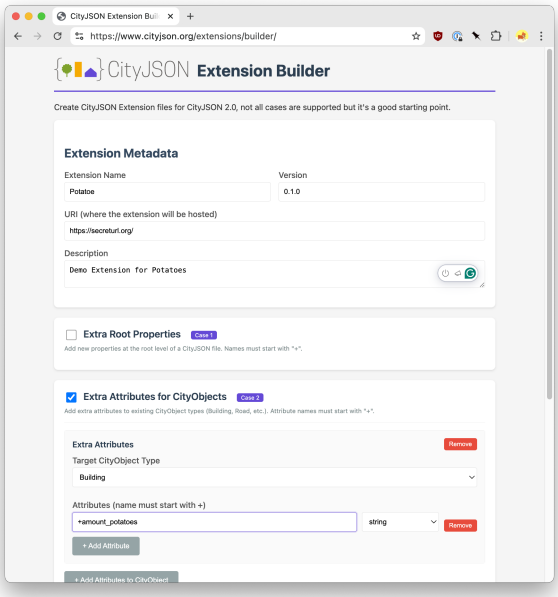

Figure 2. The web application to build Extension files.

To lower this barrier, I have developed a web application that guides practitioners through the process interactively. It is currently hosted on the official CityJSON website (`https://www.cityjson.org/extensions/builder/`) and shown in Figure 2. The user selects one of the four extension cases (new city objects, new attributes, new semantic surfaces, or new root-level properties), inputs names and attribute details, and sees the resulting JSON file update interactively at the bottom of the page. When satisfied with the result, the user can download the file.

Not all advanced cases are supported—deeply nested custom types or complex cross-references may still require manual editing after download—but the application covers the most common scenarios and produces a valid working Extension file that practitioners can immediately test and refine. The application also serves as a learning tool: by observing the generated JSON, users unfamiliar with JSON Schema can understand the structure required and progressively move towards manual editing for more complex cases.

4. A public registry of CityJSON Extensions

A further barrier to adoption has been hosting. Practitioners report difficulties obtaining a stable public URL for their Extension file, encountering *cross-origin resource sharing* (CORS) restrictions[2], and deciding where to publish and maintain it over time. As shown in Figure 3, a CityJSON file references its Extension by a URL, and if that URL is not publicly accessible (or is blocked by CORS policies), the Extension and the CityJSON file itself cannot be validated by third-party software and the user has no documentation for the new attributes and classes.

To address this, I have built a public registry, fully hosted on GitHub, in which any Extension can be submitted and maintained within a single versioned repository. The registry can be accessed at `https://github.com/cityjson/extensions/`.

[2] `https://en.wikipedia.org/wiki/Cross-origin_resource_sharing`

Table 2. Known CityJSON Extensions, from the public registry and the scientific literature.

	Extension	Purpose	Reference(s)	Schema public?	In registry?
1	Energy (space heating)	Space heating demand calculation	Tufan et al. (2022)	Yes	Yes
2	LCC (topology)	Linear cell complex topology	Vitalis et al. (2019)	Yes	Yes
3	Perception	Visual perception of buildings	Lei et al. (2024)	Yes	Yes
4	Noise	Noise emission simulation	—	Yes	Yes
5	MultiRoofs	EU roof modelling project	—	Yes	Yes
6	Quality	Data quality modelling	—	Yes	Yes
7	Shed	Template/example	—	Yes	Yes
8	Point clouds	Point cloud support	Nys et al. (2021)	Yes	No
9	Historical cities	Historical city models	Vaienti et al. (2022)	No	No
10	Dynamizer	Dynamic data integration	Boumhidi et al. (2024), Jeddoub (2026)	Yes	Yes
11	Spatial planning	3D spatial plans (LADM)	Guler (2023)	Yes	No
12	val3dity	Geometry validation reports	–	Yes	Yes

```
1  {
2    "type": "CityJSON",
3    "version": "2.0",
4    "extensions": {
5      "noise": {
6        "url" : "https://cityjson.github.io/extens
         ↪  ions/noise/2.0.0/noise.ext.json",
7        "version": "2.0.0"
8      },
9      "solar-potential": {
10       "url" : "https://someurl.org/solar-potenti
         ↪  al.ext.json",
11       "version": "0.8.1"
12     }
13   },
14   "CityObjects": {...},
15   "vertices": {...},
16 }
```

Figure 3. Example of a CityJSON file referencing two Extensions by URL.

To kick-start the registry, I have contacted the owners of currently available Extensions and copied their latest version to the registry. In cases where the files were for previous versions of CityJSON (e.g. v1.1) or contained errors, I have modified and fixed the files so that they are compatible with CityJSON v2.0. For each Extension, I have either used the example files that were provided or built one manually. An example file allows users to quickly understand how the Extension can be used in practice, and is important for adoption and reuse by others.

The registry contains at the moment 9 Extensions (Table 2). Each Extension is a self-contained file in its own folder; versioning is kept with different folders following semantic versioning; and each version has metadata, one or more example files, and a licence. The Extension file is hosted publicly with a URL that will not be modified and is structured this way: `https://cityjson.github.io/extensions/{name}/{version}/{name}.ext.json`. The approach is inspired by the *STAC Extensions registry* (STAC Extensions, 2024), which follows the same model of community-maintained, openly hosted extension files.

Extension files hosted on the registry can be used directly in CityJSON files by referencing their URL. This means that the official CityJSON validator (see `https://validator.cityjson.org`) can automatically fetch and validate any Extension file hosted on the registry.

Participation is encouraged but not mandatory: a CityJSON file may reference any reachable URL (see line 10 of Figure 3), and practitioners remain free to host their Extensions elsewhere. However, the registry offers several advantages over self-hosting: stable URLs that will not break, no CORS issues, discoverability through a single point of reference, and the presence of example files that serve as worked examples for newcomers.

A further benefit of the registry is that the submission process itself acts as a lightweight quality check. Each submission must include the Extension file, a metadata file, at least one example CityJSON file, a README, and a licence. This ensures that every Extension in the registry is documented, usable, and legally reusable by others.

Using an Extension hosted in the registry. To use an Extension hosted in the registry, a user simply needs to: (1) identify the Extension in the registry; (2) reference the Extension schema in the CityJSON file; (3) follow the Extension's documentation and examples. The URL to use is `https://cityjson.github.io/extensions/{name}/{version}/{name}.ext.json`, and line 6 of Figure 3 shows an example reference in a CityJSON file for the Noise Extension.

Submitting a new Extension. Submitting a new Extension to the registry follows a standard GitHub pull-request workflow:

1. Fork the registry repository.
2. Create a new versioned directory under `extensions/name/version`.
3. Add the five required artefacts: the Extension schema file (`name.ext.json`), a metadata file (`extension.toml`), a README documenting the Extension and its use cases, at least one example CityJSON file, and a licence file.
4. Ensure the Extension name is unique, lowercase, and hyphenated (e.g. `park-benches`).
5. Commit and push the changes to the fork.
6. Open a pull request with a clear description of the Extension.

Once merged, the Extension file becomes publicly accessible at a stable URL of the form `https://cityjson.github.io/extensions/{name}/{version}/{name}.ext.json`, and can be referenced by other applications using the `extensions` field in the CityJSON header (see Figure 3 line 4).

New versions are added as separate directories, following semantic versioning[3], so that existing datasets referencing older versions remain valid.

5. Discussion and future work

The limited adoption of CityJSON Extensions is, I believe, caused by several interrelated factors. First, there is a cascading effect: few Extensions exist, so practitioners have few examples to learn from, which discourages the creation of new ones. The registry aims to break this cycle by collecting all known Extensions in one place and providing worked examples. Second, the "schema-less" nature of JSON means that practitioners can add arbitrary attributes to CityJSON files without creating a formal Extension, and many do. While this is a strength of CityJSON, it undermines the incentive to invest effort in creating a documented Extension. Third, creating and maintaining an Extension is primarily a governance and documentation concern rather than a technical necessity: the data can be used without one. This means that the motivation to create an Extension often comes from organisations that need to exchange data with external parties or comply with standards, rather than from individual practitioners.

The builder and the registry address the practical barriers—the difficulty of writing JSON Schema by hand and the difficulty of hosting and discovering Extension files—but they do not address the fundamental question of whether formal Extensions are always necessary. For many use cases, especially internal ones, adding attributes directly and documenting them in a data dictionary is sufficient. Extensions become valuable when data must be exchanged between organisations, validated automatically, or processed by third-party software that needs to understand the schema.

I see several directions for future work. A curated collection of real-world Extensions, such as the one in the registry, could serve as training material for large language models (LLMs) to assist practitioners in authoring new Extensions. Coetzee et al. (2020) argue that accessibility and usability of geospatial software and data should improve to engage broader communities, and LLM-assisted authoring could further lower the barrier. I also plan to enhance the Extension builder to add the generation of synthetic data based on the schema, so that users can test their Extensions without needing to create a real dataset. This could be done by modifying or using the existing tool `cityjson-fake`[4].

References

Agugiaro, G., Padsala, R., 2025. A proposal to update and enhance the CityGML Energy Application Domain Extension. *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, X-4/W6-2025, 1–8.

Akahoshi, K., Ishimaru, N., Kurokawa, C., Tanaka, Y., Oishi, T., Kutzner, T., Kolbe, T. H., 2020. i-Urban Revitalization: Conceptual modelling, implementation, and visualization towards sustainable urban planning using CityGML. *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, V-4-2020, 179–186.

Bachert, C., León-Sánchez, C., Kutzner, T., Agugiaro, G., 2024. Mapping the CityGML Energy ADE to CityGML 3.0 using a model-driven approach. *ISPRS International Journal of Geo-Information*, 13(4), 121.

Biljecki, F., Kumar, K., Nagel, C., 2018. CityGML Application Domain Extension (ADE): Overview of developments. *Open Geospatial Data, Software and Standards*, 3(1).

Biljecki, F., Lim, J., Crawford, J., Moraru, D., Tauscher, H., Konde, A., Adouane, K., Lawrence, S., Janssen, P., Stouffs, R., 2021. Extending CityGML for IFC-sourced 3D city models. *Automation in Construction*, 121, 103440.

Boumhidi, K., Nys, G.-A., Hajji, R., 2024. *Integrating dynamic data with 3D city models via CityJSON Extension*. Springer Nature Switzerland, 745—758.

Braun, R., Padsala, R., Malmir, T., Mohammadi, S., Eicker, U., 2021. Using 3D CityGML for the modelling of the food waste and wastewater generation—a case study for the city of Montréal. *Frontiers in Big Data*, 4, 662011.

Coetzee, S., Ivánová, I., Mitasova, H., Brovelli, M., 2020. Open geospatial software and data: A review of the current state and a perspective into the future. *ISPRS International Journal of Geo-Information*, 9(2), 90.

El Yamani, S., Hajji, R., Billen, R., Arroyo Ohori, K., van der Vaart, J., Hakim, A., Stoter, J., 2024. Towards extending CityGML for property valuation: Property Valuation ADE. *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, X-4/W5-2024, 127–135.

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(2), 78.

Gröger, G., Plümer, L., 2012. CityGML—interoperable semantic 3D city models. *ISPRS Journal of Photogrammetry and Remote Sensing*, 71, 12–33.

Guler, D., 2023. Implementation of 3D spatial planning through the integration of the standards. *Transactions in GIS*, 27(8), 2252–2277.

Guler, D., Sun, J., Paulsson, J., 2026. Representing the spatial plans as 3D standardized geodatasets: Demonstrations from Sweden and Türkiye. *Transactions in GIS*, 30(1), e70186.

Jeddoub, I., 2026. Urban digital twins levels of integration: From conceptualisation to technical implementation. PhD thesis, University of Liège, Belgium.

Kutzner, T., Chaturvedi, K., Kolbe, T. H., 2020. CityGML 3.0: New functions open up new applications. *PFG—Journal of Photogrammetry, Remote Sensing and Geoinformation Science*, 88(1), 43–61.

Ledoux, H., Ohori, K. A., Kumar, K., Dukai, B., Labetski, A., Vitalis, S., 2019. CityJSON: A compact and easy-to-use encoding of the CityGML data model. *Open Geospatial Data, Software and Standards*, 4(4).

Lei, B., Liang, X., Biljecki, F., 2024. Integrating human perception in 3D city models and urban digital twins. *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, X-4/W5-2024, 211–218.

[3] `https://semver.org`

[4] `https://github.com/3DGI/cityjson-rs/tree/main/crates/cityjson-fake`

Liamis, T., Mimis, A., 2022. Establishing semantic 3D city models by GRextADE: The case of the Greece. *Journal of Geovisualization and Spatial Analysis*, 6, 15.

Lim, J., Biljecki, F., Stouffs, R., 2024. Integration of movement data into 3D GIS. *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, X-4/W5-2024, 219–227.

Nys, G.-A., Kharroubi, A., Poux, F., Billen, R., 2021. An extension of CityJSON to support point clouds. *International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, XLIII-B4-2021, 301–306.

OGC, 2020. OGC indoor mapping and navigation pilot: Public safety features CityGML ADE engineering report. Open Geospatial Consortium. Document 19-032.

OGC, 2021. OGC City Geography Markup Language (CityGML) part 1: Conceptual model standard. Open Geospatial Consortium inc. Document 20-010, version 3.0.0, available at `https://docs.ogc.org/is/20-010/20-010.html`.

OGC, 2023. OGC City Geography Markup Language (CityGML) part 2: GML encoding standard. Open Geospatial Consortium inc. Document 21-006r2, version 3.0.

Padsala, R., Gebetsroither-Geringer, E., Peters-Anders, J., Coors, V., 2021. Inception of harmonising data silos and urban simulation tools using 3D city models for sustainable management of the urban food, water and energy resources. *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, VIII-4/W1-2021, 81–88.

Petrova-Antonova, D., Malinov, S., Mrosla, L., Petrov, A., 2024. Towards a conceptual model of CityGML 3.0 Vegetation ADE. *International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, XLVIII-4/W10-2024, 155–161.

Saeidian, B., Hamzei, E., Tomko, M., Winter, S., 2026. A CityGML extension for 3D modeling of pedestrian mobility space. *Cities*, 178, 107302.

Saeidian, B., Rajabifard, A., Atazadeh, B., Kalantari, M., 2023. A semantic 3D city model for underground land administration: Development and implementation of an ADE for CityGML 3.0. *Tunnelling and Underground Space Technology*, 140, 105267.

Siew, C. B., Abdul Halim, N. Z., Karim, H., Mohd Zain, M. A., Looi, K. S., 2021. CityGML Application Domain Extension for 3D strata representations in the SmartKADASTER system: Towards beyond cadastre purpose in Malaysia. *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, VIII-4/W2-2021, 99–103.

Sogawa, Y., Kurokawa, C., Ishimaru, N., Kolbe, T. H., 2025. Building an ecosystem for the development, utilization, and open data of 3D city models based on CityGML: Sharing insights. Open Geospatial Consortium inc. Document 25-032, available at `http://www.opengis.net/doc/DP/building-ecosystem-3d-city-models`.

STAC Extensions, 2024. STAC Extensions registry. `https://stac-extensions.github.io/`.

Tufan, O., Arroyo Ohori, K., León-Sánchez, C., Agugiaro, G., Stoter, J., 2022. Development and testing of the CityJSON energy extension for space heating demand calculation. *International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, XLVIII-4/W4-2022, 169–176.

Uggla, M., Olsson, P.-O., Abdi, B., Axelsson, B., Calvert, M., Christensen, U., Gardevärn, D., Hirsch, G., Jeansson, E., Kadric, Z., Lord, J., Loreman, A., Persson, A., Setterby, O., Sjöberger, M., Stewart, P., Rudenå, A., Ahlström, A., Bauner, M., Hartman, K., Pantazatou, K., Liu, W., Fan, H., Kong, G., Li, H., Harrie, L., 2023. Future Swedish 3D city models—Specifications, test data, and evaluation. *ISPRS International Journal of Geo-Information*, 12(2), 47.

Vaienti, B., Guhennec, P., di Lenardo, I., 2022. A data structure for scientific models of historical cities: extending the CityJSON format. *Proceedings 6th ACM SIGSPATIAL International Workshop on Geospatial Humanities*, ACM, 20—23.

Vaienti, B., Petitpierre, R., di Lenardo, I., Kaplan, F., 2023. Machine-learning-enhanced procedural modeling for 4D historical cities reconstruction. *Remote Sensing*, 15(13), 3352.

van den Brink, L., Stoter, J., Zlatanova, S., 2013. UML-based approach to developing a CityGML application domain extension. *Transactions in GIS*, 17(6), 920–942.

Vitalis, S., Arroyo Ohori, K., Stoter, J., 2019. Incorporating topological representation in 3D city models. *ISPRS International Journal of Geo-Information*, 8(8), 347.

Wilhelm, L., Donaubauer, A., Kolbe, T. H., 2021. Integration of BIM and environmental planning: The CityGML EnvPlan ADE. *Journal of Digital Landscape Architecture*, 6, 332–343.

Yang, S., Hou, M., Fan, H., 2024. CityGML Grotto ADE for modelling niches in 3D with semantic information. *Heritage Science*, 12, 135.

Yao, Z., Nagel, C., Kendir, M., Willenborg, B., Kolbe, T. H., 2025. The new 3D City Database 5.0—Advancing 3D city data management based on CityGML 3.0. *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, X-4/W6-2025, 241–248.