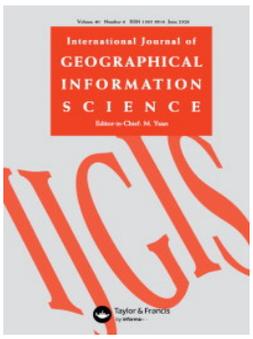

International Journal of Geographical Information Science

ISSN: 1365-8816 (Print) 1362-3087 (Online) Journal homepage: www.tandfonline.com/journals/tgis20

Leveraging knowledge graphs and semantic web technologies for validating 3D city models

Alper Tunga Akın, Ziya Usta, Jantien Stoter, Ken Arroyo Ohori & Çetin Cömert

To cite this article: Alper Tunga Akın, Ziya Usta, Jantien Stoter, Ken Arroyo Ohori & Çetin Cömert (2026) Leveraging knowledge graphs and semantic web technologies for validating 3D city models, International Journal of Geographical Information Science, 40:6, 1867-1893, DOI: 10.1080/13658816.2025.2578723

To link to this article: https://doi.org/10.1080/13658816.2025.2578723

INTERNATIONAL JOURNAL OF GEOGRAPHICAL INFORMATION SCIENCE
2026, VOL. 40, NO. 6, 1867–1893
https://doi.org/10.1080/13658816.2025.2578723

Taylor & Francis
Taylor & Francis Group

RESEARCH ARTICLE

Leveraging knowledge graphs and semantic web technologies for validating 3D city models

Alper Tunga Akın[a,b], Ziya Usta[a,c], Jantien Stoter[a], Ken Arroyo Ohori[a] and Çetin Cömert[b]

[a]3D Geoinformation Group, Delft University of Technology, Delft, Netherlands; [b]Department of Geomatics Engineering, Faculty of Engineering, Karadeniz Technical University, Trabzon, Turkey; [c]Department of Geomatics Engineering, Faculty of Engineering, Artvin Çoruh University, Artvin, Turkey

ABSTRACT

The widespread use of three-dimensional (3D) city data plays a significant role in various applications, such as mixed reality, infrastructure facility management, solar potential analysis, navigation, and so on. Ensuring high spatial and semantic quality in these endeavours is crucial to gathering proper results. Ensuring quality means verifying that the data adheres to relevant standards. Although these relevant standards are openly published, there are issues with the names of interoperability and reusability in academic studies and software development efforts. In this study, these issues are addressed using semantic web technologies. Most 3D city models (3DCMs) are treated as knowledge graphs (KG) with this approach. The main contribution of the study is a web-based interoperable tool for validation of CityGML Level of Detail 2 (LOD2) 3DCMs, which is compatible with relevant standards. Besides, an open-source 3DCM-to-KG converter and an open validation ontology are published as by-products while accomplishing the main goal. By virtue of the KG approach, the 3DCM KG becomes capable of carrying its own validation constraints, which come from the validation ontology. With these efforts, this study provides a practical, interoperable solution to improve the quality and usability of 3DCMs and validation plans, fostering consistency across applications while aligning with established standards in the field.

ARTICLE HISTORY
Received 15 May 2025
Accepted 18 October 2025

KEYWORDS
3D city models; data quality; semantic web technologies; knowledge graphs; validation

1. Introduction

Geographic or spatial data constitutes 80% of the ∼2.5 exabytes of data produced daily in the world, and 60% of the data held in existing databases is spatial data (Burns and Thatcher 2015, VoPham *et al*. 2018). This sheer amount of spatial data underpins numerous applications, from navigation systems to urban planning, where the need for detailed, accurate, and structured representations of the physical world is ever-growing. Given this dominance of geospatial data, 3D city models (3DCMs)

CONTACT Alper Tunga Akın atakin@tudelft.nl; alpertunga@ktu.edu.tr

emerge as a critical tool, transforming raw spatial information into structured, three-dimensional representations of urban environments. 3DCMs are digital representations of urban objects with three-dimensional geometry, foremost buildings, and are essential subjects for a large range of tasks which are vital for our understanding and planning of cities today (Biljecki *et al*. 2015). 3DCMs should ensure high spatial and semantic quality to provide accurate results from these tasks. For example, in a solar analysis, defining surfaces that are not roofs used as roofs or errors in the surface geometries will be misleading in terms of the solar panel placement. Besides, the use of invalid models in disaster management, such as post-disaster rescue efforts, can be problematic in terms of public safety by providing incorrect information about building accessibility or evacuation routes. In telecommunications, incorrect building heights or locations can cause flawed signal propagation analyses, leading to inefficient placement of antennas or 5 G base stations. As another example, in urban planning, invalidities in 3D models may lead to suboptimal zoning or infrastructure decisions. Therefore, purchasing and using invalid data can cause decision-making errors, economic and other drawbacks (Krämer *et al*. 2007), and ensuring quality has been a crucial research topic in the field. With the mentioned use of 3D city models across various applications and domains, CityGML was introduced as a standard data model and an exchange format (Gröger *et al*. 2012). According to ISO 19157 (2013) (Geographic information–Data quality), there are multiple dimensions of quality, such as completeness, positional accuracy, thematic accuracy, temporal quality, and logical consistency. While the data quality assessments on positional accuracy (Dukai *et al*. 2021, Gabara and Sawicki 2021, Singla and Trivedi 2022) handle the accuracy of coordinates compared to real-world locations, data completeness, thematic accuracy, logical consistency, and temporal accuracy efforts require a validation with respect to a standard data model, which is CityGML in the 3DCM field.

The validation on CityGML datasets generally consists of three main tasks, which are schema validation, geometry validation, and semantic validation. Schema validation is the validation of CityGML documents against CityGML schema documents, and it can be considered a well-known solution due to its being widely performed in the literature. Geometry validation is the validation of city objects in a CityGML dataset against the definitions in the ISO:19107 Spatial Schema Standard (ISO 2003, Biljecki *et al*. 2016). This standard provides a description set for representing and manipulating geographic entities through vector-based geometric primitives, such as points, curves, surfaces, and solids, and topological relationships, ensuring that the geometry of city objects adheres to consistent and mathematically defined structures. As an example of the geometry validation control, it can be prescribed that GML rings are topologically closed, and the first and last points of GML rings have to be identical. Semantic validation is performed even if the objects in the CityGML data pass schema validation and geometric validation to detect if they contain unrealistic representations in object classes or attribute fields. An example of a semantic error is that the value in the attribute field expressing the measured height of a building does not match the height value calculated from the coordinate arrays, or a wall or balcony surface is represented as a roof. The structured list of which of all these controls will be applied on a CityGML file and in what order is called a validation plan (Ledoux and Wagner 2016).

Studies before 2013 do not include semantic validation (Ledoux and Meijers 2011, Alam *et al*. 2013, Ledoux 2013). These studies only dealt with the geometric validation of city models. The semantic validation concept was obviously emphasised in the study done by Wagner *et al*. (2013) for the first time. In addition to performing semantic validation, this study emphasises interoperability and the fact that data exchange should satisfy the needs of the data provider and the client.

With the experiment (Quality Interoperability Experiment, QIE, 2016) carried out as a joint activity between OGC (Open Geospatial Consortium), SIG3D (Special Interest Group 3D), and EuroSDR (European Spatial Data Research), done under the editorial of Ledoux and Wagner (2016), validation checks of CityGML 3DCMs are classified and tested by various software. In their study, requirements for the validation of CityGML-formatted city models are defined in detail. The tested validation requirements for CityGML data consist of semantic checks for the 'Building' thematic class in addition to schema and geometry validation. While the study shows an extensible requirement coding system for other thematic classes, the tested requirements in this study can be considered as a minimum consensus basis, as buildings are essential elements of 3DCMs. Although this study is a well-organized guide in the field, there is still a lack of agreement with the minimum basis in the applied validation plans, even in most current studies. The complexity of CityGML schemas, the huge number of thematic domain extensions, and limited collective efforts can be seen as the most obvious reasons for the lack of such agreement. This situation hinders interoperability because different methodologies are not able to produce an identical validation result for the same 3DCM due to the lack of consensus. Most of the current studies on the quality are focusing on the positional accuracy of models regarding ground truth data (Dukai *et al*. 2021, Gabara and Sawicki 2021, Singla and Trivedi 2022). Dukai *et al*. (2021) also employed the val3dity (Ledoux 2018) software to perform geometry validation, besides the positional accuracy. The study of Chadzynski *et al*. (2023) performs several semantic checks without depending on any basis, such as QIE. They also did not perform any geometric checks. So, while semantic validation requires being valid geometrically, the results of the performed checks are not meaningful. In (Pauwels *et al*. 2024), they perform a validation process on IFC-formatted (Industry Foundation Class) BIM (Building Information Model). As stated in the study, there is a lack of standardization work in the BIM field for interoperability, in which quality requirements are defined. Therefore, this study is in a state of a prototype.

The first realised application of QIE is the study (Coors *et al*. 2020). They adopted a validation plan based on QIE for the context of heating demand simulation, using CityDoctor (CityDoctor2 2025) software. With the last version of CityDoctor software, a realized tool for QIE is freely available to use. The error codes and the minimum validation requirements in QIE are implemented in the latest version of CityDoctor. Besides, the validation results are stored in the same CityGML file through DataQualityADE (Betz and Coors 2021). The implementation is a remarkable progress for interoperability, while the rest of the available tools do not serve identical results even if they are applying the same checks (Ledoux and Wagner 2016). The difference between error codes and results for the same checks also hinders the reusability of validation results because comparing and evaluating these results together in the same process at

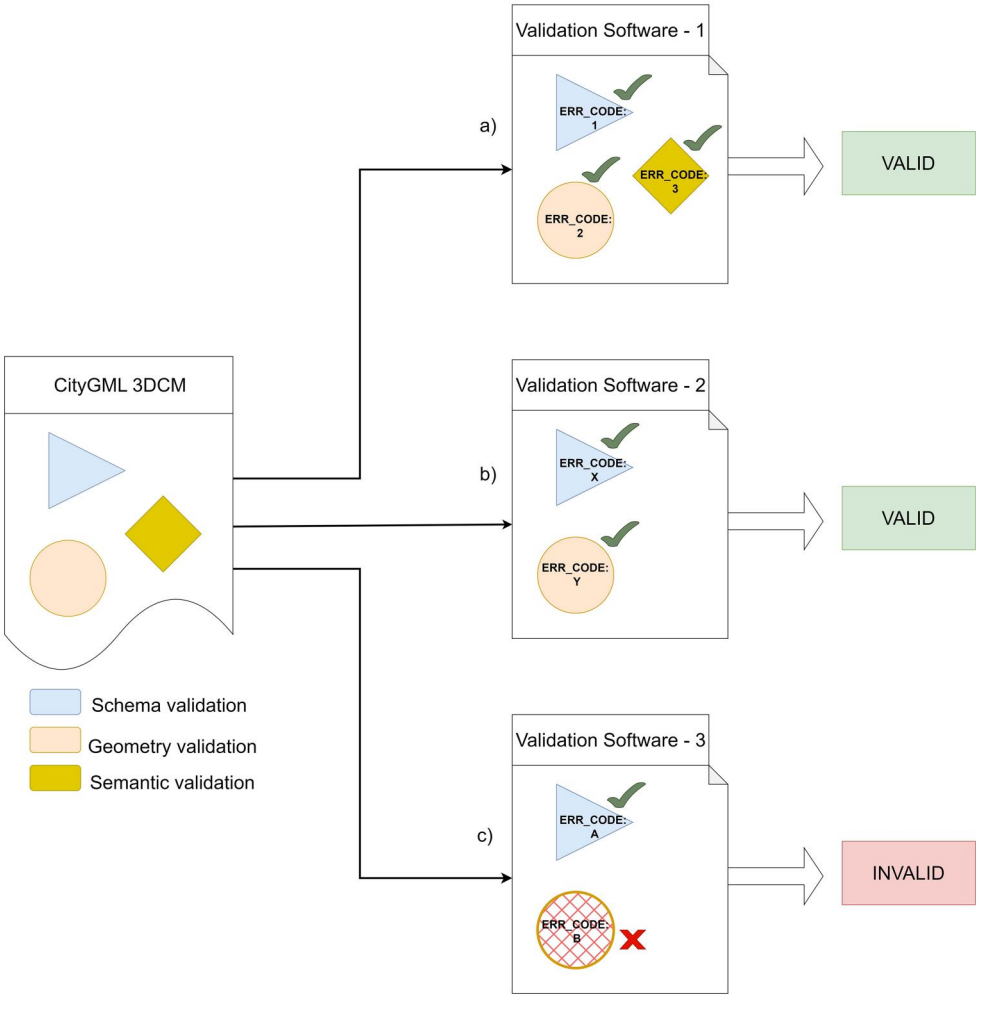

Figure 1. Validation results of three fictional validation software against the same 3DCM data. Software-1 and Software-2 claim the 3DCM is valid, but Software-2 does not include semantic validation. Software-1 and Software-2 include the same kind of checks but produce different results. This shows the need for more consensus between applied checks in the context of geometry validation. Besides, while different providers are using different software, melting their results into the same pot requires a code-matching phase due to the difference in error codes, even if the applied checks are the same.

different times sequentially requires additional code-matching processes. The interoperability and reusability problems can be demonstrated in Figure 1.

In this study, we have aimed to solve interoperability and reusability problems by leveraging the knowledge graph (KG) approach and semantic web (SW) technologies. With this approach, validation plans can be directly transferred and accessible online with the associated 3DCM KG as validation graphs. Section 1.1. describes the relationship between semantic web technologies and the validation process in detail while exposing the effect of the KG approach on the solution of the interoperability and reusability problems. We have described our solution and developed web-based

software in Section 2. Section 3 discusses the experimental results according to the current state of other relevant solutions in the literature. Finally, future plans and prospects of the adopted approach are put forward in Section 4.

1.1. Knowledge graphs, semantic web, and validation

Semantic web (SW) is an extension of the well-known document-based web, which enables the cooperation of people and computers using the information with well-defined meaning (Lassila *et al*. 2001, Cömert *et al*. 2010). Today, we know this cooperation as interoperability, and the mentioned well-definition of meaning is possible thanks to the links between resources. The SW paradigm uses RDF (Resource Description Framework) language, or format, for interlinking between knowledge resources (World Wide Web Consortium 2014). These interlinked resource networks defined in RDF format are also known as knowledge graphs (KGs), which are directed and labelled graphs (Pauwels *et al*. 2017).

RDFs are constructed with triples, which are subject, predicate, and object sequences. Subjects and objects denote entities that represent objects from the world with URIs (Unique Resource Identifiers), and predicates denote relationships between these entities. Therefore, subjects and objects become nodes, and predicates become edges of an RDF-formatted KG. KGs are used either to represent application domains or individual relations in any domain (Nardi and Brachman 2003). If a KG represents classes and properties of an application domain, the KG is named as an ontology, a T-Box, or a namespace (Kostovska *et al*. 2021). We define the ontologies using OWL (Web Ontology Language), an RDF derivative syntax which is a W3C recommended standard for the modelling of ontologies (Hitzler *et al*. 2009). Suppose a KG represents relations between individuals in any domain, represented by a T-Box (the T stands for 'terminology'), or ownership of attribution, the KG is named an A-Box (the A stands for 'assertions'). This manner enables an A-Box to carry its relevant definitions through a T-Box over the web, so this semantic richness of data transfer between different information systems or applications in various domains eases the interoperability. This reliably compatible transfer mechanism is even more useful for the data among applications or situations that include unforeseen schemas (Van den Brink *et al*. 2013, Vinasco-Alvarez *et al*. 2020).

Thanks to the information transfer capability between domains of KGs, there are a variety of studies that mention augmented queries in multi-sourced or multi-domain data, or multi-domain information merging. These studies mainly imply ontological enrichments on CityGML data model to allow more complex semantic queries than conventional ones restricted to the data model borders. 3DCityKG software (Nguyen 2024, 3DCityKG 2025) maps CityGML models into KGs to enable change detection, multi-modal route planning, and querying building interiors and urban paths using flexible graph-database techniques. Similarly, another KG framework developed by Ding *et al*. (2025) generates an RDF knowledge graph from 3DCityDB (Yao *et al*. 2018) using declarative mapping, enabling integration with OpenStreetMap and semantic querying across datasets. In parallel, Lam *et al*. (2024) developed a framework for GIS–BIM integration built on RDF graphs representing both CityGML and IFC data, and

merging them through ontology-based matching to realize unified graph querying across geospatial and building information domains. The capability of KGs for different-purpose querying, the data validation, is also currently being researched in the architecture, engineering, and construction (AEC) industry. While most of the efforts in the literature are focused on BIM data (Werbrouck *et al*. 2019, Hagedorn and König 2020, Hagedorn *et al*. 2023, Pauwels *et al*. 2024), there is still a research gap for a noteworthy effort for CityGML models. The most notable reasons for this research gap are the technical complexity of handling CityGML's rich semantics and the absence of officially standardized and consensually accepted shared ontologies, or suitable vocabularies for ontology development. While the relevant organizations or institutions still had not concluded such formalization, experiments like QIE 2016 can be used as starter evidence for such proof of the concept by converting it into a shared ontology. To address this, we adopt a KG-based approach for CityGML validation using the Shapes Constraint Language (SHACL) (Knublauch and Kontokostas 2017), a W3C-recommended standard for validating RDF-formatted KGs (Cimmino *et al*. 2020), thereby filling this methodological gap with a realized application of QIE 2016.

Before the explanation of how SHACL handles validation tasks, it should also be noted why it stands front among its alternative technologies, and why it is chosen for this study. One of these alternatives is SPARQL query-based validation (Kontokostas *et al*. 2014, Chadzynski *et al*. 2023). This approach provides flexibility to create custom queries for validation, but does not provide a standardized schema-based framework, and quickly becomes difficult to maintain for large ontologies. Another alternative is ShEx (Shape Expressions). ShEx provides a syntax for describing desired RDF patterns (Prud'Hommeaux *et al*. 2014). ShEx provides a simple syntax for pattern checks, but lacks the same extensibility and integration, and has less mature open tools and ecosystem compared to alternatives. The last alternative, which should be mentioned, is SWRL (Semantic Web Rule Language) (Horrocks *et al*. 2004). SWRL is based on the Open World Assumption (OWA) of OWL ontologies. Under OWA, the lack of information is treated as 'unknown', which is an open gap for inference, but can be problematic when enforcing strict validation rules. SHACL depends on the Closed World Assumption (CWA), and is more suitable for validation tasks, where missing or inconsistent data should be flagged as violations rather than ignored as 'unknown'. This fundamental difference makes SHACL the most well-fitted tool for this study, while SWRL can be more appropriate for use cases that need open-ended or open-to-interpretation query results, such as search engines (Knublauch and Kontokostas 2017, Cimmino *et al*. 2020).

The SHACL engine validates RDF graphs by defining shapes, which are sets of constraints, and by comparing these constraints against nodes of the RDF graph. The constraints in a shape are mandatory or restrictions that RDF data must satisfy and are defined according to an ontology, which is also where the RDF graph nodes are defined (What is SHACL? 2025, Cimmino *et al*. 2020). A set of SHACL shapes is also known as a shape graph; hence, a SHACL document can also be considered as another KG to validate the RDF-formatted data graph (Pareti and Konstantinidis 2021). For the validation process, constraints in an SHACL shape are compared to the

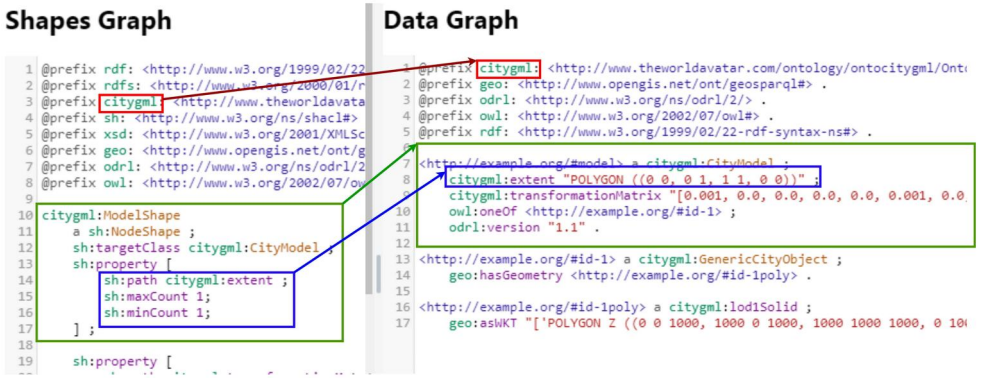

Figure 2. Prefixes at the top of both graph documents represent URIs of the used ontologies. The prefix "citygml" (highlighted in red) belongs to the ontology in which data and shape graphs are both defined. The shape "citygml: ModelShape" (highlighted in green) contains constraints to validate the individual, or instance, that belongs to the CityModel class in the "citygml" ontology. The constraint "citygml: extent" (highlighted in blue) contains the restrictions, which are about the count of occurrences for the "extent" property belonging to the CityModel instance.

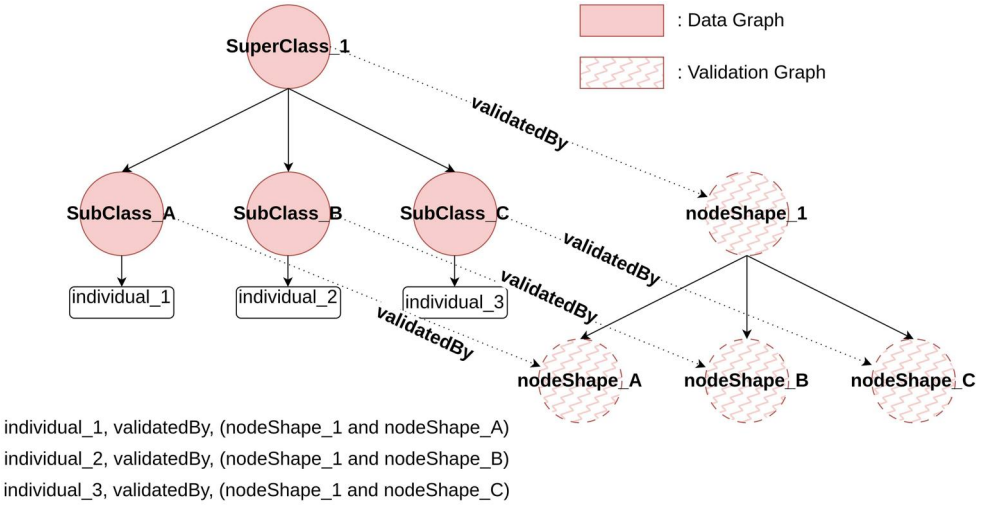

Figure 3. The connectivity between data and shape graphs. Each shape in the shape graph validates an individual, or a target node, within the same hierarchy as the data graph.

associated data graph node. This associated node is also known as the target node. If the SHACL processor finds an inconsistency between a shape and a node, it reports a violation for the target node regarding the shape in question in the validation report. The relation between SHACL shapes and data graph nodes is demonstrated in Figure 2, and the unity of connected data and shape graphs is shown in Figure 3.

By virtue of the explained mechanism, in this study, we have adopted the KG approach to validate LOD2 (Level of Detail 2) CityGML Buildings, considering the validation constraints in QIE 2016. With this approach, it can be enabled to carry a 3DCM KG with its own shape graph, which contains validation constraints. Besides, it could

also be enabled to process multidomain data in different formats into the same application frame, thanks to the RDF meeting point and the data being self-descriptive. This approach will play a key role in ensuring interoperability by informing the data client about what the data is being validated against and restricting the data provider about the standards according to which the data should be presented. In addition to the merits mentioned above, semantic web technologies allow for the inference of new information using the existing relationships in the data (Zeshan *et al*. 2023). This enables the discovery of implicit relationships or, in our context, implicit validation rules that are not explicitly stated in the KG. This capability also opens a way to recover missing relations or eliminate inconsistencies from ontology statements (Vinasco-Alvarez *et al*. 2021). With the given fundamentals of the KG approach, the main contributions of the study can be listed as follows:

- A comprehensive KG-based validation that includes both geometric and semantic aspects of the data to achieve interoperability,
- Enabling the sharing of the validation aspects and results alongside the data itself online for reusable results and validation plans,
- A framework that is extensible for any other use cases and can infer implicit relations in ontologies.

To achieve the main contributions above, the following developments have been concluded:

- An open validation ontology to formalize the validation needs of LOD2 CityGML Buildings, based on QIE 2016,
- An open CityJSON-to-RDF converter that satisfies the properties of the validation ontology with a custom computational geometry functions list,
- Automated shapes graph generation from the ontology using open tools,
- A web interface developed with Pythonic web tools to prove the methodology and to visualize the results, which is also shared openly.

Our methodology, which is explained in detail in Section 2, consists of the main steps below to implement this KG approach;

- Creating a 'Validation Ontology' based on QIE 2016,
- Extracting the shape graph from 'Validation Ontology' using automated web tools,
- Converting the CityJSON data into an RDF graph,
- Validation of the RDF graph against the shape graph,
- Usage of the pipeline through the web interface with different CityJSON datasets,
- Interpretation of the validation report content

The CityJSON encoding of the CityGML data model is chosen for this study thanks to its comfortable mapping to Python dictionaries and its less nested structure than GML encoding (Ledoux *et al*. 2019). Besides, since JSON is the standard for web data exchange, CityJSON fits naturally with REST APIs, JavaScript applications, and modern

web frameworks. Even if the CityJSON is chosen as an encoding or a data format, the CityGML data model is still the basis of this study. Furthermore, while many city applications are still using the CityGML format, there are also open converters between CityGML and CityJSON (citygml-tools 2025). Hence, there is no hindrance to developing the proposal over the CityJSON format.

2. Methodology

2.1. General pipeline

The general pipeline of the implemented validation process is demonstrated in Figure 4. According to the pipeline, the client accesses the validation software and uploads their CityJSON-formatted city model data. Following file upload, the client should choose the ontology against which the data will be validated. In our implementation, regarding our needs, the only ontology option is the validation ontology, which is created based on QIE 2016 (2. component in Figure 4). After this selection, the selected ontology and the CityJSON data are sent to the server for the validation process. On the server side, the CityJSON file is converted to an RDF file, in other words, 3DCM KG (1. component in Figure 4). The KG is validated against the SHACL graph generated regarding the validation ontology via an automated OWL to SHACL web tool (3. component in Figure 4). The validation process produces a human-readable validation report that includes all geometric and semantic violations for every geometric object in a city model, such as Ring, Polygon, and Shell (4. component in Figure 4). Then, this report is sent back to the client for examination. The rest of the section continues with explanations of the components of the architecture.

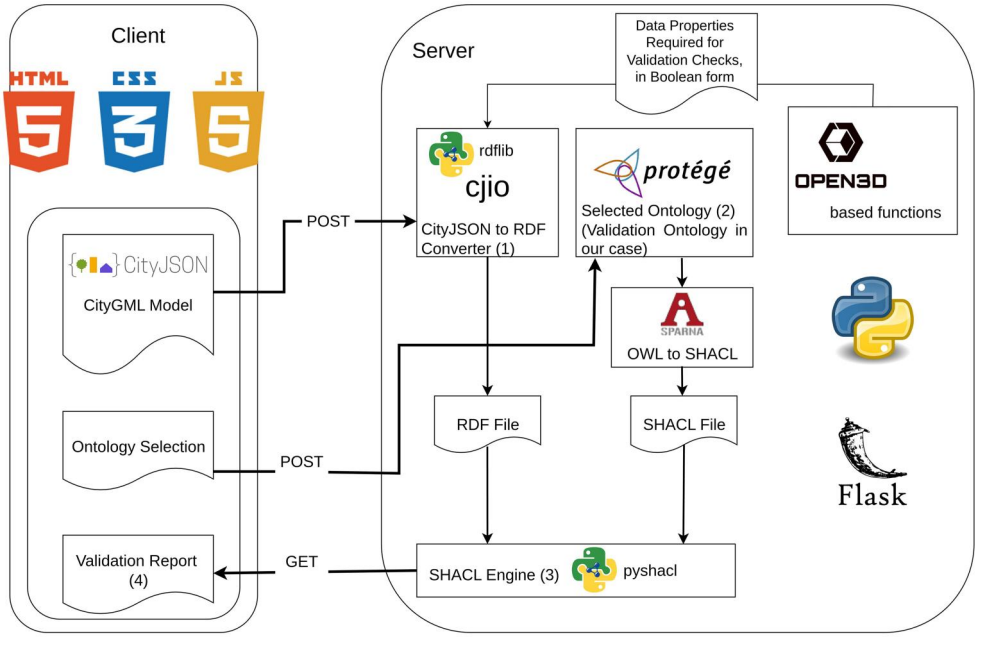

Figure 4. The general pipeline of the study.

2.2. Creation of the validation ontology

At this phase, as a crucial need for the validation, a validation ontology has been created, which includes the validation check definitions in QIE 2016. This ontology plays an essential role in generating both SHACL and RDF graphs. Regarding the definitions in QIE 2016, the validation ontology must also be inherited from GeoSPARQL Ontology (2025), CityGML 2.0 (Chadzynski *et al*. 2021), and Simple Features (Simple Features Ontology 2025) ontologies in addition to RDF, RDFS, and OWL built-in namespaces. It should be noted that the CityGML 2.0 ontology has been chosen because it is the most mature and refined ontology in the literature (Chadzynski *et al*. 2021, CityGML 2.0 OWL 2025, Quek *et al*. 2024). Hence, the ontologies of GeoSPARQL, CityGML 2.0, and Simple Features were used as base T-Boxes.

Before describing the hands-on practice of the ontology preparation, it should be detailed the alignment of parent ontologies and the validation ontology. The validation ontology inherits semantic classes and their hierarchical structure from the CityGML ontology. Besides, as already used in each parent ontology, it inherits the triple linking for data model creation, the basic class and property hierarchies, and the logical constraints (like equivalence, disjointness, or restrictions) elements from RDF, RDFS, and OWL built-in namespaces. The geometry serialization for defining the geometry sequences into a KG as WKT (Well Known Text) is inherited from the GeoSPARQL ontology, and additionally, 2D geometric primitive definitions are inherited from the Simple Features ontology. The rule definitions for validation of CityGML 2.0 LOD2 Buildings in the QIE 2016 require the ontological definitions of 3D spatial relations (Figure 5). While 3D

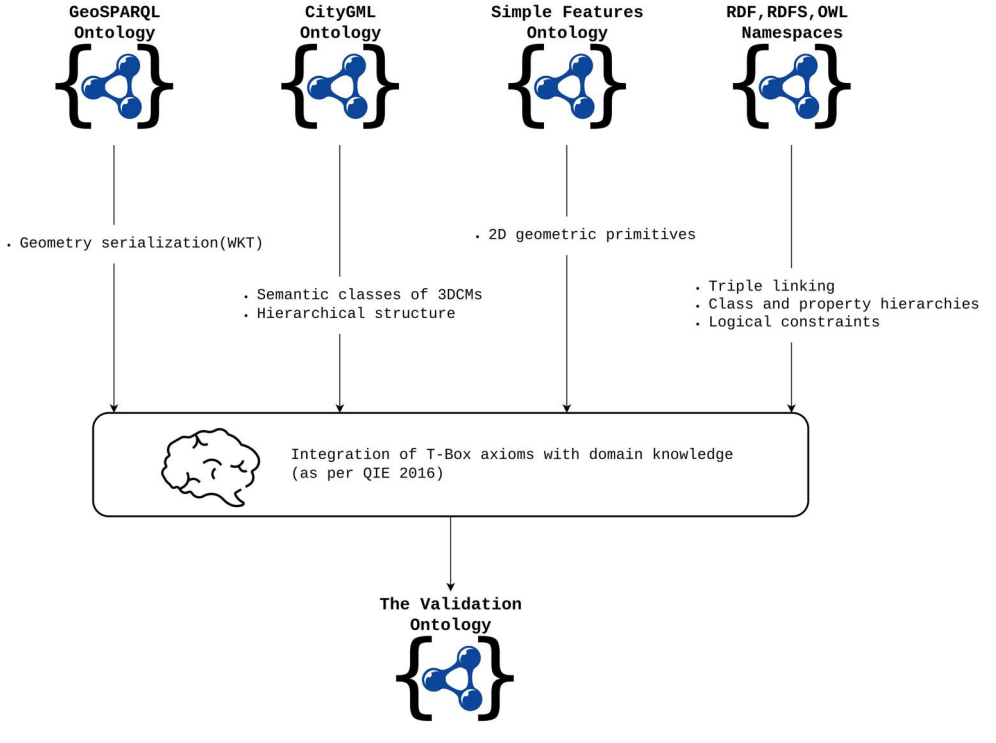

Figure 5. Inheritance aspects of parent ontologies.

primitive definitions can be obtained from CityGML ontology and 2D primitive definitions can be obtained from Simple Features ontology, no ontology includes the definitions of 3D spatial relations yet, such as a 3D extension for GeoSPARQL ontology. To handle this hurdle, the Boolean results of such relations encoded into data graph nodes as data properties will be checked by Boolean restrictions in the validation ontology, as shown in Figure 4. This encoding is detailed in Section 2.3. This strategy stands as a dimensional reduction from 3D into Boolean literals to achieve the same goal without any information loss. After obtaining the required definitions from parent ontologies, the definitions are shaped into a new ontology regarding the rule definitions in the QIE 2016 with domain expertise via the Protege ontology editor (Musen 2015).

At first, the base ontologies mentioned before, which are RDFS, OWL, CityGML 2.0, and Simple Features, were imported into a workspace of the Protege ontology editor. The required data properties of QIE 2016 were included in the workspace by linking the relevant classes of base ontologies. The properties were divided into 2 classes, which are geometric and semantic properties regarding the rule classification in QIE 2016. The geometric properties were also divided into 3 subclasses which are shell properties, ring properties, and polygon properties. All properties have values as Boolean literals, as True or False. Lastly, the additional class, 'validity', states the data's validity status and is created to store validation results after the validation process. After the consequent property classification and property-based class linking processes, the created ontology can be exported in any desired format, such as OWL, Turtle, JSON-LD, N-triples, etc. The JSON-LD (JSON for linking data), which is the chosen format for this study, is gaining popularity in the field thanks to the JSON format's wide usage in web applications. Table 1 in Appendix A presents the properties of the validation ontology together with their linked base ontology classes, while Table 2 provides their definitions with reference to the QIE 2016 study.

The semantic validation requirements include controlling the semantics of 3DCMs, object classes, hierarchies, and attribution, supported by geometric properties, such as surface normals. For instance, the Z-component of a RoofSurface's surface normal should point in the positive direction within a ±85° tolerance. Accurate computation of this property requires geometrically valid surfaces, which are topologically correct,

Table 1. Two samples of ontological restrictions and corresponding SHACL shapes.

Ontologic property restriction definitions	SHACL shapes
:isWatertight rdf:type owl:DatatypeProperty; rdfs:subPropertyOf:shellProperties. … <citygml#SolidType > rdf:type owl:Class; rdfs:subClassOf < citygmll#GeometricPrimitiveType>, [rdf:type owl:Restriction; owl:onProperty:isWatertight; owl:hasValue "true"^^xsd:boolean],	<citygml#SolidType-isWatertight> a sh:PropertyShape; sh:hasValue true; sh:path < valid#isWatertight>.
:tooFewPolygons rdf:type owl:DatatypeProperty; rdfs:subPropertyOf :shellProperties. … <citygml#SolidType > rdf:type owl:Class; rdfs:subClassOf < citygmll#GeometricPrimitiveType>, [rdf:type owl:Restriction; owl:onProperty :tooFewPolygons; owl:hasValue "false"^^xsd:boolean],	<citygml#SolidType-tooFewPolygons> a sh:PropertyShape; sh:hasValue false; sh:path < valid#tooFewPolygons>.

Table 2. Observables of application runtime.

Server configuration	City model	Objects	Size (KBs)	Size (RDF; KBs)	Time (min)
CPU: Intel Core i7-7700HQ 2.80 GHz	Den Haag	2498	16251	22673	10
RAM: 32 GB DDR4	Vienna	1322	5504	24905	13
	Rotterdam	853	5769	9737	5

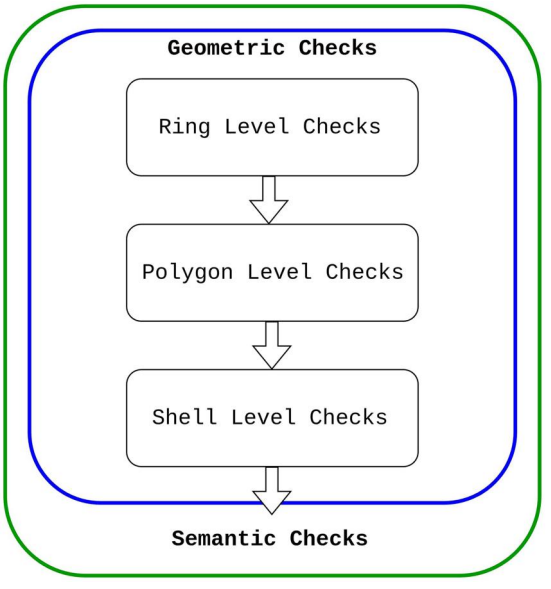

Figure 6. The hierarchy between the checks defined in the validation ontology.

planar polygons that are counter-clockwise oriented and non-intersecting. To ensure counter-clockwise orientation, a polygon must contain at least one closed outer ring consisting of at least three non-collinear points and no self-intersections. Based on this, it becomes evident that there is a hierarchy, or order of operations, in the validation checks, and the semantic validation cannot be performed without ensuring correct object topologies and other geometric properties. Figure 6 illustrates this hierarchy of validation controls.

2.3. CityJSON to RDF conversion

This phase consists of consequent processes of CityJSON parsing, creating triples, and linking triple elements with relevant definitions in the validation ontology. The parsing of the CityJSON file is carried out using the cjio Python library, which is created for CityJSON file management (Cjio's Documentation 2025). The CityJSON file is converted into a DataFrame with this parsing process to ease the triple extraction for the RDF output. The nature of DataFrames is similar to Python dictionaries, which is also similar to JSON files; hence, it becomes easier to create subject-predicate-object combinations from key-value pairs, such as 'key-is a-value'. The Rdfpandas Python library is used to extract triples from DataFrames (RdfPandas 2025). In this extraction, the row indices of the DataFrame become subjects, the column names become predicates, and the

corresponding values become objects. The triple store is an RDF-like property graph without any resource linking in this state (Hartig 2014). While all processes till this state can be automated using the mentioned libraries before linking triple elements and the validation ontology, further information from the CityJSON data must be extracted to satisfy the properties of the validation ontology. This is needed because the cjio library only transfers attributes to the DataFrame, not geometries and thematic classes (Akın and Cömert 2024). The CityJSON-to-RDF converter includes additional operations to overcome this problem. The thematic classes and parent-child relations are extracted with the additional Python functions that traverse the CityJSON data to gather this information.

To extract and include geometries into triples, the open3d Python library is used (Zhou *et al*. 2018). Regarding the data properties, which stand for 3D relational restrictions, in the validation ontology, the required geometric information is extracted and attached to the triple store as Boolean values. The open3d library provides the necessary 3D computational geometry components to create additional functions to calculate geometric Boolean properties from the coordinate sequences (Polygon Z WKTs) obtained from the CityJSON data. After the calculation of all required properties, all triple elements in the store are linked to the relevant definitions in the validation ontology. As a result of the linking process, the triple store of CityJSON data becomes a 3DCM KG regarding the validation ontology and can be exported in any RDF format, such as Turtle, JSON-LD, N-triples, etc. The RDFLib Python library is chosen for the RDF file creation (RDFLib 2025). Figure 7 includes the workflow of the RDF creation process and a part of a created RDF file.

The bottom side of Figure 7 includes a chunk of triple samples from the created RDF files. The red-highlighted triples include attributional, hierarchical, and object class information. These triples are gathered from DataFrames generated by the cjio library, and by CityJSON file traversing for extracting hierarchical, parent-child, and hasGeometry relations. The green highlighted triples include geometric properties that are required for checking validation requirements, such as watertightness, having self-intersections, being vertex manifold, and so on. These geometric properties are calculated using the open3d library-based functions. Following the triple extraction, all generated triples are aggregated into a graph, and then the elements of these triples (subject-predicate-object) are linked into relevant T-Boxes. The 'valid:' prefix in the figure states the validation ontology, and the part after the colon states the validation properties and corresponding calculated values. Similarly, the 'geo:' prefix states the GeoSPARQL ontology, and the 'citygml:' prefix states the CityGML ontology. As the data part in the figure, the object with the 'ex:GUID_DBDABF53-7DD5-4C2F-BE7F-51F29A0CBA16_1' URI is an instance of the BuildingPart class of the CityGML. The object has geometries as six surfaces (hasGeometry). The object's 'geometry type' is Lod2Solid of the CityGML data model. The object's parent is the Building object with another URI. The object's watertightness is calculated as 'True' regarding the validation ontology. Further inspections or exemplifications are possible through the figure or the repository of the project, which is already given in the Conclusions section.

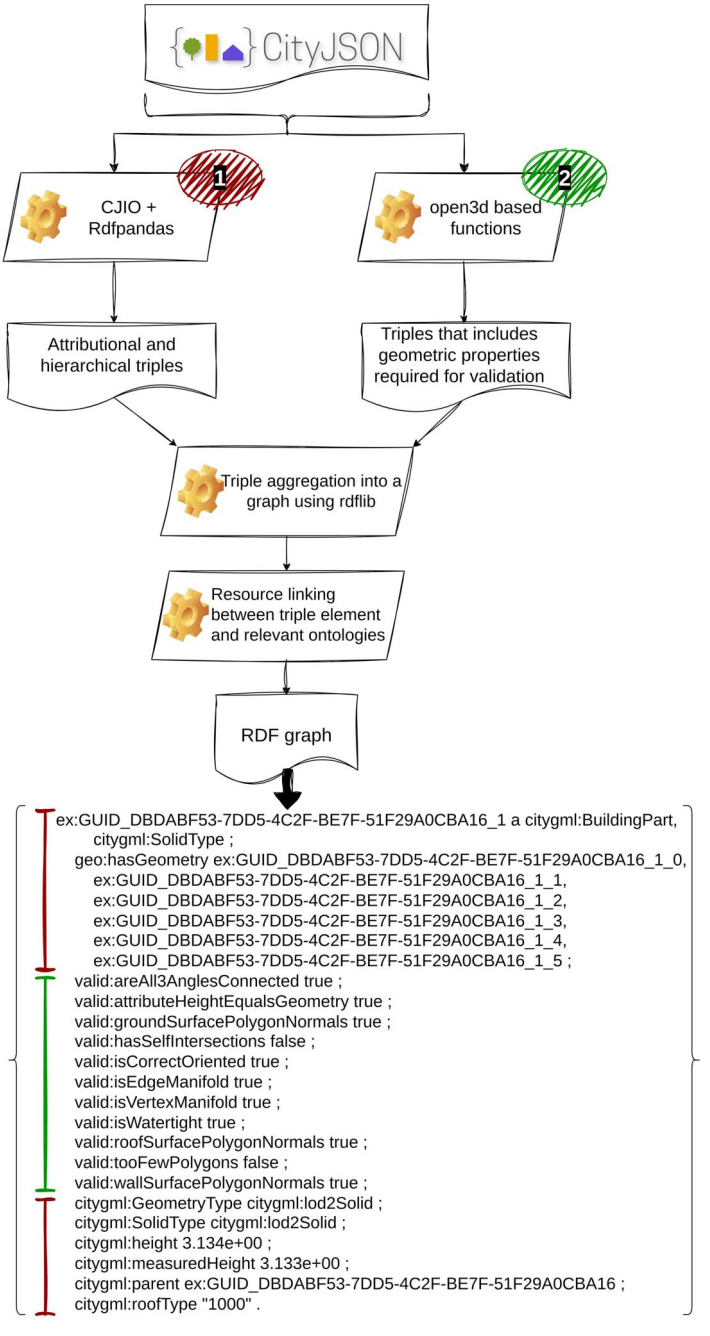

Figure 7. RDF creation workflow and a sample part of RDF data.

2.4. Generating the SHACL graph and carrying out the validation

The SHACL graph generation process is actually a conversion from property restriction axioms in the ontology, which are given in Table 1 for our purpose, to the SHACL shapes. The process can also be summarised as a conversion between statements like

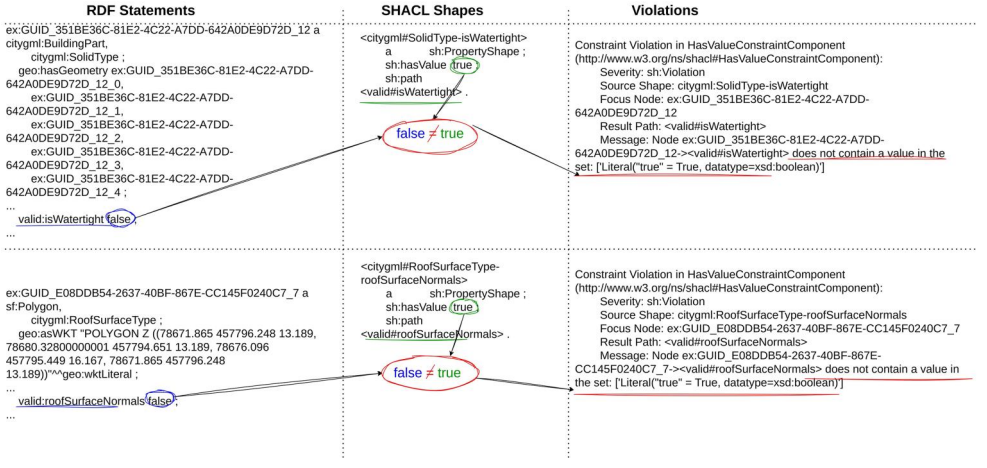

Figure 8. Samples from validation processes carried out using the pySHACL engine.

'Object-has value-X' and 'Object-must have a value-X' to enable the detection of violations in the data graph using the SHACL engine.

The conversion between the ontology and the SHACL graph is certainly accomplished in an automated manner without any manual editing process. The 'owl2shacl' conversion tool by SPARNA (owl2shacl 2025) is used for an automated process. The owl2shacl tool also enables the inference of implicit rules from the validation ontology. This ability allows for the representation of hidden relations between classes and properties in the shape graph, even if they are not stated as independent triples in the validation ontology or parent ontologies. SHACL samples from the owl2shacl are validated using the SHACL shape validator (SHACL shape validator 2025) against the SHACL specification, and they passed the validation.

Table 1 shows two samples from the restriction ontology axioms and the corresponding SHACL shapes. The SHACL graph, which includes such shapes, is used for the validation process. The validation process is carried out using the pySHACL library, which handles SHACL graph manipulations and validation processes (pySHACL 2025). The final validation report includes such violations by individuals of every class which has property restrictions in the validation ontology. In this way, invalid objects of RDF data can be inspected by the client thanks to detailed information for each individual. Figure 8 shows a few samples from validation processes between RDF statements and the relevant SHACL shapes.

2.5. Usage of the pipeline via the web interface

The web interface of the application is developed using Python's Flask library (Flask 2025) to manage web services, alongside core web technologies, such as HTML (HyperText Markup Language) and JavaScript (Figure 9). The pipeline starts by uploading the data online. Users begin by selecting a CityJSON file from their local file system and uploading it to the interface. Once uploaded, the user selects an ontology to determine the validation criteria. At present, the only available ontology is the validation ontology described earlier. After these initial steps, the data and selected

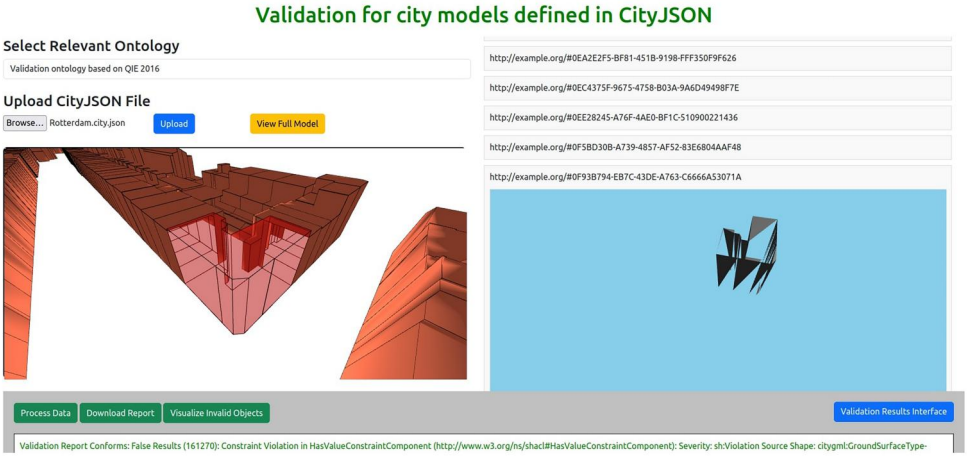

Figure 9. The user interface of the application. Users upload a CityJSON file, select the relevant ontology, and trigger the validation through the Process Data button. The full 3D city model can be inspected using the View Full Model option, while invalid objects are listed on the right with their URIs. Clicking on an object highlights it in red in the 3D view on the left, enabling the investigation of geometric errors. The results panel provides access to the validation report and its interface, that explained in Section 2.6.

ontology are transmitted to the server by clicking the 'Process Data' button. On the server side, a SHACL graph for the validation ontology and an RDF graph of the CityJSON data are generated. The validation process is then executed, and the resulting validation report is stored with a URL in the 'validity' field of the RDF data. This report is also sent back to the client, enabling them to download and review it.

The web interface offers tools to visually examine both invalid city objects and the complete city model. Once CityJSON data is uploaded, users can view the entire city model by selecting the 'View Full Model' button located on the left side of the interface. After the validation process, a list of invalid Building and BuildingPart object URIs will appear on the right panel. Users can investigate the missing or damaged triangles causing the geometric issues by clicking on the respective object URIs. Simultaneously, clicking an object URI highlights that object with a red color within the full model view. The 3D rendering of these objects is powered by the Three.js JavaScript library (Three.js 2025). The presentation of the validation report in a user-friendly form is accessible through the 'Validation Results Interface' button, it is explained in Section 2.6.

2.6. Interpretation of the validation report content

The generated validation report includes the conformance result, source shapes, focus nodes, and violation messages. Source shapes are shapes, constraint sets, and include check IDs about schema, geometric, and semantic controls. The conformance result is a text field that is filled with a Boolean value stating whether the data passed the checks or not. If the data fails against only one of the checks, the conformance result becomes False. Focus nodes are affected nodes of the data graph by each source shape, which are actually individuals of the city model. Violation messages include the

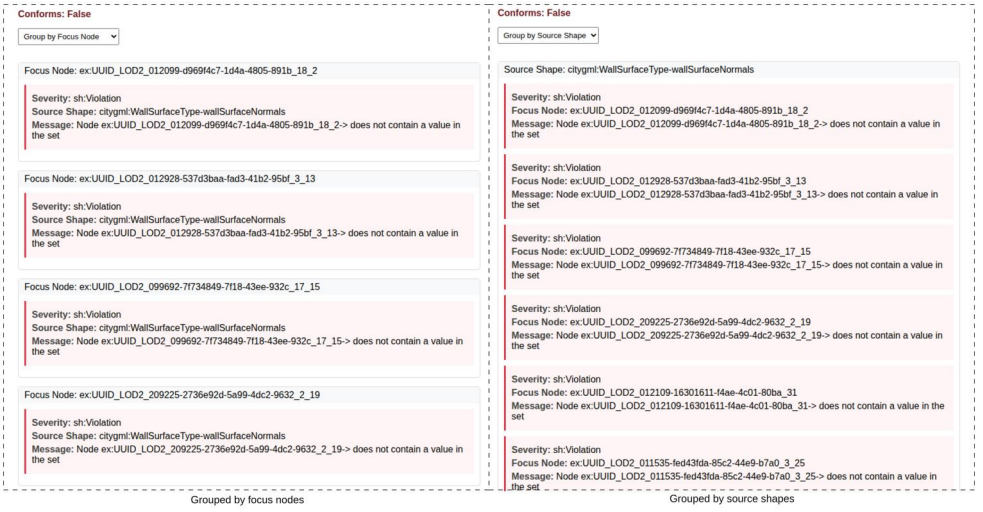

Figure 10. Different presentations of validation reports. The results are grouped by focus nodes on the left and by source shapes on the right.

reason for the check failure for each source shape, attached to focus nodes. These messages are defined in the shape graph.

Even if the violation messages are in natural language, the main form of the validation report is not proper for human understanding. To enhance this, the report is served to the user with two options, which are the grouping by source shapes and the grouping by focus nodes. This grouped presentation and enhanced styling, rather than a raw text document display, ease the user's interpretation of the content (Figure 10). Besides that, the geometric violations, geometry-related source shapes, are also displayed on the 3D model viewer by highlighting. All these functionalities make the application more user-friendly than other related work.

3. Results and discussion

The described pipeline indicates that an interoperable validation process is possible for both geometrical and semantical aspects by virtue of using the KG approach. The data is uploaded online with the validation ontology and the report after the upload and validation processes. By the automated real-time creation of the shape graph for the data, the data brings its own validation rules to the client by preventing the client from planning a rule set. By storing the validation report with a time stamp in a node of the data graph, possible new clients of the same data or data providers are also informed about the validity of the data. The application can be downloaded and compiled openly from (3DCM Quality 2025). Table 2 shows the server configuration, application runtimes, object counts, and file sizes of untextured LOD2 CityJSON test models. These models are published openly from (Datasets 2025). Table 3 includes percentages of affected Building or BuildingPart objects for the validation violations, or invalidities, detected by the developed solution, which the test models have.

Table 3. Detected invalidities for each test model (T: total instances; V: violations; P: percentage).

	Invalidity percantages per test model (our solution)								
	Den Haag			Vienna			Rotterdam		
Invalidity type	T	V	P	T	V	P	T	V	P
Solid level geometric invalidities	1990	3	0.15	2204	456	20.69	853	807	94.61
Semantic surface invalidities	16,209	44	0.27	43,260	203	0.47	15,482	1051	6.79
	Invalidity percantages per test model (val3dity)								
Solid level geometric invalidities	1990	185	9.29	2204	73	3.31	853	207	24.26
Semantic surface invalidities	–	–	–	–	–	–	–	–	–

Considering the file sizes and the runtimes in Table 2, the product application can be plugged into real-world data validation routines of concerned institutions with more powerful server configurations and user authentication. Regarding the runtime limits of popular cloud computing services, runtime results for the models are acceptable without implementing any acceleration technique if cloud systems are preferred instead of physical servers (Akin *et al*. 2024). However, real-world scenarios often involve larger, more complex models with higher object counts, where practical limits are dictated by server specifications, such as memory, CPU, and request management strategies; and client-side constraints like web browser memory and bandwidth. To address these challenges, techniques like tiling can efficiently deliver large datasets to clients, while clustering and hardware-based acceleration on servers can enhance processing capabilities for models with high object count. By creatively balancing server capacity, client capabilities, and dataset size, potentially through innovative tiling strategies or optimized acceleration methods, the application can support real-world validation scenarios, ensuring scalability and performance across diverse configurations.

Before the examination of the validation results in Table 3, it should be considered that the SHACL results are not totally human-friendly for reading, especially for people who are not closely familiar with such a mechanism. Hence, the results are converted into a more readable form in the table for better understanding. The granularity level was a crucial decision for a better understanding. For urban applications, usually we are concerned about solid level validity, or city object level, because the ring and polygon level invalidities are usually the atomic reasons for solid level invalidities. For example, the most concerned validity aspect will likely be the watertightness for a flood simulation, and connection problems between triangles or false holes that came from incorrectly oriented rings are possible reasons for the watertightness. Based on this, we summarized the percentages of geometric invalidities in solid level granularity rather than thousands of polygon level invalidities. Besides, as a novel aspect of our study, we also share the semantic invalidities, or semantically wrong defined surfaces, in polygonal surfaces, but at the polygon level. The interface of the solution provides validation results in both the SHACL engine-generated form and also a more human-readable form, with individual or error-grouped form.

Table 3 includes the semantic invalidities that cannot be identified by the currently in-use validation software. The most notable aspect of this functionality is that both semantic and geometric invalidities can be revealed and stored in connection with associated individuals online. Beyond the table, the most frequently encountered semantic invalidity across all models is the misdefinition of ground and roof surfaces.

These errors are detected by examining the orientations of surface normals and comparing them to directional restrictions in the validation ontology by the SHACL engine. The SHACL engine compares the Boolean properties that are calculated and embedded into the RDF graph by open3d-based functions of the CityJSON-RDF converter to the Booleans in the ontology restrictions. As is widely used in such studies, there are snap tolerance and normal deviation threshold values that are used by the functions. In this study, the snap tolerance value is defaultly taken as 0.01, and the normal deviation threshold is ±5° for GroundSurface, WallSurface, OuterCeilingSurface, and ±85° for OuterFloorSurface and RoofSurface as default. While there are no imposed tolerance values in QIE 2016, the chosen ones in this study are influenced by other existing studies, like the val3dity and (Biljecki *et al.* 2016). Hence, the calculated Boolean properties depend on these default thresholds. The geometric invalidity ratio can differ across validation software due to these threshold variations, especially in planarity controls, which include normal deviation control, as can be seen in Table 3. To accommodate this flexibility, thresholds can be dynamically selected based on the inherent nature of the input data, such as the model's reference system, coordinate precision, or the desired precision for specific use cases. Users can adjust these thresholds through the interface, or they can be explicitly imposed via ontology extensions if the application is adapted for different scenarios.

The developed solution stands out from its peers in terms of comprehensiveness and interoperability. As mentioned in Section 2, the most notable studies in the literature lack a standard to reach consensus, provide both geometrical and semantical validation, and provide reusable validation reports. Table 4 includes a detailed comparison of these studies and our solution. The geometric checks run by the val3dity depend on the geometric primitive definitions in ISO 19107. But this study does not include a result storage strategy to share and reuse the validation results alongside the input data. At this point, the CityDoctor enables this reusability through QualityADE. The CityDoctor's geometric validation implements the geometric part of QIE 2016. However, CityDoctor only accepts CityGML as the input format, which may pose a limitation for the software's long-term usability, particularly given the developer-friendly characteristics of CityJSON and its widespread adoption in actual practice. Following the chronology, we encounter the study of Chadzynski *et al.* (2023). This study performs some sort of semantic checks, but even the validity of these semantic checks is debatable. Because any semantic checks require geometric validity, the properties used for semantic checks are calculated from valid geometries. By the way, the whole check process of this study does not depend on any standards. The study of Pauwels *et al.* (2024) includes geometric and semantic checks for IFC models. Besides, the study uses SW technologies to better interpret the used rules

Table 4. Comparison of the proposal and the existing methods (GV: geometric validation; SV: semantic validation).

Study	Standardization	GV	SV	Reusability
val3dity	✓	✓	✗	✗
CityDoctor	✓	✓	✗	✓
Chadzynski *et al.* (2023)	✗	✗	–	✗
Pauwels *et al.* (2024)	✗	✓	✓	✗
Ours	✓	✓	✓	✓

and to benefit from the flexibility of multiple ontologies, as in our study. But, as underlined in the limitations of the study, there is a need for a standard for concise interoperability. Our study addresses the interoperability issue by using a well-defined rule definition set like QIE 2016, and enables both dynamic rules checking and reusability by the KG approach. The KG approach enables the dissemination of the data with both the related validation plan and the validation results attached as sub-nodes. Furthermore, the approach also allows inference of implicit relations for a more inclusive validation.

The most obvious limitation of this study lies in its confinement to the CityGML Building schema and LOD2 models. Both the validation ontology and the RDF converter were specifically designed for LOD2 buildings. To expand the application's scope to encompass other application schemas within the CityGML data model and higher LODs, it would be necessary to extend the current validation ontology and enhance the RDF converter to align with this updated framework. While this constitutes a limitation, it should be noted that the application addresses gaps in the existing literature and aligns with the predominant focus of most studies on LOD2 buildings. Accordingly, this study can be regarded as providing a proof-of-concept for future research and further advancements in the field. If the application or general framework is considered for an extension, upper LODs and any other schemas from the CityGML base profile can be included in the validation ontology as additional nodes. In such a scenario, LOD2 Buildings would be a subgraph of a more extensive ontology. In addition to the schematic limit, the study addresses the data quality dimensions in accordance with ISO 19157, namely completeness, thematic accuracy, and logical consistency. Temporal accuracy and positional accuracy, which require independent validation against auxiliary data sources, are beyond the scope of this work. However, their consideration remains important for a full assessment of 3D city models.

As mentioned in the explanation of the validation ontology creation, there is no ontology that provides 3D spatial relations. We applied a dimensional reduction trick to eliminate this hurdle. In this study, we achieved our goals with this reduction and the other well-defined parts of the parent ontologies. However, even if the current strategy has worked for the LOD2 3DCM validation, the lack of 3D spatial relation definitions can be disadvantageous for any possible use cases, ontology extension attempts. While a possible extension of GeoSPARQL to 3D is being discussed at developer forums, such an extension can enable the construction of SPARQL queries needed for complex or 3D relations. It is not directly noted as a future work plan, but experiments or possible use case applications for such an extension can be tried further.

Beyond the explicit validation rules defined in the ontology, the inference ability of implicit rules plays a beneficial role in enhancing the validation capability. This ability mainly shows itself during the shape graph generation. The following items show several samples, which are interpreted in natural language to enable human understanding of the inference ability. The owl2shacl conversion tool generates these inferred rules to satisfy the internal logical consistency in the shape graph.

- Every instance of 'city: ReliefFeatureType' must have at least one 'city:reliefComponent'.
- Every instance of 'city: TextureType' must have at least one 'city:wrapMode'.

- Every instance of 'city: AbstractCityObjectType' must have at least one 'city: Appearance'.
- Every instance of 'city: CityModelType' must have at least one 'city:appearanceMember'.
- Every instance of 'city: TexturedSurfaceType' must have at least one 'city: Appearance'.

In the scenario where the above rules are not inferred, the check of these properties would not be possible. Even if the defined restrictions in the validation ontology are sufficient for the LOD2 Buildings, any other possible use case application that uses these properties would give faulty results due to the lack of checking. Therefore, the inference ability extends the inclusiveness of the validation process. Although the inference capability and the inclusiveness of the owl2shacl are only tested on an LOD2 Building schema-dependent ontology, the performance of further extension to another use case should also be tested before a fully-trusted application on owl2shacl's inference engine. Because owl2shacl is an automated tool, and the quality of its results is directly dependent on the logical consistency of the ontology inputs. Hence, changing this tool with another alternative or developing a native one in response to the needs can be considered if it is needed.

To sum it up, the following targeted enhancements can be implemented to ensure a wider scalability and practical relevance of the proposed approach. Generalizing the validation ontology involves incorporating additional CityGML application schemas as sibling graph nodes, such as transportation, vegetation, or water bodies, by defining new classes and properties. This would enable validation across diverse urban datasets. Additionally, creating new SHACL shapes to support higher LODs and inter-object invalidities, such as detecting spatial overlaps between adjacent buildings or infrastructure, can be achieved using automated tools like owl2shacl to translate ontology constraints into enforceable shapes. Modifying the CityJSON-to-RDF converter requires updating the parsing logic within the cjio and open3d libraries to extract complex geometries and thematic attributes from additional CityGML schemas, ensuring robust RDF graph generation. In fact, if an ontology includes 3D spatial relations for substitution with GeoSPARQL will enable, the native dependency of open3d can be bypassed using the relation definition in a 3D-augmented GeoSPARQL ontology. These steps, grounded in the framework's existing interoperability, enable its adaptation to broader urban applications, enhancing its utility.

4. Conclusions

This study develops a KG approach with SHACL-based validation for 3DCMs, providing an interoperable web tool that supports standardized, interoperable, and reusable procedures and outcomes. The work contributes methodologically, technically, and conceptually to advance 3DCM interoperability with potential applications in other domains.

Methodologically, the study establishes a framework for constructing 3DCM KGs that embed their own validation ontologies, ensuring data is self-descriptive and

transparent. By leveraging SHACL, the approach allows KGs to carry explicit validation constraints, enabling data-sharing parties to understand and verify the standards applied.

Technically, the development of an open-source web tool, supported by a CityJSON-to-RDF converter and a QIE 2016-based validation ontology, enables real-time validation and visualization of 3DCMs. The converter transforms CityJSON data into RDF graphs, while the web interface, built with Python's Flask and Three.js, provides user-friendly access to validation reports and 3D model inspections. All developments, including the open ontology, RDF converter, the SHACL graph, and the application with its source code, are shared as open-source resources in terms of reproducibility and sustainable goals.

Conceptually, the study establishes a constructed template for connecting 3DCMs to broader web-based data ecosystems. While the tool's design allows adaptation to other domains through tailored ontologies and converters, it supports integrated data systems that improve decision-making across various domains. In the future, where all structured or unstructured data will come together in a unified information source, just as a Guide to the Galaxy, making the multi-domain worldwide data interlinked and alive on the web with such a mechanism will play a crucial role in the decision-making and solution-producing processes for experts in all fields. Considering the extensive use of geospatial data over all other types of data on the web, it is important to carry out and disseminate such studies in the GIS field to accomplish the unification path stated above.

This method and tool are designed for a wide range of users. GIS professionals, urban planners, and geospatial data providers can use it to ensure they have high-quality 3D city models (3DCMs) for tasks like urban planning, disaster management, and infrastructure analysis. CityGML users, especially those who are working with LOD2 models, can benefit from checking both the geometry and semantics of their data according to the QIE 2016 validation checks. Beyond that, researchers and developers in the AEC industry who work with the semantic web can take advantage of the framework's flexibility to validate BIM or other geospatial datasets. Municipalities and organizations responsible for data standardization can also use this approach to ease data-sharing processes and improve interoperability across different urban data platforms.

This work also shares common ground with BIM data quality control efforts, which focus on ensuring reliable and interoperable digital building permit procedures. BIM standards, such as IFC and COBie, for example, enable data consistency across design, construction, and operation phases. Similar to BIM's rule-based checks, this study uses SHACL-based validation to enforce standards and maintain both geometric and semantic integrity in 3DCMs. While BIM efforts usually focus on building-level details, frequently in third-party software ecosystems, this study uses an open-source, KG-based method to address city-scale models, emphasizing data integration and the larger urban environment. Extending the ontology and converter in the future could help bridge GIS and BIM workflows, potentially creating unified urban-building data ecosystems.

Our future plans include the healing of the 3D objects that are detected as invalid using the workflow explained above. The next nearest move is to implement overlap

and intersection tests for neighbour city objects like BuildingParts, Buildings, or other objects that are defined in other thematic classes except the Building schema. Due to the current process of only checking the geometric validity of individual objects separately, the validation report does not include any violations of inter-object geometric invalidities. The other purpose of this plan is to create an end-to-end pipeline that accomplishes both the validation and the healing processes on the same 3D city KG to keep the model up-to-date and valid with minimum user interruption. The literature has various studies and software to handle the healing process on 3D objects, but, regarding our preliminary inspections, it can be seen that there is still no mature and fully automated method. To exemplify such healing, the Buildings or BuildingParts that are detected as non-watertight by the validation process can be reconstructed by keeping the semantic information. Afterwards, the relevant node of the data graph, which includes the non-watertight object, can be updated as 'valid' without a complete from-scratch validation procedure. Considering a web-based solution, total-reconstructor global healing methods will suit better rather than locating and healing at the atomic level in terms of computational complexity and response times. Probably, the biggest challenge will be the maintenance of the surface semantics after such reconstruction. After completing a mature validation and healing pipeline, one of our next steps will also be the visualisation of the invalid and healed geometries together to make the pipeline a comprehensive framework for 3DCM quality management.

Disclosure statement

No potential conflict of interest was reported by the author(s).

Funding

This work was supported by the Scientific and Technological Research Council of Türkiye (TÜBİTAK) with application numbers 1059B142301188 and 1059B192402172.

Notes on contributors

Alper Tunga Akın is a PhD candidate in Department of Geomatics Engineering, Graduate School of Natural and Applied Sciences, Karadeniz Technical University, and he is a guest researcher in 3D Geoinformation research group, Delft University of Technology. His research interests focus on 3D city models, geospatial knowledge graphs, data validation processes. He contributed to conceptualization, methodology, software, investigation, data curation, validation, visualization, writing–original draft and writing–review and editing, and funding acquisition.

Ziya Usta is an assistant professor in Department of Geomatics Engineering, Artvin Çoruh University, and he is a guest researcher in 3D Geoinformation research group, Delft University of Technology. His research interests focus on 3D city models, 3D visualization techniques, web management of 3D geospatial data. He contributed to conceptualization, methodology, investigation, validation, visualization, writing–original draft and writing–review and editing, and funding acquisition.

Jantien Stoter is Department Head Urbanism and full professor in 3D Geoinformation, Delft University of Technology. She (co) leads/has led many national and international projects on

GeoBIM, 3D city modeling, and urban applications, and has fulfilled several roles in the Open Geospatial Consortium and the European Spatial Data Research Association. She contributed to conceptualization, methodology, supervision, validation, and writing–review and editing.

Ken Arroyo Ohori is a lecturer and researcher at the 3D geoinformation group of the Delft University of Technology in the Netherlands. His research covers various aspects of geographic information, with a focus on the use of novel techniques to process data geometrically and topologically. He contributed to conceptualization, methodology, supervision, validation, and writing–review and editing.

Çetin Cömert is Section Head of Cartography and a full professor in the Department of Geomatics Engineering, Karadeniz Technical University. His research interests include the geographic information systems, 3D city models, data infrastructures, and semantic web technologies. He contributed to conceptualization, methodology, supervision, investigation, validation, writing–original draft, and writing–review and editing.

ORCID

Alper Tunga Akın http://orcid.org/0000-0002-4535-9143
Ziya Usta http://orcid.org/0000-0003-2232-2011
Jantien Stoter http://orcid.org/0000-0002-1393-7279
Ken Arroyo Ohori http://orcid.org/0000-0002-9863-0152
Çetin Cömert http://orcid.org/0000-0002-2019-6990

Data and codes availability statement

The repository of the study is shared at https://doi.org/10.17605/OSF.IO/KNWB5.

References

3DCityKG, 2025. Available from: https://github.com/tum-gis/3dcitykg [Accessed 19 August 2025].
3DCM Quality, 2025. Available from: https://github.com/alpertungakin/3DCMQuality/tree/main/Valid_app_gui [Accessed 19 August 2025].
Akın, A.T., and Cömert, Ç., 2024. "Cityjson2rdf": A converter for producing 3D city knowledge graphs. *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, XLVIII-4/W9-2024, 15–19.
Akin, A.T., Usta, Z., and Cömert, Ç., 2024. Elaborating and performing optimization approaches for web-based 3D spatial analysis of 3D city models. *Journal of Spatial Science*, 69 (4), 1225–1239.
Alam, N., *et al.*, 2013. Towards automatic validation and healing of citygml models for geometric and semantic consistency. *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, II-2/W1, 1–6.
Betz, M. and Coors, V., 2021. An application domain extension for storing validation results of citygml structures. *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, VIII-4/W1-2021, 11–16.
Biljecki, F., *et al.*, 2015. Applications of 3D city models: state of the art review. *ISPRS International Journal of Geo-Information*, 4 (4), 2842–2889.
Biljecki, F., *et al.*, 2016. The most common geometric and semantic errors ın citygml datasets. *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, IV-2/W1, 13–22.
Burns, R. and Thatcher, J., 2015. Guest editorial: what's so big about big data? Finding the spaces and perils of big data. *GeoJournal*, 80 (4), 445–448.

Chadzynski, A., *et al.*, 2021. Semantic 3D city database—an enabler for a dynamic geospatial knowledge graph. *Energy and AI*, 6, 100106.
Chadzynski, A., *et al.*, 2023. Semantic 3D city interfaces—intelligent interactions on dynamic geospatial knowledge graphs. *Data-Centric Engineering*, 4, e20.
Cimmino, A., Fernández-Izquierdo, A., and García-Castro, R., 2020. Astrea: automatic generation of SHACL shapes from ontologies. *Lecture Notes in Computer Science*, 12123, 497–513.
CityDoctor2, 2025. Available from: https://transfer.hft-stuttgart.de/pages/citydoctor/citydoctorhomepage/en/ [Accessed 19 August 2025].
CityGML 2.0 OWL, 2025. Available from: https://cui.unige.ch/isi/onto//citygml2.0.owl [Accessed 19 August 2025].
citygml-tools, 2025. Available from: https://github.com/citygml4j/citygml-tools [Accessed 19 August 2025].
Cjio's Documentation, 2025. Available from: https://cityjson.github.io/cjio/index.html [Accessed 19 August 2025].
Coors, V., Betz, M., and Duminil, E., 2020. A concept of quality management of 3D city models supporting application-specific requirements. *PFG–Journal of Photogrammetry, Remote Sensing and Geoinformation Science*, 88 (1), 3–14.
Cömert, Ç., *et al.*, 2010. Semantic web services for implementing national spatial data infrastructures. *Scientific Research and Essays*, 5 (7), 685–692.
Datasets, 2025. Available from: https://www.cityjson.org/datasets/ [Accessed 19 August 2025].
Ding, L., *et al.*, 2025. Integrating 3D city data through knowledge graphs. *Geo-Spatial Information Science*, 28 (2), 780–799.
Dukai, B., *et al.*, 2021. Quality assessment of a nationwide data set containing automatically reconstructed 3D building models. *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, XLVI-4/W4-2021, 17–24.
Flask, 2025. Available from: https://flask.palletsprojects.com/en/stable/ [Accessed 19 August 2025].
Gabara, G. and Sawicki, P., 2021. Quality evaluation of 3D building models based on low-altitude ımagery and airborne laser scanning point clouds. *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, XLIII-B2-2021, 345–352.
GeoSPARQL Ontology, 2025. Available from: http://www.opengis.net/ont/geosparql [Accessed 19 August 2025].
Gröger, G., *et al.*, 2012. OpenGIS® city geography markup language (CityGML) encoding standard. Version 1.0.0, OGC 08-007r1.
Hagedorn, P. and König, M., 2020. Rule-based semantic validation for standardized linked building models. *In*: *International conference on computing in civil and building engineering*. Cham: Springer International Publishing, 772–787.
Hagedorn, P., Pauwels, P., and König, M., 2023. Semantic rule checking of cross-domain building data in information containers for linked document delivery using the shapes constraint language. *Automation in Construction*, 156, 105106.
Hartig, O., 2014. Reconciliation of RDF* and property graphs. arXiv preprint arXiv:1409.3288.
Hitzler, P., Krotzsch, M., and Rudolph, S., 2009. *Foundations of semantic web technologies*. New York: Chapman and Hall; CRC. https://www.taylorfrancis.com/books/mono/10.1201/9781420090512/foundations-semantic-web-technologies-pascal-hitzler-markus-krotzsch-sebastian-rudolph.
Horrocks, I., *et al.*, 2004. SWRL: a semantic web rule language combining OWL and RuleML. *W3C Member Submission*, 21 (79), 1–31.
International Organization for Standardization. 2013. ISO 19157:2013: geographic information–data quality. Available from: https://www.iso.org/standard/32575.html.
ISO, T., 2003. 211. ISO 19107 geographic information–spatial schema.
Knublauch, H. and Kontokostas, D., 2017. Shapes constraint language (SHACL). *W3C Candidate Recommendation*, 11 (8), 1.
Kontokostas, D., *et al.*, 2014. Test-driven evaluation of linked data quality. *In*: *Proceedings of the 23rd ınternational conference on world wide web*, 747–758.

Kostovska, A., *et al.*, 2021. Option: optimization algorithm benchmarking ontology. *In*: *Proceedings of the genetic and evolutionary computation conference companion*, 239–240.
Krämer, M., Haist, J., and Reitz, T., 2007. Methods for spatial data quality of 3D city models. *In*: *Eurographics Italian chapter conference*, 167–172.
Lam, P.-D., *et al.*, 2024. Digital twin smart city: Integrating IFC and CityGML with semantic graph for advanced 3D city model visualization. *Sensors*, 24 (12), 3761.
Lassila, O., Hendler, J., and Berners-Lee, T., 2001. The semantic web. *Scientific American*, 284 (5), 34–43.
Ledoux, H. and Meijers, M., 2011. Topologically consistent 3D city models obtained by extrusion. *International Journal of Geographical Information Science*, 25 (4), 557–574.
Ledoux, H. and Wagner, D., 2016. OGC® CityGML quality ınteroperability experiment.
Ledoux, H., 2013. On the validation of solids represented with the ınternational standards for geographic ınformation. *Computer-Aided Civil and Infrastructure Engineering*, 28 (9), 693–706.
Ledoux, H., 2018. val3dity: validation of 3D GIS primitives according to the international standards. *Open Geospatial Data, Software and Standards*, 3, 1.
Ledoux, H., *et al.*, 2019. CityJSON: a compact and easy-to-use encoding of the CityGML data model. *Open Geospatial Data, Software and Standards*, 4 (1), 1–12.
Musen, M.A., 2015. The protégé project: a look back and a look forward. *AI Matters*, 1 (4), 4–12.
Nardi, D. and Brachman, R.J., 2003. An introduction to description logics. *Description Logic Handbook*, 1, 40.
Nguyen, S.H., 2024. *Automatic detection and ınterpretation of changes in massive semantic 3D city models*. Doctoral dissertation. Technische Universität München.
owl2shacl, 2025. Available from: https://github.com/sparna-git/owl2shacl [Accessed 19 August 2025].
Pareti, P. and Konstantinidis, G., 2021. A review of SHACL: from data validation to schema reasoning for RDF graphs. *Reasoning Web International Summer School*, *Lecture Notes in Computer Science (LNISA)*, 13100, 115–144.
Pauwels, P., van den Bersselaar, E., and Verhelst, L., 2024. Validation of technical requirements for a BIM model using semantic web technologies. *Advanced Engineering Informatics*, 60, 102426.
Pauwels, P., Zhang, S., and Lee, Y.C., 2017. Semantic web technologies in AEC industry: a literature overview. *Automation in Construction*, 73, 145–165.
Prud'Hommeaux, E., Labra Gayo, J.E., and Solbrig, H., 2014. Shape expressions: an RDF validation and transformation language. *In: Proceedings of the 10th international conference on semantic systems*, September, Leipzig, Germany. New York, NY: Association for Computing Machinery, 32–40.
pySHACL, 2025. Available from: https://github.com/RDFLib/pySHACL [Accessed 19 August 2025].
Quek, H.Y., *et al.*, 2024. Dynamic knowledge graph applications for augmented built environments through "The World Avatar". *Journal of Building Engineering*, 91, 109507.
RdfLib, 2025. Available from: https://rdflib.readthedocs.io/en/stable/ [Accessed 19 August 2025].
RdfPandas, 2025. Available from: https://rdfpandas.readthedocs.io/en/latest/ [Accessed 19 August 2025].
SHACL shape validator, 2025. Available from: https://www.itb.ec.europa.eu/shacl/shacl/upload [Accessed 19 August 2025].
Simple Features Ontology, 2025. Available from: http://www.opengis.net/ont/sf [Accessed 19 August 2025].
Singla, J.G. and Trivedi, S., 2022. 3D building reconstruction and validation using high-resolution stereo data. *Current Science*, 122 (8), 900.
Three.js, 2025. Available from: https://threejs.org/ [Accessed 19 August 2025].
Van den Brink, L., Janssen, P., and Quak, W., 2013. *From geo-data to linked data: automated transformation from GML to RDF*. Amersfoort: Linked Open Data-Pilot Linked Open Data Nederland. https://research.tudelft.nl/en/publications/d8-from-geo-data-to-linked-data-automated-transformation-from-gml/.
Vinasco-Alvarez, D., *et al.*, 2020. *From citygml to owl*. Doctoral dissertation. LIRIS UMR 5205.

Vinasco-Alvarez, D., *et al.*, 2021. Towards limiting semantic data loss in 4D urban data semantic graph generation. *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, VIII-4/W2-2021, 37–44.

VoPham, T., *et al.*, 2018. Emerging trends in geospatial artificial intelligence (geoAI): potential applications for environmental epidemiology. *Environmental Health*, 17 (1), 40.

Wagner, D., *et al.*, 2013. Geometric-semantical consistency validation of CityGML models. *In*: *Lecture notes in geoinformation and cartography*. Berlin, Heidelberg: Springer, 171–192. https://link.springer.com/chapter/10.1007/978-3-642-29793-9_10#copyright-information

Werbrouck, J., *et al.*, 2019. A checking approach for distributed building data. *In*: *31st forum bauinformatik*. Berlin: Universitätsverlag der TU Berlin, 173–181.

What is SHACL?, 2025. Available from: https://www.ontotext.com/knowledgehub/fundamentals/what-is-shacl/ [Accessed 19 August 2025].

World Wide Web Consortium, 2014. RDF 1.1 Primer.

Yao, Z., *et al.*, 2018. 3DCityDB-a 3D geodatabase solution for the management, analysis, and visualization of semantic 3D city models based on CityGML. *Open Geospatial Data, Software and Standards*, 3 (1), 1–26.

Zeshan, F., *et al.*, 2023. An IoT-enabled ontology-based ıntelligent healthcare framework for remote patient monitoring. *IEEE Access*, 11, 133947–133966.

Zhou, Q.Y., Park, J., and Koltun, V., 2018. Open3D: a modern library for 3D data processing. arXiv preprint arXiv:1801.09847.

Appendix A

Ontological Content, https://osf.io/952sy/overview?view_only=347ecf585ae84034b54ec5679a966483