Comparative Analysis of Approaches for Geometric Data Representation in RDF

Diellza Elshani[1,*], Ali Nahkaee[1], Anthony A. Arrascue[2], Haris Isakovic[2], Navid Hedayati[2], Janakiram Karlapudi[2] and Thomas Wortmann[1]

[1]Institute for Computational Design and Construction, Department for Computing in Architecture (ICD/CA), Faculty of Architecture and Urban Planning, University of Stuttgart, Keplerstraße 11, 70174, Stuttgart, Germany
[2]GROPYUS Technologies GmbH, Hauptstraße 27, 10827 Berlin, Germany

Abstract

This paper presents a comparative analysis of three approaches for representing and processing geometric data within RDF frameworks: (1) RDF-based Geometry Modeling, (2) Extending Triplestore and Querying Capabilities, and (3) Integrating RDF-compliant Systems with Native Geometry Stores or Computer-Graphics Libraries. Each approach is evaluated based on expressiveness, complexity, geometric support, processing capabilities, validation, scalability, interoperability, and adherence to standards. The study highlights the strengths and limitations of these methods, particularly in the context of Building Information Modeling (BIM). The findings provide insights into the trade-offs between RDF-native and hybrid solutions, offering guidance on their applicability for handling complex geometric data in BIM workflows.

Keywords

RDF, GeoSPARQL, geometric validation, spatial data, interoperability

1. Introduction

The Architecture, Engineering, and Construction (AEC) industry relies heavily on Building Information Modeling (BIM) to digitally represent buildings and infrastructure. In this model, geometric data plays a central role, supporting accurate positioning, integration, and alignment of building components [1]. Beyond static representations, geometric data must be queryable and modifiable to facilitate real-time decision-making, automated rule-checking, carrying out various analysis and simulations, and multi-disciplinary collaboration [2]. Geometric data in AEC projects is represented using various schema and file formats, such as the Industry Foundation Classes (IFC) schema, the STEP-based IFC file format, and the Drawing Exchange Format, each associated with distinct geometric kernels. [3]. These variations often lead to inconsistencies and data loss when transitioning between different software tools.

The decentralized structure of the AEC industry, encompassing diverse disciplines such as architecture and engineering, is subject to challenges such as fragmented collaboration, difficulties in data sharing, and the need for effective coordination across multi-disciplinary domains [4]. The adoption of Semantic Web standards and in particular the Resource Description Framework (RDF) for geometric representation, or to link data to their geometry representation, has demonstrated to have the potential to alleviate these challenges, improving interoperability, reducing data inconsistencies, and streamlining collaboration [5, 6, 7]. However, integrating geometric reasoning into co-design workflows remains difficult due to the inherent complexity of AEC data and the expressiveness limitations of Semantic Web technologies [8, 9]. Existing challenges include the lack of a uniform approach for describing and integrating geometric components in RDF, and the complexity of geometry descriptions, which often results in excessive verbosity [1]. Moreover, RDF lacks native support for advanced geometric processing and support for 3D geometries [1], further emphasizing the need for more sophisticated methods to handle BIM geometry

LDAC 2025: 13th Linked Data in Architecture and Construction Workshop, July 09–11, 2025, Porto, Portugal
*Corresponding author.
diellza.elshani@icd.uni-stuttgart.de (D. Elshani); ali.nahkaee@icd.uni-stuttgart.de (A. Nahkaee); anthony.arrascue@gropyus.com (A. A. Arrascue); haris.isakovic@gropyus.com (H. Isakovic); navid.hedayati@gropyus.com (N. Hedayati); janakiram.karlapudi@gropyus.com (J. Karlapudi); thomas.wortmann@icd.uni-stuttgart.de (T. Wortmann)
0000-0003-2902-341X (D. Elshani); x (A. A. Arrascue); 0 (H. Isakovic); x (N. Hedayati); x (J. Karlapudi)

efficiently [6, 1]. Processing and analyzing geometric data in BIM thus remains a major challenge. Many existing systems, particularly non-proprietary BIM platforms, lack tools for efficient geometry validation, spatial reasoning, and advanced computations such as spatial relationships [4, 6]. These limitations hinder critical tasks such as model accuracy verification, clash detection, and performance simulations, which are essential for improving design accuracy and reducing costly errors in construction projects.

Thus, the lack of geometric processing capabilities in standard semantic technologies is often identified as a major limitation [3]. This analysis addresses challenges of geometry representation and evaluates the effectiveness of various technologies in integrating geometric processing capabilities into the Semantic Web stack. In particular, hybrid architectures that integrate additional components to support the storage, validation, and processing of geometric data have also been explored [6]. These hybrid approaches aim to provide a more robust and practical foundation for non-proprietary BIM-based workflows, ultimately bridging the gap between Semantic Web technologies and advanced geometry processing [5, 2, 7].

In Section 2 we describe an industrial case study that illustrates the aforementioned limitations and provides the motivation for this research. The methodology we used is described in Section 4. Section 5 describes results of the comparative analysis. Finally, in Section 6, we elaborate on the results and implications for the case study. We conclude the paper with a short outlook and outline of future work.

2. Case Study: Serial production of sustainable residential buildings

The construction industry and building maintenance count as a major source of greenhouse gas (GHG) emissions, around 37% [10]. At the same time, Germany has faced a 99% increase in building costs in the last decade, while the average construction time increased by 95% [11]. A global housing deficit is steadily increasing; it reached 400.000 units in Germany. We argue that serial production of buildings can bee seen as potential solution for these challenges.

Semantic knowledge representation as a data management methodology offers the necessary data integration, interoperability, and overall data linking capabilities to implement serial production of buildings. Buildings are represented using models that leverage ontologies as a universal modeling language. By utilizing RDF's flexible data structure, teams across different disciplines can progressively enrich the building model with specialized domain knowledge. Moreover, a systematic geometric data representation is essential aspect for interoperability of any BIM [12]. It varies from a stage of development and complexity of modeled object. For example, 2D polygons are used to define spatial footprints and other planar representations, while 3D geometric models are crucial for description of wall elements, manufacturing plans and further individual robotic tasks. In some cases geo-coordinates define a site's absolute position, other elements rely on reference-based coordinate systems. Furthermore, some sub-models in the hierarchy require only Boundary Representation (BRep), which defines objects based on their bounding surfaces and control points, and other Constructive Solid Geometry (CSG)[13], which constructs objects using primitive shapes combined through Boolean operations. Thus, the geometry representation method depends on the stage of development a building. Planning requires different geometrical complexity than parametric design of complex wall elements, or robot motion planning. From this perspective we have identified key functionalities that are essential for improving geometric data management, validation, and processing:

R1: Support for Multiple Geometry Representations
The system must accommodate various geometry representations, e.g. BRep and CSG, as well as coordinate systems, e.g. local project coordinates, and GIS data. Additionally, a hierarchical geometry structure is required to accurately model elements relative to a reference coordinate grid. These capabilities ensure that the model remains flexible and interoperable across different teams and use-cases. Given the premise of our study, which emphasizes the needs of serial production of residential buildings, we have deliberately excluded complex geometries such as curved surfaces, multi-curves, tessellated geometries, etc. Instead, our focus has primarily been on simple geometries, such as geometric

primitives, simple solids, and CSGs, since they sufficiently express the geometric data required for industrialized production of timber-elements and the overall building system they belong to.

R2: Reliable Validation of Geometric Data
As different teams contribute to the building model, it is critical to ensure that all geometric data is consistently structured and follows predefined conventions. While individual teams may validate their own datasets, a global validation mechanism is needed to enforce uniform standards throughout the model. For example, a polygon representation must always specify its points in counterclockwise order, with the first and last points being identical. Since parsers for serialized geometries, such as Well-Known Text (WKT), may not strictly enforce these conventions, inconsistencies can arise. Therefore, a comprehensive validation tool is required to perform a unified consistency check.

R3: Geometry-Based Computational Operations
Processes require the ability to perform operations on geometric data, such as calculating the surface area of walls or the volume of solid elements. Moreover, translation is an essential operation to align local project coordinates with real-world geographic coordinates by shifting the origin between coordinate systems. These computations are essential for cost estimation, sustainability analysis, and resource planning within the building model. While these operations could be performed offline by extracting the data and processing it externally using e.g. programming languages and libraries, integrating these capabilities directly into the query engine would significantly improve efficiency and streamline workflows. In addition, the RDF-based project dataset is simply an output of the design processes, which operate independently of the graph data. This means the RDF data does not require real-time updates with every design iteration. Instead, various design tools produce RDF datasets in predefined data-drops, which are planned to capture the project's status at key milestones throughout its lifecycle.

R4: Understanding Spatial Relationships Between Building Elements
A critical requirement is the ability to analyze spatial relationships between building components. For instance, determining whether two wall elements are touching or whether a device is within a wall relies on geometric processing. This capability is essential for two primary reasons: 1) to express enhanced validation constraints, e.g., to ensure that two walls do not touch if they should not, or to validate other spatial rules; 2) to support advanced use-cases, e.g. enabling building management tasks, such as device replacement, where accurately locating them is crucial. Spatial relationships typically occur at a higher level of abstraction, such as between rooms or large building elements. A simplified high-level geometric representation is preferred, as it enhances data processing and cross-domain integration. Combining geometric information with semantic power of graph data unifies data from different domains, such as spatial layouts, and IoT devices, and enables discovery of additional relationships.

3. State-of-the-art

Existing work has laid a strong foundation for representing and managing geometric data using Semantic Web technologies across domains such as GIS, BIM, and 3D city modeling. A range of vocabularies and ontologies have been proposed to express geometry in RDF, including GeoSPARQL, NeoGeo, W3C Geo, and AGO. Atemezing et al. [14] survey these standards, highlighting their varying support for geometry types like POINT, LINESTRING, and POLYGON, as well as their focus on interoperability. Regalia et al. [15] compare GeoSPARQL, NeoGeo, and AGO in terms of expressiveness and reasoning, while Bonduel et al. [16] examine strategies like embedding geometry as RDF literals and referencing external geometry files, introducing the File Ontology for Geometries (FOG) ontology to manage such links. OMG Ontology for Managing Geometry is an ontology for linking object geometry with metadata, allowing multiple versions and tracking changes over time. [17].

McGlinn et al. [18] categorize geometric data into 2D geospatial, simplified 3D, and semantically rich 3D geometries, identifying tensions between modeling needs, visualization, and computation on

the Web. Wagner et al. [1, 19] propose a classification of approaches for linking geometric resources in RDF: RDF-based descriptions, JSON-LD serializations, literal-based encoding, and links to external files. However, these classifications focus on data structure and do not address geometry querying, processing, or scalability. Bonsma et al. [20] introduce the GEOM ontology for complex 3D shapes such as NURBS and Boolean operations but acknowledge the limited coverage of geometric semantics and performance bottlenecks.

Validation methods for RDF geometry models remain underexplored; for instance, Stolk et al. [21] apply SHACL to check data types and structural constraints in ifcOWL models but do not validate geometric correctness. Ioannidis et al. [22] provide a benchmark dataset of 2D geospatial points (WGS84) to evaluate triplestores with GeoSPARQL support, focusing on spatial selection queries like `geof:sfWithin`, though their work is limited to 2D geometries. Full-system approaches such as Hor et al. [23] and Vinasco-Alvarez et al. [24] demonstrate RDF transformation pipelines for CityGML and IFC models, enabling integrated queries over GIS and BIM data. However, they remain limited to data transformation and retrieval, lacking geometry processing capabilities and robust validation.

Despite these contributions, current approaches largely emphasize representation and integration while overlooking processing, validation, and performance challenges, particularly for complex 3D geometries. Our work addresses this gap by evaluating existing paradigms through the lens of geometric processing, validation needs, and scalable querying in RDF-based systems.

4. Comparative Analysis Methodology

4.1. Approach Definition

Our comparative analysis categorizes existing approaches into three key paradigms: **A) RDF-based Geometry Modeling, B) Extending Triplestore and Querying Capabilities, and C) Integrating RDF-based Systems with Native Geometry Stores and Computer-Graphics Libraries**. Each category has been evaluated based on a set of defined criteria and practical examples, allowing us to systematically assess their suitability for addressing the above-mentioned challenges. The three paradigms are illustrated in Figure 1 and will be explained in the following sections.

A) RDF-based Geometry Modeling: Our study covers RDF ontologies like GeoSPARQL [25], and ifcOWL [26]. These assessed ontologies have different degrees of expressivity and alone do not provide any processing capabilities for geometric data. One of the requirements from the use-case is the capability of supporting multiple geometry representations. This is particularly relevant in scenarios where the primary purpose of the geometric model is data exchange, with the actual processing of the geometry handled by external systems. As a result, the models used must offer a high degree of expressiveness. Hence, this analysis is essential to understand which approaches support this, or whether multiple ontologies need to be combined.

B) Extending Triplestore and Querying Capabilities: Some approaches such as GeosSPARQL go beyond the modeling scope and propose extensions to RDF-graph triplestores. These extensions enable spatial functions and make the querying of spatial data possible, e.g. by introducing specialized indexes or by means of SPARQL custom-functions [5]. Unlike the previous approach, this is more applicable in a scenario where the processing of geometric data is carried out within the graph database, rather than being delegated to an external system. It also has the benefit of ensuring uniform processing of geometric data, regardless of the systems that may rely on that data.

C) Integrating RDF-based Systems with Native Geometry Stores or Computer-Graphics Libraries: Since, the two above-mentioned paradigms might not be a suitable solution for all use-cases, we have explored integration strategies between RDF-triplestores and other systems more capable of carrying out geometry-related tasks. One example is PostGIS, an extension for the relational database

PostgreSQL, which at a first glance offers comprehensive 3D and 2D geometric operations, benefiting from advanced database capabilities [27]. There are plenty of computer-graphics libraries as well, which have been designed to model complex geometries, but are harder to integrate with a system based on the Semantic Web stack. The integration between hybrid systems obviously increases the overall system complexity, and hence, we aim to deeply understand the implications.

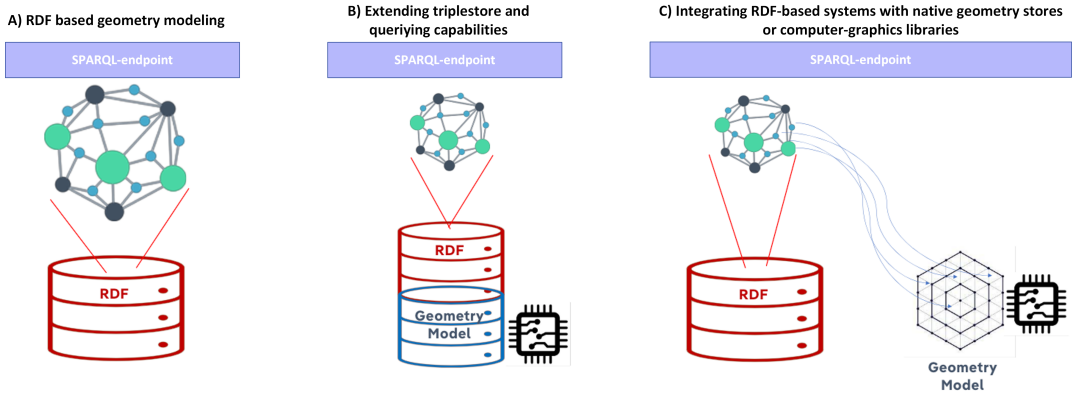

Figure 1: Three paradigms for enabling geometric representation in RDF.

4.2. Evaluation Criteria

To evaluate different approaches for representing and processing geometric information in RDF, we define a set of key criteria that reflects both academic standards and practical requirements observed in industry. These criteria are grouped for clarity and space optimization in the summary tables[1].

The criteria were chosen based on practical properties such as integration effort, maintainability, scalability, and tool interoperability. Evaluation was conducted based on modeling and querying tasks performed using actual buildings and corresponding RDF datasets. Verbosity and complexity were evaluated on RDF-based building projects with focus on comparing structural overhead and ease of use. Processing capabilities and interoperability were assessed through experiments on geometry-related operations and integration of RDF outputs in other tools. A qualitative scale is used to assess each criterion, as detailed in Section 5. While the evaluation remains qualitative, it is grounded in hands-on experience with RDF modeling, tool integration, and limitations encountered in practice.

Note that the requirements from the case study presented in Section 2 are related to these selected evaluation metrics. R1 (Support of Multiple Geometry Representations and Coordinate Systems) is mainly covered by *expressiveness* and *interoperability*. R2 (Reliable Validation of Geometric Data) essentially corresponds to *validation support*. R3 (Geometry-Based Computational Operations) and R4 (Understanding Spatial Relationships Between Building Elements) match *processing capabilities* and *support for geometry translation*.

5. Comparative Analysis Evaluation and Results

In Section 4, we introduced three paradigms to address the requirements derived from our case study. This section presents a comparative evaluation of these paradigms using the criteria defined earlier. To support consistent interpretation across all tables, we apply a unified qualitative scale: none, low, moderate, and high. These terms indicate increasing levels of capability or integration. For example, none means no support or reliance on manual workarounds, while high reflects mature, scalable, and integrated functionality. This scale is used throughout Tables 2, 3, and 4.

[1]Grouped as: Expressiveness and Interoperability; Geometry Support and Verbosity; Processing Capabilities and Geometry Translation; Degree of Standardization; and Complexity and Scalability. See Tables 2, 3, and 4.

Table 1
Evaluation Criteria for Geometric Information Approaches

Evaluation Metric	Description
Expressiveness	This metric assesses how well the language or framework can represent complex geometric structures. A more expressive system allows for a more detailed representation of geometries, including intricate relationships between elements. It also assesses whether it supports essential geometric modeling paradigms such as *Boundary Representation (BRep)* and *Constructive Solid Geometry (CSG)*[13]. Supporting both paradigms allows for greater flexibility in representing different types of building elements.
Complexity	Evaluates the conceptual and structural complexity of the resulting model. An ideal approach should be intuitive and easily understood by users, ensuring that domain experts and non-experts alike can work with the model efficiently.
Verbosity	Measures the level of verbosity required to define geometric elements. Some representations require highly detailed and explicit specifications, which may increase redundancy and hinder usability, while others provide more concise definitions.
Interoperability	Examines how well the approach integrates with existing BIM tools, data formats, and industry standards. High interoperability ensures seamless data exchange and reuse across different systems and platforms.
Processing Capabilities	Determines whether the approach supports necessary geometric operations such as computing areas, volumes, intersections, and spatial relationships, ensuring efficiency in 3D geometric operations.
Validation Support	Evaluates whether the approach provides built-in mechanisms to validate geometric data. Validation is critical for ensuring data consistency and correctness, particularly in 3D modeling scenarios where errors may lead to significant issues in downstream applications.
Degree of Standardization / Technology Adoption Rate	Examines whether the approach is based on widely adopted industry standards or proprietary implementations or it is widely adopted in specific communities, e.g. AEC, databases, etc. A higher degree of standardization facilitates long-term sustainability, compatibility, and industry acceptance. Standardization supports interoperability in AEC but may limit flexibility and innovation, especially when standards lag behind current practices or are difficult to integrate into existing workflows.
Support for Geometry Translation	Assesses whether the approach supports geometry translation across reference systems, a key requirement for hierarchical geometric models. Geometry translation ensures spatial consistency and facilitates integration into larger BIM ecosystems.
Scalability	Assesses how well the system scales when handling large BIM models and complex geometric datasets. Scalability is crucial for real-world applications where performance and efficiency are key factors in adopting a particular solution.

5.1. RDF-based Geometry Modeling

Several ontologies have been developed to model geometric representations or related aspects, such as coordinate systems, or geometry's metadata, but alone they lack processing and validation capabilities. Our evaluation includes ifcOWL, GeoSPARQL, OntoBREP, OMG (Ontology for Managing Geometry), W3C Geospatial Vocabulary, Basic Geo (WGS84 lat/long) Vocabulary, SWEET Ontology, FOG Ontology, Geometry Metadata Ontology (GOM), and GEOM. Most of these ontologies cover only secondary aspects of geometry representation, such as geometry metadata or are just linking frameworks. Below we summarize the main ontology's features.

- **ifcOWL** [28] – Developed to provide an RDF-based representation of *Industry Foundation Classes (IFC)*, the primary standard for BIM data exchange. It includes geometric constructs such as *extrusions, Boolean operations, and tessellated models*, but lacks built-in validation and computational geometry capabilities. It supports both BRep as well as CSG. However, the resulting model is very verbose [29] and it does not address the problem of inconsistencies between different representations [13].
- **GeoSPARQL** [25] – Created by the *Open Geospatial Consortium (OGC)* to model spatial data and topological relationships within GIS systems. It supports *2D geometries*, such as *points, lines, and polygons*, and provides a *query language for spatial operations*, but is not designed for detailed 3D geometries. GeoSPARQL supports common geometry serialization formats (e.g., WKT), and the Geography Markup Language (GML), embedded as literals.

- **OntoBREP** [30] – Developed specifically to represent Boundary Representation (BRep) models in RDF. It provides a structured way to define solid objects using faces, edges, and vertices, aligning closely with CAD/BIM applications. However, it does not natively support *operations such as intersections, unions, or geometry validation.*
- **SWEET Ontology** [31] – The *Semantic Web for Earth and Environmental Terminology (SWEET)* ontology suite introduces `GeometricalObject_3D`, which categorizes three-dimensional geometric shapes such as *cavities, cones, polyhedra, rings, shells, and spheres.* While useful for scientific spatial modeling, it lacks a *BIM-specific* integration with CAD/BIM tools.
- **RDF GEOM Ontology** [32] – Designed to facilitate geometric data exchange between different geometric editors, particularly BIM modelers. It supports a wide range of geometric concepts, from simple shapes to complex NURBS surfaces, and includes Boolean operations for modeling. GEOM provides a more comprehensive geometry modeling framework compared to many of the other ontologies reviewed.

While several ontologies exist to represent and link geometric data, most focus on metadata, spatial referencing, or linking RDF resources rather than providing a full geometry processing framework. The OMG and the FOG Ontology are designed to link geometry descriptions in various formats with non-geometric RDF resources. GOM also describes secondary aspects of geometry data such as coordinate systems, length units, file size and software of origin. Moreover, some ontologies are more focused on GIS concepts such as W3C Geospatial Vocabulary and Basic Geo (WGS84 lat/long) Vocabulary.

Table 2
Comparison of RDF-Based Geometry Modeling Approaches with Revised Evaluation Criteria

Category	Expressiveness and Interoperability	Complexity and Verbosity	Degree of Standardization	Scalability
IFC and ifcOWL	Highly expressive, integrates BIM and linked data (High)	Supports BRep and CSG but highly verbose (Low)	IFC-based, aligned with buildingSMART standards (High)	High complexity, limited scalability due to verbosity (Low)
GeoSPARQL (Ontology)	Moderate expressiveness, integrates with spatial databases (Moderate)	Limited validation, moderate verbosity (WKT/GML overhead) (Moderate)	OGC standard, widely adopted (High)	Limited support for spatial transformations, lacks full 3D capabilities (Low)
OntoBREP	Detailed boundary representations, moderate interoperability (Moderate)	Minimal validation support, high verbosity (Low)	No official standardization (Low)	High complexity, low scalability due to boundary model complexity (Low)
GEOM (Ontology)	Covers various geometric concepts, lacks absolute coordinates (Moderate)	No validation beyond basic geometry definitions, moderate verbosity (Moderate)	No official standardization (Low)	Moderate complexity, moderate scalability, focuses on structured geometric data (Moderate)
Sweet Ontology	Extensive 3D classification, moderate interoperability (High)	Limited validation, moderate verbosity due to classification details (Moderate)	Widely adopted in the Earth science community (High)	High complexity, moderate scalability (Moderate)

Table 2 presents a comparison of RDF-based geometry modeling approaches based on the revised evaluation criteria[2]. To clarify the differences between the vocabularies, we provide a minimal example in Appendix A, showing how the same geometric object—a rectangular wall—is represented using ifcOWL, GeoSPARQL, and OntoBREP. This highlights the structural expressiveness and verbosity of each approach. Among the ontologies which enable geometric modeling, IfcOWL stands out as the most comprehensive ontology for geometry representation. It is the only ontology capable of

[2]In this table, the criterion "Processing Capabilities and Support for Geometry Translation" has not been considered, as it is not applicable to this category. Approaches that extend beyond modeling and provide processing capabilities, such as GeoSPARQL, have been split into their ontology and query language components, and will be further discussed in the following section.

representing both BRep and CSG. Additionally, it is the only one that supports hierarchical geometry structures with local placements, allowing transformations to be defined in a chain of reference points through transformation matrices. This makes it possible to accurately compute the absolute location of geometric elements. While IfcOWL provides the most detailed and structured representation of geometry, it still lacks built-in computational capabilities for processing and validating geometric data, a gap that remains unaddressed across all ontologies evaluated.

5.2. Extending Triple-Store Capabilities

In addition to choosing an expressive ontology for RDF-based geometric representations, a triplestore can be extended to provide additional processing functionality. This section explores solutions which go in this direction, focusing on two major works: GeoSPARQL and BimSPARQL. We discuss their respective functionalities, strengths, and limitations, particularly in handling 3D geometries.

GeoSPARQL Beyond its modeling aspects (described in the previous section), GeoSPARQL introduces a standardized set of spatial functions for querying and manipulating geometries. These include operations for computing distances, spatial relationships (e.g., intersection, containment, adjacency), and transformations (e.g., buffering, simplification). While GeoSPARQL allows storing three-dimensional coordinates, it does not fully support 3D geometry processing. The standard follows a 2.5D approach, i.e. Z-coordinates are ignored in calculations. Consequently, true volumetric computations, such as intersection detection for 3D objects, are not supported. To ensure RDF data consistency, GeoSPARQL incorporates SHACL Shapes for validation. Despite its strengths in handling 2D spatial data, GeoSPARQL lacks robust support for 3D geometric reasoning, limiting its direct applicability in BIM workflows.

BimSPARQL BimSPARQL extends SPARQL for querying BIM data in RDF, specifically for building models in ifcOWL [7]. Unlike GeoSPARQL, it is an academic research proposal on incorporating building-related semantics. It introduced domain-specific functions for schema-level semantics, instance-level semantics, and spatial reasoning. BimSPARQL was implemented using SPIN (SPARQL Inferencing Notation), a rule-based extension of SPARQL that allows defining procedural logic. Unfortunately, our attempt to use BimSPARQL revealed that the project is no longer functional. The codebase no longer compiles, as its dependencies on the SPIN framework are outdated and unavailable.

GEOM Geometry Kernel The GEOM Geometry Kernel is a low-level library used across multiple products, providing core functionality for BREP modeling, NURBS, Boolean operations (including CSG), triangulation, and geometric calculations such as volume, area, and perimeter. The Parametric Geometry Modeling Kernel integrates with the Geometry Modeling Kernel, supports IFC, CityGML, etc., and stores parametric data in RDF, RDFS, OWL, and Turtle formats. To the best of our knowledge, there is no publication explaining whether they extend SPARQL with geometric operations or if these computations are done outside the RDF and Semantic Web stack. Since the tool is commercial and not open source, we are unable to test it, so we do not include it in our comparison table.

While extending triplestores with spatial capabilities offers a promising approach for handling geometric data, existing solutions have notable limitations. GeoSPARQL provides a solid foundation for 2D spatial queries but lacks support for true 3D geometric processing. BimSPARQL, which aimed to bridge this gap by introducing building-specific geometric reasoning, is unfortunately no longer maintained or functional. This highlights the technological gap in RDF-based triplestores when it comes to advanced 3D geometry handling.

5.3. Integrating RDF-compliant Systems with Native Geometry Stores or Computer-Graphics Libraries

This paradigm consists of externally storing geometric data in systems such as specialized databases rather than embedding it directly as RDF-triples. By linking an RDF building graph to a spatial database,

Table 3
Comparison of RDF-Based Triple-Store Extension Approaches with Revised Evaluation Criteria

Category	Expressiveness and Interoperability	Processing Capabilities and Support for Geometry Translation	Complexity and Verbosity	Degree of Standardization	Scalability
GeoSPARQL (Query Language)	Moderate expressiveness, integrates with spatial databases (moderate)	Provides spatial operations (distance, containment, adjacency) but lacks volumetric computations (moderate)	Supports 2D geometries, moderate verbosity (WKT/GML overhead), lacks full 3D support (moderate)	OGC standard, widely adopted (high)	Moderate complexity, scales well for 2D spatial data but lacks 3D geometry processing (moderate)
BimSPARQL	High expressiveness for BIM semantics but not standardized (high)	Allows querying of building components, spatial reasoning, and limited geometric processing (moderate)	Supports BIM-related queries but lacks built-in 3D validation (low)	Not standardized, academic research prototype (low)	High complexity, poor scalability; no longer maintained and not functional (low)

complex geometrical information can be efficiently managed and retrieved without overloading RDF datasets [33]. Depending on the type of external storage, the integration strategy could vary. For external stores with a JDBC driver, it shall be possible to create a virtual graph using OnTop and to use SPARQL federated queries to combine the geometric data with the building data. If no JDBC driver is available, more complex data flows are required, which might involve query rewriting or ETL processes. Among the available solutions, PostGIS—an extension of PostgreSQL—emerges as the only viable and fully developed system for integrating spatial databases with RDF-based knowledge graphs.

5.3.1. Geometry Processing in PostGIS

PostGIS is a spatial extension for PostgreSQL that provides robust support for geospatial data, enabling it to function as a spatial database for GIS applications. PostGIS stores geometries in WKT and Well-Known Binary (WKB) formats and offers a rich set of spatial functions for 2D and 3D geometry operations, including topological relationships, spatial transformations, and analytical queries. PostGIS supports a wide range of functions for 2D spatial analysis, including `ST_Within` to determine if one geometry is completely within another, `ST_Distance`, which computes the minimum Cartesian distance between two geometries. Other functions are `ST_Intersects`, `ST_Contains`, `ST_Union`.

A critical limitation of many RDF-based approaches is their inability to handle true 3D geometries. PostGIS partially overcomes this limitation by providing native 3D spatial functions, making it a viable solution for BIM-related processing. Key 3D operations include `ST_3DDistance`, which computes the shortest distance between two 3D geometries, and `ST_3DWithin`, which checks if two 3D geometries are within a specified distance. Additionally, PostGIS provides limited 3D geometry validation functions, e.g. `ST_IsSolid(geometry A)` checks if a given geometry is a valid solid. PostGIS also supports geometric transformations, enabling the manipulation of spatial data beyond simple storage. These transformations include `ST_Translate`, which moves a geometry by a specified offset, `ST_Affine`, which applies an affine transformation (scaling, translation, rotation). Other functions in this category are `ST_Rotate`, `ST_RotateX`, `ST_RotateY`, `ST_RotateZ`, and `ST_Scale`.

Integrating RDF data with native geometry stores like PostGIS offers an efficient way to manage complex spatial data. PostGIS is the only mature solution supporting both 2D and 3D geometries, enabling advanced spatial queries, indexing, and processing—critical for BIM applications. However, it remains a relational database and lacks native RDF support. While RDF-to-PostGIS mappings exist, there's no standard for executing SPARQL directly on geometry objects.

Unexpected Behavior in Some 3D Functions Computation in PostGIS. PostGIS provides advanced 3D spatial functions, many of which rely on `LWGEOM_mindistance3d` and `LWGEOM_maxdistance3d` functions. The `LWGEOM_mindistance3d` function computes the shortest Euclidean distance by evaluating vertex pairs rather than true surfaces, leading to inaccuracies when the nearest point isn't explicitly a vertex. Similarly, `LWGEOM_maxdistance3d` finds the greatest separation between vertices, which may not reflect the actual maximum distance for concave geometries.

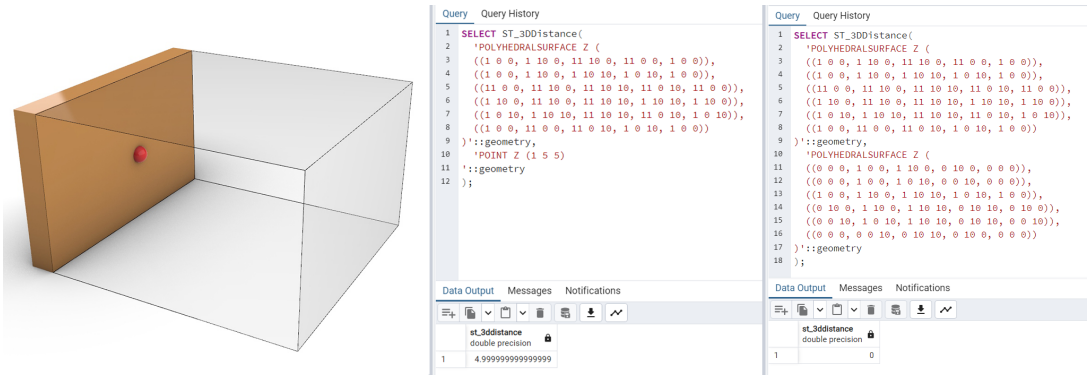

Figure 2: Unexpected 3D distance computation in PostGIS.

These limitations affect functions like `ST_3DDistance`, `ST_3DWithin`, `ST_3DFullyWithin`, and `ST_3DIntersects`. For instance, in Figure 2, a red sphere is placed against a glass wall, where the expected minimum distance is zero. However, `ST_3DDistance` returns 4.99999, failing to capture surface contact. In contrast, when measuring the distance between a solid wall and the glass room, PostGIS correctly returns zero because the evaluated vertices coincide with the boundary. Since PostGIS relies on discrete vertices rather than continuous surfaces, users should be cautious when applying 3D spatial functions in precision-critical applications.

5.3.2. Computer Graphics Libraries

Beyond ontologies and semantic web technologies, various computer-graphics libraries provide essential tools for geometric manipulation, analysis, and validation. They support tasks such as computational geometry operations, spatial analysis, visualization, and geometric validation, making them valuable for BIM applications and broader fields like GIS. We assessed the following libraries: **Shapely, PyVista, Trimesh, SFCGAL, CGAL, and OpenCascade**. Each of these tools provides unique capabilities.

CGAL. The Computational Geometry Algorithms Library (CGAL) [34] is a robust C++ library that provides high-precision geometric algorithms for 2D and 3D processing. It is one of the most expressive libraries, supporting both BRep and CSG, making it particularly well-suited for computational geometry applications that require complex modeling capabilities. However, due to its mathematical rigor and algorithmic depth, CGAL has a high level of complexity and a steep learning curve. CGAL includes extensive processing capabilities that offer robust Boolean operations, spatial reasoning, and validation tools. It also provides high-precision validation techniques for detecting geometric inconsistencies.

OpenCascade. OpenCascade [35] is an open-source CAD kernel for 3D solid modeling and geometry processing. It supports BRep and CSG representations, making it well-suited for CAD applications. However, due to its focus on industrial CAD applications, OpenCascade is relatively complex, requiring expertise in geometric modeling. It is also one of the most verbose libraries, as defining and manipulating geometries requires explicit specifications. OpenCascade assures high interoperability and extensive processing capabilities including Boolean operations, feature recognition, and geometric analysis.

Other. SFCGAL [36] extends CGAL with an additional parser for WKT and other features. It is used by PostGIS and it has similar limitations for 3D operations. From python libraries Shapely, PyVista, and Trimesh, only PyVista supports surface meshes and volumetric data, but it does not cover all functionality.

Summary. The evaluated libraries offer a range of capabilities for geometric processing, each excelling in different aspects. SFCGAL extends 3D processing within relational databases, whereas CGAL and

OpenCascade provide the most comprehensive solutions for computational geometry and solid modeling. Among these, CGAL and OpenCascade stand out as the most expressive libraries, fully supporting BRep and CSG representations, making them the best choices for advanced BIM applications requiring detailed geometric computations (see Table 4).

Table 4
Comparison of Geometry Processing in PostGIS and Computer Graphics Libraries with Revised Evaluation Criteria

Category	Expressiveness and Interoperability	Processing Capabilities and Support for Geometry Translation	Complexity and Verbosity	Degree of Standardization	Scalability
PostGIS	High expressiveness, integrates with spatial databases and RDF stores (high)	Provides advanced spatial queries, but lacks comprehensive 3D volumetric processing (moderate)	Supports 2D and limited 3D geometries, moderate verbosity (WKT/WKB) (moderate)	OGC standard, widely adopted (high)	Moderate complexity, highly scalable for 2D but limited scalability for complex 3D processing (moderate)
Shapely	Limited to 2D GIS applications, high interoperability with geospatial tools (low)	Basic spatial operations (e.g., intersection, union), lacks 3D support (low)	Supports 2D geometries only, concise definitions (high)	Follows OGC Simple Features standard (high)	Low complexity, scales well for small datasets but not optimized for large-scale geometry (moderate)
PyVista	High expressiveness for 3D visualization, integrates well with VTK (high)	Includes mesh transformations, smoothing, and reconstruction but lacks reasoning functions (moderate)	Supports surface meshes and volumetric data, moderate verbosity (moderate)	No official standardization, commonly used in scientific computing (low)	Moderate complexity, highly scalable for large 3D models in visualization and simulation (high)
Trimesh	Moderate expressiveness, optimized for mesh-based modeling (moderate)	Offers Boolean operations, ray tracing, and collision detection (moderate)	Supports triangular meshes only, concise definitions (high)	No official standardization, widely used in robotics and 3D modeling (low)	Low complexity, highly scalable for large mesh-based datasets (high)
SFCGAL	Expressive for solid modeling, tightly integrated with PostGIS (high)	Provides 3D Boolean operations, volumetric computations, and spatial validation (high)	Supports full 3D geometries, moderate verbosity (moderate)	Follows OGC spatial processing standards (high)	High complexity, scales well for database-driven spatial processing (high)
CGAL	Highly expressive, supports both BRep and Mesh (high)	Offers robust Boolean operations, spatial reasoning, and geometric validation (high)	Supports complex 2D and 3D geometries, high verbosity (low)	Well-established in computational geometry research (high)	High complexity, highly scalable for precision-demanding geometry applications (high)
OpenCascade	Extremely expressive, designed for CAD/BIM workflows (high)	Advanced CAD functionalities including feature recognition, Boolean operations, and transformations (high)	Supports detailed BRep and CSG models, very high verbosity (low)	Industry standard for CAD applications (high)	Very high complexity, highly scalable for industrial and large-scale CAD applications (high)

6. Conclusion and Future Work

The analysis presented in this paper highlights the advantages and limitations of the approaches covered in the three explored paradigms for integrating geometric data into RDF-based systems. Storing multiple representations of geometric data, performing geometric operations, validating geometric objects, and computing derived data (such as the volume) is essential part of any future BIM framework. The findings of this study can be summarized as follows:

RDF-based Geometry Modeling. Many ontologies exist for representing geometric data in RDF, but only ifcOWL is specifically designed to support multiple geometry representations. Furthermore, IFC's geometric modeling follows a hierarchical approach, meaning that footprints and other geometric elements are always defined relative to a reference point using local placements. Beyond the frameworks, ensuring interoperability with BIM-related processing tools requires the adoption of standardized geometry serialization formats, such as WKT. However, a key limitation remains: there is no standard serialization format for Constructive Solid Geometry (CSG), which restricts its broader adoption and the interoperability between different geometry tools.

Extending Triplestore and Querying Capabilities. Standards such as GeoSPARQL introduce spatial functions that enhance triplestores with spatial querying capabilities. While this improves queryng functionality around geometries, our evaluation reveals that geometric operations, such as area or volume, are still constrained to 2D geometries. Validation is only partially supported for 2D data and is entirely absent for 3D geometries. Given that full 3D support is indispensable for the serial production case study, these limitations make triplestore extensions insufficient as a standalone solution.

Integrating RDF-based Systems with Native Geometry Stores or Computational Geometry Libraries. We evaluated a widely adopted spatial database solution, PostGIS, which implements SQL-based and spatial standards. These solutions offer better 3D support than any RDF-based alternatives but still have significant gaps. For example, there are no built-in functions to answer full 3D spatial queries, such as determining whether a device is contained within a 3D room. Similarly, validation mechanisms for 3D geometries remain lacking, and while some computations for 3D volumetric analysis are possible, they remain limited compared to specialized geometric libraries.

Among all reviewed approaches, only external computational geometry libraries, particularly CGAL and OpenCascade, seem to fully satisfy all functional requirements for 3D geometries. However, these libraries are also the most complex to use. Furthermore, as these are computer-graphics libraries, with a higher integration effort. A feasible approach would be to implement SPARQL custom functions that invoke external code written using these libraries. Even such a seemingly simple integration strategy poses challenges, as most RDF triplestores are written in Java, and CGAL and OpenCascade are implemented in C++. Such an integration would be feasible using technologies such as the Java Native Interface (JNI). However, even if this functionality was successfully implemented, performance trade-offs remain a major concern. For instance, queries for containment operations (e.g., checking whether an object is inside another) are inherently quadratic in the number of geometric objects stored in the system. Thus, global optimization and pre-processing techniques might still be required to ensure efficient performance.

Summary This study provides valuable insights into the advantages and challenges of different paradigms for modeling geometries in RDF and extending RDF-based systems with geometric processing and validation capabilities. It reveals that there are trade-offs between complexity, interoperability, performance, and scalability, which must be carefully considered based on specific application requirements. However, current RDF-based systems lack the flexibility needed to tailor strategies such as horizontal scaling, data partitioning, spatial indexing, and caching for geometric data. Most platforms rely on fixed implementations with limited configurability, requiring users to operate within the constraints of predefined system architectures. Our findings also align with ongoing efforts to extend semantic geospatial standards—particularly the development of future versions of GeoSPARQL.

In the future, we will continue exploring how to extend RDF-based building models with geometric information. We plan to implement a prototype based on SPARQL custom-functions and computer-graphics libraries that supports higher-level geometric functions and integrates them into real-world applications. In addition, we will conduct a comprehensive performance and scalability evaluation to ensure the proposed solution meets the computational demands of large-scale BIM applications.

Acknowledgments

This research was supported by GROPYUS AG, Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under Germany's Excellence Strategy – EXC 2120/1 – 390831618, and the German Federal Institute for Research on Building, Urban Affairs, and Spatial Development, on behalf of the Federal Ministry of Housing, Urban Development and Building, through the Zukunft Bau research funding program (Grant Nos. 10.08.18.7-24.25).

Declaration on Generative AI

The authors used ChatGPT-4 to improve grammar. All content was reviewed and edited by the authors, who take full responsibility for the final manuscript.

References

[1] A. Wagner, M. Bonduel, P. Pauwels, U. Rüppel, Representing construction-related geometry in a semantic web context: A review of approaches, Automation in Construction 115 (2020) 103130. doi:10.1016/j.autcon.2020.103130.

[2] J. T. Ó'Donovan, D. O'Sullivan, K. McGlinn, A method for converting ifc geometric data into geosparql, in: LDAC, 2019. URL: https://api.semanticscholar.org/CorpusID:198164007.

[3] G. Sibenik, I. Kovacic, Assessment of model-based data exchange between architectural design and structural analysis, Journal of Building Engineering 32 (2020). doi:10.1016/j.jobe.2020.101589.

[4] A. Borrmann, M. König, C. Koch, J. Beetz (Eds.), Building Information Modeling: technology foundations and industry practice, Springer, Cham, Switzerland, 2018. doi:10.1007/978-3-319-92862-3.

[5] R. Battle, D. Kolas, Geosparql: Enabling a geospatial semantic web, in: Proceedings of 12th International Semantic Web Conference (ISWC), 2011.

[6] P. Pauwels, S. Zhang, Y.-C. Lee, Semantic web technologies in AEC industry: A literature overview, Automation in Construction 73 (2017) 145–165. doi:10.1016/j.autcon.2016.10.003.

[7] C. Zhang, J. Beetz, B. de Vries, Bimsparql: Domain-specific functional sparql extensions for querying rdf building data, Semantic Web 9 (2018) 829–855. doi:10.3233/SW-180297.

[8] D. Elshani, A. Lombardi, D. Hernandez, S. Staab, A. Fisher, T. Wortmann, Aec co-design workflow for cross-domain querying and reasoning using semantic web technologies, Automation in Construction 176 (2025) 106226. doi:https://doi.org/10.1016/j.autcon.2025.106226.

[9] D. Elshani, T. Wortmann, S. Staab, Towards better co-design with disciplinary ontologies: Review and evaluation of data interoperability in the aec industry, CEUR Workshop Proceedings (2022).

[10] United Nations Environment Programme - Global Alliance for Buildings and Construction, The Importance of Building Decarbonization and Benefits for the SDGs, 2024. URL: www.un.org/sites/un2.un.org/files/technical_brief_unep_global_alliance_synergies_conference.pdf.

[11] Pestel Institut gGmbH and Arbeitsgemeinschaft für zeitgemäßes Bauen e. V., Bauen und Wohnen in der Krise: Aktuelle Entwicklungen und Rückwirkungen auf Wohnungsbau und Wohnungsmärkte, 2023. URL: https://www.mieterbund.de/app/uploads/fileadmin/public/Studien/Studie_-_Bauen_und_Wohnen_in_der_Krise.pdf, accessed: 2025-02-10.

[12] J. Zhang, Towards systematic understanding of geometric representations in bim standard: An empirical data-driven approach, in: Proceedings of the ASCE Construction Research Congress, ASCE, 2018, pp. 96–105.

[13] A. H. Liu, C. Ellul, Quantifying geometric changes in bim-gis conversion, ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences 10 (2022) 185–192. doi:10.5194/isprs-annals-X-4-W2-2022-185-2022.

[14] G. Atemezing, R. Troncy, Comparing vocabularies for representing geographical features and their geometry, Terra Cognita 2012 Workshop (2012) 3. URL: https://ceur-ws.org/Vol-901/paper1.pdf.

[15] B. Regalia, K. Janowicz, G. McKenzie, Revisiting the representation of and need for raw geometries on the linked data web, in: Linked Data on the Web (LDOW@WWW), 2017.

[16] M. Bonduel, A. Wagner, P. Pauwels, M. Vergauwen, R. Klein, Including widespread geometry formats in semantic graphs using rdf literals, in: Proceedings of the 2019 European Conference on Computing in Construction, volume 1 of *Computing in Construction*, European Council on Computing in Construction, Chania, Greece, 2019, pp. 341–350. doi:10.35490/EC3.2019.166.

[17] A. Wagner, M. Bonduel, P. Pauwels, R. Uwe, Relating geometry descriptions to its derivatives on the web, in: Proceedings of the 2019 European Conference on Computing in Construction,

volume 1 of *Computing in Construction*, European Council on Computing in Construction, Chania, Greece, 2019, pp. 304–313. doi:10.35490/EC3.2019.146.

[18] K. McGlinn, A. Wagner, P. Pauwels, P. Bonsma, P. Kelly, D. O'Sullivan, Interlinking geospatial and building geometry with existing and developing standards on the web, Automation in Construction 103 (2019) 235–250. doi:10.1016/j.autcon.2018.12.026.

[19] A. Wagner, M. Bonduel, J. Werbrouck, K. McGlinn, Geometry and geospatial data on the web, in: P. Pauwels, D. Shelden, J. Brouwer, D. Sparks, S. Nirvik, T. P. McGinley (Eds.), Buildings and Semantics: Data Models and Web Technologies for the Built Environment, 1st edition ed., CRC Press, London, 2022, pp. 69–99. doi:10.1201/9781003204381-5.

[20] P. Bonsma, I. Bonsma, A. E. Ziri, E. Iadanza, F. Maietti, M. Medici, F. Ferrari, R. Sebastian, S. Bruinenberg, P. M. Lerones, Handling huge and complex 3d geometries with semantic web technology, IOP Conference Series: Materials Science and Engineering 364 (2018) 012041. doi:10.1088/1757-899X/364/1/012041.

[21] S. Stolk, K. McGlinn, Validation of ifcowl datasets using SHACL, in: M. Poveda-Villalón, A. Roxin, K. McGlinn, P. Pauwels (Eds.), Proceedings of the 8th Linked Data in Architecture and Construction Workshop, Dublin, Ireland, June 17-19, 2020 (virtually hosted), volume 2636 of *CEUR Workshop Proceedings*, CEUR-WS.org, 2020, pp. 91–104.

[22] T. Ioannidis, G. Garbis, K. Kyzirakos, K. Bereta, M. Koubarakis, Evaluating geospatial rdf stores using the benchmark geographica 2, Journal on Data Semantics 10 (2021) 189–228. doi:10.48550/arXiv.1906.01933.

[23] A.-H. Hor, G. Sohn, Design and evaluation of a bim-gis integrated information model using rdf graph database, ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences 8 (2021) 175–182. doi:10.5194/isprs-annals-VIII-4-W2-2021-175-2021.

[24] D. Vinasco-Alvarez, J. Samuel, S. Servigne, G. Gesquière, From CityGML to OWL, Technical Report, LIRIS, UMR 5205, 2020. URL: https://hal.science/hal-02948955v1, accessed via HAL open archive.

[25] Open Geospatial Consortium (OGC), Geosparql, https://www.ogc.org/standards/geosparql, 2024. Created by OGC to model spatial data and topological relationships within GIS systems.

[26] P. Pauwels, K. McGlinn, Buildings and semantics: Data models and web technologies for the built environment, 1st ed., CRC Press, 2022. doi:10.1201/9781003204381.

[27] R. Obe, L. Hsu, PostGIS in Action, Third Edition, Manning Publications, 2021. URL: https://www.manning.com/books/postgis-in-action-third-edition.

[28] buildingSMART International, ifcowl, https://standards.buildingsmart.org/IFC/RELEASE/IFC4/ADD2_TC1/HTML/schema/ifcowl/, 2018. Developed to provide an RDF-based representation of Industry Foundation Classes (IFC). accessed: 2025-02-11.

[29] D. Guo, E. Onstein, A. D. L. Rosa, An improved approach for effective describing geometric data in ifcowl through WKT high order expressions, in: C. Grueau, R. Laurini, L. Ragia (Eds.), Proceedings of the 7th International Conference on Geographical Information Systems Theory, Applications and Management, GISTAM, 2021. doi:10.5220/0010532302290236.

[30] A. Perzylo, N. Somani, M. Rickert, A. Knoll, An ontology for cad data and geometric constraints as a link between product models and semantic robot task descriptions, in: 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2015, pp. 4197–4203.

[31] NASA Jet Propulsion Laboratory, Sweet ontology, https://sweet.jpl.nasa.gov/, 2024. A component of the Semantic Web for Earth and Environmental Terminology (SWEET) ontology suite.

[32] RDF Ltd., RDF Ltd., https://rdf.bg/, 2025. Provides solutions for creating interactive 3D and 4D geometrical data, supporting various standards including IFC and STEP. accessed: 2025-02-11.

[33] P. Pauwels, A. Costin, M. H. Rasmussen, Knowledge Graphs and Linked Data for the Built Environment, Springer International Publishing, Cham, 2022. doi:10.1007/978-3-030-82430-3_7.

[34] The CGAL Project, CGAL Documentation, 2025. URL: https://www.cgal.org/, accessed: 2025-02-11.

[35] Open Cascade SAS, Open CASCADE Technology Documentation, 2025. URL: https://dev.opencascade.org/resources/documentation, accessed: 2025-02-11.

[36] L. Bartoletti, R. Delhome, J. Felder, F. Fougères, Sfcgal documentation, 2025. URL: https://sfcgal.gitlab.io/SFCGAL/, accessed: 2025-02-11.

Appendix A: Example Geometry Representation Across RDF Vocabularies

To illustrate the differences in expressiveness and structure among the RDF vocabularies evaluated in Section 5.1, we provide a simplified example of a rectangular wall represented in three selected ontologies: ifcOWL, GeoSPARQL, and OntoBREP. The geometry is a basic 4-meter wide, 3-meter tall wall element, located at a specific coordinate system origin.

1. ifcOWL Representation (CSG-based)

The geometry is modeled using an extruded area solid in the context of an IFC schema expressed as RDF:

Listing 1: ifcOWL representation of a wall using parametric extrusion

```
@prefix :<http://www.example.com/> .
@prefix ifc:<https://standards.buildingsmart.org/IFC/DEV/IFC4/ADD2_TC1/OWL#> .

:Wall_1 a ifc:IfcWallStandardCase ;
     ifc:Representation :WallGeom_1 .

:WallGeom_1 a ifc:IfcProductDefinitionShape ;
        ifc:Representations :ShapeRep_1 .

:ShapeRep_1 a ifc:IfcShapeRepresentation ;
        ifc:Items :Extrusion_1 .

:Extrusion_1 a ifc:IfcExtrudedAreaSolid ;
         ifc:SweptArea :RectProfile_1 ;
         ifc:Depth "0.2"^^xsd:double ;
         ifc:ExtrudedDirection :ZDir .

:RectProfile_1 a ifc:IfcRectangleProfileDef ;
          ifc:XDim "4.0"^^xsd:double ;
          ifc:YDim "3.0"^^xsd:double .
```

2. GeoSPARQL Representation (WKT-based)

The same wall can be abstracted as a 2D polygon footprint without vertical information:

Listing 2: GeoSPARQL representation of the wall footprint using WKT

```
@prefix geo:<http://www.opengis.net/ont/geosparql#> .
@prefix :<http://www.example.com/> .

:Wall_1 a geo:Feature ;
     geo:hasGeometry :WallGeom_1 .

:WallGeom_1 a geo:Geometry ;
        geo:asWKT "POLYGON((0 0,4 0,4 0.2,0 0.2,0 0))"^^geo:wktLiteral ;
        geo:crs <http://www.opengis.net/def/crs/EPSG/0/4326> .
```

3. OntoBREP Representation (Topological structure)

OntoBREP allows a more detailed structural description of surfaces:

Listing 3: OntoBREP representation using face-edge-vertex topology

```
@prefix obrep:<http://www.fortiss.org/kb/ontobrep.owl#> .
@prefix :<http://www.example.com/> .

:Wall_1 a obrep:Solid ;
      obrep:hasFace :Face_1 .

:Face_1 a obrep:Face ;
      obrep:hasEdgeLoop :Loop_1 .

:Loop_1 a obrep:EdgeLoop ;
      obrep:hasDirectedEdge (:Edge_1 :Edge_2 :Edge_3 :Edge_4) .

:Edge_1 a obrep:Edge ; obrep:hasStart :V1 ; obrep:hasEnd :V2 .
:Edge_2 a obrep:Edge ; obrep:hasStart :V2 ; obrep:hasEnd :V3 .
...

:V1 a obrep:Vertex ; obrep:coordinates "0 0 0" .
:V2 a obrep:Vertex ; obrep:coordinates "4 0 0" .
...
```

Discussion

- **ifcOWL** provides high-level parametric representations, supporting domain-specific semantics but requiring deeper IFC knowledge.
- **GeoSPARQL** simplifies spatial representation via literals and is easy to query but typically remains limited to 2D geometries.
- **OntoBREP** captures topological precision and surface relationships, but it is verbose and lacks native support for geometric computation or visualization.

This example comparison complements the evaluation in Table 2 and clarifies the expressiveness and usability trade-offs discussed in Section 5.1.