Knowledge representation for neuro-symbolic digital building twin querying

Stéphane Reynaud[1,2,*], Anthony Dumas[1] and Ana Roxin[2]

[1]B27-AI, R&D, 2 rue René Char, 21000 Dijon, France
[2]Laboratoire d'Informatique de Bourgogne (LIB EA 7534), Université de Bourgogne, 21000 Dijon, France

Abstract

The complexity of modern construction projects necessitates collaboration among diverse stakeholders and the handling of substantial data volumes. Significant investments in digitization have occurred globally to address this complexity, emphasising the need for interoperability among standards and diverse knowledge sources. The emergence of Digital Building Twins (DBTs) further underscores the importance of integrating heterogeneous data to create comprehensive digital representations of buildings. DBTs enable real-time data integration and support various phases of architectural development, offering practitioners access to historical, present, and predictive data. In this context, our research focuses on enhancing the accessibility and interpretability of DBT data through natural language querying. Leveraging domain-specific ontology and advanced AI techniques, our approach facilitates efficient communication between users and DBTs, enabling rapid extraction of specific building details. This paper presents our methodology, including knowledge representation, semantic analysis, and information extraction, along with evaluation results demonstrating its effectiveness in improving DBT querying performance.

Keywords

Digital Building Twin (DBT), Building Information Modelling (BIM), Artificial Intelligence (AI), Neuro-symbolic AI, Ontology, Knowledge representation, Knowledge base question answering (KBQA)

1. Introduction

Construction projects are complex and require collaboration between various stakeholders to develop increasingly intelligent structures. Smart constructions entail handling substantial data volumes, which must be organised for services like digital archives, data analysis, and predictive interpretation. Significant investments in digitisation occur globally to address the complexity of information required for decision-making in today's construction industry, necessitating interoperability among the underlying standards used. Historically, GIS (Geographic Information Systems) addresses the modelling of the environment surrounding a construction, while BIM (Building Information Modelling) tackles the construction itself. To have a complete model of constructions and their environment, one must integrate GIS and BIM standards, thus blurring traditional distinctions between the two domains, highlighting the need to remove barriers to

AI4DT&CP@IJCAI2024: The Second Workshop on AI for Digital Twins and Cyber-Physical Applications in conjunction with 33rd International Joint Conference on Artificial Intelligence, 3rd-9th August 2024, Jeju, South Korea
*Corresponding author.
sreynaud@b27.fr (S. Reynaud); adumas@b27.fr (A. Dumas); ana-maria.roxin@u-bourgogne.fr (A. Roxin)
0009-0003-3117-7765 (S. Reynaud); 0009-0009-1204-5488 (A. Dumas); 0000-0001-9841-0494 (A. Roxin)

information sharing across disciplines, ultimately optimising investments and enhancing data quality [1].

Additionally, Digital Building Twins (DBT) represent a crucial aspect of modern construction processes. A DBT entails real-time communication among sensors to create a comprehensive digital representation of a building at a specific moment, reflecting the evolving understanding of construction in the digital era. Typically relying on Building Information Modelling (BIM), a DBT represents the virtual counterpart of a physical structure, replicating its real attributes and conditions [1]. DBTs enable seamless real-time data integration and support multiple phases of architectural development, from design to operational and maintenance stages. DBTs allow AEC (Architecture, Engineering, Construction) practitioners to gain access to historical, present, and predictive data, empowering them to explore diverse optimisation possibilities. In the era of Industry 4.0, DBTs are instrumental in smart buildings, streamlining data aggregation, optimising operations, and facilitating predictive maintenance processes [2].

To best understand the challenges associated with accessing information from a DBT, let us consider the example of a large building with a surface of 10,000 square meters and 50 stories, which is a structure that typically has numerous emergency exits. Querying information about these exits, including their dimensions and compliance status, involves using some software tool to navigate through different levels and inspect potential existing door labels such as "fire exit" or "emergency exit." This task is time-consuming, taking at least a dozen minutes for experienced users and up to 30 minutes for less seasoned practitioners, particularly if thoroughness is required. Additionally, obtaining dimensions of door frames often requires guesswork, as this information may not be readily available.

Our approach aims to efficiently address such queries by providing specific building details that may not be directly represented in the existing standard schema of the DBT. The approach emphasises the effectiveness and user-centric design of the enhanced information retrieval system presented in [3] by enabling users to obtain this information in seconds rather than minutes.

Following our definition of a DBT, this article presents our approach for an intuitive "dialogue" between humans and DBTs by rendering the data within a DBT accessible and interpretable via machine processing. Our approach allows expert and novice users to ask queries in natural language by leveraging domain-specific and semantically rich ontology crafted by domain experts, i.e. the OB27AI ontology. As such, it leverages advanced capabilities derived from the domain of Artificial Intelligence (AI), namely symbolic AI (through knowledge representation), deep learning and Natural Language Processing (NLP).

In this paper, we focus on improving the performance of queries addressed in natural language over knowledge representations of DBTs. Our contributions are summarised as follows:

- We describe a complete approach for natural language querying of DBTs
- We define a new knowledge representation of DBTs in the form of the OB27AI ontology and detail its design and contents
- We enhance our approach presented in [3] in terms of semantic analysis and information extraction
- We specify our implementation for querying the DBT along with its related phases, i.e. query analysis, query parsing, grounding, extraction, generation, execution and

transformation
- We evaluate the performance improvement of our approach compared to [3], both on a general level (in terms of recall, precision and F1 score) and on the level of each phase.

The article is structured as follows: section 2 provides some definitions forming the necessary scientific background for our research, and section 3 resumes existing research done in the related domains. Our approach is introduced in section 4, while its implementation is provided in section 5. Last but not least, section 6 gives the results of the evaluation of our prototype, and section 7 concludes this paper.

2. Scientific background

The following sections provide the necessary definitions and references to understand our approach's components.

2.1. Modelling & querying knowledge

Semantic Web (SW) aims to make heterogeneous information accessible to a machine, allowing to turn data into machine-processable knowledge. SW uses an ensemble of W3C standards for annotating data present on the Web and enabling knowledge graphs (KG). Based on the existing Web standards, e.g. URI (Universal Resource Identifier) for resource identification and HTTP as a universal access protocol, SW standards build upon RDF (Resource Description Framework) for representing simple facts about resources (identified by HTTP URIs) and upon the OWL (Web Ontology Language) family for describing more complex knowledge in the form of ontologies. In an ontology, knowledge is expressed as "Subject - Predicate - Object" triples and can be queried or modified using SPARQL (SPARQL Protocol and RDF Query Language). Additionally, Linked Data (LD) has been defined by the W3C[1] as four principles for enhancing the share and reuse of such annotated data to form a global network of knowledge accessible and comprehensible by a network of machines. This link between different silos of knowledge can be expressed in different ways, depending on the need. If it is necessary to align knowledge for inference purposes (e.g. owl:equivalentClass), the reasoning mechanics provided by OWL are to be preferred but can be costly; if the link to be established is strictly semantic, using the vocabulary of an existing ontology such as SKOS is sufficient [4].

The OWL family comprises several profiles for ontology description languages based on Description logic (DL) with different levels of expressivity. They allow for defining complex knowledge representations, combining concepts, relationships among concepts and constraints over these elements, generally expressed as logical rules or necessary and sufficient conditions over class definitions. Given their symbolic and logical nature, it is possible to reason upon such knowledge representations by applying specific algorithms (called reasoners) that allow implicit axioms from the explicit ones contained in the ontology to be inferred. KGs are obtained when using LD principles to connect elements from different ontologies.

[1] https://www.w3.org/

2.2. OpenBIM

Building Information Modelling (BIM) has become vital in the last decade in AEC (Architecture, Engineering and Construction). Following ISO 23386:2020 [5], BIM is defined as the "use of a shared digital representation of an asset to facilitate design, construction and operation processes to form a reliable basis for decisions". OpenBIM is a collaborative process supported by the BuildingSMART International organisation, emphasising the interoperability of construction project data through open standards and common vocabularies.

Industry Foundation Classes (IFC) [6] is the openBIM standard exchange format for exchanging building and infrastructure digital representations in the AEC industry. Initially designed for sharing and exchanging product data, the different effective implementations of the IFC data model have proven their limitations for querying and analysis tasks[7]. Indeed, the primary goal of IFC is to effectively represent basic concepts [8] and descriptive building information, often characterised by complex structures. This complexity has led to challenges in querying and managing IFC instance data. While promoted as a vector for interoperability, IFC lacks the flexibility to integrate and process data from multiple sources [9].

OpenBIM also defines the exchange requirements for every type of business process and every BIM design stage through Information Delivery Manuals (IDMs) [10]. IDMs describe the framework for implementing Model View Definitions (MVDs)[2], representing a subset of the information strictly necessary for a specific business process [11]. For the IFC 2x3 TC1 specification, the most common MVD is the "Coordination View 2.0", which enables BIM requirements to be matched to each discipline, such as architecture, MEP and structure, represented respectively by the CV2.0-Arch, CV2.0-MEP and CV2.0-Struct views. Once the MVD has been extracted from the IFC file, the IDM file is used within the visualisation software to check the conformity of the MVD concerning the requirements criteria. Finally, the BIM Collaboration Format (BCF), encapsulated in IFC, is used for reporting purposes to alert the various stakeholders and ensure that the project runs smoothly.

2.3. Interoperability

As seen in section 2.2, the AEC industry has wide heterogeneity in existing knowledge and data. The diversity and variation in information types, structures, formats, and semantics pose interoperability challenges, as defined by ISO/IEC 2382:2015 [12].

Current standards outline three primary levels of interoperability: data, syntactic, and semantic. These levels are interconnected and progressively enhance each other, with lower tiers furnishing the necessary components for the functionalities of higher levels [13]:

- Data interoperability, as defined by ISO/IEC 20944-1:2013 [14], relates to the comprehensive spectrum of activities involving data creation, interpretation, computation, utilisation, transmission, and interchange.
- Syntactic interoperability, as defined by ISO 22378:2022 [15], refers to the capability of two or more systems or services to exchange structured information seamlessly. It ensures that the formats of the exchanged information are comprehensible to the participating systems, as outlined by ISO/IEC 19941:2017 [16].

[2]https://technical.buildingsmart.org/standards/ifc/mvd/mvd-database/

- Semantic interoperability, as defined by ISO/TS 18308:2011 [17], refers to the ability of data exchanged between systems to be understood in terms of fully specified domain concepts. It enables systems or services to automatically interpret and utilise exchanged information accurately, as outlined by ISO 22378:2022 [15]. Furthermore, it ensures that the participating systems comprehend the meaning of the data model within the context of a specific subject area, as outlined by ISO/IEC 19941:2017 [16].

Data and syntactic interoperability are already extensively addressed by standard protocols (e.g. TCP/IP, HTTP) and the adoption of syntax standards (e.g. XML, HTML). Semantic interoperability, though, is yet to be fully addressed by existing standards and approaches.

2.4. Natural Language Processing

Although BIM is considered state-of-the-art in the AEC industry, its adoption rate remains relatively low [18]. This slow adoption highlights the need to explore methods to enhance users' interaction with BIM models, such as using Natural Language Processing (NLP).

NLP is a branch of AI that enables computers to understand, interpret, and generate human language meaningfully. NLP utilises techniques from various fields, including linguistics, computer science, and machine learning, to bridge the gap between human communication and computational systems. Contrary to the formal knowledge models provided by ontologies, language models are ambiguous and vague, while the relation between symbols to objects is not clearly defined as in DL. Language models are defined as a "probability distribution describing the likelihood of any string" [19]. Given these issues, approaches based on neural networks have encountered a large success in this field. Providing an extensive state-of-the-art of these goes beyond the scope of this article, but to best understand our contribution, two approaches are worth presenting: the Transformer architecture and Pre-trained Language Models (PLMs).

The Transformer architecture, outlined in [20], introduces the concept of "self-attention", allowing each word to consider all others, disregarding their positions. It enables the language model to prioritise words effectively (using positional embedding), which is crucial for generating accurate outputs if we consider the above definition of a language model. Comprising encoder and decoder layers with self-attention and feed-forward neural networks, the Transformer replaces traditional recurrent and convolutional layers, improving parallelisation and capturing long-range dependencies more efficiently. It also incorporates positional encodings to maintain sequence order information [19].

Built on the Transformer architecture, several approaches called Pretrained Language Models (PLMs) have been published, i.e. BERT (Bidirectional Encoder Representations from Transformers) or GPT (Generative Pretrained Transformer). As an extension of the Transformer architecture, PLMs are a form of transfer learning that uses large and general text datasets to train an initial NLP model with self-supervised learning objectives. The initial NLP model is refined using smaller domain-specific text datasets [19].

There are three broad categories of PLM transformers:

- Encoder-only (e.g. Bidirectional Encoder Representations from Transformers aka BERT) are meant to generate contextual representations of text. We aim to test SpaCy LLM,

based on OpenLLaMA[3] which uses Large Language Model Meta AI (LLaMA), developed by Meta.

- Decoder-only (e.g. Generative Pre-trained Transformer aka GPT) are purposed to generate new text based on a context. We plan to test Falcon[4], developed by TII (Technology Innovation Institute) UAE and GPT, developed by OpenAI, which is only available on-premises in version 2[5] (now in version 4).
- Encoder-decoder (e.g. Text-to-Text Transfer Transformer aka T5) are geared toward transforming text. We seek to test Flan-T5[6], developed by Google, a T5 fine-tuned using instruction-based prompts.

NLP can be split into two main domains [21]: Natural Language Understanding (NLU) and Natural Language Generation (NLG). NLU deciphers human language, parsing inputs into a structured format for machines. It includes tasks like entity recognition, syntactic, and semantic analysis. NLG[22], on the other hand, creates coherent text from structured data, performing tasks such as report generation and personalized messaging. Together, NLU and NLG cover language processing from comprehension to production.

3. Related work

This section summarises existing standards and works in the literature about our approach's different components.

3.1. Knowledge modelling for the digital building twin

To address the limitations of the IFC standard as mentioned in section 2.2, bSI organisation promotes the ifcOWL ontology, which serves as a foundational domain ontology for the AEC industry [23].

ifcOWL provides an OWL2-DL semantic representation of the IFC schema, which can be used not only to access information in an IFC model, but also to link it with external data not explicitly described in the IFC schema. Due to the verbose structure underlying the IFC data model (more than 900 classes), the resulting OWL structure has a high OWL2-DL expressivity (SROIQ(D)) [23], which inevitably leads to the collapse of reasoning engine performance due to the number of axioms and assertions to be processed [24].

When considering the use of ifcOWL for handling and interpreting DBT data, several types of research have highlighted the need for less expressive structures [25] as well as for adapted querying approaches [7]. Additionally, several methodologies have been published addressing the simplification of DBT data conversion from IFC into an ontology, such as IFC2LD[7],

[3]https://github.com/openlm-research/open_llama
[4]https://falconllm.tii.ae/
[5]https://openai.com/research/gpt-2-1-5b-release
[6]https://github.com/google-research/t5x/blob/main/docs/models.md#flan-t5-checkpoints
[7]https://github.com/Web-of-Building-Data/ifc2ld.git

IFCtoRDF[8], IFCtoLBD[9], KGG[10] [26]. In addition, numerous works have led to ontologies with narrower scopes and/or simplified structures [25, 27, 28].

3.2. Approaches for aligning ontologies

"Ontology alignment is the process of identifying correspondences between semantically related entities of different ontologies" [29] to address semantic heterogeneity and enable semantic interoperability. This process leads to identifying various types of relationships among entities (e.g. equivalence, subsumption, disjointness).

As we have already seen, there is a wide disparity in the data around the field of AEC construction engineering. Generally speaking, ontologies answer how to structure all this data, enabling it to be expressed as knowledge. Moreover, depending on the formalism used (RDF / OWL / etc.), they enable us to define a balance between the expressiveness and decidability of the knowledge they describe [30], to express the axioms and rules of behaviour of each of the entities within the domains in which they are represented, and to apply a logical reasoning mechanism to them.

Aligning 2 or more ontologies to map the entities (ABox / TBox) of each domain under study addresses the problem of semantic heterogeneity in the data [29, 31].

There are 4 main approaches to ontological alignment [32]:

1. Terminological analysis methods focus on lexical, syntactic or linguistic similarities [33]. They consist of comparing labels qualifying knowledge based on the notions of synonymy, polysemy, translation or according to methods for calculating frequencies of occurrence (TF-IDF, bags of words, etc.).
2. Approaches based on internal ontological structures (metadata, cardinalities, constraints, attributes, etc.) and external structures (taxonomies) [34].
3. Instance-based approaches - extensions use statistical and data analysis methods to determine a level of sharing of standard entity-to-entity information and resources between the ontologies to be aligned.
4. Semantic similarity approaches are based on the meaning and context of entities. With the rise of Large Language Models, many researchers are looking into ontology alignment based on generative AI [35], as they are particularly well suited to natural language-based information retrieval.

Ontology matching approaches are evaluated by the OAEI (Ontology Alignment Evaluation Initiative)[11], which brings together all the research on the subject. Ontology alignment remains a real challenge for the scientific community.

3.3. Knowledge Base Question Answering

Knowledge Base Question Answering (KBQA) systems aim to find answers to queries expressed in natural language from a Knowledge Base (KB). Present methodologies for KBQA can be

[8]https://github.com/pipauwel/IFCtoRDF
[9]https://github.com/jyrkioraskari/IFCtoLBD
[10]https://github.com/kyriakos-katsigarakis/openmetrics/tree/master/openmetrics-kgg
[11]http://oaei.ontologymatching.org/

broadly categorised into two main groups:

1. Semantic Parsing Based Methods: These approaches focus on a semantic parser that translates questions into intermediate logic forms. These logical forms are then executed against the KB to retrieve relevant answers [36].
2. Information Retrieval Based Methods: These methods involve retrieving potential answers from the topic-entity-centric sub-graph within the KB. Subsequently, the answers retrieved are ranked based on their relevance to the questions asked [37].

Recently, there has been a notable surge in research interest towards combining these two approaches for KBQA [38]. Indeed, the neuro-symbolic reasoning (NSR) approach gains attention from the efficacy and enhanced interpretability offered by semantic parsing and its symbolic backbone and the powerful capabilities of neural networks to capture information from unstructured data surrounding topics of interest as in information retrieval [39].

When considering KBQA systems for querying DBT data, the semantic parsing of natural language queries involves identifying IFC concepts, their relationships, and value restrictions. As the existing ifcOWL ontology does not provide natural language descriptions or labels for its elements (i.e. IFC concepts), the IFC Natural Language Expression (INLE) ontology [40] was developed to add these necessary constructs, therefore supplementing ifcOWL with natural language representations of IFC concepts, providing notably synonyms, hyponyms, abbreviations, and morphological variations. Moreover, this INLE ontology has been developed as a seed ontology for model-specific semantic interpretation of natural language queries, focusing on all building elements (e.g., IfcWall, IfcDoor, IfcBeam) and 2 types of spatial elements (IfcBuildingStorey and IfcSpace) from IFC 2x3 TC1, comprising 121 classes, 58 object properties, 446 individuals, and 3071 axioms in all.

3.4. Approaches for interoperability

As seen in section 2.1, SW promotes using standardised vocabularies, ontologies and linked data principles to enable machine-readable and interoperable information. Still, semantic heterogeneity is persistent as knowledge representation is done through independent ontologies with no existing links. Ontologies being formal knowledge models, one can consider standard ISO 11354-1:2011 [41] interoperability approaches, i.e. integration, unification, or federation, for addressing the related semantic interoperability issues.

Integration harmonises elements/systems into a cohesive common whole, ensuring they conform to a common format for seamless interaction. It prioritises interoperability from the outset, ideal for new system designs requiring adherence to a common form. Specifically regarding ontologies, integration involves defining a new ontology and defining outgoing links to the elements of existing ontologies.

Unification establishes a shared meta-model for semantic equivalence, aiding translation between different models. The meta-model ranges from a foundational reference vocabulary to an intricate ontology. It maps all alternative models to this common framework, enabling translation but potentially leading to some loss of information. Studies show it enhances interoperability among AEC stakeholders [42, 43].

Federation is decentralised, allowing entities to maintain autonomy without conforming to a common format. Interoperability is achieved through mappings and agreements despite differences in vocabularies or methodologies. Unlike integration and unification, it doesn't require a common format or meta-model, allowing each entity to retain autonomy. This approach suits contexts with diverse or intricate vocabularies and methodologies. As such, it has higher design and implementation costs and provides the greatest flexibility. FOWLA (Federated Architecture for OWL Ontologies) is one of the few examples of an ontology federation approach [44].

4. Our approach

4.1. Overview of the contribution

We aim to enhance the general performance of the approach presented in [3] in terms of precision, recall and F1 score. The overall system architecture is presented in 1 below. We highlight in green the new elements added in this version compared to the previous one.

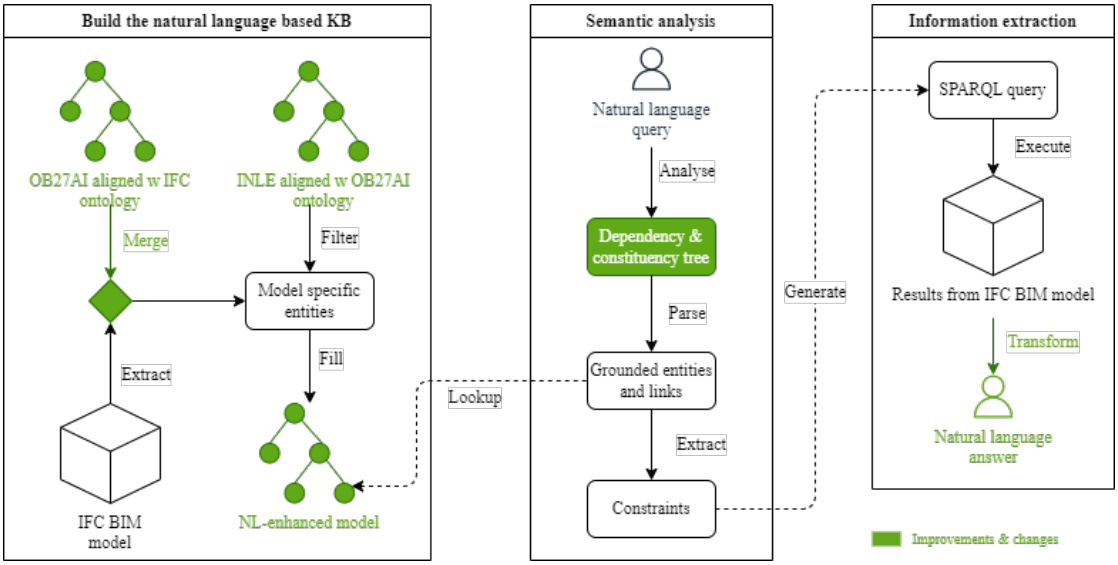

Figure 1: Our new approach vs the one presented at SITIS [3]).

The main modifications consist of a new domain ontology OB27AI (detailed in section 4.2) along with a dependency tree providing a syntactic understanding of the input sentence through constituency parsing coupled with an NL-enhanced model for grounding query elements (detailed in section 4.3).

4.2. Knowledge modelling

To do so, we decided to develop a new domain ontology that would complement it, with input from domain experts, to have a finer-grain description of the desired subdomains while keeping the ontology simple enough for existing reasoners to work effectively.

There are many methods to develop a new domain ontology (e.g. TOVE, Seven-Step, SENSUS): among these methods, the Seven-Step [45] is chosen as it's widely used and fits our approach. It adapts Gruber's five main principles of ontology construction [46].

This methodology is iterative, and often, steps are revisited as the development process evolves. Each step builds on the previous ones, contributing to a comprehensive and robust ontology that accurately represents the knowledge in a particular domain. Additionally, continuous validation with domain experts and potential ontology users is essential throughout these steps to ensure its accuracy and usability.

4.2.1. Our domain ontology

The following paragraphs detail the methodology for conceiving our domain ontology (OB27AI). According to the Seven-Step methodology, we first determine our ontology's domain and scope. We chose to focus on the following subdomains to encompass the DBT and its wider context and surroundings:

- Structural works: the major works of a building (e.g. foundations, load-bearing walls and beams).
- Heating, Ventilating and Air Conditioning (HVAC): all the fluids (e.g. water, air, gas) within a building.
- Highways and miscellaneous external works: outside of a building (e.g. access and pathways, roads and parking areas, drainage networks, tanks), and the outdoor structural works (e.g. preparation of the site for foundations, fittings, excavation)
- Electrical works: requirements and layout of electrical equipment, high-voltage (e.g. electrical power, lighting), low-voltage (e.g. telephone network, computer network), and fire safety (e.g. detectors, fire alarms)
- Materials: e.g. concrete, steel, aluminium, wood, cement.
- Woodwork: similar to structural works but specific to wood-based structural elements.
- All trade works: covers all of the above, as well as all the "second works" of a building (e.g. partitions, non-load-bearing walls, doors and windows, paint, skirting boards).

For each domain above, domain experts enumerate the important terms and structure them into a taxonomy built top-down from a single high-level concept. They follow simple recommendations about naming conventions and guidelines regarding the expected structure. A group review of these taxonomies is then done to help the domain experts finalise them, to ensure the level of detail is in line with our objectives, and to check there are no duplicates of concepts across domains. The taxonomies are then processed automatically to fill the ontology. A manual review is done afterwards to check the consistency of the ontology, adapting it when needed while not deviating from the taxonomies. As outlined in the Seven-Step methodology, the whole process is iterative.

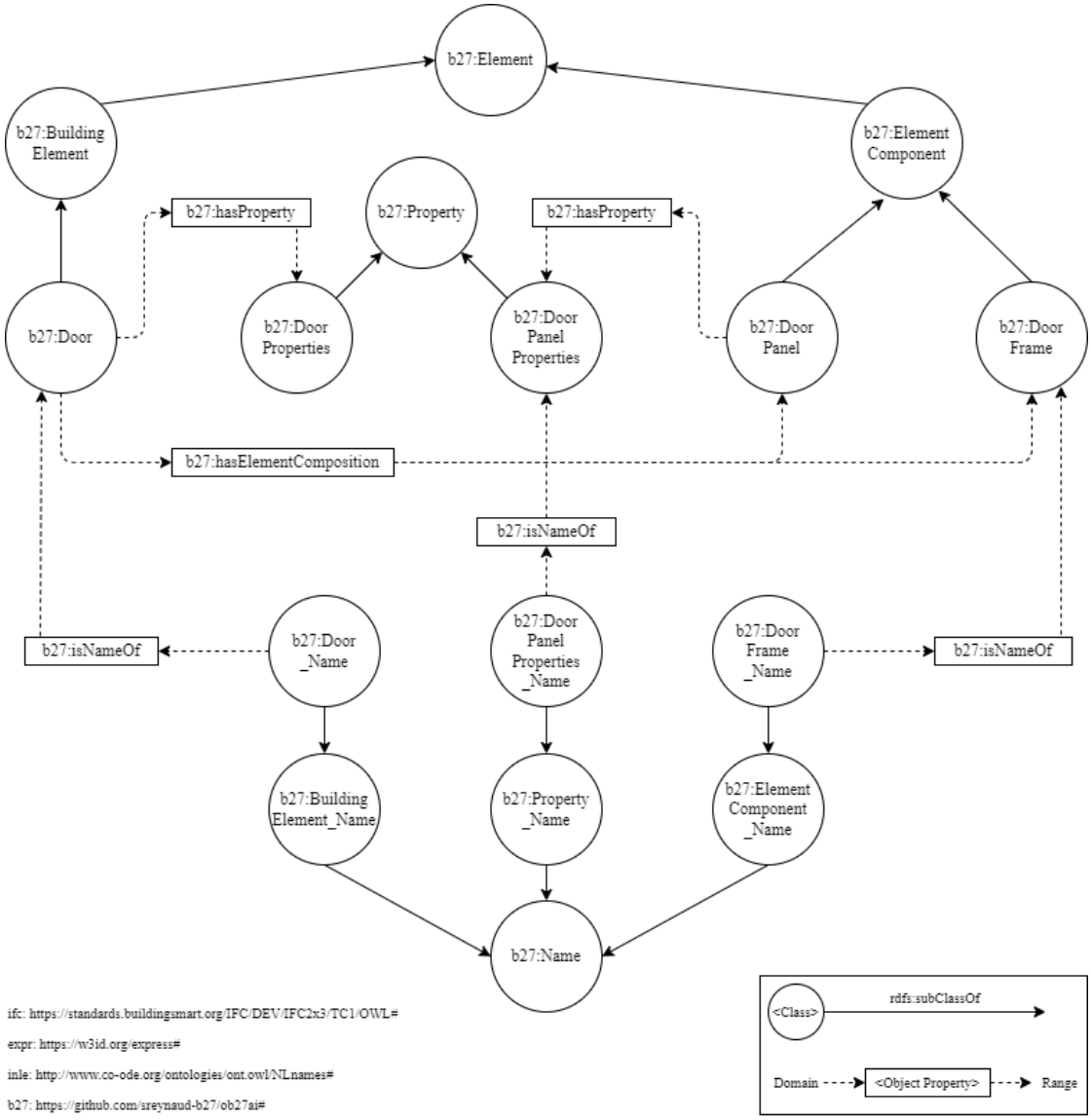

Figure 2: Simplified view of the TBox of our OB27AI ontology.

The TBox represents the entities and their relationships on a conceptual level, as shown in Fig. 2. The classes and the hierarchies we choose are similar to those found in the IFC ontology (version 2x3 TC1), though the following differences and additions drive our work:

- More classes are defined to materialise components of larger systems (e.g. door frame) for finer granularity.
- The properties are simpler and straighter to keep the ontology more human-readable.
- The natural language description of objects is prominent in the "b27:Name" class and its

sub-classes, as it's a key point to access the model and interact with the DBT.

The "b27:Name" is the root for all natural language representations of the concepts we chose to focus on in our domain ontology, providing synonyms, hyponyms, hypernyms, abbreviations and morphological variations, along with a similarity index related to the main (i.e. origin) term representing a concept. As we aim to go deeper with a finer grain in our chosen subdomains than the INLE ontology, we rely on this ontology (and its alignments) for a natural language expression of the broader IFC concepts that may not be covered by our ontology. This approach has been chosen to ease the interoperability with ifcOWL [23] and INLE [40]. We provide an example snippet of the description of a "b27:Name_DoorPanel", a subclass of "b27:Name" in Listing 1.

Listing 1: Example of a "b27:Name" subclass

4.2.2. Ontology alignment

Following the Seven-Step methodology, we identified the different ontologies we could reuse. ifcOWL is the most comprehensive ontology for IFC, which, in our approach, is complemented by INLE for handling natural language. As we create a new ontology, we maintain interoperability with ifcOWL and INLE through unification for a faster and tighter alignment. To do so, we embed the shared meta-model in our OB27AI ontology. A future venue would be a federation approach for interoperability, allowing our ontology to be fully independent, transposing the existing meta-model into mappings and agreements, and moving it out of our ontology.

We choose to proceed with alignment based on the internal structures of ifcOWL and INLE, which allows greater control over the inferences in the context of global reasoning. Our work is easier because INLE has been built as an extension of ifcOWL. Therefore, they are both already aligned. The considered ontologies have very few individuals in their original state, i.e., they are not linked to a model. Therefore, we focus on the schema of these ontologies, i.e., their classes and properties.

Our methodology has been inspired by Euzenat et al. [29] and consists of the following steps.

1. Adjust the structure of our ontologies to facilitate easier association.
2. Assess the similarities to establish the matching along these axes:
    - Linguistic: considers the textual similarities between entity labels
    - Structural: considers the structure of the ontologies, i.e. the classes, their hierarchies, and their relationships.
    - Semantic: considers the meaning of terms by relying primarily on domain experts.
3. Generate the meta-model to be embedded within our OB27AI ontology for most mapping based on constraints, equivalence, and subsumption. A more complex mapping part requires establishing inference rules outside our OB27AI ontology.
4. Validate the alignment using reasoning tools for inconsistencies and logical errors. A manual review is done with the domain experts to finalise the validation.

5. Refine the alignment each time a new taxonomy is incorporated into our OB27AI ontology by repeating this process on the extended ontology.
6. Integrate the alignments to produce our OB27AI ontology aligned with ifcOWL, and a modified INLE ontology aligned with ours (Fig. 3).

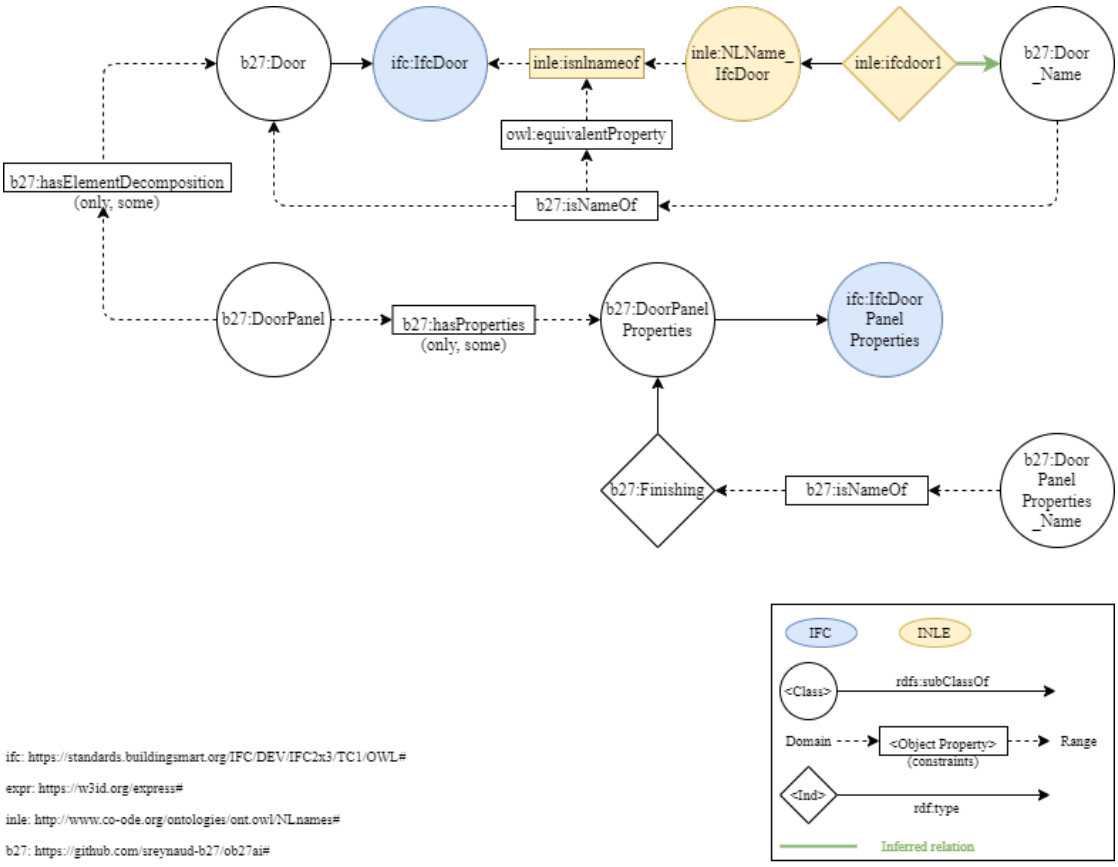

Figure 3: Example of alignment between our OB27AI ontology, IFC and INLE.

As briefly seen on Fig. 3, the INLE ontology has some generic individuals (e.g. "inle:ifcdoor1"). Our OB27AI ontology has such instances for properties, predefined values of properties, materials, etc., that are shown with more details in Fig. 4

However, most individuals are related to the names (and their variants, i.e. synonym, hyponym, lemma, etc.) in natural languages used by the domain experts, including common terms. Those are used to attach labels to the objects (e.g. door panel, door leaf), their properties (e.g. door aperture), the values of properties (e.g. automatic or manual door operation), etc.

As illustrated in Fig. 5, based on the labelling done in a building model (e.g. "Ridge beam"), the precise type of object (e.g. RidgeBeam) from our ontology is retrieved for an individual that is otherwise typed as a broader IFC class (e.g. IfcBeam). This process allows us to add

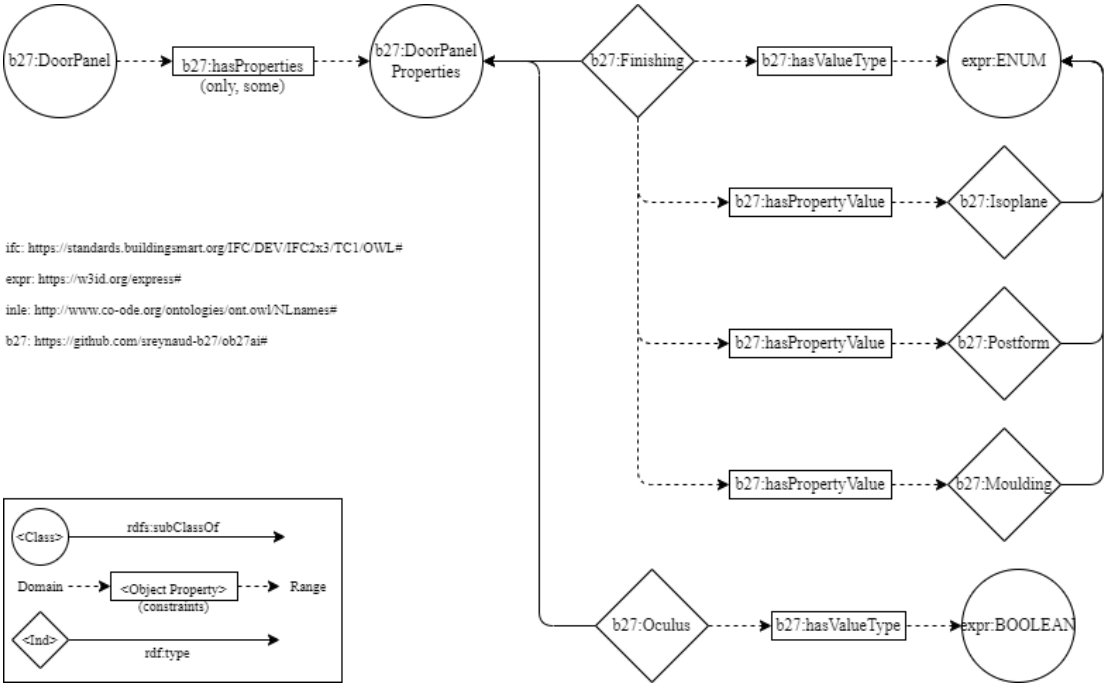

Figure 4: Example of generic individuals in our OB27AI ontology.

constraints of all kinds (e.g. material, dimension, position, property, etc.) on these individuals, enabling us to reach our goal for a fine-grained description within the subdomains we chose.

We provide an example snippet another example of alignment around "b27:Door" in Listing 2.

Listing 2: Example of an OB27AI alignment

The complete list of alignments can be found as a link set on the GitHub - donner le lien.

The complete definition of these alignments represents ongoing work and goes beyond the scope of this article. Besides corresponding to a correct application of Linked Data principles, such alignments provide the advantage of inheriting from properties and constraints defined in the other ontologies, thus allowing us to further reason on DBT knowledge. For example, aligning our classes with the BOT ontology allows us to take advantage of the different topological properties defined in BOT (Building Topology Ontology) ref - https://w3c-lbd-cg.github.io/bot/, thus allowing us to decompose the DBT into different spaces. Additionally, such alignments allow other ontologies to reuse the natural language annotation of building elements, as permitted by the B27AI ontology.

4.3. Natural Language (NL) querying

In our previous approach [3], we built a KBQA system based on semantic analysis methodology, where the named entity recognition (NER) phase is a mixture of neural networks to decompose

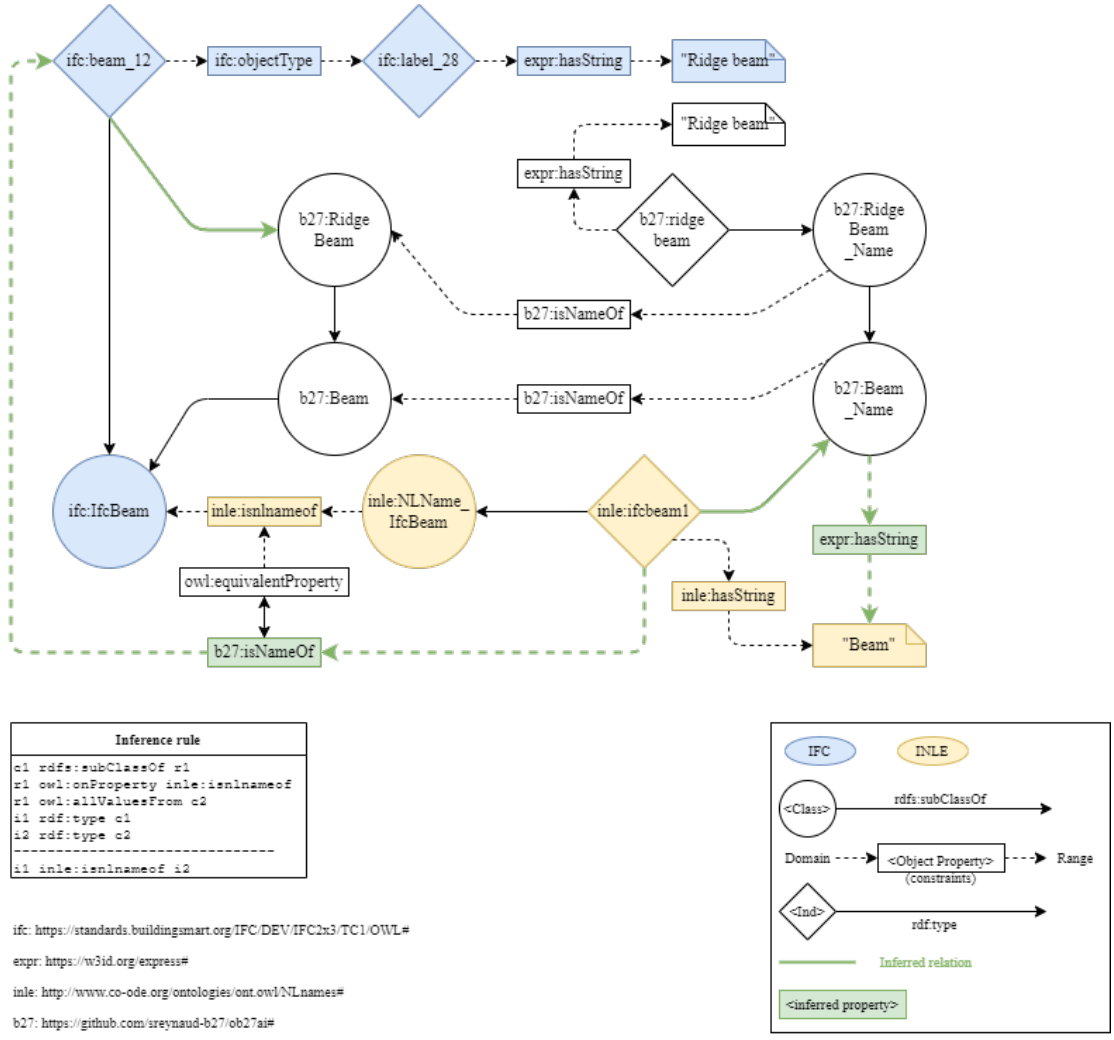

Figure 5: Example of an individual in a DBT described by our OB27AI ontology, IFC and INLE.

the user's query and symbolic reasoning to ground entities in the KG.

To improve our KBQA system, different approaches for the NER phase needed to be tested. The parsing of the input sentence is done on the semantic level using a convolutional neural network (CNN), providing a dependency tree where nodes represent the words, and the directed edges represent dependencies between the words. Each dependency is labelled with a relationship type that describes the nature of the dependency (e.g., subject, object, modifier).

We aim to test adding a syntactic understanding of the input sentence through constituency parsing, where the output reflects the syntactic structure of a sentence in terms of a hierarchical tree of phrases, sub-phrases, and words.

Additionally, we aim to test different types of neural networks widely used for NLU. Indeed,

as mentioned in section 2.4, Transformer neural networks have gained considerable attention in NLP-related tasks, as they show superior performance in handling of context and sequence than CNNs and recurrent neural networks (RNNs) [20], due to their attention mechanisms and ability to model bidirectional contexts efficiently [47].

Our KBQA system outputs the items in the DBT corresponding to a query. This raw dry output contrasts sharply with the user's natural language query. Therefore, we decide to use NLG to represent the output in a more user-friendly way, testing template-based sentence generation and pre-trained language models (PLM) for text generation.

In both the NLU and NLG tasks, our goal is to find the optimal balance between the relevance of the output and the speed to obtain it.

5. Implementation

To build and maintain our ontology and its alignments, we use Protégé[12], an open-source tool from Stanford University that facilitates creating, maintaining, and editing ontologies. It offers a graphical interface for modelling ontologies using OWL and supports testing inferences and running SPARQL queries.

5.1. Ontology's population

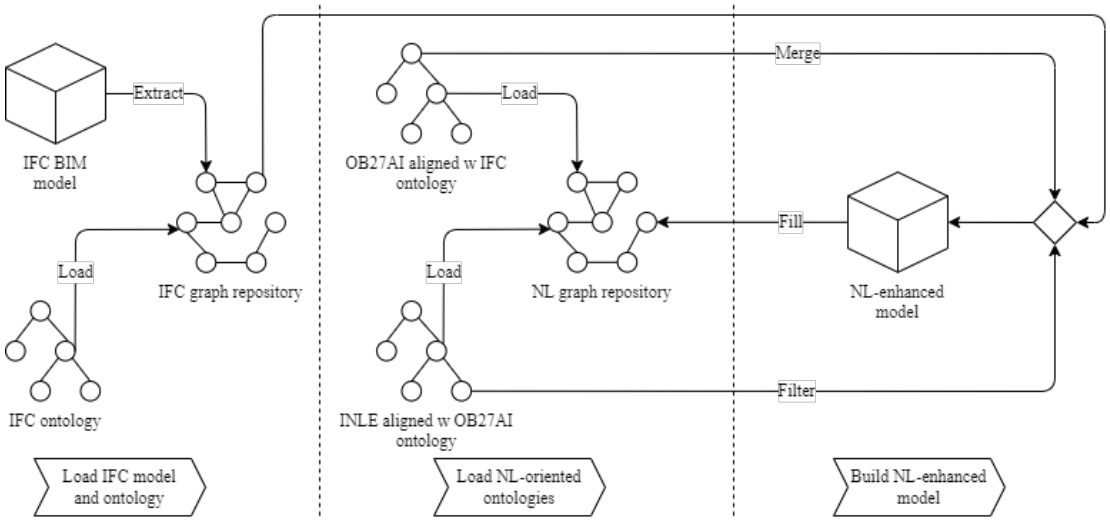

Figure 6: Process to populate our Knowledge Graphs

Another output built from the work done by the domain experts is embedding properties and constraints into the DBT. Specifically, tools such as Autodesk Revit[13], offer the capability to

[12]https://protege.stanford.edu/
[13]https://www.autodesk.com/products/revit/

add custom fields, predefined values, and sometimes even some basic rules (e.g. EXAMPLE) to the model. This constitutes the preparatory step for practitioners to build and/or obtain a DBT enriched with our domain ontology.

To populate our KGs, the first step is to create an IFC-based repository in our graph DB (DataBase), loaded with the ifcOWL (IFC 2x3 TC1) ontology from the official file (using the turtle serialization format). The DBT is exported to an IFC raw file (STEP format), which is then transformed into an RDF file (using IFCtoRDF) to allow the model to be imported into the IFC-based repository of the graph DB. A small duplex apartment building contains 454,359 statements (364,216 explicit and 90,143 inferred) and has an expansion ratio of 1.25.

The second step is to create an NL-based repository in the graph DB, loaded with our OB27AI ontology aligned with IFC and the INLE ontology aligned with our ontology: both are imported from their respective RDF files. A small duplex apartment building contains 34,727 statements (18,856 explicit and 15,871 inferred) and has an expansion ratio of 1.84.

The third step is to build and fill an NL-based representation of the DBT from the IFC-based repository. Once our OB27AI ontology (aligned with IFC) is merged with the IFC-based graph representation of the model, the entities and concepts of interest are filtered based on the INLE ontology (aligned with ours), to fill the NL-based repository with the NL-enhanced model.

These three steps, shown in Fig. 6, are done using Python scripts and the GraphDB[14] APIs (Application Programming Interfaces).

GraphDB, edited by OntoText, is our chosen graph database engine and knowledge management tool, equipped with reasoning and inferencing capabilities and compliant with RDF and SPARQL.

Python is selected as the programming language due to its versatility and the extensive array of libraries and frameworks available, both third-party and native, in the relevant areas.

The owlready2[15] module, licensed under GNU LGPL v3, is a Python module for loading, browsing, and modifying ontologies, as well as for searching, inferencing, and executing SPARQL queries.

The minimalist and lightweight Flask[16] framework is employed for developing web services in Python. It provides the essential elements required to create lightweight web applications.

Conda[17] is an open-source package manager that creates isolated Python environments, each containing packages and dependencies specific to various tools and experiments.

Project management and sharing are facilitated through GitHub, Microsoft's online platform that uses the Git version control system.

Microsoft's Visual Studio Code (VS Code) is the source code editor. It is lightweight, fast, and versatile, with particularly strong integration of Python, Conda, and Git.

5.2. NL querying

Our approach for elaborating an NL answer from an NL query comprises 7 steps and is shown in Fig. 7.

[14] https://www.ontotext.com/products/graphdb/
[15] https://owlready2.readthedocs.io/en/latest/
[16] https://pypi.org/project/Flask/
[17] https://anaconda.org/anaconda/conda

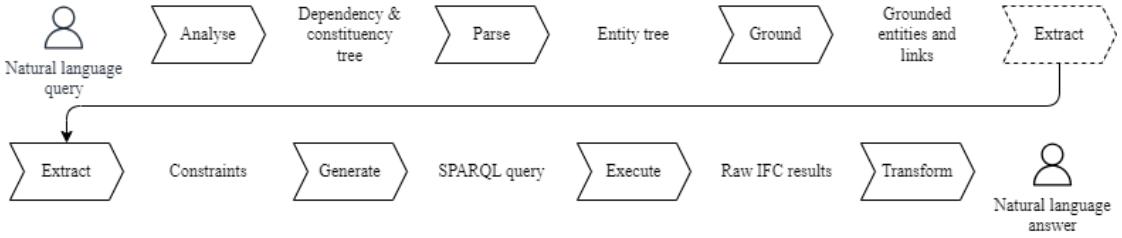

Figure 7: Steps to parsing the natural language query

The "Analyse" step uses the user's question as input, triggering multiple NLU analyses to obtain one dependency and one constituency tree combined into a single tree representing the query.

The "Parse" step traverses the dependency and constituency tree to recognise potential entities. It organises them in branches of different lengths, following the overall structure of the input tree to generate an entity tree. This entity tree comprises all the possible branches between two entities, including their alternatives above a threshold based on semantic similarities with the terms from the origin sentence.

The "Ground" step traverses the entity tree to look for the entities in the NL KG while also searching for potential links between such entities in this KG. These grounded entities and links (aka grounded triples) are stored in a tree structurally identical to the one of the entity tree, though lighter as some branches have been pruned, either for the lack of possible relationships between entities or because their local relevance dropped below the threshold to keep them. The value for local relevance is computed on semantic similarities of entities and distances between classes and domains or ranges.

The "Extract" step explores the grounded triples, expands them in local SPARQL queries to be executed against the IFC KG, and ranks them against the triple score computed for the pruning in the previous step. The goal is to prune the grounded tree further when a query does not return a result, except for comparison (e.g., walls higher than 3m). Each branch's best constraint (aka triples grounded in the IFC KG) is kept for the output list.

The "Generate" step stitches the constraints together in a shallow tree-like structure to account for the potential hierarchy of sub-queries to form the final SPARQL query.

The "Execute" step executes the final SPARQL query against the IFC KG and collects the raw list of IFC results.

The "Transform" step parses the raw list of IFC results to obtain the labels and types of the objects from the DBT so that very simple sentences can be created automatically using templates.

These sentences can be presented to the user, though more importantly, they can be used, in combination with the user's query for context, to generate more complex sentences using PLMs (introduced in section 2.4).

As input to these models, either the keywords from the questions and the answers are provided, or the question along the template-generated answer and an instruction. For example, with the question "List the exterior doors with a width larger than 1m." (#21 in appendix A),

there are 2 items in the ground truth:

- The raw keywords are "2 door is external has property width larger than 1 meter".
- The template-based generated answer is "There are 2 items of type door. These items are 'Door_6597' and 'Door_6702'.".
- For models that can incorporate prompts (i.e. Falcon, GPT, Flan-T5), the full input is then: "Question: List the exterior doors with a width larger than 1m.
  Answer: There are 2 items of type door. These items are 'Door_6597' and 'Door_6702'.
  Generate a sentence to answer the question:"

For NLU tasks to understand the user's query, the primary open-source library we used is spaCy[18], renowned for its high execution speed and precision in processing text. Available for Python, spaCy is designed to handle various NLP tasks, including tokenization, lemmatization, parsing, and building dependency trees.

For the specialised and precise task of constructing constituency trees that represent questions, the BeNePar[19] (Berkeley Neural Parser) is used for its exceptional efficiency [48]. Originally developed at the University of Berkeley, this parser is designed to conduct syntactic analysis of sentences using deep neural networks, enhancing the accuracy of such analyses. It captures complex syntactic dependencies, enabling precise analysis in multiple languages. BeNePar is integrated with spaCy.

Hugging Face[20] provides open-source libraries that offer easy access to various pre-trained models for diverse NLP tasks alongside a platform for sharing models and datasets.

6. Results & evaluation

The tests were conducted on the two versions of our KBQA system, using a set of 22 questions like "Search for all the load bearing beams with slope equal to 0." (see appendix A for full list), the ground truth having been manually extracted from the DBT.

6.1. KBQA performance

When the first version was evaluated against this new set of questions, longer and more complex, the recall dropped from 95.0% to 50.5%, and the F1 score from 0.974 to 0.671, mostly because this version has been using the small spaCy CNN model "en_core_web_sm", a model 35 times smaller than its transformer counterpart.

The two versions in our tests (see Table 1) have been configured to use the same NLP neural network from the spaCy transformer library ("en_core_web_trf"). The tests were performed on a computer with an Intel ® i5 CPU (2.5 GHz), 16 GB RAM, and the Windows 10 64-bit system.

Our precision is lower by 2.7 points due to many false positives in a single question (#11 in appendix A). The recall is improved by 18.6 points, while the F1 score, more balanced, shows an improvement of 8.9 points.

[18] https://spacy.io/
[19] https://github.com/nikitakit/self-attentive-parser
[20] https://huggingface.co

Table 1
Performance metrics (%)

	NSR-BIM v1 [3]	NSR-BIM v2 (Ours)	Variation (%)
Recall	75.8	94.4	+20
Precision	93.7	91.0	-3
F1 score	83.8	92.7	+10

Table 2
Population execution time (s)

Step	NSR-BIM v1 [3]	NSR-BIM v2 (Ours)	Variation (%)
1 - Load IFC model and ontology	21	11	-39
2 - Load NL-oriented ontologies	2	1	-50
3 - Build NL-enhanced model	9	6	-33
Total per model	32	18	-44

The time taken for knowledge data preparation, shown in Table 2, is drastically reduced by 44%, mostly thanks to the use of a graph database instead of file-based storage.

Table 3
Query processing execution time (s)

Step	NSR-BIM v1 [3]	NSR-BIM v2 (Ours)	Variation (%)
1 - Analyse	0.34	0.36	+6
2 - Parse	0.01	0.08	+700
3 - Ground	-	1.17	n/a
4 - Extract	0.09	0.44	+389
5 - Generate	0.02	0.01	-50
6 - Execute	0.55	0.03	-95
7 - Transform	-	0.01	n/a
Total per single question	1.01	2.10	+107

The query processing time, shown in Table 3, is further detailed below.

The step #1 is slightly longer as a constituency tree is produced in version 2 while not in version 1, although most of the time taken in the analysis of the question is linked to the NLP model used ("en_core_web_trf"): using a smaller and simpler model ("en_core_web_sm") results in a duration of 0.01s instead of 0.34s, though the trees obtained are syntactically and semantically too weak and faulty when longer and more complex sentences are used.

Step #2 in version 2 is 8 times longer (0.08s vs 0.01s) than in version 1 because two trees have to be parsed and combined, and mainly because semantic similarities against terms in the ontologies are computed at this stage.

The step #3 is new to version 2, as no grounding against KG was done in version 1. When this step was first added to the first version, it took 3.52s on average, more than twice the

current time, as the existing tree traversing algorithm was not designed to handle that many tree branches and triples.

The step #4 is more than 4 times longer, as there are more extracted constraints, each tested against the IFC KG in version 2. Those constraints were taken as they were without further pruning in version 1: when introducing constraints testing against the KG in version 1, the average time climbed to 0.75s as too many constraints were tested.

Step #5 is twice as fast in version 2, as step #4 already performed the expansion of constraints into SPARQL sub-queries to enable executing them in the IFC KG.

The step #6 is almost 20 times faster, as version 2 uses a graph database instead of files.

Step #7 is new to version 2: the time measures have been done using template-based answer generation, not transformer-based PLM.

The overall time to process a query has doubled on average in version 2, largely due to the intermediate grounding of the whole tree against both NL and IFC KGs. For example, the question "List the exterior doors with a width larger than 1m." (#21 in appendix A), which used to have 2 branches with 12 triples in the first version, has 6 branches with 57 triples in the second version, each of them being tentatively grounded in the KGs in this second version.

Nevertheless, in version 1, this question #21 found no answers as the 2 branches completely separated "width" from "door": both branches were directly and separately attached to the root. This issue, widespread in test questions, was fixed by creating a larger and deeper tree representing the question: in the case of question #21, some branches were now linking both concepts of "width" and "door". However, having larger and deeper trees required better pruning to avoid the rise of wrong answers, hence the grounding and extraction steps (#3 and #4) that would test the branches and their triples against NL and IFC KGs.

Another example of a question that was failing in version 1 was "What are the dimensions of the door frame of all the building's emergency exits?". This question is not part of the test set because version 1 could not possibly find its answer: indeed, this is the kind of question where a concept (e.g. "door frame") doesn't exist explicitly in the model. Therefore, such a concept is inferred through our OB27AI ontology aligned with IFC, resulting in finding a relevant answer with our second version.

Regardless of implementation details, one major venue to improve the execution speed could be to use Cython, which allows compiling Python code in C or C++ and the direct call to functions written in C or C++. Another major venue could be loading the NLP neural network model to a GPU. There is also room for improvement in the grounding queries made for each possible triple.

A contextual performance comparison between our first version and MOP-SP [40], which is outside our scope here, shall be found in [3].

6.2. NLG evaluation

To generate a user-friendly answer, we tested 5 models of Transformer-based PLM (see Table 4), available on-premises (as opposed to cloud-based), with both qualitative (rated "very poor", "poor", "good" or "very good") and quantitative (typical time to generate the answer) criteria. When possible, the prompt and the question in full were added as input to the template-based generated sentence.

For example, with the question "List the exterior doors with a width larger than 1m." the template-based generated answer is "There are 2 items of type door. These items are 'Door_6597' and 'Door_6702'." and the PLM generated answers range from "Two door width more than 1 meter, exterior." (very poor) to "There are two exterior doors with a width larger than 1 meter, 'Door_6597' and 'Door_6702'." (very good).

Table 4
Transformer-based PLM evaluation for NLG

Family	Category	Model	Quality	Typical time (s)
SpaCy LLM	encoder-only	spacy.OpenLLaMA.v1[21]	Very poor	25
Falcon	decoder-only	tiiuae/falcon-7b-instruct[22]	Very good	45
GPT-2	decoder-only	gpt2-medium[23]	Poor	15
T5	encoder-decoder	t5-base[24]	Very poor	10
Flan-T5	encoder-decoder	google/flan-t5-base[25]	Good	1

7. Conclusion and future work

Integrating multiple knowledge sources and standards becomes increasingly imperative as the construction industry progresses into the digital era. This fusion blurs traditional distinctions between domains and emphasises the necessity of interoperability to facilitate seamless information sharing across disciplines. Moreover, the advent of Digital Building Twins (DBTs) represents a pivotal advancement in modern construction processes, enabling real-time data integration and supporting multiple architectural development phases. Our research contributes to this evolving landscape by presenting an approach that enhances the accessibility and interpretability of DBT data through natural language queries. By leveraging a domain-specific ontology and advanced AI techniques, our methodology facilitates efficient communication between humans and DBTs, allowing users to extract specific building details rapidly and accurately. Our comprehensive approach, encompassing knowledge representation, semantic analysis, and information extraction, represents a significant step forward in advancing the capabilities of DBT querying systems.

Future work Several strategies can be implemented when considering future work. Firstly, grounding can be improved by setting a trust threshold to prevent systematic grounding. Secondly, the number of NLP models loaded should be limited; for example, one could be employed for constituency/dependency tree (NLU) and another for answer generation (NLG). Additionally, maintaining context appears needed when accommodating multiple questions about the same topic or object of interest. Further, extracting and sorting partial results can provide explanations or clues leading to the final answer. To broaden the scope and robustness of the system, an idea would be to expand test sets with questions sourced from practitioners

[21] https://spacy.io/usage/large-language-models
[22] https://huggingface.co/tiiuae/falcon-7b-instruct
[23] https://huggingface.co/openai-community/gpt2-medium
[24] https://huggingface.co/google-t5/t5-base
[25] https://huggingface.co/google/flan-t5-base

across diverse domains, along with generating various syntactic and semantic variations for each. Lastly, as mentioned in section 4.2.2, aligning the OB27AI ontology with more existing ontologies can contribute to its effectiveness and relevance across different domains while enhancing the interpretability of the DBT, i.e. taking advantage of the topological knowledge present in ontologies such as BOT (Building Ontology Topology) [49] or geoSPARQL[26].

Acknowledgments

We want to express our gratitude to B27-AI for their financial and material contributions to this study and to the French National Agency of Research and Technology ANRT for their CIFRE subvention. Our thanks equally go to the Faculty of Science and Technology at the University of Burgundy for their academic assistance and to the anonymous reviewers for their valuable feedback.

References

[1] A. Roxin, W. Abdou, W. Derigent, Interoperable Digital Building Twins Through Communicating Materials and Semantic BIM, SN COMPUT. SCI. 3 (2021) 23. URL: https://doi.org/10.1007/s42979-021-00860-w. doi:10.1007/s42979-021-00860-w.

[2] O. A. Al-Mufti, O. A. Al-Isawi, L. H. Amirah, C. Ghenai, Digital Twinning and ANN-based Forecasting Model for Building Energy Consumption, in: 2023 Advances in Science and Engineering Technology International Conferences (ASET), 2023, pp. 1–8. URL: https://ieeexplore.ieee.org/document/10180899. doi:10.1109/ASET56582.2023.10180899, iSSN: 2831-6878.

[3] S. Reynaud, A. Dumas, A. Roxin, Neuro-symbolic approach for querying BIM models, in: 2023 17th International Conference on Signal-Image Technology & Internet-Based Systems (SITIS), 2023, pp. 62–69. URL: https://ieeexplore.ieee.org/document/10472789. doi:10.1109/SITIS61268.2023.00019.

[4] A. Stellato, Dictionary, Thesaurus or Ontology? Disentangling Our Choices in the Semantic Web Jungle, Journal of Integrative Agriculture 11 (2012) 710–719. URL: https://www.sciencedirect.com/science/article/pii/S2095311912600604. doi:10.1016/S2095-3119(12)60060-4.

[5] ISO 23386:2020, Building information modelling and other digital processes used in construction — Methodology to describe, author and maintain properties in interconnected data dictionaries, 2020. URL: https://www.iso.org/obp/ui/en/#iso:std:iso:23386:ed-1:v1:en.

[6] ISO 16739-1:2024, Industry Foundation Classes (IFC) for data sharing in the construction and facility management industries, 2024. URL: https://www.iso.org/standard/84123.html.

[7] T. M. d. Farias, A. Roxin, C. Nicolle, A rule-based methodology to extract building model views, Automation in Construction 92 (2018) 214–229. URL: https://www.sciencedirect.com/science/article/pii/S0926580517306738. doi:10.1016/j.autcon.2018.03.035.

[8] P. Pauwels, D. Van Deursen, R. Verstraeten, J. De Roo, R. De Meyer, R. Van de Walle, J. Van Campenhout, A semantic rule checking environment for building performance

[26] https://opengeospatial.github.io/ogc-geosparql/geosparql11/geo.ttl

checking, Automation in Construction 20 (2011) 506–518. URL: https://www.sciencedirect.com/science/article/pii/S0926580510001962. doi:10.1016/j.autcon.2010.11.017.

[9] C. Zhang, J. Beetz, d. Vries, BimSPARQL: Domain-specific functional SPARQL extensions for querying RDF building data, Semantic Web (2018) 1–27. doi:10.3233/SW-180297.

[10] ISO 29481-1:2016, Building information models — Information delivery manual — Part 1: Methodology and format, 2016. URL: https://www.iso.org/obp/ui/en/#iso:std:iso:29481:-1:ed-2:v1:en.

[11] ISO 29481-3:2022, Building information models — Information delivery manual — Part 3: Data schema, 2022. URL: https://www.iso.org/obp/ui/en/#iso:std:iso:29481:-3:ed-1:v1:en.

[12] ISO/IEC 2382:2015, Information technology — Vocabulary, 2015. URL: https://www.iso.org/obp/ui/en/#iso:std:iso-iec:2382:ed-1:v2:en.

[13] H. Kubicek, R. Cimander, H. J. Scholl, Layers of Interoperability, in: Organizational Interoperability in E-Government: Lessons from 77 European Good-Practice Cases, Springer Berlin Heidelberg, Berlin, Heidelberg, 2011, pp. 85–96. URL: https://doi.org/10.1007/978-3-642-22502-4_7. doi:10.1007/978-3-642-22502-4_7.

[14] ISO/IEC 20944-1:2013, Information technology — Metadata Registries Interoperability and Bindings (MDR-IB) — Part 1: Framework, common vocabulary, and common provisions for conformance, 2013. URL: https://www.iso.org/obp/ui/en/#iso:std:iso-iec:20944:-1:ed-1:v1:en.

[15] ISO 22378:2022, Security and resilience — Authenticity, integrity and trust for products and documents — Guidelines for interoperable object identification and related authentication systems to deter counterfeiting and illicit trade, 2022. URL: https://www.iso.org/obp/ui/en/#iso:std:iso:22378:ed-1:v1:en.

[16] ISO/IEC 19941:2017, Information technology — Cloud computing — Interoperability and portability, 2017. URL: https://www.iso.org/obp/ui/en/#iso:std:iso-iec:19941:ed-1:v1:en.

[17] ISO 18308:2011, Health informatics — Requirements for an electronic health record architecture, 2011. URL: https://www.iso.org/obp/ui/en/#iso:std:iso:18308:ed-1:v1:en.

[18] D. Walasek, A. Barszcz, Analysis of the Adoption Rate of Building Information Modeling [BIM] and its Return on Investment [ROI], Procedia Engineering 172 (2017) 1227–1234. URL: https://linkinghub.elsevier.com/retrieve/pii/S1877705817306501. doi:10.1016/j.proeng.2017.02.144.

[19] S. J. Russell, P. Norvig, Artificial Intelligence: A Modern Approach, Pearson, 2021.

[20] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, I. Polosukhin, Attention is All you Need, in: I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, R. Garnett (Eds.), Advances in Neural Information Processing Systems, volume 30, Curran Associates, Inc., 2017. URL: https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf.

[21] D. Khurana, A. Koli, K. Khatter, S. Singh, Natural language processing: state of the art, current trends and challenges, Multimed Tools Appl 82 (2023) 3713–3744. URL: https://link.springer.com/10.1007/s11042-022-13428-4. doi:10.1007/s11042-022-13428-4.

[22] H. K. Skrodelis, A. Romanovs, N. Zenina, H. Gorskis, The Latest in Natural Language Generation: Trends, Tools and Applications in Industry, in: 2023 IEEE 10th Jubilee Workshop on Advances in Information, Electronic and Electrical Engineering (AIEEE), 2023, pp. 1–5. URL: https://ieeexplore.ieee.org/document/10134841. doi:10.1109/AIEEE58915.

2023.10134841, iSSN: 2689-7342.

[23] P. Pauwels, W. Terkaj, EXPRESS to OWL for construction industry: Towards a recommendable and usable ifcOWL ontology, Automation in Construction 63 (2016) 100–133. URL: https://www.sciencedirect.com/science/article/pii/S0926580515002435. doi:10.1016/j.autcon.2015.12.003.

[24] W. Terkaj, P. Pauwels, A Method to Generate a Modular ifcOWL Ontology, 2017. URL: https://www.semanticscholar.org/paper/A-Method-to-Generate-a-Modular-ifcOWL-Ontology-Terkaj-Pauwels/1c6411c9f9295dfc9503ab1a9e0055cead6804f3.

[25] T. M. Farias, A. Roxin, C. Nicolle, IfcWoD, Semantically Adapting IFC Model Relations into OWL Properties, ArXiv (2015). URL: https://www.semanticscholar.org/paper/IfcWoD%2C-Semantically-Adapting-IFC-Model-Relations-Farias-Roxin/8161912ecf67ffd6d3f94d0cb0754f77f335ce56.

[26] M. Bonduel, J. Oraskari, P. Pauwels, M. Vergauwen, R. Klein, The IFC to linked building data converter - current status, 2018. URL: https://www.semanticscholar.org/paper/The-IFC-to-linked-building-data-converter-current-Bonduel-Oraskari/945040b77455d09b3f37f975fb18ab12624982d6.

[27] P. Pauwels, A. Roxin, SimpleBIM: From full ifcOWL graphs to simplified building graphs, in: eWork and eBusiness in Architecture, Engineering and Construction: ECPPM 2016, CRC Press, 2016. Num Pages: 8.

[28] 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 for Computing in Construction, European Council on Computing in Construction, 2019, pp. 341–350. URL: http://hdl.handle.net/1854/LU-8633665. doi:10.35490/ec3.2019.166, iSSN: 2684-1150.

[29] J. Euzenat, P. Shvaiko, Ontology Matching, Springer, Berlin, Heidelberg, 2013. URL: https://link.springer.com/10.1007/978-3-642-38721-0. doi:10.1007/978-3-642-38721-0.

[30] C. T. Hoyt, A. Konotopez, C. Ebeling, PyBEL: a computational framework for Biological Expression Language, Bioinformatics 34 (2018) 703–704. URL: https://doi.org/10.1093/bioinformatics/btx660. doi:10.1093/bioinformatics/btx660.

[31] X. Xue, H. Wang, X. Zhou, G. Mao, H. Zhu, Matching heterogeneous ontologies with adaptive evolutionary algorithm, Connection Science 34 (2022) 811–828. URL: https://doi.org/10.1080/09540091.2021.1991278. doi:10.1080/09540091.2021.1991278, publisher: Taylor & Francis _eprint: https://doi.org/10.1080/09540091.2021.1991278.

[32] F. Ardjani, D. Bouchiha, M. Malki, Ontology-Alignment Techniques: Survey and Analysis, IJMECS 7 (2015) 67–78. URL: http://www.mecs-press.org/ijmecs/ijmecs-v7-n11/v7n11-8.html. doi:10.5815/ijmecs.2015.11.08.

[33] M. Gulić, I. Magdalenić, B. Vrdoljak, Ontology Matching Using TF/IDF Measure with Synonym Recognition, in: T. Skersys, R. Butleris, R. Butkiene (Eds.), Information and Software Technologies, Springer, Berlin, Heidelberg, 2013, pp. 22–33. doi:10.1007/978-3-642-41947-8_3.

[34] S. Zghal, S. Ben Yahia, E. Mephu Nguifo, Y. Slimani, SODA : Une approche structurelle pour l'alignement d'ontologies OWL-DL, in: B. D. e. B. P. Gargouri Faïez (Ed.), 1ères Journées Francophones sur les Ontologies (JFO'07), Sousse, Tunisia, 2007, pp. 1–20. URL:

https://hal.science/hal-00445724.

[35] Y. He, J. Chen, H. Dong, I. Horrocks, Exploring Large Language Models for Ontology Alignment, in: Posters and Demos ISWC 2023, Athens, Greece, 2023. URL: http://arxiv.org/abs/2309.07172, arXiv:2309.07172 [cs].

[36] J. Berant, A. Chou, R. Frostig, P. Liang, Semantic Parsing on Freebase from Question-Answer Pairs, in: D. Yarowsky, T. Baldwin, A. Korhonen, K. Livescu, S. Bethard (Eds.), Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, Association for Computational Linguistics, Seattle, Washington, USA, 2013, pp. 1533–1544. URL: https://aclanthology.org/D13-1160.

[37] A. Bordes, S. Chopra, J. Weston, Question Answering with Subgraph Embeddings, in: A. Moschitti, B. Pang, W. Daelemans (Eds.), Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), Association for Computational Linguistics, Doha, Qatar, 2014, pp. 615–620. URL: https://aclanthology.org/D14-1067. doi:10.3115/v1/D14-1067.

[38] J. Gomes, R. C. de Mello, V. Ströele, J. F. de Souza, A study of approaches to answering complex questions over knowledge bases, Knowledge and Information Systems 64 (2022) 2849–2881. URL: https://doi.org/10.1007/s10115-022-01737-x. doi:10.1007/s10115-022-01737-x.

[39] J. Zhang, B. Chen, L. Zhang, X. Ke, H. Ding, Neural, symbolic and neural-symbolic reasoning on knowledge graphs, AI Open 2 (2021) 14–35. URL: https://www.sciencedirect.com/science/article/pii/S2666651021000061. doi:https://doi.org/10.1016/j.aiopen.2021.03.001.

[40] M. Yin, L. Tang, C. Webster, S. Xu, X. Li, H. Ying, An ontology-aided, natural language-based approach for multi-constraint BIM model querying, Journal of Building Engineering 76 (2023) 107066. URL: https://linkinghub.elsevier.com/retrieve/pii/S2352710223012457. doi:10.1016/j.jobe.2023.107066.

[41] ISO 11354-1:2011, Advanced automation technologies and their applications — Requirements for establishing manufacturing enterprise process interoperability — Part 1: Framework for enterprise interoperability, 2011. URL: https://www.iso.org/obp/ui/en/#iso:std:iso:11354:-1:ed-1:v1:en.

[42] P. Hagedorn, M. Block, S. Zentgraf, K. Sigalov, M. König, Toolchains for Interoperable BIM Workflows in a Web-Based Integration Platform, Applied Sciences 12 (2022) 5959. URL: https://www.mdpi.com/2076-3417/12/12/5959. doi:10.3390/app12125959, number: 12 Publisher: Multidisciplinary Digital Publishing Institute.

[43] L. Höltgen, F. Cleve, P. Hagedorn, Implementation of an Open Web Interface for the Container-based Exchange of Linked Building Data, 2021.

[44] T. Mendes de Farias, A. Roxin, C. Nicolle, FOWLA, A Federated Architecture for Ontologies., 2015. doi:10.1007/978-3-319-21542-6.

[45] N. Noy, D. Mcguinness, Ontology Development 101: A Guide to Creating Your First Ontology, Knowledge Systems Laboratory 32 (2001).

[46] T. R. Gruber, A translation approach to portable ontology specifications, Knowledge Acquisition 5 (1993) 199–220. URL: https://www.sciencedirect.com/science/article/pii/S1042814383710083. doi:10.1006/knac.1993.1008.

[47] J. Devlin, M.-W. Chang, K. Lee, K. Toutanova, BERT: Pre-training of Deep Bidirectional

Transformers for Language Understanding, in: J. Burstein, C. Doran, T. Solorio (Eds.), Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), Association for Computational Linguistics, Minneapolis, Minnesota, 2019, pp. 4171–4186. URL: https://aclanthology.org/N19-1423. doi:10.18653/v1/N19-1423.

[48] N. Kitaev, D. Klein, Constituency Parsing with a Self-Attentive Encoder, in: Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), Association for Computational Linguistics, Melbourne, Australia, 2018, pp. 2676–2686. URL: https://www.aclweb.org/anthology/P18-1249. doi:10.18653/v1/P18-1249.

[49] M. H. Rasmussen, M. Lefrançois, G. F. Schneider, P. Pauwels, BOT: The building topology ontology of the W3C linked building data group, Semantic Web 12 (2021) 143–161. URL: https://content.iospress.com/articles/semantic-web/sw200385. doi:10.3233/SW-200385, publisher: IOS Press.

A. Test questions

To evaluate the versions of our KBQA system, we used the following list of questions:

1. Find external walls on floor 2 with heights smaller than the height of A201.
2. Search for external walls on the floor 2.
3. Select the windows with a width greater than 4m.
4. List all single swing right doors.
5. How many rooms are in this house?
6. Identify all the beams with a span larger than 5m.
7. Find all the doors with a height greater than 2.10 m.
8. Locate the walls in brick.
9. Search for the slabs in wood.
10. Select the footing at ground 0.
11. Find all spaces connected to B201 and contained in floor 2.
12. Select all stairs on level 1.
13. Find the area of A103.
14. Are there load-bearing roofs?
15. Locate walls whose width is greater than 0.400m.
16. List windows whose sill height is greater than 1.5m.
17. Find the room that has the largest area.
18. Search for all the load-bearing beams with slope equal to 0.
19. Select the windows on the roof.
20. Locate all the stairs.
21. List the exterior doors with a width larger than 1m.
22. Find the doors with automatic operation.