2023 European Conference on Computing in Construction
40th International CIB W78 Conference
Heraklion, Crete, Greece
July 10-12, 2023

AUTOMATED GENERATION OF SPARQL QUERIES FROM SEMANTIC MARK-UP

Nicholas Nisbet[1 2], Zijing Zhang[1], Ling Ma[1]
[1]University College London, London, UK,
[2]AEC3

Abstract

Regulations and requirement documents contain normative knowledge that needs to be compared against actual and proposed built assets, if safety and social expectations are to be met. Previous work has shown that semantic mark-up of normative documents can be consumed directly by a rule-engine or can be automatically transformed to a number of existing rule representations. This work investigates the feasibility of automatically transforming examples of normative documents into SPARQL and testing the result against typical building information models. The desirability of using SPARQL is discussed.

Introduction

Regulations and requirement documents contain normative knowledge that needs to be compared against actual and proposed built assets, if safety and social expectations are to be met. Requirements, Applications, Selections and Exceptions (RASE) (Nisbet et al., 2008) is a method of semantic markup up of plain text based on the identification on the semantic and logical roles of phrases and sections, from which operable knowledge can then be deduced. It was initially developed to support knowledge capture for normative compliance checking.

The RASE method addresses three kinds of knowledge. The initial motivation for its development has been to capture normative content, as found in laws, regulations and requirement documents. Subsequently it has also been used to capture definitive knowledge as found in dictionaries, classifications and formulae. A third area of application is to capture the knowledge found in descriptive and narrative content as found in written and database representations of the built environment. In some cases, this content may be ‘fictitious’ when it describes a plausible future state of the built environment. For normative knowledge, each objective and metric are typed as one of four types: Requirement, Applicability, Selection or Exception. Instead of Requirement metrics, declarative knowledge uses Register metrics and descriptive/narrative knowledge uses Reported metrics. (Nisbet et al., 2022b). A semantic dictionary can be used to map between normative terminology and the descriptive/narrative terminology. This ensures that there is no requirement for BIM content to be restructured to reflect the regulatory content nor for the regulatory content to be expressed in the vocabulary of a Building Information Model (BIM), as proposed by Soliman-Junior et al. (2022).

The RASE mark-up is applied to text and tables in documents using four colours which highlight four distinct roles for phrases and sections. The four roles are Requirements/registers/reports, Applications, Selections and Exceptions. This identifies the RASE knowledge ontology as a recursive tree structure of objectives and metrics, where objectives contain metrics and other objectives. Metrics are simple atomic queries that can be evaluated, by domain experts or by enquiring of a target domain model. The method is able to capture the logical meaning of an entire document, section, paragraph or clause, not just individual sentences. This makes RASE mark-up particularly relevant to automated code compliance checking, where regulations are tested against BIM information to detect non-compliant aspects. SPARQL is a query language that can be used to retrieve information from semantic web resources, such as graph-based BIM models. By automatically converting regulatory clauses from normative documents into SPARQL queries, graph-based BIM models can be validated for compliance checking. This approach can help ensure that BIM models are compliant with relevant regulations and standards, and can ultimately improve the quality of the models and ultimately of the built environment. Semantic mark-up is being deployed in a wide range of research and industry scenarios. “AEC3 Require1” consumes semantic mark-up directly, without the use of any intermediary rule language (Nisbet et al., 2022b). In the UK, ‘RegBIM’ (Beach et al., 2013) and the Digital Compliance ‘DCOM’ project (DCOM, 2023), semantic mark-up was added by regulatory domain subject matter experts. The semantic markup was then transformed into Java Drools, compiled and evaluated against a substantial building information model representing a 300-pupil secondary school.

The sematic web and in particular RDF/OWL has been labeled as the gold standard of ‘description logics’ (Crotts, 2022; Kostylev et al, 2015). It is increasingly popular for construction research, as existing building information models can be transformed using online tools implementing mixtures of ifcOWL (Pauwels et al, 2016) and other simplified ontologies such as BOT (Rasmussen et al, 2021).

Simple Protocol and RDF Query Language (SPARQL, 2023) was chosen in preference to other semantic query syntaxes for RDF/OWL such as Shape Expressions (SHEX, 2023) because SPARQL has support for logical operators. SPARQL can be embedded in Shapes Constraint Language (SHACL, 2023), effectively using a query to validate an information set.

There has been previous work on automated generation of SPARQL queries. Some have been based on limited source domains using keywords (Im et al., 2014) or controlled templates (Shekarpour et al., 2013), both of which necessarily involve a degree of pre-structuring. Jung et al. (2020) has investigated the use of Natural Language Processing (NLP) analysis to generate queries but this approach has not generated the confidence in the results appropriate to regulatory or requirements enforcement tasks.

This work is aimed at identifying the feasibility of the automated generation of SPARQL queries from semantic mark-up. It is a further exploration of the feasibility of representing potentially complex logical queries arising from semantic markup in another query language, without resorting to handcrafting of the query or of the target domain either of which may undermine the generality of the solution.

Semantic web technologies including RDF/OWL (RDF/OWL, 2023) represent knowledge as triples, typically described as subject, predicate (relationship or property) and object, where both subject and object can be primitive values or references to other triples. Hence RDF/OWL triples are in RASE terms a combination of an Application (the subject) and a Requirement (or Register or Report) in the predicate and object.

Method

The research objective is to explore whether SPARQL can be generated systematically from normative documents with the gain in accuracy and completeness. This research established specific criteria arising from the objective of creating such a transparent and fully automated process. The automated process is detailed below in the Artifact section. The mapping was refined repeatedly in the Experiments described below until the four criteria had been met. The criteria for a solution are:

a. A SPARQL query should be generated in a layout that supports review and comment, even though RASE semantic mark-up is proposed as the primary format for review prior to the mapping to SPARQL. Clarity of the resulting SPARQL query is intended to encourage endorsement and improvement of the transformation process.
b. The mapping of RASE to SPARQL should produce a result that be used directly without further editing or revision. This is necessary to meet the objective of 'automation' and to eliminate the deployment of tacit 'craft' knowledge as seen in exercises focused on a single regulation clause such as Pauwels et al's (2015) investigation into an acoustic regulation.
c. The rules within the mapping should be expressed explicitly so that refinements can be proposed and adopted. The mapping should, if possible, be executed using a single pass.
d. The SPARQL query should be executed and give reviewable results. Beyond producing a result, execution performance was not a criterion.

A valid SPARQL query must declare the target ontologies, narrow the target model, filter the selected objects and report them. The generated query is made of a WHERE statement to identify the relevant information, a BIND statement to generate a logical result and an optional FILTER statement to select only 'True' or 'False' results. SPARQL therefore presents a number of challenges compared to the RASE knowledge ontology.

Experiments

Rasmussen et al (2021) developed a semantic BIM interface which is freely available for use (LD-BIM, 2023). This accepts IFC and generates a simplified RDF/OWL representation. The visual interface displays the geometry of the objects found in the IFC. Objects identified by SPARQL queries can be highlighted alongside the tabulated results. The application offers a number of simple SPARQL queries but the interface also allows for any query to be composed or copy-and-pasted in, validated and executed.To explore the feasibility of a RASE to SPARQL mapping and to discover any limitations, two experiments were conducted. The first addressed a simple case to confirm the design principles around creating a generic transformation and the second then sought to scale the solution to considering a complete and complex regulatory clause, and assess whether a satisficing approach had been found. The second experiment includes combinations of many types of objectives and metrics, and a variety of measure types within the metrics, such as logical, textual and numeric tests.

Experiment 1

A simple sentence "Doors should be at least 850mm wide" was given RASE semantic mark-up. Figures 1 and 2 show the markup and an automatically derived concept graph.

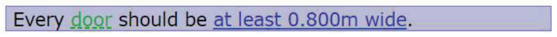

Every door should be at least 0.800m wide.

Figure 1 : Simple RASE semantic mark-up

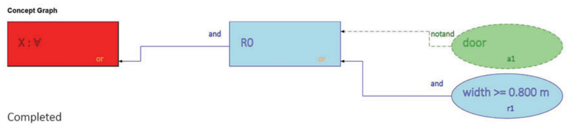


Figure 2: Automatically generated concept graph

Previous mappings developed for concept graphs, predicate logic and other representations have used XSLT applied to the source HTML (Nisbet et al., 2022). In each case a recursive depth-first tree traversal algorithm is used. This algorithm generates and reacts to specific events such as 'drop down a level of the tree hierarchy' 'return up a level' or 'consider next item on this level' as they are discovered within the semantic markup of the HTML.

```
# Example 1: Doors should be at least 0.800m wide

PREFIX xsd:  <http://www.w3.org/2001/XMLSchema#>                    #1
PREFIX ifc:  <http://ifcowl.openbimstandards.org/IFC2X3_Final#>
PREFIX bot:  <https://w3id.org/bot#>
PREFIX inst: <https://web-bim/resources/>

SELECT DISTINCT ?this ?result  WHERE {                              #2 #3
     ?this a bot:Element .                                          #4
     OPTIONAL {?this inst:widthPSetRevitTypeDimensions ?r1 } .      #5 #6
BIND ( (
     ( ! ( EXISTS{ ?this a ifc:IfcDoor } )                          #7 #8
     ||  ( ?r1 >= '0.800'^^xsd:double ))                            #9
     ) AS ?result) .
FILTER ( ?result != 'true'^^xsd:boolean) .                          #10
}
ORDER BY ?this
```

Figure 3: Automatically generated SPARQL query

The varying part of the query template is highlighted in bold in Figure 3. The remainder of the text of the query is invariant. This highlighting shows that there are two bodies of varying text, so it is necessary to make two passes through the normative source, the first to define the explicit variables and a second to generate the logical statement bound to the result.

Several challenges arise in generating a SPARQL query. These issues are noted as comments such as '#1' Each resolution was then included in an automated process.

1. The appropriate header identifying the selected ontologies must be provided. The LD-BIM interface presents the necessary references and the transformation reproduces them directly.

2. In order to associate any results with the graphical interface, the 'this' reference must be present in the results, as this is used by the RDF/OWL model, by any encasing SHACL statement and by the visual graphics.

3. A decision must be made as to what other values are reported. Whilst further specific values, such as 'type' and 'width', can be reported during development, in the general case there could be a very large number of relevant properties. Once operational, there is no need to report any of these additional properties. So only 'this' and 'result' are included by default.

4. Normative judgements are made against identifiable objects which may have a number of significant attributes. Hence the target realm of 'objects' must be distinguished from other triples containing 'relationships' or 'property assignments'. This is achieved by selecting only the 'bot:Element' type.

5. The connection must be made between terms in the normative metrics and terms in the descriptive model. RASE offers two means to achieve this: Any metric can have the determining property, comparator and target recorded explicitly or reference can be made to a semantic dictionary to obtain a mapping. In this case a proprietary attribute is used for 'width'. The ifcOWL mapping has suppressed the standard IFC attribute 'OverallWidth', so a proprietary property name is used instead namely 'widthPSetRevitTypeDimensions'.

6. Variables must be defined for any non-type attributes. This is done in the selection body because the syntax for the BIND structure does not support the naming of specific attributes. RASE markup provides a readily available identifier for the metric that can be expected to be unique. These values are marked as 'OPTIONAL' so as to postpone any impacts of undefined properties on the handling of issues related to Closed or Open World Assumptions (CWA/OWA) until the evaluation of the result in the BIND statement.

7. Type definition in RDF/OWL is a special property, often with multiple inheritance being represented in multiple values in the 'type array'. This means that 'type' must be identified as a special case since it has a unique syntax. It is important not to assume that 'type' is always present, and even if present it may be part of a selection of a number of different types. In this case the concept of 'door' exists in both the normative domain and in the descriptive domain. However, in the descriptive domain 'doorness' is a special attribute embedded in the type definition 'IfcDoor'.

8. The logical structure of the query must be represented. The logical structure of the sample regulation is derived from interpreting the semantic role for objectives and metrics, found as Requirement, Applicability, Selection and Exception, into the mathematical primitives AND '&&', OR '||', and NOT '!'. This logical structure is assigned to a variable called 'result' using the BIND construct. In a production environment this could be a more descriptive 'clause_X_result', especially if the results are to be 'INSERT'ed or 'CONSTRUCT'ed, giving the results persistence in the triple store.

9. The comparator operator found in any metric must be represented. The semantic dictionary can aid the mapping of phrases such as 'at least' to the equivalent mathematical symbol '>='. Since the units have been adjusted to SI in the target model, metrics based on millimetres need to be factored.

10. A decision must be made as to whether the instances evaluating to 'True' or 'False' are to be returned for reporting. In general, we expect normative evaluation

to generate 'true' for passing and 'false' for failing, but the implications of 'true' or 'false' outcomes may be more subtle in an adversarial context such as a dispute between a 'conservationist' and a 'developer'. The FILTER syntax is used to make the choice as explicit as possible. Here, the query is used to find the hopefully relatively small number of non-compliant elements.

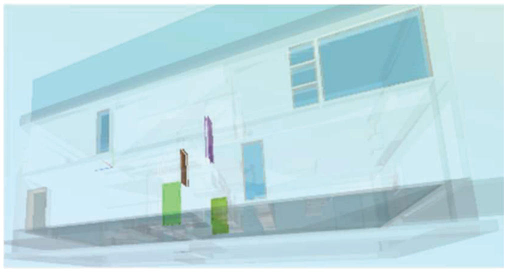


Figure 4: LD-BIM highlighting the four 'non-compliant' doors

Figure 4 shows that the generated SPARQL query is valid, and has highlighted four 'non-compliant' doors. The query was then tested further in other cases including when information is unknown. Comparisons were made of the expected truth table and the outcomes reported by the LD-BIM interface in Table 1.

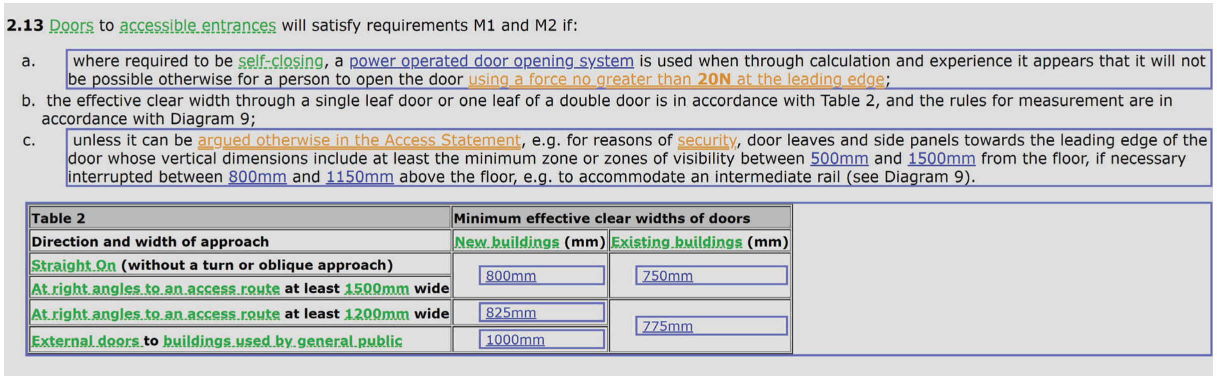
Table 1: Expected and SPARQL results for nine test cases ** indicates 'Query returned no results'*

Case	Is a Door	Width > 0.800m	Expected Result	Actual result
1	True	True	True	True
2	True	Unknown	Unknown	*
3	True	False	False	False
4	Unknown	True	True	*
5	Unknown	Unknown	Unknown	*
6	Unknown	False	Unknown	*
7	False	True	True	True
8	False	Unknown	True	*
9	False	True	True	True

Table 1 shows that in all straightforward cases (1,3,7 and 9) the correct outcomes are obtained, using the default Duplex model as an example. By modifying the query or the model, scenarios were created and tested where one or other of the two metrics are 'Unknown'. Misleading results are produced in these cases (2,4,5,6 and 8). Case '8' is of particular concern where a non-door should evaluate to 'true' even if its 'width' is unknown. This is an example where 'short circuit 'McCarthy' execution' is necessary, if descriptive models are not be overloaded with unnecessary and irrelevant attributes, purely to enable the execution algorithms within SPARQL.

Experiment 2

The semantic markup from a substantial clause of the UK Building Regulation Approved Document M (ADM, 2023) was used. This includes several requirements and a table of expected widths. Figure 5 shows the marked-up document.



2.13 Doors to accessible entrances will satisfy requirements M1 and M2 if:

a. where required to be self-closing, a power operated door opening system is used when through calculation and experience it appears that it will not be possible otherwise for a person to open the door using a force no greater than **20N** at the leading edge;

b. the effective clear width through a single leaf door or one leaf of a double door is in accordance with Table 2, and the rules for measurement are in accordance with Diagram 9;

c. unless it can be argued otherwise in the Access Statement, e.g. for reasons of security, door leaves and side panels towards the leading edge of the door whose vertical dimensions include at least the minimum zone or zones of visibility between 500mm and 1500mm from the floor, if necessary interrupted between 800mm and 1150mm above the floor, e.g. to accommodate an intermediate rail (see Diagram 9).

Table 2	**Minimum effective clear widths of doors**	
Direction and width of approach	**New buildings (mm)**	**Existing buildings (mm)**
Straight On (without a turn or oblique approach)	800mm	750mm
At right angles to an access route at least 1500mm wide		
At right angles to an access route at least 1200mm wide	825mm	775mm
External doors to buildings used by general public	1000mm	

Figure 5: Regulatory clause with RASE semantic mark-up

The transformation scaled successfully to generate the query, shown abbreviated in Figure 6, using twenty-five metrics and thirty logical checks.

```
# Example 2: UK Government ENG
# Approved Document M2 clause 2.13 (2015)
PREFIX xsd:  <http://www.w3.org/2001/XMLSchema#>
PREFIX ifc:
<http://ifcowl.openbimstandards.org/IFC2X3_Final#>
PREFIX bot:  <https://w3id.org/bot#>
PREFIX inst: <https://web-bim/resources/>

SELECT DISTINCT ?this ?result  WHERE {
     ?this a bot:Element .
OPTIONAL{ ?this
inst:partOfAccessibleEntranceUKDoorcapture ?a01 }  .
#    several further measures
OPTIONAL{ ?this inst:clearWidthUKDoorcapture ?r09 }  .
BIND ( (
     ! ( EXISTS{ ?this a ifc:IfcDoor  } &&   ( ?a01 ))
||
#    several further comparisons and logical tests
     ( ?r09 >= '1.000'^^xsd:double )))))
) AS ?result) .
FILTER ( ?result != 'true'^^xsd:boolean) .
}
ORDER BY ?this
```

Figure 6: SPARQL query (part)

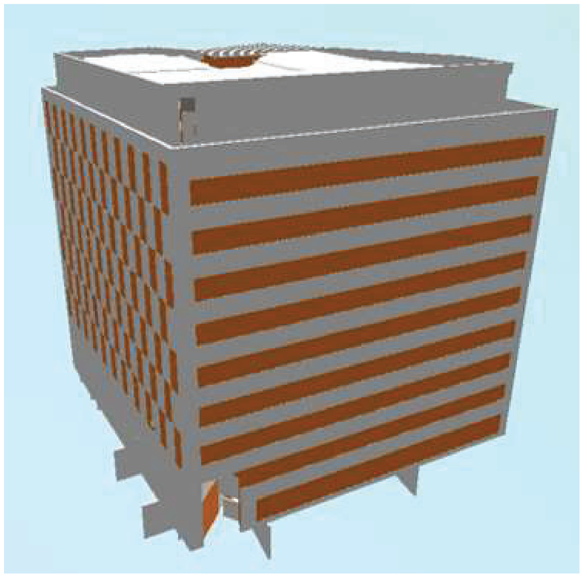


Figure 7: Office building concept (courtesy of Mace plc)

A further issue arose in that the example regulation requires wider entrances in new-build buildings as against the refurbishment of existing buildings. The 'is new' property is assigned to 'true' on the overall building 'IfcBuilding' entity. This prevents the query which is focused on the door element from generating a result. Figure 7shows the test case where the main entrance of a proposed office building is too narrow, but the SPARQL query is unable to generate a result because the 'is new' property is unknown. This is discussed further below.

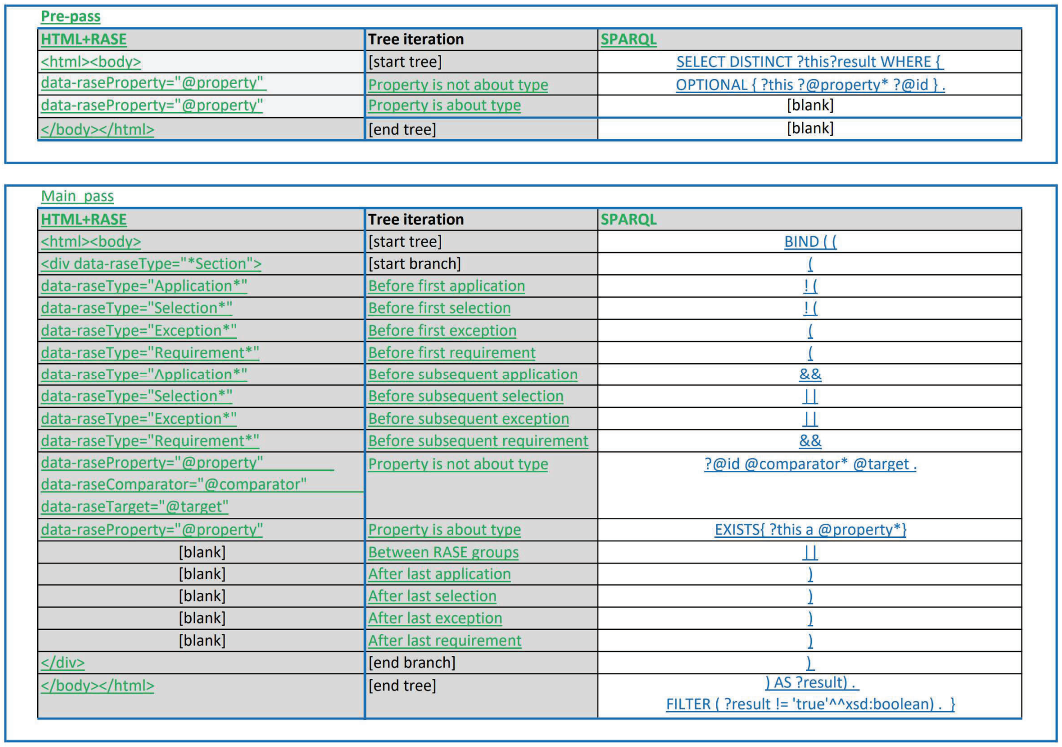
Artifact

The mapping rules using HTML/RASE, depth-first tree traversal events and SPARQL syntax were tabulated to explicitly capture the mapping process for review and to control the mapping process (Figure 8). Further columns (not shown) hold the further mappings for propositional logic, predicate logic, and other knowledge schemas.
Green (with dotted underline) indicates tests for Application. In this case all cells containing Applications are 'header' or 'sider' cells. Blue (with solid underline) indicates defined output. Each pass is represented by a separate Requirement Section, indicated as a blue box (in solid line).

An example rule extracted from the table might be that on a 'main pass', with 'SPARQL' expected and on encountering the event 'before a subsequent exception' the required output is '||'. SPARQL is exceptional in that its syntax treats tests about 'type' differently from all other property values, requiring additional rule rows in the table. Tests about 'type' are detected by examining a separate dictionary entry for the term 'door' where its representation in SPARQL is the type specification 'ifc:IfcDoor'.

Example execution

The input to the process is an HTML document with RASE markup highlighting a normative knowledge, along with the HTML document with RASE mark-up highlighting the mapping rules. Figure 1 shows a simple example with a single Application and a single Requirement. Performing the depth-first tree traversal detects 14 events in sequence in the HTML/RASE and responds to each of these with appropriate SPARQL output, as shown in Figure 9. Terms found in the normative document are translated using a separate definitive dictionary handling the English language context, the IFC context and the ifcOWL/BOT context. The resulting valid SPARQL output was already shown in Figure 4.



Pre-pass

HTML+RASE	Tree iteration	SPARQL
<html><body>	[start tree]	SELECT DISTINCT ?this?result WHERE {
data-raseProperty="@property"	Property is not about type	OPTIONAL { ?this ?@property* ?@id } .
data-raseProperty="@property"	Property is about type	[blank]
</body></html>	[end tree]	[blank]

Main pass

HTML+RASE	Tree iteration	SPARQL
<html><body>	[start tree]	BIND ((
<div data-raseType="*Section">	[start branch]	(
data-raseType="Application*"	Before first application	! (
data-raseType="Selection*"	Before first selection	! (
data-raseType="Exception*"	Before first exception	(
data-raseType="Requirement*"	Before first requirement	(
data-raseType="Application*"	Before subsequent application	&&
data-raseType="Selection*"	Before subsequent selection	||
data-raseType="Exception*"	Before subsequent exception	||
data-raseType="Requirement*"	Before subsequent requirement	&&
data-raseProperty="@property" data-raseComparator="@comparator" data-raseTarget="@target"	Property is not about type	?@id @comparator* @target .
data-raseProperty="@property"	Property is about type	EXISTS{ ?this a @property*}
[blank]	Between RASE groups	||
[blank]	After last application	)
[blank]	After last selection	)
[blank]	After last exception	)
[blank]	After last requirement	)
</div>	[end branch]	)
</body></html>	[end tree]	) AS ?result) . FILTER (?result != 'true'^^xsd:boolean) . }

Figure 8: Definitive mapping knowledge

HTML/RASE event	Tree traversal event	SPARQL output
<body>	Pre-pass (start tree traversal)	SELECT DISTINCT ?this WHERE { ?this a bot:Element .
<span id="**r1**" data-raseProperty="**width**">	Property not about type (term is translated)	OPTIONAL {?this **inst:widthPSetRevitTypeDimensions** ?**r1** }
</body>	Pre-pass (end tree traversal)	
<body>	Main pass (start tree traversal)	BIND ((
<div data-raseType ="RequirementSection">	(start branch)	(
	Before first Application	!(
<span data-raseProperty="**door**">	Property about type (term is translated)	EXISTS{ ?this a **ifc:IfcDoor** }
	After last Application	)
	Between RASE groups	||
	Before first Requirement	(
<span id="**r1**" data-raseComparator="**>=**" data-raseTarget="**0.800**">	Property not about type	(?**r1** **>=** '**0.800**'^^xsd:double)
	After last Requirement	)
</div>	(end branch)	)
</body>	(end tree traversal)	) AS ?result) . FILTER (?result != 'true'^^xsd:boolean) . } ORDER BY ?this

Figure 9: Example execution steps. (Bold text is specific to the example.)

Analysis and Discussion

RASE mark-up has been seen to be significantly more productive than conventional coding (Beach et al., 2013; DCOM, 2023). Example 1 illustrates this using a simple case, with only one Application and one Requirement, and yet the syntax of the SPARQL query and in particular the BIND construct shows that its correctness is not obvious. Before SPARQL and RDF/OWL can be considered as a candidate method, it is necessary to demonstrate that the generality of regulations including both the variety of logic constructs and the variety of metric types can be mapped to SPARQL without for example anticipating the nature of the queries by configuring the target database. Example 2 was chosen as a substantial case. It includes metrics representing multiple Requirements, Applications, Selections and Exceptions with implications for the complexity of the BIND statement. It also uses a range of parameter types including ontological, text, logical and numeric constructs.

Three issues were encountered in these experiments. The first is the impact arising from the decisions that substantial parts of the IFC schema have been omitted from the published mappings of IFC to RDF/OWL. This omits the attributes containing the 'Overall Width' of door and window entities. These choices can be remedied by revisiting the implemented mappings or falling back on non-standard and proprietary attributes handled in the semantic dictionary.

The second is considered to be potentially more serious. Previous work such as a UK Digital Compliance project (DCOM, 2023) has indicated that there may well be several thousand metrics in the UK Approved Documents, representing Requirements, Applicability's, Selections and Exceptions. It may be impractical to ensure that a descriptive model is complete before it is assessed. In this context, the 'Closed World Assumption' (Minker, 1982) can be summarized as 'any information not known to be true is taken as being false', essentially a two-value logic, whereas the Open World Assumption can be summarised as 'any information not known to be true is taken as being unknown', essentially a three-valued logic. The CWA embedded in the SPARQL language may be a hazard to the proper enforcement of legally prescribed minimum performance. No results are generated in five of the nine test cases examined in experiment 1. The CWA may be tilted onto the side of caution when considering requirements such as 'minimum door width' but is flawed when we consider that the larger part of normative text is made up of applicability's such as 'is a door'. In this case not knowing if an entity is considered to be a door leads to the width requirement not be tested.

The third issue relates to the understanding of objects in context. Facilities are managed whenever possible through the introduction of intermediate specifications such as product and space types, or intermediate groupings such as zones and systems. Many attributes describing the context are assigned to these or to the overall site, building or project objects. It may be that SPARQL queries and the 'path' constructs can affect the arbitrary number of recursive steps from an element back to the higher levels to collect this information. It is not clear how such path syntax or any 'semantic enhancement' - effectively de-normalization - could know which properties should be disseminated downwards. The implication is that the appropriate search path for a particular attribute will have to be stored in the dictionary. Such search paths are only available in SPARQL v1.1 onwards.

Limitations

The authors anticipate that the experiments can be extended and the solution further refined. For example, the work has been undertaken with no particular regard for the performance optimization of the combination of SPARQL queries, target ontologies, and the chosen demonstration platform. Pauwels et al (2016) indicates that without a process of refinement of the tools, schema and BIM model size, the performance of SW tools can be disappointing or prohibitive. Additionally, the examples have not considered the complexity of unit conversion and of predicate logic relationships, which is the subject of further work.

Conclusions

The process has successfully generated queries from textual and tabular regulations involving several dozen properties and logical relationships.

SPARQL queries can be automatically generated from RASE semantic mark-up with correct syntax and content, and that this outcome is not dependent on the length or complexity of the source normative regulation.

The mapping between knowledge representations can be held in a table which can itself be made machine operable with HTML/RASE semantic markup. This shows that the mapping can be exposed for discussion and that the craft skills around SPARQL development can be eliminated and the potential for error reduced.

However, the first example has demonstrated that the execution of automated regulatory compliance checking should use rule-engines that handle logical (three-value) outcomes correctly. However, this issue cannot be solved when using SPARQL.

References

ADM: UK Building Regulations Approved Document M https://www.gov.uk/government/publications/access-to-and-use-of-buildings-approved-document-m accessed Jan 2023

Beach, T.H., Kasim, T., Li, H., Nisbet, N. and Rezgui, Y., 2013, August. Towards automated compliance checking in the construction industry. In International Conference on Database and Expert Systems Applications (pp. 366-380). Springer, Berlin, Heidelberg.

CSS: https://www.w3.org/Style/CSS/Overview.en.html Accessed Jan 2023

Crotts, Larry Joshua., 2022, Construction and Evaluation of a Gold Standard Syntax for Formal Logic Formulas and Systems. The University of North Carolina at Greensboro ProQuest Dissertations Publishing, 2022. 29065468.

DCOM: https://www.dcom.org.uk/wp-content/uploads/2022/03/The-Digital-Compliance-Ecosystem_260222.pdf accessed Jan 2023

HTML5: https://www.w3.org/TR/2008/WD-html5-20080122/ Accessed Jan 2023

Im, S., Sohn, M. , Jeong, S. and Lee, H. J. , 2014, "Keyword-Based SPARQL Query Generation System to Improve Semantic Tractability on LOD Cloud," 2014 Eighth International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing, Birmingham, UK, pp. 102-109, doi: 10.1109/IMIS.2014.95.

Jung, H., Kim, W. Automated conversion from natural language query to SPARQL query. J Intell Inf Syst 55, 501–520 (2020). https://doi.org/10.1007/s10844-019-00589-2

Kostylev, E.V., Reutter, J.L., Romero, M. and Vrgoč, D., 2015. SPARQL with property paths. In The Semantic Web-ISWC 2015: 14th International Semantic Web Conference, Bethlehem, PA, USA, October 11-15, 2015, Proceedings, Part I 14 (pp. 3-18). Springer International Publishing.

LD-BIM: https://ld-bim.web.app/ Accessed Jan 2023

Minker, J., 1982. On indefinite databases and the closed world assumption. In 6th Conference on Automated Deduction: New York, USA, June 7–9, 1982 6 (pp. 292-308). Springer Berlin Heidelberg.

Nisbet, N. and Ma, L., 2022a. Nisbet, N. and Ma, L., 2022, July. Presentations of rase knowledge mark-up. In EC3 Conference 2022 (Vol. 3). University of Turin.

Nisbet, N., Ma, L. 2022. Using RASE to represent normative, definitive and descriptive knowledge. eWork and eBusiness in Architecture, Engineering and Construction: ECPPM 2022 Editors: Eilif Hjelseth, Sujesh F. Sujan & Raimar Scherer. Publisher: CRC Press.

Nisbet N, Wix J and Conover D. 2008. "The future of virtual construction and regulation checking", in Brandon, P., Kocaturk, T. (Eds), Virtual Futures for Design, Construction and Procurement, Blackwell, Oxfordshire. doi: 10.1002/9781444302349.ch17.

Pauwels, P., Van Deursen, D., Verstraeten, R., De Roo, J., De Meyer, R., Van de Walle, R. and Van Campenhout, J., 2011. A semantic rule checking environment for building performance checking. Automation in construction, 20(5), pp.506-518.

Pauwels, P. and Terkaj, W., 2016. EXPRESS to OWL for construction industry: Towards a recommendable and usable ifcOWL ontology. Automation in construction, 63, pp.100-133.

Rasmussen, M.H., Lefrançois, M., Schneider, G.F. and Pauwels, P., 2021. BOT: The building topology ontology of the W3C linked building data group. Semantic Web, 12(1), pp.143-161.

RDF/OWL: https://www.w3.org/OWL/ . Accessed Jan 2023

Shekarpour, S., Auer, S., Ngonga Ngomo, A.C., Gerber, D., Hellmann, S. and Stadler, C., 2013. Generating SPARQL queries using templates. Web Intelligence and Agent Systems: An International Journal, 11(3), pp.283-295.

SHEX: https://shex.io/ Accessed Jan 2023

SHACL: https://www.w3.org/TR/shacl/ Accessed Jan 2023

Soliman-Junior, J., Tzortzopoulos, P., and Kagioglou, M. (2022). "Designers' perspective on the use of automation to support regulatory compliance in healthcare building projects." Construction Management and Economics, 40(2), 123-141.

SPARQL: https://www.w3.org/TR/rdf-sparql-query/ . Accessed Jan 2023

Zhang, Z., Nisbet, N., Ma, L. and Broyd, T., 2022, October. A multi-representation method of building rules for automatic code compliance checking. In: Proceedings of the European Conference on Product and Process Modeling 2022. Trondheim, Norway.