Advanced Engineering Informatics 33 (2017) 68–88

Contents lists available at ScienceDirect

Advanced Engineering Informatics

journal homepage: www.elsevier.com/locate/aei

Full length article

A performance benchmark over semantic rule checking approaches in construction industry

Pieter Pauwels [a,*], Tarcisio Mendes de Farias [c,e], Chi Zhang [b], Ana Roxin [c], Jakob Beetz [b], Jos De Roo [d], Christophe Nicolle [c]

[a] Department of Architecture and Urban Planning, Ghent University, J. Plateaustraat 22, B-9000 Ghent, Belgium
[b] Department of the Built Environment, Eindhoven University of Technology, P.O. Box 513, NL-5600 MB Eindhoven, The Netherlands
[c] Laboratory LE2I, CNRS, Arts et Métiers, Univ. Bourgogne Franche-Comté (UBFC), Dijon, France
[d] Agfa HealthCare NV, Moutstraat 100, B-9000 Ghent, Belgium
[e] Department of Ecology and Evolution, University of Lausanne, Biophore, CH-1015 Lausanne, Switzerland

ARTICLE INFO

Article history:
Received 9 February 2016
Received in revised form 24 March 2017
Accepted 3 May 2017
Available online 18 May 2017

Keywords:
ifcOWL
Rule checking
Linked data
Reasoning
Semantic web
Benchmark

ABSTRACT

As more and more architectural design and construction data is represented using the Resource Description Framework (RDF) data model, it makes sense to take advantage of the logical basis of RDF and implement a semantic rule checking process as it is currently not available in the architectural design and construction industry. The argument for such a semantic rule checking process has been made a number of times by now. However, there are a number of strategies and approaches that can be followed regarding the realization of such a rule checking process, even when limiting to the use of semantic web technologies. In this article, we compare three reference rule checking approaches that have been reported earlier for semantic rule checking in the domain of architecture, engineering and construction (AEC). Each of these approaches has its advantages and disadvantages. A criterion that is tremendously important to allow adoption and uptake of such semantic rule checking approaches, is *performance*. Hence, this article provides an overview of our collaborative test results in order to obtain a performance benchmark for these approaches. In addition to the benchmark, a documentation of the actual rule checking approaches is discussed. Furthermore, we give an indication of the main features and decisions that impact performance for each of these three approaches, so that system developers in the construction industry can make an informed choice when deciding for one of the documented rule checking approaches.

1. Introduction

1.1. Rule checking in construction industry: application scenarios

Rule checking of building models is one of the key features required for many applications in the domain of architectural design and construction. A large share of the rule checking approaches is located in the realm of building performance checking and regulation compliance checking. Throughout the building life cycle, designs have to be checked for compliance to a vast number of different rules and constraints on international, national, local and even company-specific levels. Rule checking is often present in other application scenarios as well, including automatic query rewriting, building model conversion and subset selection.

With the advent of Building Information Modelling (BIM) tools [1], fundamentally new processes evolve that allow building information to be managed at any point in time. As acknowledged by Eastman et al. [2], more advanced BIM-based rule checking approaches are within reach as a result of this trend. Automated rule checking is defined by Eastman et al. [2] as *"software that does not modify a building design, but rather assesses a design on the basis of the configuration of objects, their relations or attributes*". According to Eastman et al. [2], who refer to the early works by Han et al. [3–5], rule-based systems are understood as systems that *"apply rules, constraints or conditions to a proposed design, with results such as 'pass', 'fail' or 'warning', or 'unknown'.*". So, a rule-based system in construction industry includes at least two critical elements: the design model and the rules.

* Corresponding author.
E-mail addresses: pipauwel.pauwels@ugent.be (P. Pauwels), tarcisio.mendesdefarias@unil.ch (T.M. de Farias), c.zhang@tue.nl (C. Zhang), ana-maria.roxin@u-bourgogne.fr (A. Roxin), j.beetz@bwk.tue.nl (J. Beetz), jos.deroo@agfa.com (J. De Roo), cnicolle@u-bourgogne.fr (C. Nicolle).

http://dx.doi.org/10.1016/j.aei.2017.05.001
1474-0346/

In the AEC industry, the BIM model is typically considered to be the preferred *design model* to start from. Many initiatives that start from such a BIM model furthermore start from a neutral representation of the building model, typically captured in the Industry Foundation Classes (IFC) [6,7], which is developed and maintained by the BuildingSMART organization [8]. The original representation of IFC in EXPRESS is very closely affiliated to a class structure in any programming language or database. Hence, rule checking has typically been implemented in hard-coded rules.

Recently, however, more and more architectural design and construction data is now also represented in the Resource Description Framework (RDF) data model [9], possibly referencing statements and concepts in the Web Ontology Language (OWL2) [10]. The underlying logical basis of OWL can promote the realization of the rule checking process outlined by Eastman et al. [2] using this additional logical basis. Also [11] indicated the usefulness of a logical basis in regulation compliance checking, as early as 2003, which was at that time implemented as an addition to plain XML. The same argument for a semantic or logical basis is also made in Hjelseth and Nisbet [12].

The *rules* come in different forms and shapes. It is certainly not our intention to discuss all these rule representations here, but we can outline a few. For the rule representation forms, Eastman et al. [2] presents three different options, namely:

1. using computer language encoded rules,
2. using parametric tables, and
3. language-driven

Of these three, especially the latter is interesting, as a language-driven approach is particularly good in providing extensibility of the rule set. As long as one can represent rules in the specific rule language that is used, one can supply the system with more rules in an on-demand fashion, which is not possible (or only to a limited extent) when using computer language encoded rules or parametric tables. Eastman et al. [2] further divides the language-driven implementation methods into two alternatives: either as a logic-based language or as a domain-oriented language. An example of the latter is the Building Environment Rule and Analysis (BERA) language that is proposed in Lee [13] and Lee et al. [14]. Another example is the rule checking approach proposed in Solihin and Eastman [15] that relies on Conceptual Graphs (CG), which are grounded in First Order Logic (FOL). This approach is an example of a predicate logic-based language for rule checking in construction industry. With its logical basis in Description Logics (DL) [16], the semantic rule checking approach as proposed in Pauwels et al. [17] is another example of such a logic-based language. It relies on semantic web technologies [18] for the implementation of a limited acoustical performance check. Also Beach et al. [19] relies on semantic web technologies for automated regulatory compliance in the construction sector.

Regardless of the representation that is used to represent the rules, a number of techniques are available to develop the rules. Rules can be developed or created manually, which is near to always the case when representing them in procedural code. When using a (logic-based) language, there is also a possibility of semi-automating the development of rules from its human language representation, as is for example investigated in Zhang and El-Gohary [20,21].

1.2. Semantic rule checking: the basics

In the previous section, we already saw that the design model and the rules are two vital elements in any rule checking process. These elements of the rule checking process have well-defined terms and definitions in any computer science context. The design model can be considered as a sample data set (e.g. an IFC building model). This data set typically follows an agreed structure, vocabulary, or class hierarchy (e.g. the IFC schema). The former, the sample data set, is commonly understood as an assertion box (*the ABox*), whereas the latter, the vocabulary, is commonly known as a terminological box (*the TBox*). ABox and TBox components are often used in scenarios other than rule checking, e.g. query interfaces, data exchange, interface design. In the context of rule checking, the rules form a third box in addition to the ABox and TBox, namely the rule box (*the RBox*).

At the core of any rule checking process are then three key components: (1) a schema that defines what kind of information is used by the rule checking process and how it is structured (the TBox), (2) a set of instances asserting facts based on the concepts defined in the TBox (the ABox), and (3) a set of rules (e.g. `IF-THEN` statements) that can be directly combined with the schema (the RBox). The key advantage in using a 'language-based' rule checking process, is that ABox, TBox and RBox are all stored in a compatible or identical (logic-based) language. A schematic display of this setup is provided in Fig. 1.

These three components are realised in various ways depending on the approach taken and the software system used. In a traditional hard-coded rule checking process, the schema is typically represented by the internal object model of the system including its class hierarchy; the instances are represented by the objects that follow this class hierarchy; and the rules are represented by interconnected procedural functions that can follow any kind of structure, while still being compatible with the class hierarchy of the system.

This is considerably different from the way in which these three components take shape in a semantic language-driven approach. Namely, in this case, the schema is typically represented by an OWL ontology, the instances are represented by the RDF graph using definitions of that OWL ontology, and the rules are logical conjunctions (`AND`) of declarative `IF-THEN` statements. Because of the logical basis of the OWL language in DL [16], the rule checking process is straightforward as soon as all the data and all the rules are available in a complete and consistent shape: inferences are generated by generic reasoning engines and results, asserted as new facts into the graph, are used, e.g. for simple visualisation in a graphical user interface (GUI).

In this article, we specifically look into the rule checking implementation method using semantic web technologies for construction industry. The great advantage of using semantic web technologies is that the schema, the instances, and the rules can all be described using one and the same data model or language. As a result, all three components benefit from the advantages given by Eastman et al. [2] for any language-driven approach, namely:

1. the possibility to easily retarget an implementation to different source formats (e.g. an alternative ontology: a Revit ontology instead of an IFC ontology);
2. portability across contexts, applications and devices, and

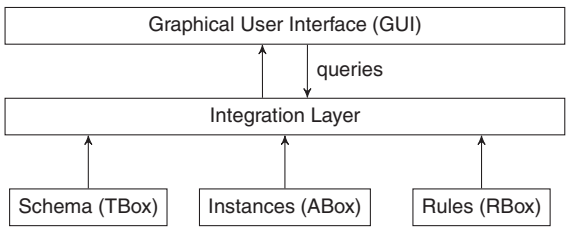

Fig. 1. Schematic view of a rule checking system and its key components.