Querying a Regulatory Model for Compliant Building Design Audit

Johannes Dimyadi, jdim006@aucklanduni.ac.nz
Department of Computer Science, University of Auckland, New Zealand

Pieter Pauwels, pipauwel.pauwels@ugent.be
Department of Architecture and Urban Planning, Ghent University, Ghent, Belgium

Michael Spearpoint, michael.spearpoint@canterbury.ac.nz
Department of Civil and Natural Resources Engineering, University of Canterbury, New Zealand

Charles Clifton, c.clifton@auckland.ac.nz
Department of Civil and Environmental Engineering, University of Auckland, New Zealand

Robert Amor, trebor@cs.auckland.ac.nz
Department of Computer Science, University of Auckland, New Zealand

Abstract
The ingredients for an effective automated audit of a building design include a BIM model containing the design information, an electronic regulatory knowledge model, and a practical method of processing these computerised representations. There have been numerous approaches to computer-aided compliance audit in the AEC/FM domain over the last four decades, but none has yet evolved into a practical solution. One reason is that they have all been isolated attempts that lack any form of standardisation. The current research project therefore focuses on using an open standard regulatory knowledge and BIM representations in conjunction with open standard executable compliant design workflows to automate the compliance audit process.

This paper provides an overview of different approaches to access information from a regulatory model representation. The paper then describes the use of a purpose-built high-level domain specific query language to extract regulatory information as part of the effort to automate manual design procedures for compliance audit.

Keywords: Knowledge management, regulatory compliance audit, compliant design workflows, regulatory knowledge, domain specific language

1 Introduction

Research in the area of computer-aided compliant design audit in the Architecture, Engineering, Construction and Facility Management (AEC/FM) domain dates as far back as four decades, when decision tables were used to aid engineering design for conformance with the AISC (American Institute of Steel Construction) specifications (Fenves et al., 1969). This led to a few expert system implementations such as SASE, SICAD, SPEX (Fenves and Garrett, 1986), but none of them survived due to the high costs of keeping up with frequent changes in the expert knowledge. Since then, there has been a new stream of projects and prototype systems in Australia, New Zealand, Singapore, Finland, Sweden, Norway, Germany, France and the USA, suggesting different rule-based compliance audit approaches.

Apart from rule-based systems, other suggested approaches to computable representation of regulatory knowledge in the AEC domain include the use of hypertext and hypermedia to aid navigating in regulatory texts, automated and semi-automated knowledge acquisition from regulatory texts by means of deontic modelling, natural language processing (NLP), document mark-up techniques, and the use of a domain ontology and semantic technologies with reasoning

capabilities (Evt et al., 1992; Pauwels et al., 2011; Salama and El-Gohary, 2013; Zhang and El-Gohary, 2013; Zhong et al., 2012).

An effective regulatory knowledge representation must allow correct and efficient on-demand access to information. This paper reviews commonly used methods of regulatory information retrieval and highlights the practicality of using a domain specific query language in conjunction with the approach developed in this research for performance-based compliance audit.

2 Representing and Accessing Regulatory Knowledge for Compliance Audits

2.1 Explicit and Implicit Regulatory Knowledge

Regulatory knowledge in the context of the AEC/FM domain includes a lot of explicit forms of knowledge, such as: prescribed design parameters, mathematical equations, rules, constraints and other normative data. This knowledge is conventionally written in natural language texts for human interpretation yet there are many ways in which a design can be made compliant with these explicit regulations. A designer may choose certain parameters and scenarios to achieve a particular compliant design. We refer to this sequence of choices as a 'path to compliance'. There are a considerable number of such paths in each regulation. If another set of parameters or scenarios were chosen, then another path may have been found and achieved. Ultimately, it is up to the designer to evaluate and decide which path to follow. Making the decision on which path to take to achieve compliance typically depends on implicit knowledge that takes into account selected design scenarios, acceptable levels of risk, and safety margins.

The implicit regulatory knowledge also considers selected building performance criteria, which are usually descriptive and not prescriptive in nature. Contemporary building design solutions are driven by high performance objectives and innovation, which often fall outside the scope of prescriptive regulatory requirements. Performance criteria usually require evaluations by means of engineering analysis or simulations, which are not easily represented in a rule-based system. Performance-based regulations allow designers to explore engineering solutions from a broad range of compliant design options, which means that the number of compliance paths is often indeterminate at the outset.

While the more explicit regulatory knowledge can generally be formalised into rules relatively easily, implicit knowledge is much more difficult to represent. There needs to be a more practical method of representing and accessing such regulatory knowledge, so that it support human input and allows interactions with engineering analysis or simulation tools commonly used by designers. In this research we argue that designers should accept the responsibility of specifying exactly which objects or attributes in a regulatory model or building model are to be checked for compliance. To automate this design compliance audit process the required mapping of objects can be specified and recorded in a compliance audit workflow for subsequent executions. Similarly, to interface with engineering analysis or simulation tools, designers would need to specify the input and output schemas of those tools as the mapping mechanism for the workflow to use and gather the required information and generate, or parse, the associated input and output data files.

2.2 Conventional Rule-based Approach

A rule-based system is the most obvious implementation approach to follow when aiming at compliance audits. This is also the basis of many computable representations of regulatory knowledge. A common method of developing a rule-based system is to manually extract and translate written rules directly into computer code, optionally using parameterisation and branching. In this approach, formalised regulatory information in the form of codified rules is then accessed internally by the programming code of the compliance audit application. A comprehensive survey of conventional rule-based compliance audit tools and prototype systems has been given in the literature (Eastman et al., 2009). Hard-coded rules are indicated to be central in these conventional systems, resulting in rules that are tightly integrated into the compliance audit system, e.g. DesignCheck, SMARTCodes, ePlanCheck, Solibri Model Checker (Eastman et al., 2009). One challenge associated with hard-coded rules as part of the compliance audit system is the inflexibility and cost to update, as it requires a system programmer to recode the system to accommodate even a minor rule change. Furthermore, a proprietary or closed rule-based system lacks the transparency that makes it possible for end users or domain experts to verify the correctness of the representation. Moreover, the

approach in which rules are hard-coded in the software may be acceptable for representing prescriptive regulatory requirements, but it is far from adequate in representing performance-based requirements due to their dynamics and qualitative, or descriptive, nature.

2.2.1 Object mapping

Systems that are designed to automatically compare objects in separate data representations typically need to rely on some object mapping schemas. As a simple example, in order to check if a doorway in a building model has an adequate width for compliance with certain regulations, the equivalent doorway object in the relevant regulatory model needs to be first identified and the required width attribute in a given condition noted. An automated compliance audit system has a challenging task to check every object and its attributes in the building model independently against the equivalent objects and attributes in the regulatory model, taking into account any condition or scenario attached to the use of those objects.

2.2.2 Rule engine based systems

Separating rules from the compliance audit core functionality improves maintainability, enables extensibility, and allows portability. In a rule engine based system, regulatory requirements are formalised into a set of IF-THEN statements and stored in a centralised database that is accessible by the rule engine. Rule engine implementations usually have reasoning capabilities that can execute one or more rules as required in a runtime production environment. There are many open standard rule engines that may be suitable to represent regulatory knowledge for compliance audit purposes, for example DROOLS, OpenRules and OpenRuleEngine, SRE (Simple Rule Engine), JESS (Java Expert System Shell), and others. In particular, DROOLS and its DROOLS Rule Language (DRL) has been suggested as the method of representing regulatory knowledge in a number of research projects on computer-aided compliance audit (Beach et al., 2013; Solihin and Eastman, 2015).

2.3 Language-based Compliance Audit Approach

An alternative to the conventional rule-based approach is to formally interpret natural language statements into computable rules in either a domain specific language or in a language that is based on first order predicate logic (Eastman et al., 2009; Pauwels et al., 2011; Rosenman and Gero, 1985). Such an approach would precisely overcome the disadvantages outlined for hard-coded rules and make regulatory knowledge and compliance audits flexible, transparent, and portable.

In the remainder of this paper, we will look into this alternative language-based approach and investigate to what extent it is possible to use either a domain specific language or a language based on first order predicate logic. As semantic web technologies (Berners-Lee et al., 2001) have a basis in Description Logics (DL) and they are used to represent building data, they can be used to test the latter approach.

3 The Logic-based Language Approach: Semantic Web Technologies

3.1 Methodology

Semantic web technologies have their appeal in allowing the structured representation of information with ontologies and in enabling the combination or linking of disparate information sources accessible on the world wide web (Berners-Lee et al., 2001). The de facto open standard knowledge representation language for authoring ontologies is the Web Ontology Language (OWL) (Horrocks, 2008), which is integrated with RDF (Resource Description Framework), a simple language originally developed to describe data or resources as targeted labelled graphs. Information in the semantic web consists of triples, which are RDF expressions constructed of subjects, predicates, and objects (Figure 1). By semantically linking all kinds of objects and subjects (resources) using predicates, large clouds of Linked Data can emerge. This information is stored in RDF triple stores, which are a specific kind of graph database. There are a number of open standard computing environments for managing RDF graphs. A commonly used open source Java framework is the Apache Jena, which has an API (Application Programming Interface) for reading RDF graphs and supports serialising triples in a number of syntaxes, including RDF/XML and Turtle. The primary query language for RDF graphs is SPARQL (SPARQL Protocol and RDF Query Language), which is a declarative language for subgraph retrieval. Like its SQL counterpart, SPARQL includes a number of reserved keywords, such as SELECT and WHERE, to build queries (Prud'hommeaux and Seaborne, 2008).

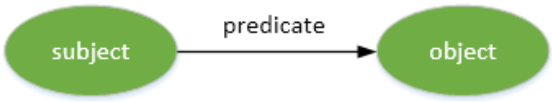

Figure 1 A triple form of a RDF expression

As semantic web technologies also allow the representation of rules and the combination of such rules with available information sources, they might also be a useful set of technologies that can be used for compliance checking of building designs (Pauwels et al., 2011). An essential component of a semantic web based compliance audit system in the AEC/FM domain is the domain ontology, which is a structured and formal representation of knowledge in that domain. The scope of the domain knowledge represented in an ontology depends on the intended application and the type of problems to be addressed. The ontology defines the object types and relations that are available for the representation of objects, which can result both in a building model and a regulation model. To enable mapping of objects between building and regulatory knowledge representations, two complementary ontologies may be used, one representing the building and another representing the regulatory knowledge.

To represent building models, the ifcOWL ontology is available (Beetz et al., 2008; BuildingSMART, 2015; Pauwels and Terkaj, 2015), which is created from the open standard IFC (Industry Foundation Classes) schema in EXPRESS. There are a number of ways to represent regulation knowledge depending on how the regulation audit is intended to take place. The development of ontology-based regulatory knowledge representation usually involves the following procedure (Yurchyshyna and Zarli, 2009):

1. Conversion of regulatory texts into a formal language such as XML (EXtensible Mark-up Language) and RDF.
2. Formalisation of regulatory requirements by capitalising on the domain knowledge
3. Semantic mapping of regulatory requirements to domain specific ontologies
4. Formalisation of compliance requirements for compliant design audit purposes

3.2 Application to a Regulatory Knowledge Model Case

The above procedure was used to develop a Regulatory Knowledge Model (RKM) for the purposes of this research project and further discussions on RKM are given in Section 4.1. The schema for RKM is defined using XSD (XML Schema Definition) representing the regulatory document structure and content. By making the knowledge inherent in the RKM available as an OWL ontology that is compatible with the ifcOWL and with a set of semantically structured rules, it should in theory be possible to only use a reasoning engine to audit a given building model for compliance.

Building a parallel OWL ontology for RKM is not the hardest part. What is important is how the rules that are defined in the XSD schema can be represented in a semantic web rule language such as N3Logic, which is used in this case. Figure 2 gives an indication of how a rule may be represented as an addition to the generated OWL ontology (not as an RDF instance graph). The Omnicode "13-31 13 00" shown in the rule is the classroom type space activity and "FLED" is the Fire Load Energy Density prescribed by the regulatory document, both are as defined in the RKM (Listing 2).

```
{
    ?spaceActivity occ:spaceActivityCode "13-31 13 00"
    ?occupancyGroup occ:hasSpaceActivity ?spaceActivity
}
=>
{
    ?designParameters compl:FLED "400"
}
```

Figure 2 Representation of rule-like information from RKM expressed in N3Logic

There are a couple of features to notice that can impact on the way in which a rule-checking process or compliance audit process is implemented. Firstly, in order for any N3Logic rule to work, the IF-part of that rule (the first two lines in the N3Logic rule in Figure 2) needs to be available and

recognisable as a graph. If a graph does not contain the predicates used or the structure represented by that IF-part, the IF-part is never valid and the rule never 'fires'. The IF-part of the rule in Figure 2 relies on the predicates *occ:spaceActivityCode*, and *occ:hasSpaceActivity*. If the building ontology and regulatory ontology are well designed, this should not be an issue. The N3Logic rule can be used entirely apart from the targeted application and can be used, just like the OWL ontology and the RDF graphs compliant with that ontology, by any number of other applications.

A second element to notice here, is that the output of the N3Logic rule, in this case the property *compl:FLED*, is added to the original RDF graph as soon as an inference engine executes the rule. This additional property can in that same inference run also use that rule, for example if it is contained in the IF-part of another rule. In other words, when an inference engine is started on a set of such rules, it generates all information that can logically be entailed from what is given. In cases where requirements and conditions are well established, an implementation using logically structured rules is likely to result in logically more sensible conclusions without the need for human intervention.

4 The Domain Specific Language

4.1 Regulatory Knowledge Model (RKM)

There has been research in the legal domain over the last two decades resulting in a number of useful initiatives for digitally sharing parliamentary, legislative, and judiciary documents. These include Akoma Ntoso (Architecture for Knowledge-Oriented Management of African Normative Texts using Open Standards and Ontologies), which is currently being standardised by the OASIS (Organisation for the Advancement of Structured Information Standards) into LegalDocML (Vitali and Zeni, 2007); CEN (European Committee for Standardisation) MetaLex; and LKIF (Legal Knowledge Interchange Format) and LegalRuleML (Athan et al., 2013), which focus more on the content and legal knowledge representations.

While these universally open legal data models are being standardised, an interim RKM schema has been developed in this research (Dimyadi et al., 2014a) to illustrate a practical computer-aided approach that focuses primarily on automating compliance audit in design procedures. The selected case study for research is the compliance audit of fire engineering performance of buildings. In particular, a performance-based regulatory document, C/VM2, which is a verification framework for fire safety design in New Zealand, has been used as the subject for the RKM representation. The high-level XSD schema of this RKM is shown in Figure 3.

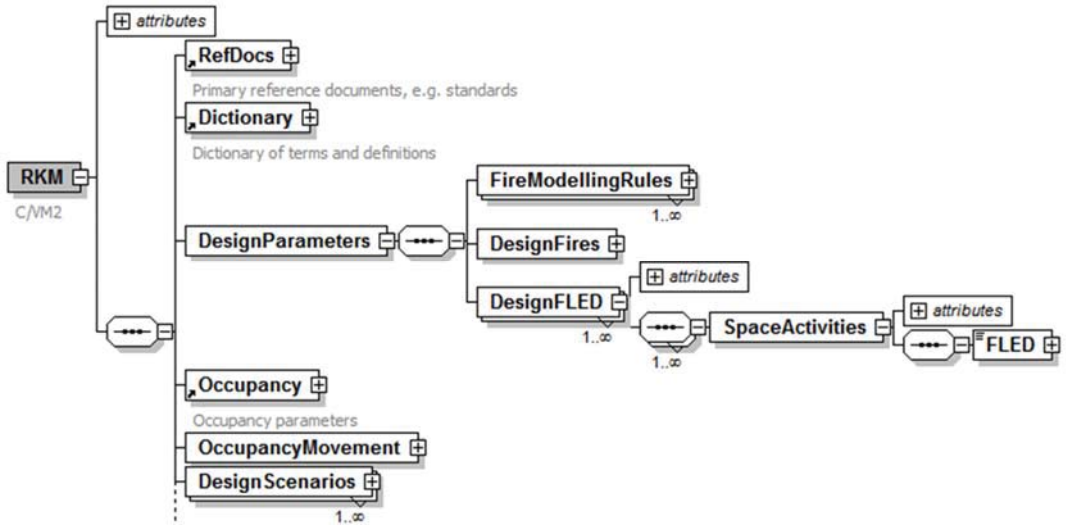

Figure 3 High-level schema of RKM for C/VM2 document for Fire Engineering Design in New Zealand

Different regulatory documents may use different terminologies and classifications for identical objects. For example, space functions or activity types may be described differently, while referring to the same activity, so these terms would need to be translated into a consistent set of codes using the same standard classification. In this case, the open standard Omniclass (CSI, 2012) classification

of spaces by function may be used. Hence, this classification system forms one of the important sources behind the specification of the RKM shown in Figure 3.

4.2 Compliant Design Procedures (CDP)

Human input is an important feature in performance-based design where the compliance audit procedure or method and the design assumptions need to be formally documented. Building designers need to specify exactly how their designs can be verified for compliance by peer reviewers or the regulatory authority. A practical approach has been developed in the current research to allow designers to describe their own Compliant Design Procedures (CDP) and any tacit knowledge using the open standard BPMN (Business Process Model and Notation) executable workflow model (Object Management Group, 2011) (Figure 4), which supports XML data exchange natively (see Listing 1).

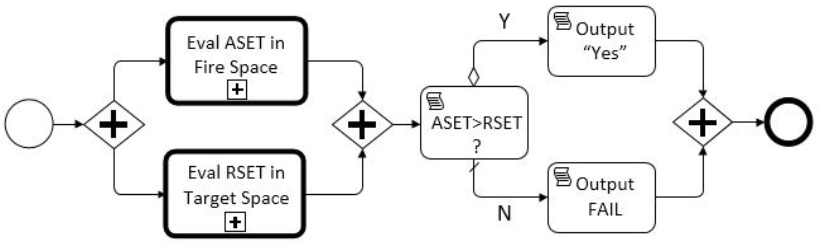

Figure 4 An exemplar executable CDP workflow described in BPMN 2.0 (Dimyadi et al., 2014b)

This approach gives designers the freedom to explore design options that are compliant with selected regulations, while not taking away their responsibility to specify the intended compliance paths followed in their design procedures. Each CDP workflow represents a pre-determined set of compliant design procedures that can be executed multiple times for different design options and across multiple projects, hence automating the manual compliance audit procedures (Dimyadi et al., 2014b). This is considered more appropriate than trying to allow a machine to derive the appropriate compliance path out of an indefinite number of options directly from regulatory texts based on a given design model. Furthermore, a CDP workflow can be used to gather the information required and generate the input data for engineering analysis or simulations. Inevitably, information will be missing from the building model that needs to be supplemented by human input. A CDP workflow allows additional human input to be specified as necessary.

Instructions in the form of query scripts can be embedded in the CDP workflow for retrieving information from building and regulatory models. These scripts can also assign values to variables, and evaluate mathematical expressions or logical statements. As in regular compliance audit procedures and design processes, designers certainly need to be familiar with the content of the RKM in order to be able to specify correct queries. This is no different from searching and using regulatory documents in the course of a traditional design process or traditional compliance audit. A high-level user interface (Figure 5) may be incorporated into a compliance audit system to provide a list of objects and attributes of the models available for query or allow designers to navigate easily through the models to find the correct objects to query. Likewise, designers need to be familiar with the content of the BIM model view, so that they are able to obtain the correct building information to process.

In practice, one possible scenario would be for a professional body representing the domain experts, such as the professional association of engineers, to develop a library of best practice CDP workflows. This would minimise the effort required by individual designers to create their own CDPs from scratch, although designers can still modify any officially published CDP workflow to suit their own design practice or a particular design option.

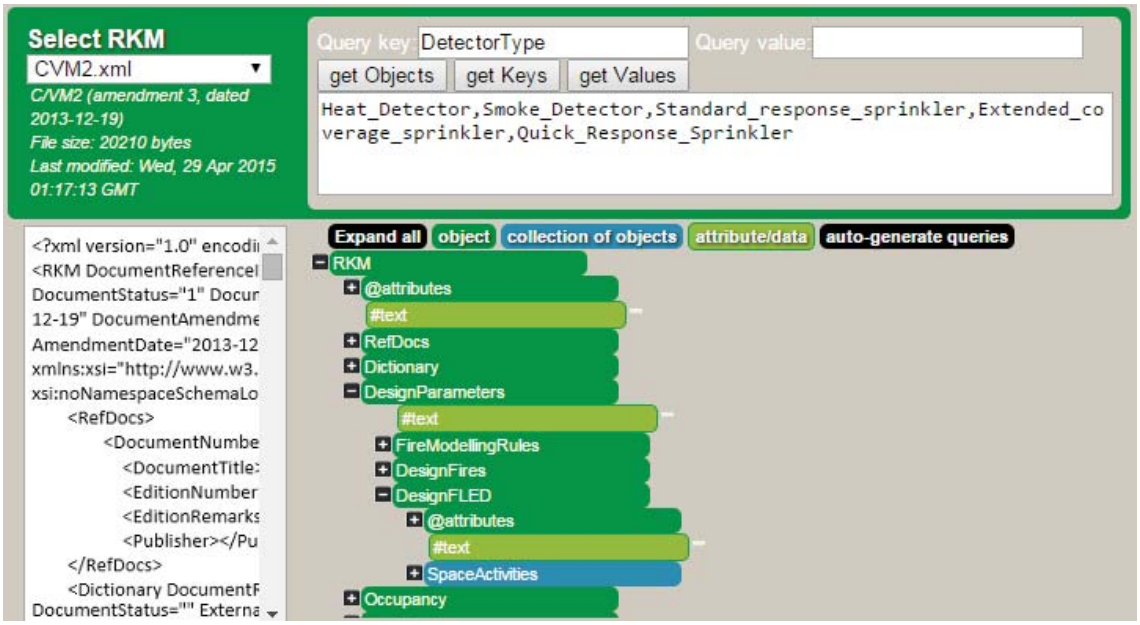

Figure 5 A user interface to navigate RKM and help identifying objects and attributes to query

4.3 Querying Regulatory Knowledge

To access and manage information stored in a relational database management system (RDBMS), an international standard special-purpose programming language SQL (Structured Query Language) (ISO/IEC 9075-1, 2008) is commonly used. The most common operation in SQL is the query, which is performed with the declarative SELECT statement and has the optional FROM and WHERE and other keywords. While SQL is the de facto standard for querying data in a relational database, XQuery (XML Query Language) and LINQ (Language Integrated Query) are two languages commonly used to query data in XML documents. XQuery is the W3C (World Wide Web Consortium) standard and LINQ is a component of the Microsoft .NET framework that extends the .NET languages with query expressions.

As described earlier, one way to query information in the RKM using the CDP approach is to embed instructions in the form of scripts in the CDP workflow. However, it would be unreasonable to expect a building designer to be conversant with standard computer scripting or query languages such as JavaScript, so a much simpler, more intuitive and easy to learn domain specific query language, referred to as Regulatory Knowledge Query Language (RKQL) has been developed as part of this research.

4.3.1 Regulatory Knowledge Query Language (RKQL)

RKQL has been developed to hide the low-level technical functionality from the end user and provides a simple specification to aid building designers to write high-level scripts that can be embedded into a task in the CDP workflow easily. RKQL mainly uses the keyword **GET** with **FROM** and **WHERE** clauses to retrieve information from the RKM. A similar syntax can also be used to query information from the Fire Compliance Model (FCM), which is a view or subset of a BIM model for compliant fire engineering design audit.

The Extended BNF (Backus-Naur Form) notation has been used to describe the syntax and grammar of RKQL (Figure 7). EBNF is an extension of BNF, which is a meta language commonly used to express a context-free grammar of a formal language such as a computer programming language. The grammar of RKQL can also be represented as a set of syntax diagrams as shown in Figure 6. A query to get a specific value from an object in RKM can simply be written as: **GET** *object* **FROM RKM WHERE** *condition*. By default, the type of object is assumed to be **DATA**. However, optionally, RKQL allows one to specify other types of object to get, e.g. **EQUATION** or **RULE**. In a compliance audit system, RKM is usually pre-selected so that the path to its physical location is known, otherwise a full path of the RKM location may be specified (Figure 6). To evaluate a specific rule in RKM, one simply writes **EVAL RULE** *ruleId,* where ruleId is a unique ID of the rule.

```
RKQLStatement ::= getStatement | evalStatement | setStatement
getStatement ::= getClause fromClause whereClause?
getClause ::= 'GET' objectType? nameExp
fromClause ::= 'FROM' sourceExp
sourceExp ::= 'FCM' | (fullpathExp 'IN')? 'RKM'
objectType ::= 'DATA' | 'EQUATION' | 'RULE' | 'ENTITY'
whereClause ::= 'WHERE' conditionalExp
conditionalExp ::= nameExp comparisonOperator (valueExp | variable | booleanLiteral)
comparisonOperator ::= ( '=' | '!=' | '<' | '>' | '<=' | '>=')?
nameExp ::= variable | letter+
fullpathExp ::= ('/' nameExp)
valueExp ::= integer | real
setStatement ::= ('SET') nameExp '=' valueExp | expression
evalStatement ::= 'EVAL' 'RULE'? nameExp
expression ::= ('+' | '-') term+
term ::= factor ( ('*' | '/') factor )+
factor ::= integer | real | variable | function | limits | '(' expression ')'
variable ::= identifier
function ::= identifier '(' expression (',' expression) ')'
limits ::='{' expression '}'
identifier ::= letter (digit | letter | '_')+
booleanLiteral ::= 'true' | 'false'
letter ::= 'a-z' | 'A-Z'
digit ::= '0-9'
integer ::= digit+
real ::= integer real_decimals
real_decimals ::= '.' integer
```

Figure 7 Extended Backus-Naur Form (BNF) notation of RKQL domain specific query language

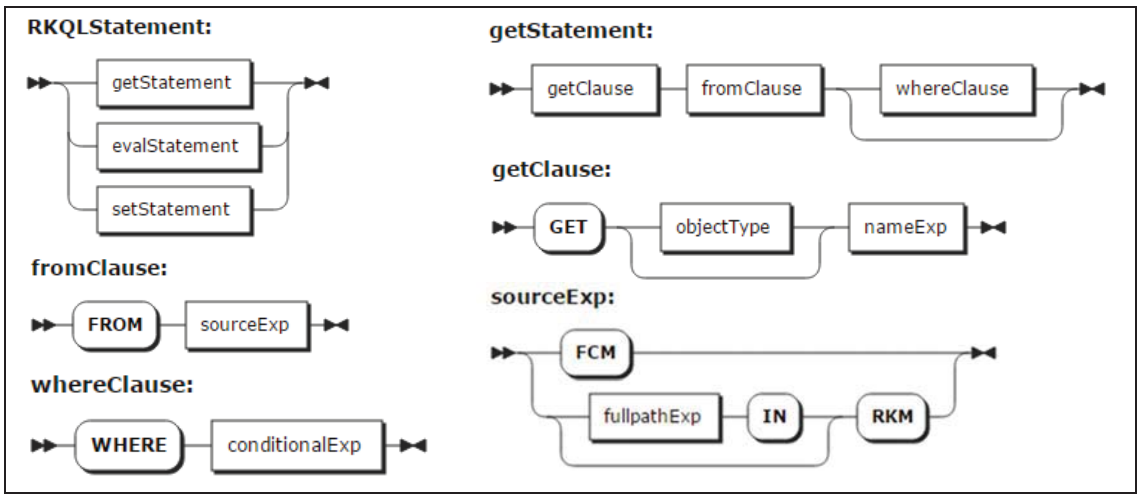

Figure 6 Some syntax diagrams of RKQL domain specific query language

A systematic process of an automated building compliance audit may be on a floor level by floor level basis starting at the top level down. Every space object on each level is then processed in turn and subject to the calculations specified in the CDP workflow. For example, given a space activity type such as "Offices" or "13-55 11 00" in Omnicode, the corresponding prescribed FLED value for that space is 800 MJ/m^2. An exemplar RKQL script to retrieve FLED from RKM given a set of conditions is shown in Listing 1. Listing 2 shows part of the RKM where the information to be retrieved by the query script in Listing 1 is utilised.

Listing 1 Exemplar RKQL script embedded in a script task of a CDP workflow

```
<scriptTask isForCompensation="false" id="P0_p2034" name="FLED" arcabim:unit="MJ/m^2">
    <incoming>P0_p2033</incoming>
    <outgoing>P0_p2073</outgoing>
    <script>GET FLED FROM RKM WHERE SpaceActivities.SpaceActivityCode = spaceActivityType</script>
</scriptTask>
```

Listing 2 Excerpt from the RKM of C/VM2 compliance document for the New Zealand Building Code (NZBC)

```
<DesignFLED DocumentReferenceId="2.3.3" TableReferenceId="2.2">
    <SpaceActivities  SpaceActivityCode="13-31 13 00" SpaceActivityDescription="Classrooms" SpaceActivit
        <FLED Unit="MJ/(m^2)" Multiplier="1" MaxFLED="0.0" MinFLED="0.0" Remarks="" >400</FLED>
    </SpaceActivities>
    <SpaceActivities  SpaceActivityCode="13-55 11 00" SpaceActivityDescription="Offices" SpaceActivityRe
        <FLED Unit="MJ/(m^2)" Multiplier="1" MaxFLED="0.0" MinFLED="0.0" Remarks="" >800</FLED>
    </SpaceActivities>
    <SpaceActivities  SpaceActivityCode="13-59 00 00" SpaceActivityDescription="Factory" SpaceActivityRe
        <FLED Unit="MJ/(m^2)" Multiplier="1" MaxFLED="0.0" MinFLED="0.0" Remarks="" >1200</FLED>
    </SpaceActivities>
```

5 Discussion and Summary

In this paper we have presented several methods commonly used to represent and access digital regulatory knowledge for compliance audit purposes. The traditional methods based on proprietary or hard-coded rule-based representations were quite successful in their implementations but they have the disadvantages of being costly to maintain and inflexible to changes. Many of these systems did not survive the test of time although today a few commercial tools are still adopting such 'black-box' strategies.

There is a need for an open standard regulatory knowledge representation that allows efficient access to regulatory information. Semantic web technology provides a means to represent regulatory knowledge with reasoning capabilities that can automatically access relevant information for compliance audit based on a set of pre-defined rules. While this logic-based approach may provide a way to automate some of the more established requirements and conditions, there are aspects of regulatory compliant design that still rely on tacit knowledge and intuition, which is best handled by a human. Furthermore, qualitative performance-based criteria requires engineering analyses to address, which is not yet suitable for automation. The current research focuses on allowing a human designer to specify exactly how compliance can be achieved by recording the procedures in a CDP workflow that can then be executed in a computer system for multiple design options and across different projects with consistent results. For usability a domain specific language, RKQL, has been developed to allow a building designer or engineer to specify queries and other high-level computer instructions with ease and intuitively. All low-level technical specifications are hidden from the user and handled by the compliance audit process engine.

Acknowledgements

The first author gratefully acknowledges the funding assistance from the New Zealand Building Research Levy, which is administered by Building Research Association of New Zealand (BRANZ). The second author acknowledges the funding and support by the Special Research Fund (BOF) of Ghent University.

References

Athan, T., Boley, H., Governatori, G., Palmirani, M., Pasckhe, A., and Wyner, A. (2013). OASIS LegalRuleML. In *The International Conference on Artificial Intelligence and Law*. Rome, Italy.

Beach, T. H., Kasim, T., Li, H., Nisbet, N., and Rezgui, Y. (2013). Towards Automated Compliance Checking in the Construction Industry. *Database and Expert Systems Applications*, *8055*, pp. 366–380.

Beetz, J., van Leeuwen, J., and de Vries, B. (2008). IfcOWL: A case of transforming EXPRESS schemas into ontologies. *Artificial Intelligence for Engineering Design, Analysis and Manufacturing*, *23*(01), pp. 89. doi:10.1017/S0890060409000122

Berners-Lee, T., Hendler, J., and Lassila, O. (2001). The Semantic Web. *Scientifc American*, *284*(5), pp. 29–37.

BuildingSMART. (2015). Linked Data Working Group. Retrieved May 20, 2015, from http://www.buildingsmart.org/standards/standards-organization/groups/linked-data-working-group/

CSI. (2012). *OmniClass - A Strategy for Classifying the Built Environment Table 13 – Spaces by Function.* Construction Classification System.

Dimyadi, J., Clifton, C., Spearpoint, M., and Amor, R. (2014a). Computer-aided Compliance Audit to Support Performance-based Fire Engineering Design. In *Proceedings of 10th International Conference on Performance-based Codes and Fire Safety Design Methods.* Gold Coast, Queensland.

Dimyadi, J., Clifton, C., Spearpoint, M., and Amor, R. (2014b). Regulatory Knowledge Encoding Guidelines for Automated Compliance Audit of Building Engineering Design. In *Proceedings of the ICCCBE/CIB W78*. Orlando, Florida, USA.

Eastman, C., Lee, J., Jeong, Y., and Lee, J. (2009). Automatic rule-based checking of building designs. *Automation in Construction*, *18*(8), pp. 1011–1033. doi:10.1016/j.autcon.2009.07.002

Evt, S. K., Khayyal, S., and Sanvido, V. E. (1992). Representing Building Product Information using Hypermedia. *Journal of computing in Civil Engineering*, *6*(1), pp. 3–18.

Fenves, S. J., and Garrett, J. H. (1986). Knowledge based standards processing. *Artificial Intelligence in Engineering*, *1*(1), pp. 3–14. doi:10.1016/0954-1810(86)90029-4

Fenves, S. J., Gaylord, E. H., and Goel, S. K. (1969). *Decision table formulation of the 1969 AISC specification*. University of Illinois Engineering Experiment Station. College of Engineering. University of Illinois at Urbana-Champaign, USA.

Horrocks, I. (2008). Ontologies and the semantic web. *Communications of the ACM*, *51*(12), pp. 58–67. ACM Press. doi:10.1145/1409360.1409377

ISO/IEC 9075-1. (2008). *ISO/IEC 9075-1:2008 Information technology - Database languages - SQL - Part 1: Framework (SQL/Framework)*. International Organisation for Standardisation.

Object Management Group. (2011). Business Process Model and Notation (BPMN), *version 2*, pp. 1–538. Object Management Group. Retrieved from http://www.omg.org/spec/BPMN/2.0/

Pauwels, P., and Terkaj, W. (2015). EXPRESS to OWL for Construction Industry: Towards a Recommendable and Usable ifcOWL Ontology. *Automation in Construction*, (Submitted).

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. Elsevier. doi:10.1016/j.autcon.2010.11.017

Prud'hommeaux, E., and Seaborne, A. (2008). SPARQL Query Language for RDF. *W3C Recommendation 15 Jan 2008*. Retrieved May 14, 2015, from http://www.w3.org/TR/2008/REC-rdf-sparql-query-20080115/

Rosenman, M. A., and Gero, J. S. (1985). Design codes as expert systems. *Computer-aided design*, *17*(9), pp. 399–409. Elsevier.

Salama, D. A., and El-Gohary, N. M. (2013). Towards Automated Compliance Checking of Construction Operation Plans Using a Deontology for the Construction Domain. *Journal of Computing in Civil Engineering*, pp. 1–51. ASCE. doi:10.1061/(ASCE)CP.1943-5487.0000298

Solihin, W., and Eastman, C. (2015). Classification of rules for automated BIM rule checking development. *Automation in Construction*, *53*, pp. 69–82. Elsevier B.V. doi:10.1016/j.autcon.2015.03.003

Vitali, F., and Zeni, F. (2007). Towards a country-independent data format: the Akoma Ntoso experience. In *Proceedings of the V legislative XML workshop*, pp. 67–86. Florence, Italy.

Yurchyshyna, A., and Zarli, A. (2009). An ontology-based approach for formalisation and semantic organisation of conformance requirements in construction. *Automation in Construction*, *18*(8), pp. 1084–1098. Elsevier B.V. doi:10.1016/j.autcon.2009.07.008

Zhang, J., and El-Gohary, N. M. (2013). Semantic NLP-Based Information Extraction from Construction Regulatory Documents for Automated Compliance Checking. *Journal of Computing in Civil Engineering*, pp. 1–39. doi:10.1061/(ASCE)CP.1943-5487.0000346

Zhong, B. T., Ding, L., Luo, H. B., Zhou, Y., Hu, Y. Z., and Hu, H. M. (2012). Ontology-based semantic modeling of regulation constraint for automated construction quality compliance checking. *Automation in Construction*, *28*, pp. 58–70. doi:10.1016/j.autcon.2012.06.006