Modularized BIM Data Validation Framework Integrating Visual Programming Language with LegalRuleML

11

Pedram Ghannad, Yong-Cheol Lee, Johannes Dimyadi, and Wawan Solihin

Abstract

A building design must satisfy diverse requirements including building codes, owner's specifications, design guidelines, and project requirements. In addition, there is a growing need for an automated design evaluation process involving intelligent checking and reporting capabilities that addresses the inefficiency and error-prone nature of the current manual checking practice. To leverage the automated rule checking procedure, we need to overcome two existing key challenges, which are the inherent complexity of rules and the impracticability of checking methods. To address these challenges, this research proposes a node-based visual language approach integrated with the emerging open standard LegalRuleML, which allows the flexibility in defining and executing design rules in a machine-readable and implementable format. The approach effectively facilitates the entire rule-checking process including the rule interpretation from natural language-based requirements to machine-readable forms, rule categorization, rule parameterization, and checking execution with a BIM model. The LegalRuleML-based visual programming language approach for rule checking will help automatically and iteratively evaluate the quality and defects of information conveyed in a given building model interactively as an essential part of design process.

Keywords

Building information modeling • BIM data checking • Visual Programming language • LegalRuleML

11.1 Introduction

A building design must fulfill a myriad of requirements including building codes, normative standards, owner's specifications, design guidelines, and project requirements. These requirements consist of various types of rules and execution plans. Rule-checking is an integral part of the design process to evaluate and maintain all the requirements iteratively throughout the entire project. However, the conventional compliance checking practice is laborious, time-consuming, and error-prone. Since rule-checking is frequently a costly bottleneck of the project delivery process, its automation can be a method to save a significant amount of time and cost for the project [5]. The role of automated rule checking has been critically recognized prior to the advent of BIM and it has been investigated as early as the 1960s [4]. In recent decades, the advancement of the BIM technology has offered the opportunities for establishing automated rule-checking systems and their interactive implementation interconnected with a BIM authoring tool.

P. Ghannad · Y.-C. Lee (✉)
Louisiana State University, Baton Rouge, USA
e-mail: yclee@lsu.edu

P. Ghannad
e-mail: gpedra1@lsu.edu

J. Dimyadi
The University of Auckland, Auckland, New Zealand

W. Solihin
Georgia Institute of Technology, Atlanta, USA

I. Mutis and T. Hartmann (eds.), *Advances in Informatics and Computing in Civil and Construction Engineering*,
https://doi.org/10.1007/978-3-030-00220-6_11

LegalRuleML [12] is derived from RuleML [2] with extended features which aim to specifically represent legal documents in a formalized format. This open standard XML has been developed to formalize logical content of norms, guidelines, and codes being used in diverse domains such as Engineering, Commerce, Law, etc.

Rules can be stated in natural language, in some formal notation, or in a combination of both. Currently, encoding codes from natural language to formal rules is a manual process. However, development of LegalRuleML in conjunction with natural language processing (NLP) and other artificial intelligence (AI) methods may be able to make most of the encoding process fully automated.

Using the open standard LegalRuleML for the rule-checking process generally helps convert natural language-based codes and rules into machine-computable forms that can be automatically analyzed and implemented in appropriate software. Since it has been successfully adopted in various domains, the AEC industry also needs to get an attention on its application to address the complex requirements and heterogeneous structure of a building project.

A visual language can be described as a "formal language with a graphical notation", which employs a modular system of signs and rules with visual elements instead of textual ones on the semantic and syntactic level [14]. Visual Programming Languages (VPL), which utilize visual elements can be interpreted much faster and easier by humans. In recent years, VPL has been established particularly in the field of a building design. Known software products in the domain of a building design are the Grasshopper for Rhinoceros3D, Dynamo for Autodesk Revit, and Marionette for Vectorworks. Although these applications primarily focused on expediting the 3D parametric modeling, their features have been significantly extended by further functionalities [13].

To explore the potentials of formal representation of design rules for VPL-based rule checking, this study involves the investigation of the LegalRuleML format and its execution on VPL. Because of the flexibility and usability, the node-based programming approach for automated rule checking will help users properly coordinate necessary features and readily execute a large number of rules with modularized rule sets iteratively for validating a BIM model.

To achieve this goal, this research includes the study of a formal procedure of rule translation from compliant design requirements to LegalRuleML and its possible execution plan using the graphical scripting language. This study adopted Marionette, which supports a graphical programming language embedded in Vectorworks, one of popular BIM authoring tools. To build the rule translation and implementation frame-works using LegalRuleML and visual programming, this study focuses on the simple and fundamental design requirements needed during the early design phase. The case studies and implementation details using the early design's requirements are described in the section of R304 (Minimum room areas) of the International Residential Code (IRC) published by International Code Council (ICC) [8] showed the flexibility of automated rule checking using LegalRuleML-based modularized rule parameters.

Using Marionette with Vectorworks in conjunction with LegalRuleML, the pro-posed approach elevates automated rule-checking capabilities for iteratively evaluating a BIM model regarding conformance with a set of predefined rules as well as consistently maintaining the initial conditions and quality of a BIM model and its data. The checking libraries developed on Marionette give a clear indication of the potentials of a flexible and extensible rule checking method that can handle multiple parameters and rulesets. In other words, this approach executes diverse checking features with a limited amount of checking nodes. The use of a node-based rule definition and analysis approach also improves the transparency of the encoded rule system, which is completely readable and understandable for end users. This notable feature of node-based rule checking provides users with intuitive rule coordination opportunities that have not been available in previous rule checking approaches such as Solibri Model Checker (SMC), which is one of the most popular rule-checking commercial software. Although it has several practical limitations, SMC allows the user to understand and inspect every single processing step and adjust the checking procedures accordingly.

11.2 Literature Review

Building design is subject to various compliance checking in order to meet various types of requirements. This process should be executed for every aspect of the building design to ensure owners, designers, and end users that the product of that design fulfills their requirements and regulatory codes. This process is not only limited to the design phase and assessment of the building compliance must be carried out during other phases of its life cycle. In recent years, there has been an extensive number of researches conducted in the field of automated rule-checking for AEC industry. The traditional methods of rule-checking in AEC industry use a manual process which is significantly error-prone and inefficient. In addition, the increasing complexity of a building design makes the manual practice more time consuming and costly. Over the last three decades, numerous studies and attempts have aimed to improve and automate the process [4], but several reasons have

caused the progress to be very slow, such as the inherent complex nature of the industry, fragmentation, various stockholders, lack of motivation towards the use of new technologies [6, 7].

The most common approach for automating the compliance checking process is the rule-based system. In this approach, the natural language normative texts are manually encoded into computer-readable rules. The building model is then checked against these codified rules. Eastman et al. [5] surveyed different types of rule-based compliance checking systems, such as DesignCheck, SMARTCodes, ePlanCheck, Solibri Model Checker. Lee et al. also reviewed several rule-based platforms and applications such as dRofus, Solibri Model Checker. Revit reviewer add-on, and Invicara [9]. One of the shortcomings of these systems is their inflexibility. They prevent users from defining and executing their own rules or modify the pre-defined rulesets [10]. Another challenge with hard-coded rulesets is the inflexibility to changes and updates in response to changes in the normative requirements. This would require a system developer to maintain the codes for every minor rule update [12]. The lack of transparency is also another limitation of hard-coded systems resulting in domain experts not being able to verify or validate the checking process easily.

11.2.1 Formal Representation and Semantic Interoperability Using RuleML and LegalRuleML

Legal texts (e.g. legislation, regulations, contracts, and case law) are the source of norms, guidelines, and rules. For the domain of a building design, rules and regulations exist in different forms. International, national, local, and even project-specific rules may need to be complied with before, during and after a building is built. Today, rules are typically written in natural languages that require significant domain knowledge to "interpret" before they can be processed by machines. There are several challenges associated with nature of textual content such as exchanging specific data between parties, searching for and extracting structured information within the text, and automatically doing further process. Although patterns exist in many rule structures and clauses, they entail detailed variations that make it hard to capture all required information. The extraction of such patterns, hidden assumptions, and dependencies with other rules are challenging because they require experience with different rule applications and inductive reasoning [15]. The primary challenge for exchanging legal data is the heterogeneous terminology or jargons and its representation structure. In addition, since legal statements of each domain encompass various intents, interests, and targeted realms, their definitions, and representations are frequently abstract and unclear, leaving open-ended interpretations flexibly applicable in relevant cases and situations. Thus, legislators, legal practitioners, business managers have been impeded from comparing, contrasting, integrating, and reusing the contents of the texts, since any such activities are manual. In the current web-enabled context, where innovative eGovernment and eCommerce applications are increasingly deployed, it becomes essential to provide machine-readable forms (generally in XML) of the contents of the text.

The formalization of the appropriate and expressive conceptual, machine-readable format of the multifaceted aspects of norms, guidelines, and general legal knowledge is a fundamental factor for the successful development of rule-Checking processes. To ameliorate this issue, one study has adopted the LegalRuleML to produce a rule inter-change language for the legal domain [1]. Using the representation tools, implementers can structure the contents of the legal texts in a machine-readable format, which then feeds further processes of interchange, comparison, evaluation, and reasoning.

LegalRuleML is built on top of RuleML to facilitate modeling various classes of rules and adding the deontic logic (e.g., obligation, prohibitions, permissions), priorities and the relationships between them. Metadata of rules and normative elements can also be captured by LegalRuleML. In addition, it supports the legal isomorphism principle to maintain the connection between legal source provisions represented by its complementary standard, LegalDocML, and their formal representations or rules [3].

LegalRuleML offers facilities to model different types of norms, deontic effects (e.g., obligations, prohibitions, permissions), and supports defeasibility to resolve contradictions. In addition, it has features to capture the metadata of norms and other normative elements (such as jurisdiction, authorities, validity times, etc.) [1].

11.2.2 Graphical Scripting or Visual Programming Language in Rule Checking

Due to the low-level computation and programming in current design checking implementation, the conventional checking practice is laborious, time-consuming, and error-prone. Even though there are numerous approaches for automated code compliance checking, there are still inadequacies in flexibility, extensibility, and practicability of rule definition and execution. VPL is flow-based and composed by a set of nodes. Each node generally represents a piece of a modularized code as

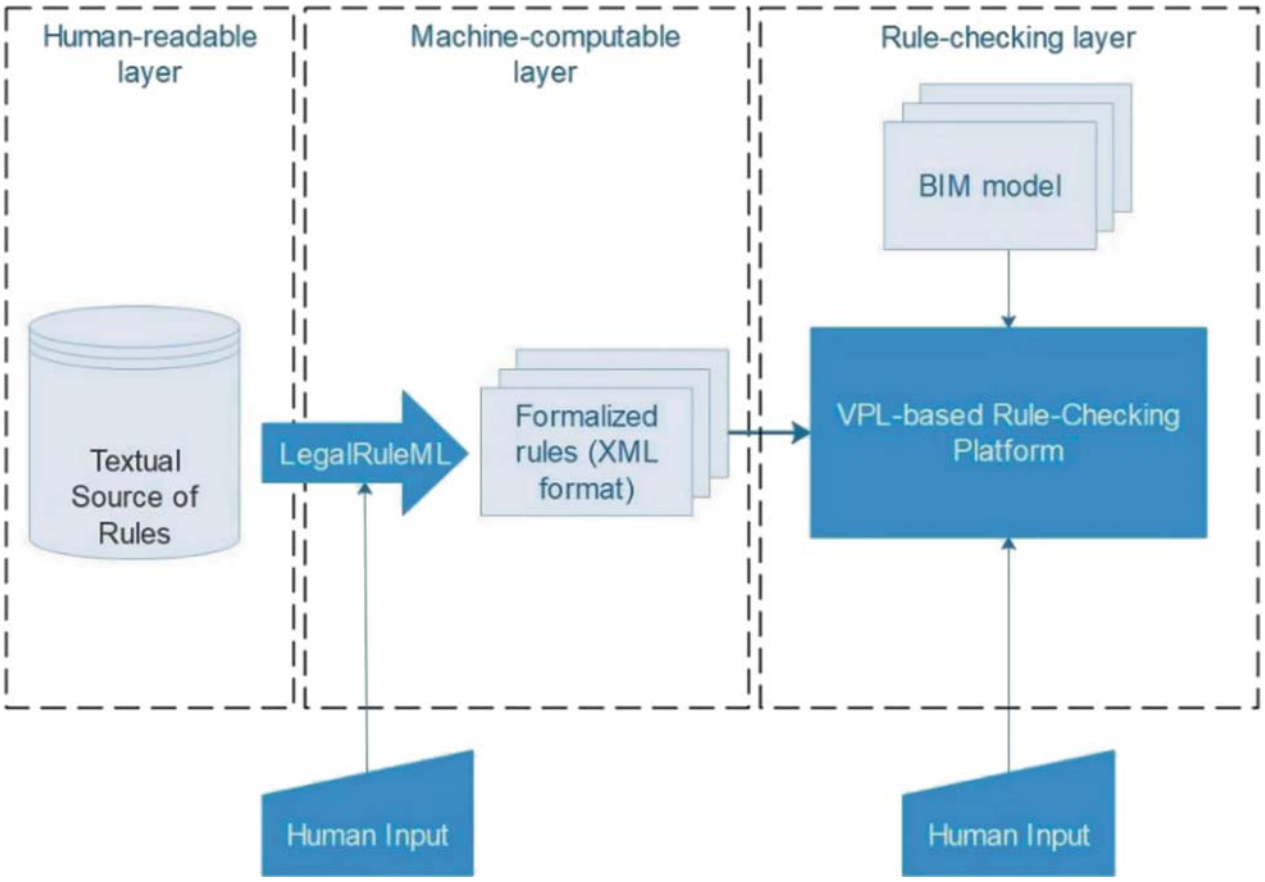

Fig. 11.1 Diagram for the proposed approach of LegalRuleML and visual programming-based rule-checking system

well as the basic unit of programming. This graphical notation also can be used to represent design rules in a machine- and human-readable language. The collection of nodes is similar to a network flowchart that provides the readability and the flexibility so that users can implement compliance checking [13].

11.3 Methodology

This study utilizes open standard LegalRuleML and node-based scripting to develop an integrated framework of a flexible rule-checking approach for the early design phase. This research study employed VPL on the BIM authoring platform, which allows users to intuitively generate and manipulate BIM objects in order to expose and use embedded building information. For a case study, the authors used Marionette which is the VPL feature of Vectorworks. The scope of rule checking includes the rule sets required for the early design phase. In order to develop a robust framework, the first step was the collection and the identification of the early design requirements and rules. The investigation of the requirements for the early design phase allows this research team to confirm that rules are mostly about the spatial and geometric controls and the relationship between different building objects. Examples of these types of rules are as follows: circulation, room size, space sorting, space ratio, geometric property, and object existence.

These rules are formally presented by the XML schema based on LegalRuleML principles. LegalRuleML is integrated with visual programming platform to categorize the rules and implement the module-based rule-checking process. The integrated frame-work addresses mapping of rule types, parameters, and reporting methods into each code and the execution of semi-automated rule-checking.

According to the objectives of this research, we develop a process diagram of the proposed approach (Fig. 11.1). The process has three layers of the human-readable layer, machine-computable layer, and rule-checking layer. The first two layers will connect to each other by utilizing features of LegalRuleML. Then Marionette as visual programming platform will integrate these two layers into the rule-checking process.

11.4 LegalRuleML and Visual Programming-Based Rule-Checking System

11.4.1 The Human-Readable Layer and Machine-Computable Layer

The first major part of rule checking is the rule definitions. Rules are typically written in natural languages that require significant domain knowledge in order to interpret them into a machine computable form. There are many ways to approach the interpretation, as mentioned earlier, but most rule checking studies focus merely on the language representation of syntax and grammar of the rules. In practice, expert knowledge is often required to interpret the meaning or semantics of the rules: the intent, base, and hidden assumptions, assumed a general knowledge of the subjects, and dependencies with other rules.

11.4.2 The Rule-Checking Layer

Based on the functionalities of rule checking, in this research study, the authors identified four steps of rule definition and implementation process: (1) object selection, (2) rule definition, (3) rule implementation, and (4) validation report. These steps are implemented in VPL (marionette in our study) because of two main reasons. Firstly, VPL does not need extensive programming knowledge. Consequently, compliance checking process can be easily designed and modified by domain experts. In addition, marionette is a VPL embedded in Vectorworks, which is a BIM authoring tool, and this can significantly facilitate retrieving and query of building model data required for compliance checking.

The nodes utilized in the development of rule checking network can be categorized according to the aforementioned four steps. To avoid complex and ambiguous networks, each step of the rule checking process can utilize predefined nested nodes as modules which propose the required and identified parameters of the rules. These modules can be used iteratively in various steps and whenever they are needed. Predefined modules are logical nodes, mathematical nodes, geometric nodes, and building model related nodes. By this method, a rule checking process will improve from three aspects: readability, flexibility, and extensibility.

11.5 Rule Classification and Parametrization

11.5.1 Rules and LegalRuleML

One of the major problems in automated rule-checking is addressing how domain expertise is utilized in the interpretation of rules which is currently carried out manually. Rule interpretation is a significant step in the process of rule checking. To address this issue, Solihin and Eastman [15] find it useful to classify rules in four general classes: class (1) that require a small number of explicit data, class (2) that require simple derived attribute, class (3) that require extended data structure, and class (4) that require "proof of solution".

Based on LegalRuleML principles rules are classified mostly into two major categories: constitutive rules and prescriptive rules. The function of constitutive rules is to define and create the so-called institutional facts [1]. Where an institutional fact is how a particular concept is understood in a specific institution. Thus, constitutive rules provide definitions of the terms and concept used in a jurisdiction. On the other hand, the scope of prescriptive rules is to dictate what are the obligations, prohibitions, permissions, etc. in a legal system, and the conditions under which we have them. Legal-RuleML has various other capabilities that can model normative documents with their distinct characteristics, such as defeasibility, superiority, suborder, penalty, etc.

A rule set consist of relevant criteria (data model) and required an operation that must be used to assess different aspects of that data model. In order to organize and execute diverse rule sets more efficiently, with a minimum number of nodes, rule parameters can be classified based on their characteristics and attributes. In order to organize the nodes for rule checking, avoiding complex and ambiguous networks, and integration of the approach a rule mapping process is needed to recognize relevant parameters in formalized rule and VPL. Then each step of the rule checking process can be implemented by using predefined nested nodes as modules which address the required and identified parameters of the rules.

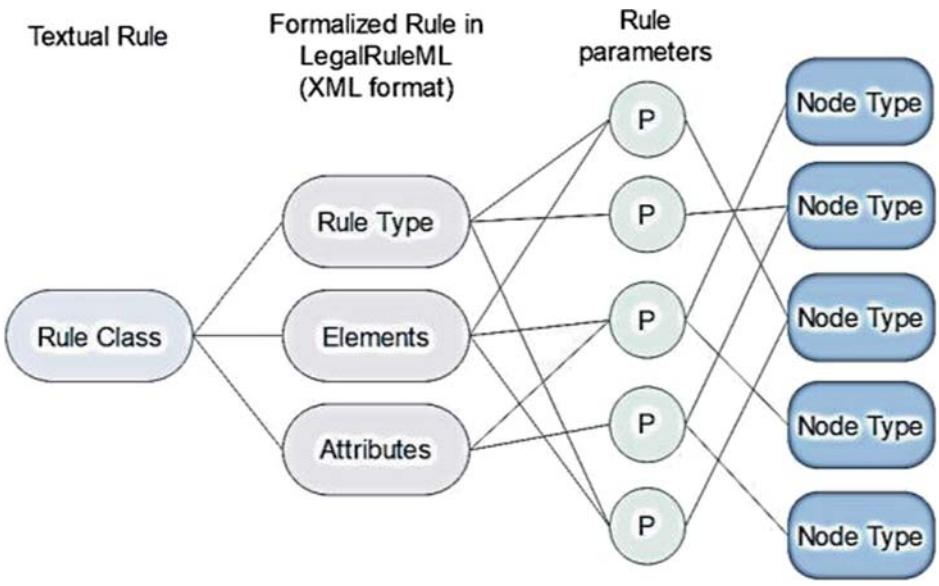

Fig. 11.2 Schematic rule mapping process

11.5.2 Rule Mapping in LegalRuleML

In order to integrate LegalRuleML with the BIM-based rule-checking process, the authors mapped compliant design rules required for the early design phase into rule classes in LegalRuleML. Rule mapping offers a robust framework for the user to design the most effective rule-checking network in VPL. In a proposed approach, it is the user's role to decide how to organize the VPL network for rule-checking implementation. This rule mapping offers a better understanding of the compliance checking process. Furthermore, this step is promising step to move toward automation of the rule-checking process. Figure 11.2 illustrates a schematic process of rule mapping.

11.6 Example Case Study

The proposed automated rule-checking approach is capable of executing the entire rule-checking process. To investigate the applicability of the approach and to accomplish the objective, a case study has been conducted using an early design requirement as described in section R304.1 (Minimum room areas) of the International Residential code [8], which stipulates:

> Habitable rooms shall have a floor area of not less than 70 square feet (6.5 m^2). Exception: Kitchens.

A single-story building model with multiple-rooms was used as a test model for this case study. The above normative textual provision is translated into XML format based on LegalRuleML principles by using the XML schema published by OASIS [11]. The logical content of above provision can be formalized into two rules, "*IRCsecR304.1Statements*" and "*IRCsecR304.1MinimumAreaException*" as follows:

```
<?xml version="1.0" encoding="UTF-8"?>
<lrml:Statements key="IRCsecR304.1Statements">
    <lrml:PrescriptiveStatement key="IRCsecR304.1MinimumArea">
      <ruleml:Rule>
        <lrml:hasStrength>
          <lrml:DefeasibleStrength iri="http://spin.nicta.com.au/spindle/ruleStrength#defeasible"/>
        </lrml:hasStrength>
        <ruleml:if>
          <ruleml:Atom>
            <ruleml:equal>
            <ruleml:Expr>
              <ruleml:Fun iri="buvo:Function"/>
              <ruleml:Var>BuildingSpaceFunction</ruleml:Var>
            </ruleml:Expr>
            <ruleml:Ind>Habitable</ruleml:Ind>
            </ruleml:equal>
          </ruleml:Atom>
        </ruleml:if>
        <ruleml:then>
         <lrml:Obligation>
            <ruleml:atom>
              <ruleml:Rel>Greater than</ruleml:Rel>
              <ruleml:Var>BuildingSpaceArea</ruleml:Var>
              <ruleml:Data>6.5 square meters</ruleml:Data>
            </ruleml:atom>
          </lrml:Obligation>
        </ruleml:then>
      </ruleml:Rule>
    </lrml:PrescriptiveStatement>

    <lrml:PrescriptiveStatement key="IRCsecR304.1MinimumAreaException">
      <ruleml:Rule>
        <ruleml:if>
          <ruleml:Atom>
            <ruleml:equal>
            <ruleml:Expr>
              <ruleml:Fun iri="buvo:Function"/>
              <ruleml:Var>BuildingSpaceName</ruleml:Var>
            </ruleml:Expr>
            <ruleml:Ind>Kitchen</ruleml:Ind>
            </ruleml:equal>
          </ruleml:Atom>
        </ruleml:if>
        <ruleml:then>
         <lrml:permission>
            <ruleml:atom>
              <ruleml:Rel>Less than</ruleml:Rel>
              <ruleml:Var>BuildingSpaceArea</ruleml:Var>
              <ruleml:Data>6.5 square meters</ruleml:Data>
            </ruleml:atom>
          </lrml:permission>
        </ruleml:then>
      </ruleml:Rule>

    </lrml:PrescriptiveStatement>
```

The above representation has one obligatory condition and one permission condition, i.e. if the function of a space is habitable the area must be greater than 6.5 ft^2. This rule can be defeated if the function of the habitable area is “kitchen”. Table 11.1 indicates the rule mapping for this rule which contains rule classification and parametrization discussed before.

Table 11.1 Rule parametrization

Textual format		LegalRuleML format	Rule parameters		Node types	
Class 2	Elements	<ruleml:if>, <ruleml:then>, <lrml: Obligation>, <lrml:permission>, etc.	Relationship	Greater than	Logical nodes	Greater than
	Attributes	Var: BuildingSpaceFunction	Target building object	Space	Building model related nodes	Selection
			Name or function of a building object	Habitable	Building model related nodes	Filter
		Var:BuildingSpaceName	Target building object	Space	Building model related nodes	Selection
			Name or function of a building object	Kitchen	Building model related nodes	Filter
		Var:BuildingSpaceArea	Geometric property of a building element	Area		Get area
		Data: 6.5 m^2	Value	6.5 m^2		

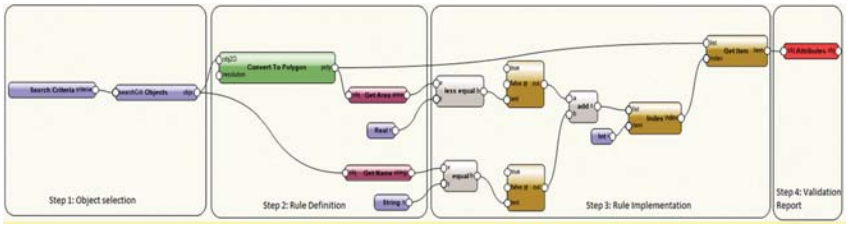

Fig. 11.3 Node network for checking minimum area rule

The output of this step is identifying required nodes for retrieving essential information from the data model to implement the effective rule-checking process.

Figure 11.3 shows the node network for executing this rule. The network carries out this geometric property checking process in four mentioned steps: object selection, rule definition, rule implementation, and validation reporting. The output of this network is a visual report that highlights the spaces which do not fulfill the requirement (Fig. 11.4).

11.7 Discussion and Conclusion

This paper proposes a new rule checking approach that integrates formal procedure of rule translation from compliant design requirements to LegalRuleML and its execution plan using the graphical scripting language. A simple example of early design phase requirements is used to demonstrate how the proposed approach carries out rule checking process from textual normative documents to final validation report of BIM model checking. The current practice of rule-checking in the AEC industry is conducted manually and is a time-consuming, error-prone, and costly process because of the extensive amount of normative requirements and the highly complex building design models. The rule-checking process consists of repetitive tasks that can be performed much more efficiently and effectively by machines. However, most of the previous attempts to automate the process have the inclination to hard-code rules into the checking system. These approaches also have shortcomings such as the lack of transparency, the need for extensive knowledge of programming, and inflexibility. This research team believes that using the open standard LegalRuleML in conjunction with VPL can address listed shortcomings and has the potential to provide unique contributions to defining and organizing rules consistently and implementing them interactively with BIM models.

Fig. 11.4 The highlighted room violates the minimum area rule

References

1. Athan, T., Boley, H., Governatori, G., Palmirani, M., Paschke, A., Wyner, A.: Oasis legalruleml. In: Proceedings of the Fourteenth International Conference on Artificial Intelligence and Law, pp. 3–12. ACM (2013)
2. Boley, H., Paschke, A., Shafiq, O.: RuleML 1.0: the overarching specification of web rules. In: International Workshop on Rules and Rule Markup Languages for the Semantic Web, pp. 162–178. Springer, Berlin, Heidelberg (2010)
3. Dimyadi, J., Governatori, G., Amor, R.: Evaluating legaldocml and legalruleml as a standard for sharing normative information in the AEC/FM domain. In: Proceedings of the Lean and Computing in Construction Congress (2017)
4. Dimyadi, J., Amor, R.: Automated building code compliance checking–where is it at. In: Proceedings of CIB WBC, pp. 172–185 (2013)
5. Eastman, C., Lee, J.M., Jeong, Y.S., Lee, J.K.: Automatic rule-based checking of building designs. Autom. Constr. **18**(8), 1011–1033 (2009)
6. El-Diraby, T.E., Kinawy, S.: Effective semantic web-based solutions for civil engineering. In: Proceedings of CIB W78 Conference, pp. 1–6 (2008)
7. Froese, T., Han, Z., Alldritt, M.: Study of information technology development for the Canadian construction industry. Can. J. Civ. Eng. **34** (7), 817–829 (2007)
8. IRC Web page: https://codes.iccsafe.org/public/document/toc/553/
9. Lee, Y.-C., Eastman, C.M., Lee, J.-K.: Validations for ensuring the interoperability of data exchange of a building information model. Autom. Constr. **58**, 176–195 (2015)
10. Lee, Y.C., Ghannad, P., Shang, N., Eastman, C., Barrett, S.: Graphical scripting approach integrated with speech recognition for BIM-based rule checking. In: Construction Research Congress, pp. 262–272 (2018)
11. OASIS Homepage: https://www.oasis-open.org/
12. Palmirani, M., Governatori, G., Rotolo, A., Tabet, S., Boley, H., Paschke, A.: Legal RuleML: XML-based rules and norms. In: Rule-Based Modeling and Computing on the Semantic Web, pp. 298–312. Springer, Berlin, Heidelberg (2011)
13. Preidel, C., Borrmann, A.: Towards code compliance checking on the basis of a visual programming language. J. Inf. Technol. Constr. (ITcon) **21**(25), 402–421 (2016)
14. Schiffer, S.: Visuelle Programmierung: Grundlagen und Einsatzmöglichkeiten, vol. 245. Addison-Wesley, München (1998)
15. Solihin, W., Eastman, C.: Classification of rules for automated BIM rule checking development. Autom. Constr. **53**, 69–82 (2015)