https://doi.org/10.1016/j.autcon.2019.102842

Rule Checking Interface Development between Building Information Model and End User

ABSTRACT

BIM-based rule checking systems are helpful for identifying potential concerns and flaws during the building design process. Currently, there is a lack of useful user interfaces for flexible rule manipulations and better tractability of the design intentions when rule violations occurred. This study aims to develop a BIM-based rule checking interface for users to address: (1) the current inflexible laborious coding process of rules, and (2) the low capability to describe rule influence comprehensively on the model designs. A research question is raised that whether the design adjustments against rule violations, such as making BIM elements changed, can be further traced their influence by an interface to prevent derived violations. A two-layered logic-based rule evaluation interface is developed to establish the rules aimed at meeting users' needs, and to identify design dependency topologically related to the involved BIM elements. A case study with a partial 7-story building section is used to evaluate the interface, and results show that the dependency between rule violations and their affected BIM elements can be successfully retrieved. It helps users to make the adjustment and identify influences of design changes to avoid further derived rule violations. A future direction of the study is capability evaluations for typical building engineers and architects to translate their design intentions and local rules by using the proposed interface with real building projects.

Keywords: Building Information Modelling (BIM); Rule Checking; Interface; First-order Logic; Construction Rules

1. Introduction

Building Information Modelling (BIM), which supports collaborative project delivery, has attained widespread attention in the architectural, engineering and construction (AEC) industry (Teng et al., 2019; Li et al., 2019). With the increasing maturity of BIM tools and opportunities they offered, people can make more sophisticated and detailed design models of the buildings promptly. However, simply relying on visual inspection to identify whether BIM models comply with the design requirements is no longer practical due to the high complexity. Complicated design logical flaws could be hidden and not always presented in drawings or properties from BIM models. More abstract and semantically rich information is necessary to be dealt with nowadays. Automation of checking, taking advantages of BIM models, where well-defined rules can be applied automatically with minimum user intervention, is increasingly needed (Solihin and Eastman 2015). BIM-based rule checking systems, the tools developed utilizing rules and properties of BIM models to check for the compliance, are evolving and getting mature. However, current BIM-based rule checking systems are implemented without flexibility and potential to discover logical conflicts behind the design. Users are not able to add, adjust or modify the rules within the system. For example, BIM software, Solibri Model Checker (SMC), contains hard-coded rules related to space management and accessibility (Preidel and Borrmann, 2015). It could be of benefit to BIM users, especially for the designers, to have more capabilities to examine whether their designs fit general regulations as well as their "own" design rules or higher-level design intentions. Also, having flexibilities to adjust the rules when it is necessary allows the creation of cost-efficient solutions considering different types of rules or the potential changes of rules. This research aims at providing an interface that is flexible and user-friendly without the need for hard coding to perform rule checking. It is expected that the interface will reduce customized hard coding and system development effort and the user may perform BIM-based rule checking based on different needs of the project stakeholders.

The paper begins with an investigation of the classification of rules depending on the levels of

interpretability. Their corresponding BIM-based rule checking systems that are in place are also reviewed. A user interface design is then proposed, which enables users to set the rules to be checked to meet their needs. Subsequently, a case study is further conducted following the user interface design to validate its usage. The user interface was examined using a partial BIM model of a project in Taiwan to demonstrate its merits.

2. BIM Rule Checking Systems

Automated rule checking or compliance analysis of buildings has been an active area of research since the 1960s (Sharpe and Oakes, 1995; Ismail et al., 2017). To facilitate a computer-interpretable checking mechanism, researchers put tremendous effort into developing representation methods to present and reason with engineering knowledge (Fenves, 1979; Hakim and Garrett, 1993). Further developments regarding the modelling of design standards are categorized into several directions, including predicate logic-based approach (Rasdorf and Lakmazaheri, 1990), object-oriented approach (Garrett and Hakim, 1992), and description logic (Hakim and Garrett, 1993). Many related knowledge-based and expert systems (Heikkila and Blewett, 1992) have been developed based on the above approaches, despite some of the drawbacks and limitations were also raised through the studies. For example, logic-based approaches are set to specify sufficient design conditions instead to represent the necessary structure for membership in a design concept, while object-oriented approaches are the opposite in what they are capable of representing (Hakim and Garrett, 1993).

Given the development trajectory of representation methods with engineering knowledge, BIM in the AEC industry provides opportunities tackling issues of indirect and fragmented information and further derives all sorts of checking applications across the project. One of the promising directions of BIM checking applications is to facilitate simulations and evaluate building designs in the earlier project stages, such as conception and final design (Eastman et al., 2011; Dimyadi and Amor, 2013). Conventionally, such evaluations were performed by specific domain experts by

manual approaches, which require considerable time to complete. The processes were tedious, error-prone and often very costly. Even when supported by additive information technology systems, platforms or tools, they usually require domain-specific criteria and analysis assuming the form of spatial requirements and numerical constraints. (Martins and Monteiro, 2013)

With BIM, automated checking interfaces and simulations can be provided for users making such processes quicker and more reliable (Han et al., 1998; Ding et al., 2006; Eastman et al., 2011; Luo and Gong, 2015). The written codes can be translated through semi-automatic encoding processes into programmable ones at a certain level of flexibility. The building designs can also be validated using the encoded rules by rule checking engines established upon BIM platforms. The methods of encoding processes at current practices, for example, include using programming languages such as EXPRESS (ISO 10303-11:2004, 2004) or BERA (Lee et al., 2015a; Preidel and Borrmann, 2015) to help users to improve the interpretability of the existing rules and formats like IFC (Industry Foundation Classes, which is a commonly-used collaboration format in BIM-based projects) schema (Ding et al., 2006). The fields of BIM-based rule checking applications may include structure design (Nawari, 2011), model correctness (SMC, 2015), constructability (Jiang and Leicht, 2015), accessibility (Lee et al., 2015b), safety (Zhang et al., 2013), facility management and maintenance, and other aspects of the project (Pahl and Beitz, 1996).

The current BIM rule checking systems can be classified into four categories, targeting the corresponding levels of checking rules based on Solihin and Eastman's review (2015). It depends on the level of interpretability the rules encapsulated. The classification is investigated in detail with state-of-the-art applications. Current limitations and potential improvements which motivate the rule checking interface design in this research have been identified.

The Class 1 rules need a single or a small number of explicit data, indicate that they can be generated directly through specific attributes or references from the dataset of BIM. For example, a certain kind of property or entity information on the components of BIM model, such as a door or window type for a firewall, should be checked through identifying whether the existing design has

chosen correct types without violations (SMC, 2015). Tools used in supporting such rule checking made use of various IFC toolkits (IFC2016). Many BIM automated rule checking platforms, such as Solibri Model Checker (SMC, 2015), Jotne EDModelServer (Jotne, 2016) and SMARTcodes (Conover, 2007), were all developed to support such level of rules.

Class 2 rules are the ones that require simple derived attribute values. These refer to those rules that were generated through extended entity with a higher level of semantic content but at the same time not creating any new data structure. The level of rules can be derived merely through the model component's properties, such as space arrangement in front of a door, reserved space, etc. Sometimes, it also requires resolving combinatorial issues of multiple rules on the derivation processes. SMC and CORENET ePlanCheck system (Khemlani, 2005) can be representatives in supporting and checking this level of rules.

Class 3 rules are the ones that require an extended data structure for a further rule checking. The spatial relationship of an overall building's design can be an example, the links of certain objects and attributes are implicit and require further operations to establish complex topological structures for the interpretation thereof (Borrmann and Rank, 2009). FORNAX (novaCITYNETS, 2015) as implemented in CORENET ePlanCheck and GSA building design circulation checking based on SMC (Lee, 2010) have such derivation but with the limited effort for checking this level of rules.

Class 4 rules do not necessarily to be used for identifying compliance. Instead, they are used to examine whether the design can be the "proof" of the solution. The applications in supporting such a level of rules are usually more concerned with the solutions having more than one acceptable answer (Solihin and Eastman, 2015). Therefore, knowledge-based approaches (Zhang et al., 2013; Jiang and Leicht, 2015) support the rule interpretation behind the checking mechanisms. This level of rules is considered as a new era of rule checking with only few research outcomes to enforce currently. The research work of Zhang et al. (2013) regarding safety rule checking can be an example of such a preliminary effort.

Based on different levels of interpretations and required functions, the relatively mature

125 platforms for automated rule checking systems are listed in Table 1. Projects, input, output, and rule
126 interpretation approaches are presented.

127 **Table 1. Rule Checking Platforms and Projects (from Eastman et al., 2009)**

Name	Projects	Descriptions	Project Input/Output & Rules Checked	Project Rule Category	Project Rule Interpretation
FORNAX (novaCITYNETS, 2015)	CORENET ePlanCheck (Singapore) (Khemlani, 2005)	FORNAX platform can be used to capture the semantic of building components at a high level for code checking. The platform provided a development and deployment environment for capturing building codes. It can also be divided into three modules: FORNAX geometry engine, FORNAX object, and FORNAX checking engine (Khemlani, 2005). Its representative system is CORENET ePlanCheck. The objective of the system is to provide a series of IT applications that could automatically check the building model for compliance with regulatory requirements, using BIM technologies. It was undertaken by the Singapore government agency in support of IFC and the BIM.	Input: IFC Model (Encapsulated by FORNAX geometry engine) Output: A web-based report to show the checking results regarding whether the model follows the building technical regulations, fire codes and so on. Rules Checked: Building code	Class 2 and 3	Hardcoded through a computer programming language. (by the programmer)
dRofus (2015)	HITOS project (Norway) (Lê et al., 2006)	dRofus provided a user-friendly interface allowing users to manage spatial data and compare planning information with the actual one from the IFC model. It also supports collaboration through the Internet for projects' data on the server. The received data is the one with progress information from different building models. It does not handle visualization or support geometry manipulation. In the HITOS project, it has been used on several college buildings for checking and managing spatial program requirements for building specific rules.	Input: BIM Model Output: The spatial program review report showed the comparison, discrepancies, requirements and actual space area values. Rules Checked: Spatial area	Class 1	N/A
Express Data	Cooperative Research	In the CRC for Construction Innovation project, Design Check is developed based on a feasibility study using SMC and EDM model	Input: IFC Model	Class 1	Using EXPRESS language (ISO 10303-11:2004, 2004) to define

Name	Projects	Descriptions	Project Input/Output & Rules Checked	Project Rule Category	Project Rule Interpretation
Manager (EDM) Model Server (Jotne, 2016)	Centre (CRC) for Construction Innovation – Design Check (Australia) (CRC, 2016; Ding et al., 2006)	server, and it has been adopted in the construction industry of Australia. The Design Check provided a graph for adjacent space accessibility inferring and checking. It encodes the design requirements of building codes as object-based rule checking system using EDM.	Output: An interactive reporting page with print-friendly functions. Users can check analysis results based on clauses or selected types of objects. Rules Checked: Accessibility		rule schema manually. (by the programmer)
SMART-codes (Conover, 2007)	International Code Council (ICC) (USA) (2015)	SMARTcodes provided functions to transfer written building codes into computer-interpretable ones. In the ICC's project, the development of SMARTcodes is mainly for automating and simplifying compliance checking against the ICC codes and different derivations of the codes including federal, state, and locally adopted versions. It has been implemented as extensions on Solibri Model Checker (Digital Alchemy, 2015) and Xabio (AEC3, 2015), supporting the rule-based building code compliance checking.	Input: IFC Model Output: The software provided checking results to the end user in various document file formats, such as HTML, PDF, RTF, XLS, and XML. Rules Checked: Building code	Class 1 and 3	Using SMARTcode Builder by defining key phrase through an established dictionary.

Name	Projects	Descriptions	Project Input/Output & Rules Checked	Project Rule Category	Project Rule Interpretation
Solibri Model Checker (SMC, 2015)	HITOS project (Norway) (Lê et al., 2006) General services administration (USA) (Lee, 2010)	SMC, as a BIM validation software, provided various built-in functions. For example, it has a library for model pre-checking, such as space overlap, property conventions, fire code, distance measurement, space program requirement. It has a module called Solibri Model Viewer which allows automatic viewing of checking issues, proprietary visualization and reporting format for design reviews. Also, SMC is developed based on JAVA and has capabilities to read an IFC model and map it to its internal structure, further facilitating access and processing (Ding et al., 2006).	Input: IFC Model Output: Textual and graphical reports in specific file formats, including PDF, XML, and XLS. Links to common BIM/CAD applications were provided. Rules Checked: Accessibility, circulation, and security	Class 1, 2 and 3	Using a parametric table to define and parameterize the rules. (by the programmer)

128

Current development trends of BIM automated rule checking systems are focused on facilitating building design validations at the early stages of a project. However, many BIM-based rule checking systems are implemented without encoding flexibility and users are not allowed to add, adjust or modify the rules. Also, little research is focused on designing an appropriate and general rule checking interface which may help to capture design knowledge from users to realize flexible rule checking and potentials of creating higher Classes interpretability for the future rule-based systems. Here the interface design is referred to a general configurable mechanism of rules to the users; it is different from other design concerns, such as the usability of interface layout arrangement or human factor considerations.

To develop an efficient and robust interface for the rule checking system, it involves various technical issues (Lee et al., 2012). One of the current issues is that the system focus is too heavy towards specific domains like the particular building code, space planning, accessibility, safety and structure (Tan et al., 2010). These designs were not generalized and caused increasing difficulty to be integrated with others. While the level of generality is the key to achieve semantic rule checking, user interfaces of rule-based systems should provide more flexibility in this matter. Besides an open system, to expand the use of rule checking methods, BIM-based rule checking requires more public approaches to develop enriched objects, enabling user-oriented rule defining mechanisms and model views. (Eastman et al. 2009; GSA, 2009)

Given the existing rule checking systems based on the rule classifications, most of the systems require defining and encoding rules through programming. When modifications or new codes at different fields are needed, they have to go through programming processes, either direct coding or indirect operations based on pre-coded mechanisms, all over again. Users are not allowed to input new values without such coding processes. These are potential issues hindering the development of automated rule checking systems for general purposes. Without flexible user interfaces to capture a higher level of semantic information directly from users, comprehensive databases cannot be established, and further analysis for creating knowledge (towards Class 3 or 4) cannot be achieved.

Therefore, research effort is required for improving the conventional approaches such as getting a higher level of semantic information and better flexibility of the encoding processes from different project perspectives. The interface created in this research is expected to be helpful for filling up the gaps mentioned above regarding semantic information capturing and flexibility.

3. Two-Layered Logic-based Rule Checking Framework

This research is focused on creating a flexible BIM-based rule encoding and checking interface, as well as on capturing higher level rules and intentions of end users, to facilitate the code checking from different project perspectives. A two-layered logic-based framework for BIM-based rule checking is proposed, and the details of the framework are described below:

3.1 Overview of the Framework:

The framework of the two-layered logic-based rule checking based on BIM can be seen as Figure 1. All BIM-based rule checking processes include firstly designing rules, establishing the associations between BIM elements and rule expressions, and performing corresponding checking afterward. Considerations can be added at two layers: rule operation and entity (BIM element) dependency.

Rule operation refers to the manipulations of rules including define, modify, eliminate and combine rule expressions. For example, local building regulations can be defined at this layer for being used by rule engines to perform checking on different BIM models. They can be modified at the same layer once the regulations are amended or the rules are applied to different countries. Normally the rules are hard-coded during pre-processing stages of rule checking. Instead, in the proposed rule operation layer, a graphical programming interface is provided for users to perform rules manipulation freely as well as to use the defined rules for checking at any time. The format of rule expressions is based on First-Order Predicate Calculus (Slaney and Paleo, 2018), also called First-Order Logic, which consists of variables, constants, and operators for indicating possible restrictions. All the expressions comply with associative laws of logic so that they can be combined

or split flexibly and at the same time maintaining logical meanings. The reason for adopting First-Order Logic with graphical programming interface at this layer is to integrate declarative and procedural knowledge representation formalisms of rules, further eliminate the limitations of lacking specific knowledge constraints and rule hierarchies. Given that object-oriented concepts can be implemented through the graphical programming interface, rule expressions, as well as advanced operations, can be encapsulated as pre-defined structures to further derive sophisticated rule expressions and logic relationships for further checking process.

Once the user is building the relationships between BIM models and rules, variables can be used to represent the associated BIM elements with their various property values. Through the queries to the BIM model (or database), rule checking results can be displayed at this layer for users' further reference.

Above the layer of rule operation, a layer called "entity dependency" is introduced, to determine how the defined rules affect their associated geometrical components (entities) in the BIM model. For example, a rule regarding the number limitation of stair risers would affect the design of stairs and further affect those of their connected floors in the BIM models. Such dependency due to the rules made at rule operation layer will be identified explicitly and automatically at this layer. A graphical representation is also adopted though users cannot manipulate the graph structure. This layer shows entities as individual nodes by retrieving their corresponding information from BIM associated elements, once rules at the rule operation layer have been established. If there are entities (nodes) being mentioned in the defined rule expressions, the edges, representing the rules at this layer, will be generated between specific node pairs. By examining the entire dependency graph at this layer, the users are expected to observe the dependency of BIM elements affected by rules for their further action. If there is a rule violation and the related design needs to be amended, the influence can be considered during the checking process to further identify any subsequent effects to other BIM elements through the graph connections. Also, the layer can be used to highlight the involved BIM elements on the model visualization platform to strengthen the users' interpretability against the

violation effects.

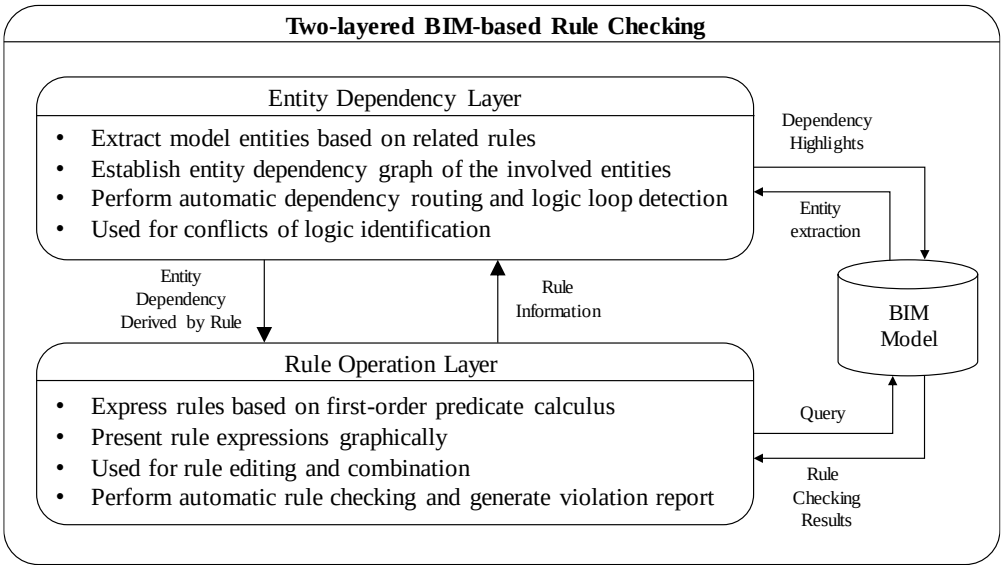


Figure 1. Rule Checking Framework

3.2 Rule Pattern Identification and Graphical Expression:

The rule definition process at the rule operation layer is based on a comprehensive survey of rule patterns. The authors focus mainly on building construction rules because they are representative ones with vital importance in building design. The data is collected locally in Taiwan given that the ease of the data collection and the applicability of the case study.

According to the different rules studied in the building construction rules (Building Technical Regulations, 2018), First-Order Logic has been identified as an appropriate pattern expression method for the abstraction of Class 1 and 2 rules. All the rule descriptions can be explicitly represented by specific sequences of the combinations of logic operators, variables and expressions. As can be seen in Table 2, a selected set of building construction rules has been classified into different rule patterns by using logic-based expressions. For example, a building construction rule states that "in building volume control areas, the area of each car park space in the building shall not be larger than 40 square meters." Then its rule pattern can be expressed as the third pattern of Table 2: "*Check if R1 <= 40*," where *R1* represents the area information retrieved from the corresponding BIM model component.

The related symbols used in the table can be seen below:

R1, R2, and R3: Real Number Value that a rule checking system can get from BIM

T: Text Value that a rule checking system can get from BIM

A1, A2, and A3: Real Number Input Value that a user input through a user interface according to the rules/regulations or codes

B1: Text Input Value that a user input through a user interface according to the rules/regulations or codes

C: The result calculated by a rule checking system

Table 2. Selected Construction Rule Patterns

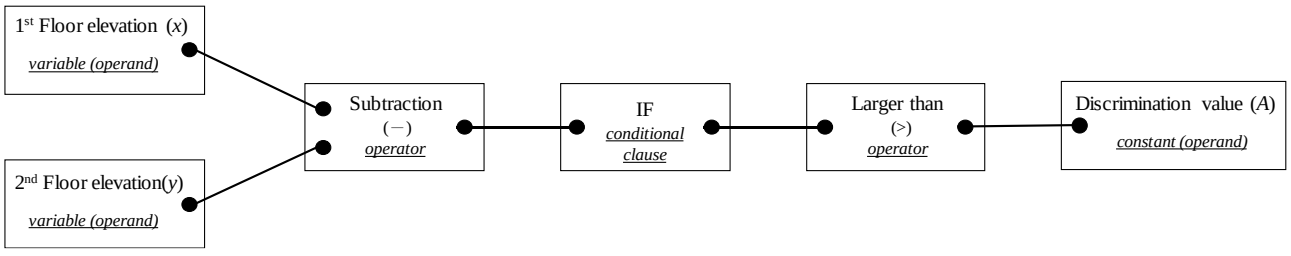
No.	Type/Pattern	Needed Operators	Needed Interface
1	*C = (R1 − R2) × A1*	=	1. Define arithmetic operators =, - and ×
		-	2. Input a real number value *A1*
		×	3. Retrieve real number values *R1* and *R2* from BIM
2	*R1 = ΣR2*	*Σ*	4. Define a summation *Σ*
	C = R3 / R1	/	5. Define a binary operator /
	Check if C > A1	>	6. Define a logical checking rule >
3	*Check if R1 <= 40*	<	7. Define a logical checking rule <=
4	*Check if R1 == R2*	==	8. Define a logical checking rule ==
5	*C = A1 * A2 / 0.8 / A3*		9. Define associative law
6	*Check if R1 / R2 == (A1 / 10) ^ A2*	^	10. Define a binary operator ^
	0 ≦A1 ≦A3		11. Scoping a value through two logical checking operators
7	*C = A1 × R1 ^ A1 – A1 × R2 + A1*		12. Define a single conditional description for determining a further action
	when R1 > A1		
	Check if R2 < C		
8	*Check if R1 >= 75 && R2 > 120*	&&	13. Define a logical AND operator &&
9	*C = 50 when*	*{*	14. Define associative conditional descriptions for further action
	{	*}*	
	A1 = 0.5 && A2 >= 5 && B1 = T		15. Retrieve a text value *T* from BIM
	}		16. Input a text value *B1*
10	*If R1 >= 30 || R2 >= 25*	||	17. Define a logical OR operator ||
	{		18. Define alternative conditions for a further judgement
	C = 36		

No.	Type/Pattern	Needed Operators	Needed Interface
	} *else if* *{* *C = 21* *}*		
11	$C = \sqrt{R1} \times A$	$\sqrt{}$	19. Define a unary operator $\sqrt{}$
12	$C = exp\ (6.61 - 0.345 \times A1)$ $A1 < 15$	*exp*	20. Define a unary operator *exp*

234

235 The rule operation layer for users to define, modify, eliminate and combine rules is thus designed
236 as a graphical programming environment. Users can simply drag necessary components, such as logic
237 operators, variables or texts, to form rule expressions. An example can be seen as Figure 2, two
238 variables, 1st floor and 2nd floor elevations, have been added to the layer and each of them is associated
239 with their corresponding elements' properties in the BIM model. They are then connected by a logic
240 operator "subtraction" as two operand inputs. The output of the subtraction can be represented as "net
241 height between 1st floor and 2nd floor" and can further be examined by a conditional statement "if"
242 and a conditional operator "larger than". A discrimination value is thus assigned and appended to the
243 conditional operator. The entire logic expression above can be interpreted as "Check if the net height
244 between 1st floor and 2nd floor is higher than the assigned discrimination value."

245



246

247 **Figure 2. Graphical Rule Pattern Expression**

248 A logic expression of the example mentioned above in text format can be referred to Eq. 1:

249 $$Largerthan(substraction(x, y), A) \quad (1)$$

where *x* and *y* denote as variables of floor elevations; *subtraction()* represents a function mapping a tuple of terms to another term which in this case a subtraction result; *Largerthan()* is a predicate variable showing a bool operation result through evaluating the given condition; and *A* is treated as a constant in the expression and used for the discrimination. In such graphical rule pattern expression, all the components and input/output connections can be configured by users and further merge or alter with other defined expressions.

3.3 Entity Dependency Graph and Logical Checking:

By analyzing the rules defined at the rule operation layer and their associations in the BIM model, it can identify specific dependency among BIM elements affected by the rules made. The BIM elements are presented at the entity dependency layer as "entities," and dependency among them are shown as connections between entities. They are formed as a graph structure automatically by computers.

Unlike the Class 1 or 2 rules presented by logic-based expressions explicitly at the rule operation layer, the dependency graph reveals abilities to describe derived and consequential data structure which can be considered when the defined rules are applied to different BIM models (Class 3 rules), such as spatial relationships. It also reveals a general topological structure with inner-relationships among intentions of the design, for users to have further alternative design decisions (Class 4 rules) based on rule checking results. Once any rule violation is observed, the related dependency based on the applied rule can be identified by the user to catch the design ideas and determine what else will be affected if further design adjustments take place.

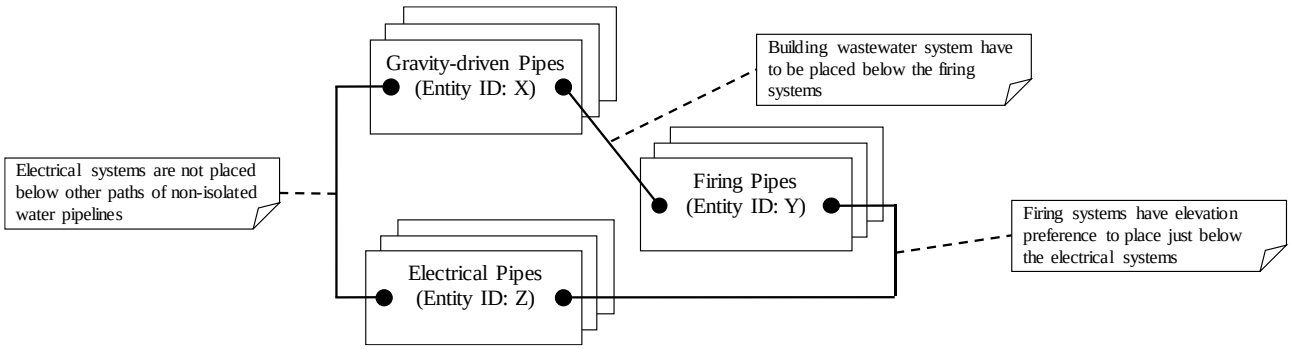


Figure 3. The Rule-affected Entity Dependency Graph

Take building Mechanical, Electrical, and Plumbing (MEP) systems as an example (as shown in Figure 3), the three batches of entities: (1) firing pipes, (2) electrical pipes and (3) gravity-driven pipes are connected due to the three rules defined for checking. The rule (dependency) between firing pipes and electrical pipes indicates that the firing system is preferred to be placed below the electrical system, due to the electric shock risk when the firing sprayers are functioning. Once a violation of such rule is observed, design changes on specific firing pipes are necessary. The user can further observe other dependency related to the firing pipes, such as the one between firing pipes and gravity-driven pipes. In this example, building wastewater pipes need to be placed below the firing pipes, and some of them should be taken care of when specific firing pipes are re-routed (the design changes due to the violation mentioned above). The dependency graph is helpful for identifying such derived design relationships among the BIM elements. It should be noticed that the rules defined in this example are not standard regulations but general design guidelines of MEP engineering. It is not necessarily as simple as this example in practice. Nevertheless, it shows the thinkable approach for users to consider the design inter-relationships due to rule effects.

As for the user-defined rules, the dependency graph also provides a checking mechanism for logical conflicts. As shown in Figure 3, for instance, a dependency loop is formed among different entities and users would have abilities to double check whether the rules defined along the loop are reasonable without contradictions. A logical definition of contradiction is that two statements are contradictory if they cannot both be true under any circumstance (Li et al., 2017). The definition is

loosened in the proposed graph for capturing human intuitions of incompatibility, given that topological relationships in every design work are mostly different. The graph is a better fit for the applications of recognizing discrepancies under the same design intention. If the rule between electrical pipes and gravity-driven pipes in this example is "Electrical systems need to be placed below the gravity-driven systems," then a contradiction exists due to an impossible implementation to fulfill all three rules along the loop. A violation would definitely occur. The user needs to either amend some of the rules along the loop or change the design to avoid the contradiction.

The two-layered logic-based rule checking framework has been adopted in the proposed interface of BIM-based rule checking system. For validating the uses of the interface, a prototype is carried out in this study. The related details of the interface, as well as the system, can be seen in the following section.

4. User Interface and System Development

The proposed BIM-based rule checking framework aims at providing a user-oriented interface which allows users to deal with the rules or design intentions flexibly. A system associated with the proposed interface is developed. For the demonstration purposes, the rules defined in the system are limited to the first five patterns listed in Table 2 above. Further effort is needed in the implementation of the other patterns in the future. More complicated logic expressions can be realized by adding more operators and defining their operator precedence.

The proposed system includes three modules as shown in Figure 4. The system is helpful for exporting the properties of all building elements from existing BIM software and import them into a database. Another database keeps a set of rule data defined by users through Rule Manipulation module. This module can be treated as an implementation of the rule operation layer. The user firstly defines the necessary rules to relate specific elements in BIM models. These are done by going through the definition of basic terms of rules which will be used as operands in the expressions. If it is necessary, derivative terms, such as terms defined by other terms, also need to be defined. Then the

rule can be generated by combining operands with operators and values. Operators indicate what kind of calculation needs to be performed for the rule checking process and discriminate values represent the limitations or restrictions used to judge whether the current design values comply with the rule. Once the rule has been formulated, the rule checking process can be executed. Another module called Compliance Analysis is thus used to call these pre-defined rules, perform the rule checking and retrieve the checking results. In the meantime, a dependency graph will be built after these operations, which realizes the entity dependency layer in the system. At the end of the process, the Review module is responsible for showing all the reports, including rule violations and the detection of the dependency, for users' further reference.

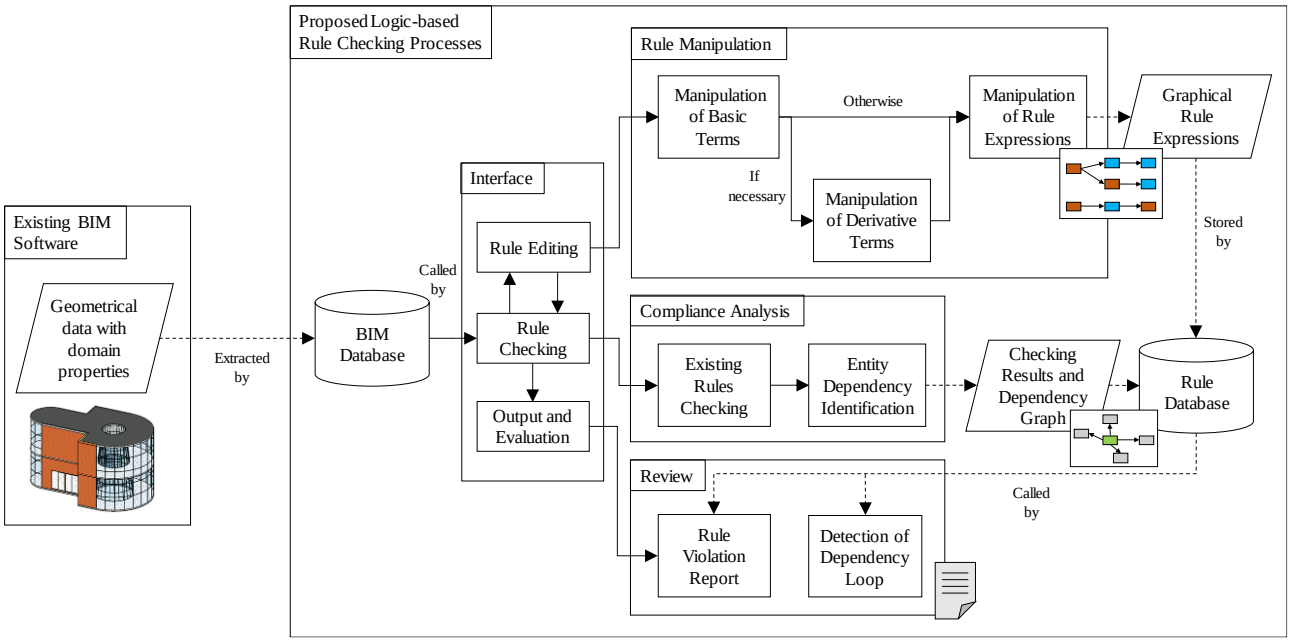


Figure 4. Three Modules in the Proposed System

Detailed functions of the individual modules can be seen as follows:

Rule Manipulation Module:

- A function to define values with arithmetic operators, including +, -, ×, and /
- A function to allow the user to input real values or the ones from BIM elements' properties

- A function to define logical checking operators, including >, =, and <
- A function to perform Σ calculations
- Moreover, the interface provides a function to name the rules and to categorization

Compliance Analysis Module:

- A function to get the values of the terms and perform logic inference according to the defined rules
- A function to compare the inference results with the discriminative values in the rule expressions, and show if the results comply with the rules
- Moreover, this module is also used to identify the related BIM elements being affected by the defined rules and to generate the dependency graph automatically

Review Module:

- A function to gather and display rule checking and dependency results in a format of reports
- A list for users to retrieve necessary checking results without tracing back to graphical representations on the interface

5. System Implementation

The proposed system is implemented by applying Microsoft Visual C# and Windows Presentation Foundation (WPF) to build up the interface and rule engine, and applying Microsoft SQL Server to create a database for BIM data storage and process. Revit, as a BIM model design platform, is used in this study. TUM.CMS.VplControl, an open source library for visual programming interface implementation (Preidel and Borrmann, 2015), is adopted for developing graphical representations of the rule manipulation and entity dependency layers. The reason to choose these implementation tools is because of their flexibility and efficiency to establish a user interface and related rule checking mechanism. The focus is put on proof of the concept, and the authors did not

utilize the state-of-the-art rule engines or generate a comprehensive rule checking system. The implementation details are described below:

5.1 Overview of the interface

The overview of the implemented interface can be seen as Figure 5. In general, the graphical representations of logic-based rules can be shown and manipulated individually on the visual programming panels created by the "+" tab (as shown in Figure 5a). Each panel can be assigned a new rule name (Figure 5b). Afterward, users can create graphical components by simply right-clicking on the panel and selecting the necessary type of the templates, such as variable, text, operator and numbers (Figure5c). The components are further connected by users through simply dragging lines among them. Once the manipulation of the rules is done, users can execute the checking process through the "Execute" button (Figure 5d). And consequently, a dependency graph will be generated and shown on a new panel with tab name as "Entity Dependency" (Figure 5e). Users can thus click on the "Report" button to see the checking results in a list (Figure 5f). All the rule violations can be traced back to related rule panels or BIM elements.

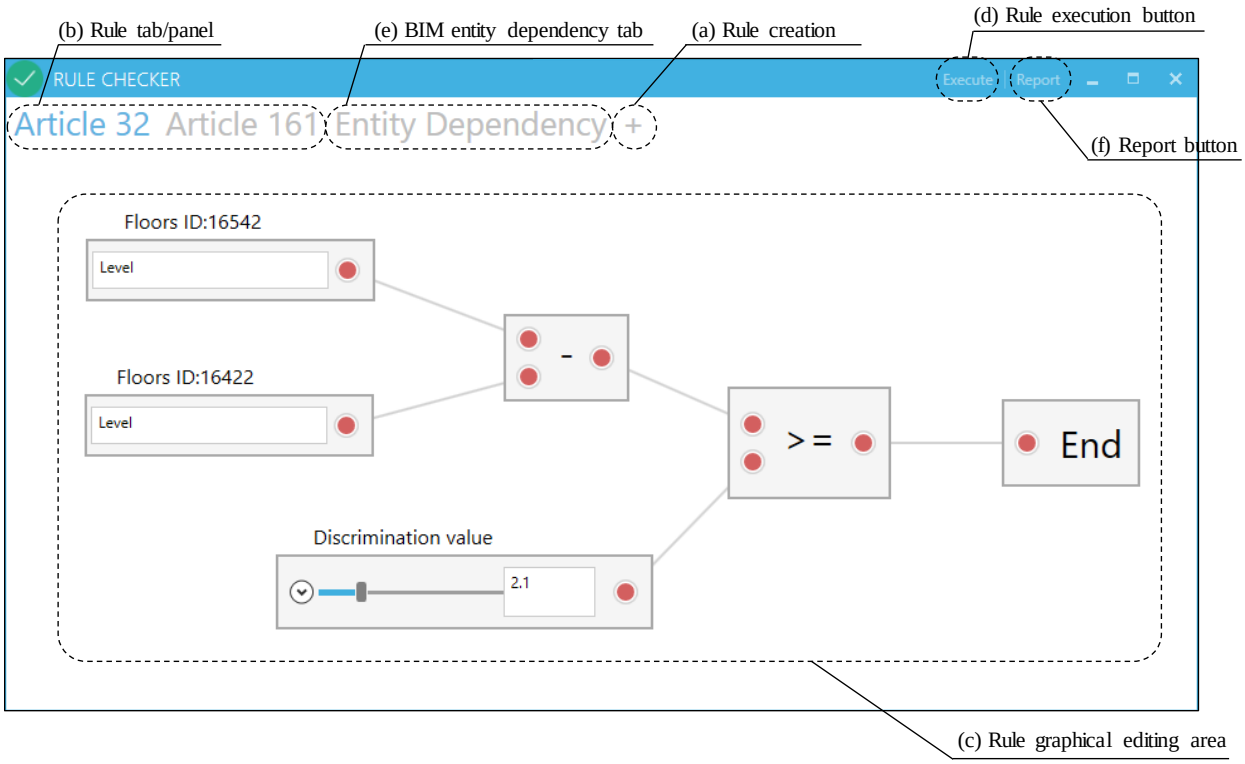


Figure 5. Overview of the Logic-based Rule Checking Interface

5.2 Defining Rules and Checking

To define a rule through the visual programming interface, users are free to generate different graphical components and assign connections to them. The rule regarding the clear height of floors is used again as an example. Its representation can be seen as Figure 6. Users need to create six graphical components: two variables "1st" and "2nd floor elevation", two operators "-" and ">=", one real number "Discriminative value" and one rule enclosure. They are connected as shown in the figure. The rule can be interpreted as "Check if the net height between 1st floor and 2nd floor is higher than 2.1 meters."

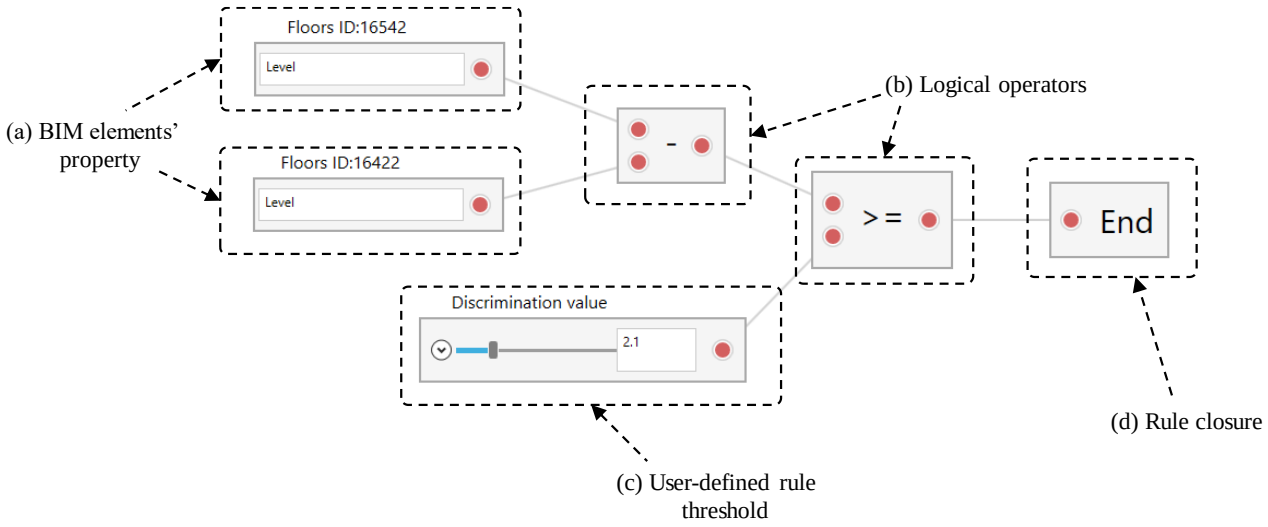


Figure 6. Graphical Representation of a Rule Description

It should be noticed that the conditional statement "if" is not used due to no alternative action needed based on this rule expression. Also, each of the variable need to be either assigned a value (or multiple values) or connected with those of their related attributes of BIM elements beforehand. In the latter one, users can double-click on the variable components. A dialog, as indicated in Figure 7, will pop up. Then users start to define the linkage by choosing corresponding "BIM Element Type," "Property," and specific instance(s) "Element ID," either through picking up the elements in the BIM model or selecting the items through a drop-down list retrieved from the database. The value of the

variable is thus gained from single or multiple BIM elements it has been linked with. Optionally, a sum value of multiple BIM elements can also be linked. If the variable has already been defined, it can be directly created by next time. Some BIM element types, as well as their properties in the database, can be referred to Table 4.

Table 4. Selected BIM Element Types and Related Properties in the Database

Element	Properties
Conduits	ID, Category ID, Demolish Phase, Construction Phase, Design Option, Service Type, Size, Remarks, Outer Diameter, Inner Diameter, Bottom Elevation, Top Elevation, Diameter, Length
CurtainPanels	ID, Category ID, Demolish Phase, Construction Phase, Design Option, Area, Instance ID, Width, Height, Remarks, Floor, Cost Item, Schedule Item
Floors	ID, Category ID, Perimeter, Demolish Phase, Construction Phase, Area, Design Option, Remarks, Structure, Glosses, Offset Height to the Story, Analysis, Estimated Reinforce Volume, Story, Volume
Stairs	ID, Category ID, Demolish Phase, Construction Phase, Design Option, Remarks, Actual Riser Number, Top Floor, Offset to Top Floor, Offset to Bottom Floor, Actual Riser Height, Width, Bottom Floor

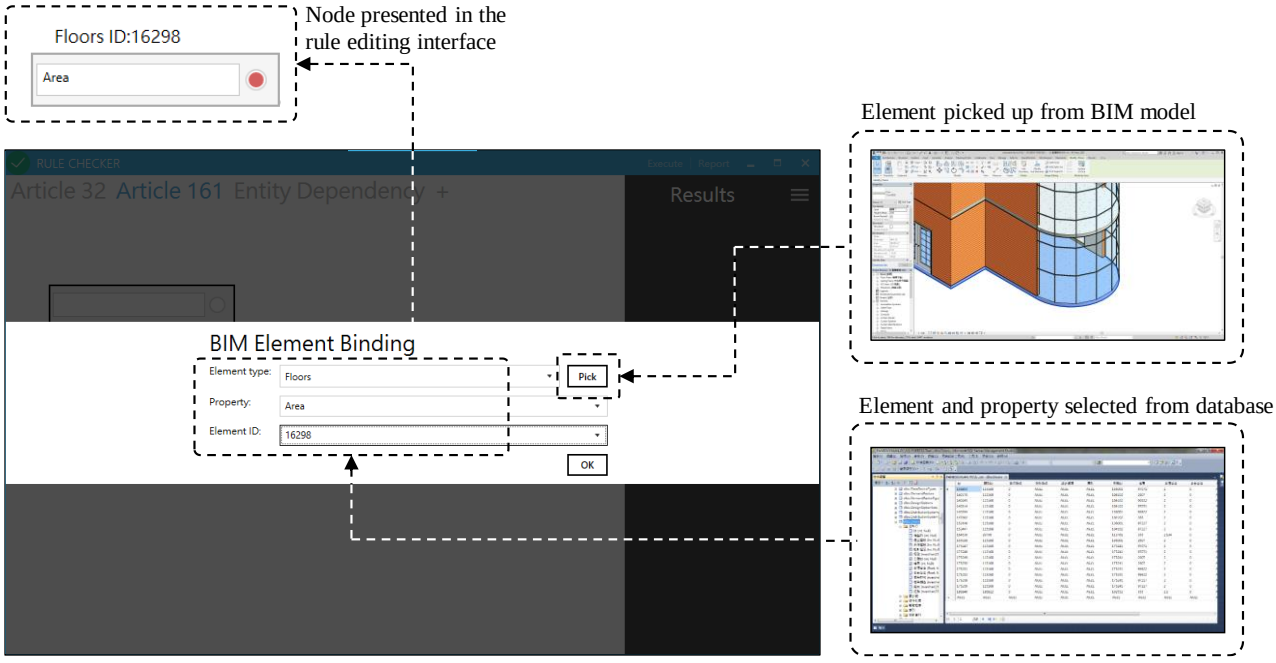


Figure 7. The Linkage between Variables and Related Properties of BIM Elements

5.3 Reporting

The report of the rule checking results shows as a list for users to see whether the current designs of the BIM models are complying with defined rules. Once the user double-clicks on the specific item of the list, detailed violation information will be shown accordingly. Moreover, the dependency of BIM elements affected by rules is also generated as well. An example of the dependency graph for the floor clear height rule is shown in Figure 8. By clicking on the connection between any two connected entities, the system will bring the users' focus to the corresponding rule panel for a further examination. Figure 9 shows general tracing actions through the reporting interface, including checking the list of results and tracing highlights through the entity dependency graph as well as individual BIM elements.

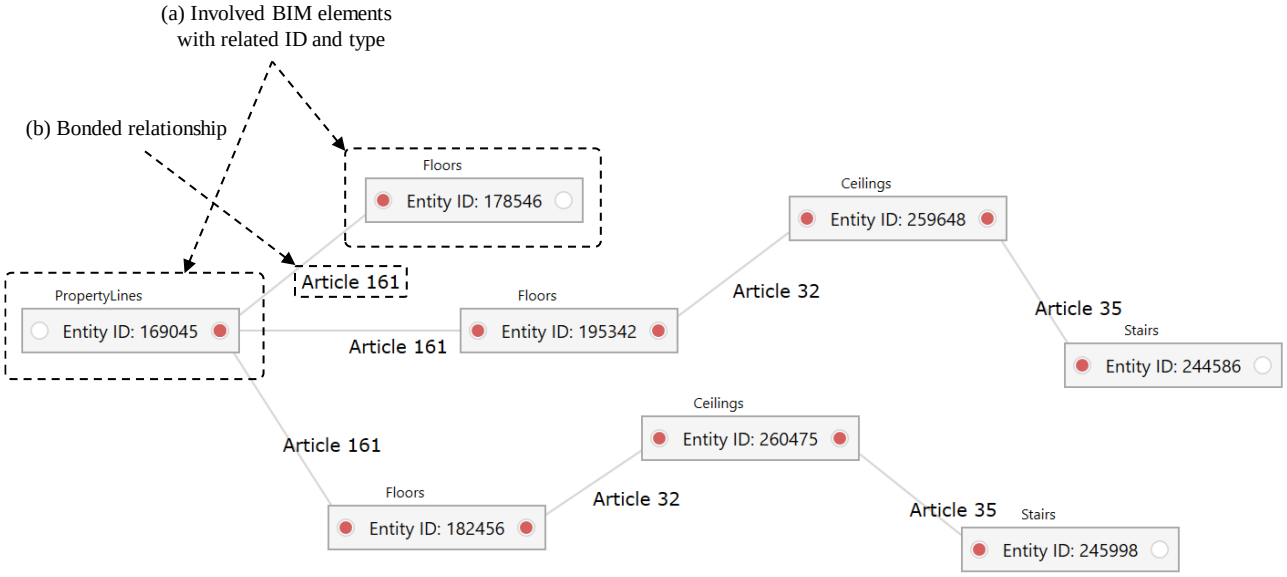


Figure 8. A Dependency Graph of Involved Rules and BIM Elements

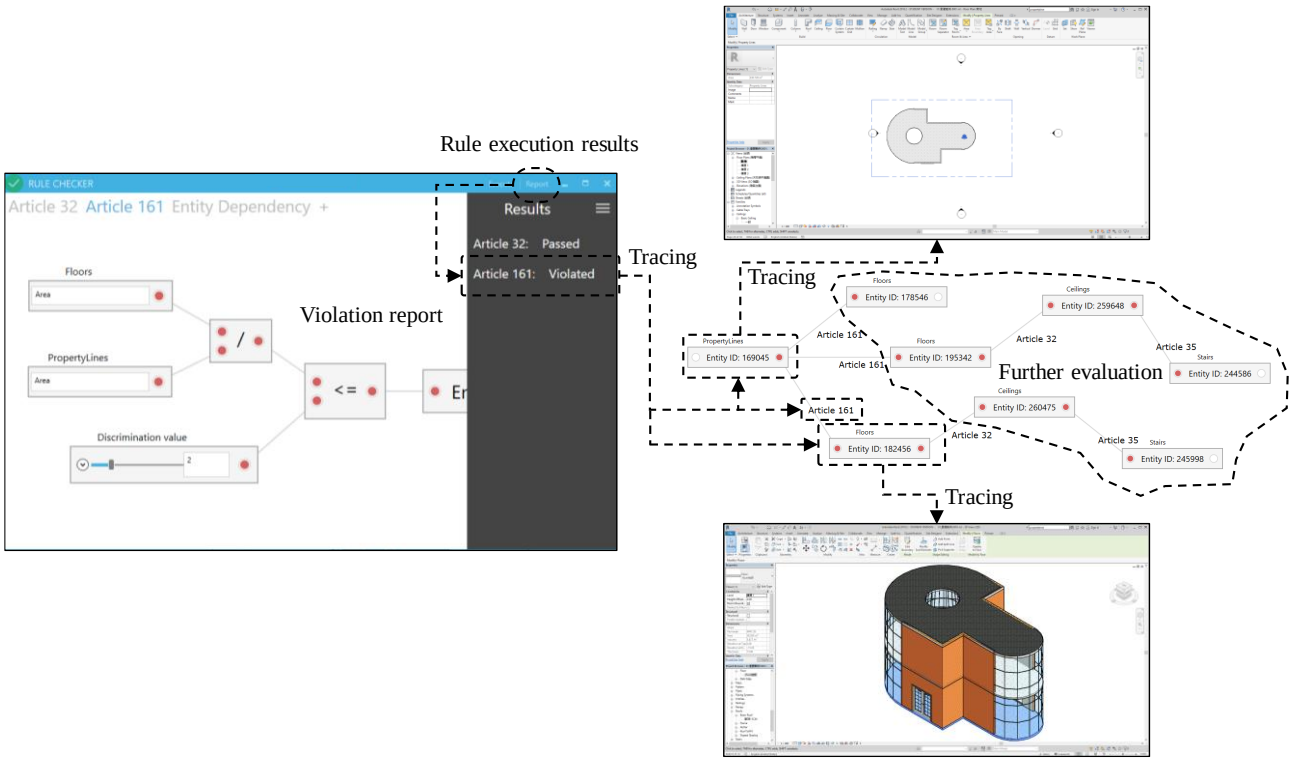


Figure 9. Rule Checking Results

6. Case Study

To demonstrate the use of the proposed rule encoding and checking interface, a revised BIM model based on an actual project in Taiwan consisting of a partial 7-story building section is used as a case study. Further details can be seen as follows:

6.1 Case Overview

The building section is shown in Figure 10. The elements of interest within the BIM model are listed in Table 5. Particular rules, based on Building Technical Regulations for Design and Construction (2018) in Taiwan and mainly related to the main structure of buildings, have been defined through the rule manipulation module of the proposed system. Due to a significant amount of BIM components with the same property values, such as floors made by different finishes but with the same elevations, only partial and representative components have been selected to link with the corresponding variables through the interface. An example of the rule definition procedure is given

and described as follows:

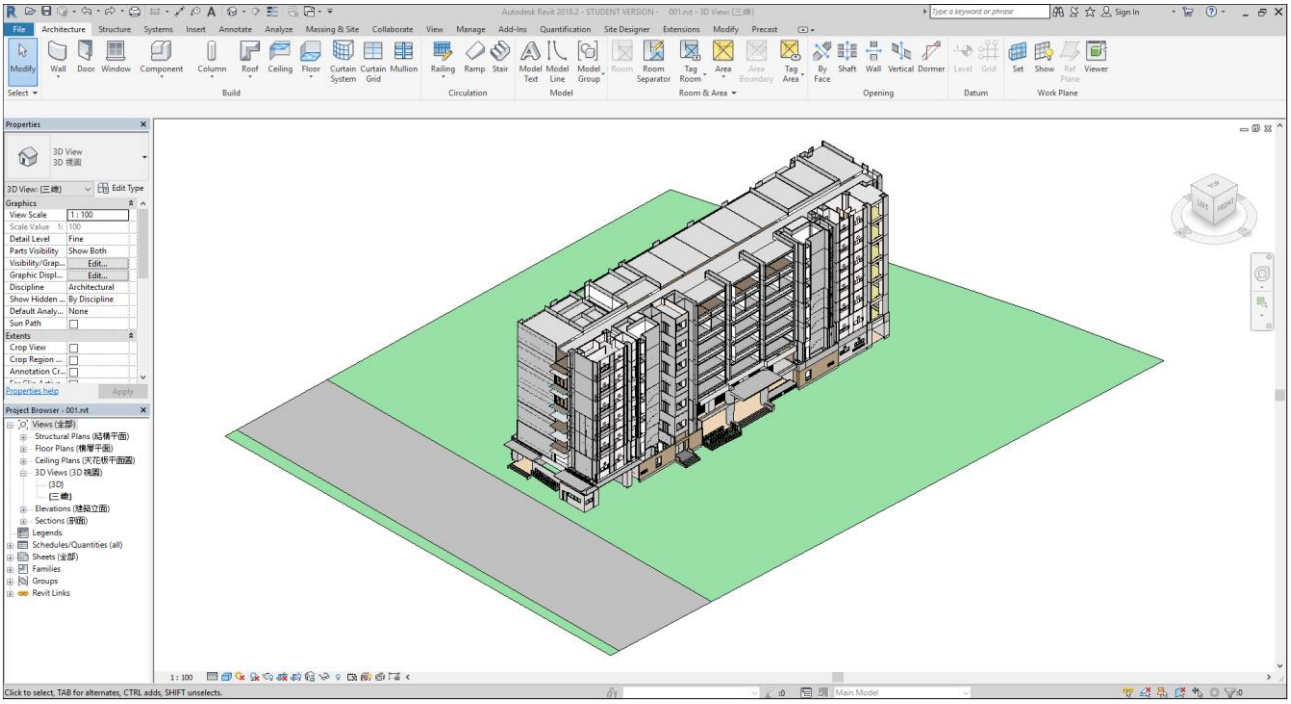


Fig. 10. The Overview of the Example Case

Table 5. BIM Elements of Interest in the Example Case

Elements	Quantity*	Remarks
Ceiling	163	Made in different materials, including light aluminum/steel plate, mineral fiber, PVC with fire-proof painting
Room	110	Areas of the rooms ranged between 4.14m^2 (Electricity distribution room) to 800.15m^2 (Office)
Floor	215	Concrete structure (150mm-800mm) with the surface made in different materials, including slate, granite, PVC/quartz tile
Wall	4010	Including concrete RC wall, light-weighted partition, curtain wall, light aluminum/steel plate, PC board
Window	93	Including fixed, 2-open, and Venetian blind. Some of the glasses are made in colors
Door	230	Including single entrance door, sliding, folding and door for the elevators. The materials have been used include timber and steel plate.
Stair	30	Interior cast-in-place stairs for different levels
PropertyLine	1	Used to define the total lot area of the building
Road	1	Nearby the Building

* enumerated by instances in the model

According to Article 161 of the Regulations, Floor Area Ratio (FAR) is the relationship between the total amount of usable floor area that a building has or has been permitted for the building compared to the total area of the lot on which the building stands. This ratio is determined by dividing the total or gross floor area of the building by the gross area of the lot or lot size. A higher ratio is more likely to indicate a dense or urban construction project. According to urban planning regulations, there are different limits of the ratio in different zones. For this example, the limit is 200%.

There are two variables to be defined:

1. "Total Floor Area," as shown in Figure 11(a): The user defines it through selecting element type "Floors" and property "Area," and then chooses "Sum All" for summing up the values of all the associated BIM elements. This variable refers to the total floor area and is defined as the summation of the area of each floor element in the model.
2. "Total Lot Area," as shown in Figure 11(b): The user selects the element type "Property Lines" and property "Area," and "Sum All" for all the elements. This variable refers to the lot size and is defined as the summation area of each lot size as determined by the Property Lines.

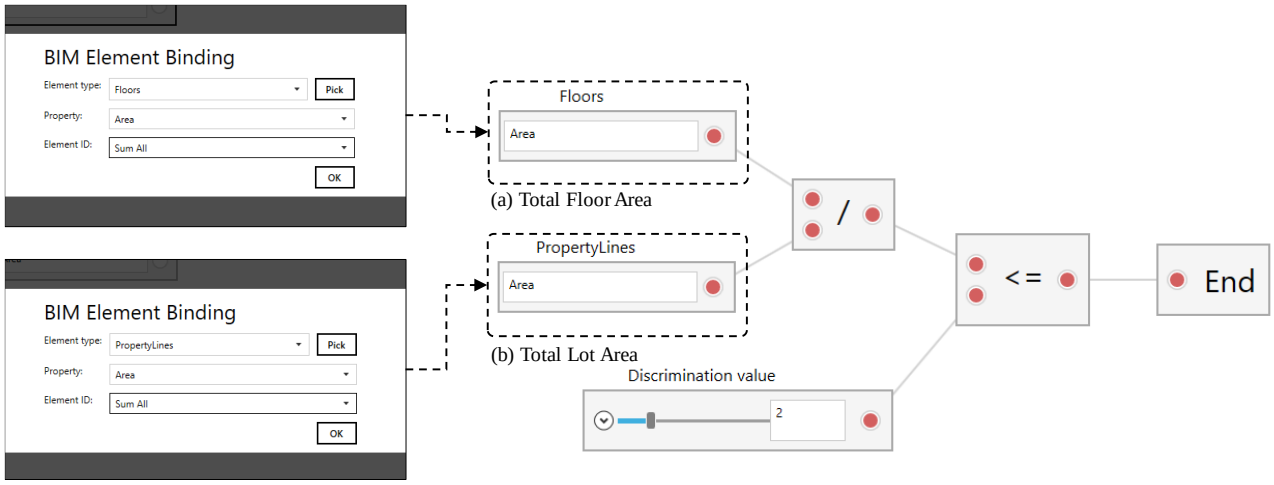


Fig. 11. Defining Variables for Floor Area Ratio Checking: (a) Total Floor Area and (b) Total Lot Area

After these two variables have been defined, the user names the rule as "Article 161" and drags necessary graphical components to form the rule. The rule can be seen in Figure 11, and it indicates "the Total Floor Area divided by Total Lot Area should be smaller or equal to (<=) a discriminative value 2," which means that the "FAR" should not be greater than two times (200%). Similarly, the user can define other rules based on a similar process. Afterward, the user presses the "Execute" button to see the rule checking results.

In this case, rules of interest are selected presuming the following fact is known in advance: their affected BIM elements have a certain level of overlapping. It is expected to see the potential entity dependency and to show the inference ability of the checking process, as long as there are rule violations which cause design changes to become necessary. A successful example is shown in Table 6. Five rules, related to height restrictions and the FAR mentioned above, made different constraints and affected the design of totally six types of BIM elements, including "Roads," "Floors," "Ceilings," "Stairs," "Beams" and "PropertyLines". Those BIM elements are linked through variables among the rule expressions. The "Roads" component is customized in this case, with a property "Width" to indicate its corresponding widthways measurement.

It took around 3 seconds for the system to traverse through all the rule expressions and conducted the checking results according to the five defined rules, with 36 representatives BIM elements being involved. The checking result of this case is shown in Figure 12, where the users got only one violation message (Article 14) on the sidebar of the interface. And the related BIM elements of such violation can be highlighted once the users double-click on the message. Furthermore, a description regarding the violation details can be displayed on the interface. In this violation message, the width of the road is 9.2 meters. After a series of calculations based on Article 14 shown in Table 6, the width is further multiplied by 1.5 and plus by 6; we have a value equal to 19.8. According to Article 14, this value is supposed to be larger than the height of the building, which is 22.4 meters in this case. While it does not make the case, a violation is valid.

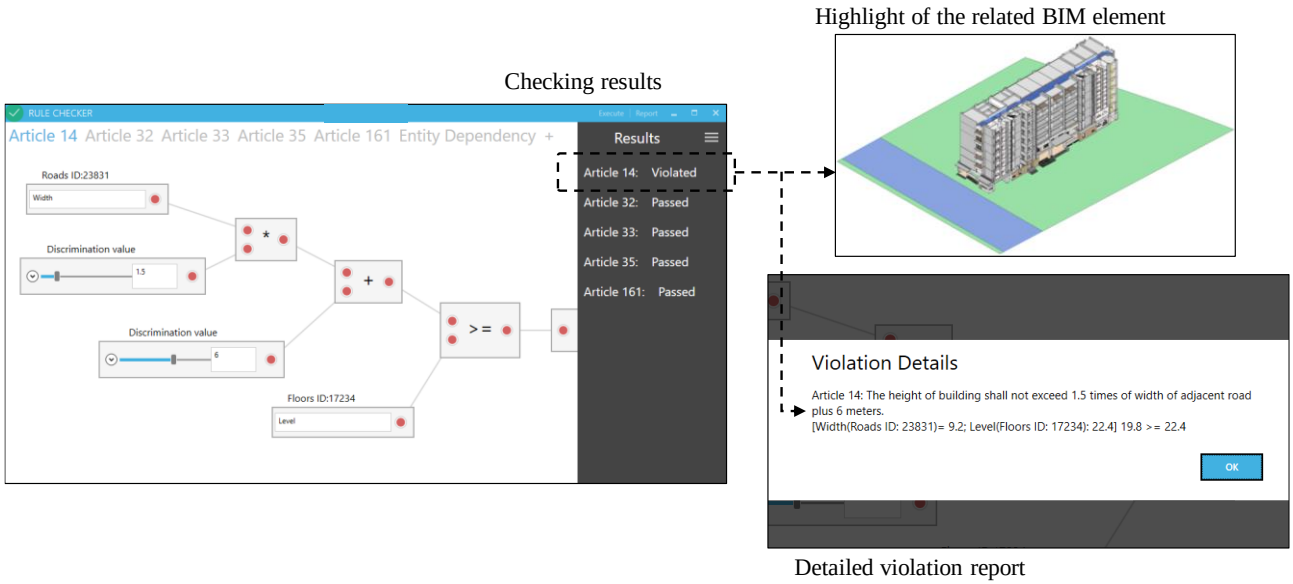


Fig. 12. The Rule Checking Results based on Five Selected Building Rules in this Case

6.2 Discussion

After the study based on the above case, rule checking results, as well as entity dependency are presented. Observations and discussions regarding the case demonstration have been summarized as follows:

1. By showing the inner-relationships of BIM elements affected by rules, it reveals that the proposed interface, given the rule formulation in the case, has the potential to achieve the checking at higher rule classes. The influence of rule compliance can be identified based on the generated entity dependency graph, such as the example shown in Table 6. For instance, the dependency exists between the floor elevation attributes of the building and the width attributes of the nearby road. Based on Article 14 of General Architectural Design Specifications in Taiwan: "the height of the building in building volume control areas shall not be larger than 1.5 times of the width of the road nearby plus 6 meters." In a case of violation, a reduction of building height may influence the elevations of the seven building floors. Based on the clear height restriction mentioned previously (Article 32), no further reduction can be made if the elevations of the floors, in this case, are already the lower bounds of the restriction (2.1 meters). Additional design changes, even the reduction of the numbers

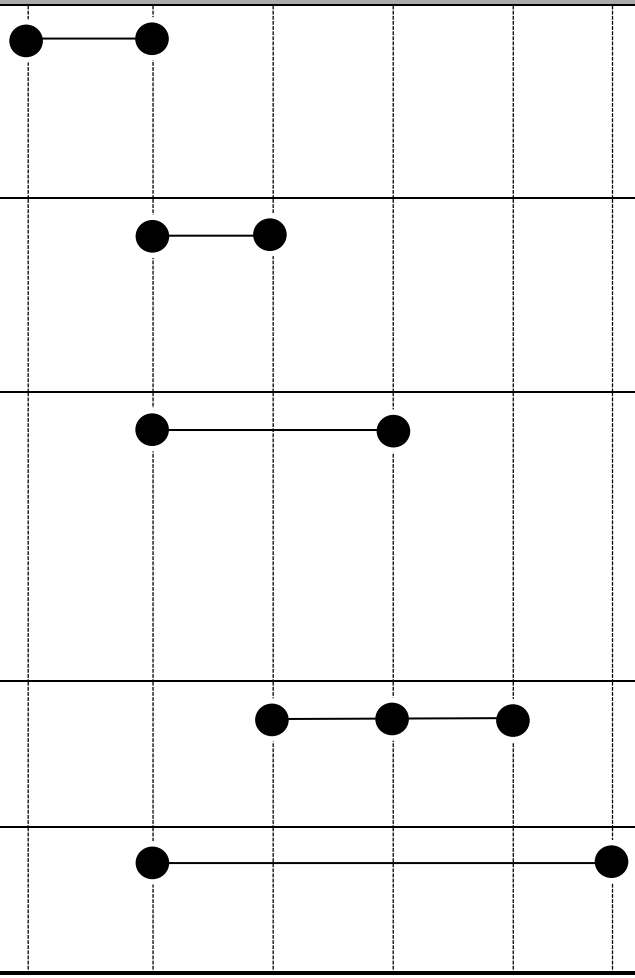
of floors, need to be considered to avoid the potential impossibility. Once the changes have been made, the dependency graph is also helpful for tracing forward to other related BIM components, such as Ceilings, Stairs, Beams, and PropertyLines in the table, as well as their related rules (Article 33, 35 and 161) for further examinations. A study through a real construction project with design checking team can quantify the values of the proposed interface in the future.

2. It is feasible to conduct a flexible rule manipulation and checking process within a relatively short period by using the proposed interface, though the effort put on linking variables with BIM elements may still be a concern if the elements are not well-organized in a complex project in practice. As for the BIM model and five building rules used in this case, the checking process can be done within 3 seconds. However, the user needs to set up the connections between variables and BIM elements in advance, and it goes complex easily if the numbers of elements or rules are increasing. It reflects the difficulties in maintenance mentioned by Hakim and Garrett (1993) regarding the potential drawbacks of logic-based rule expressions. To lighten the difficulties, currently, the proposed interface allows the logic-based expressions to be easily added, modify, and combine with different logic elements for creating more complicated patterns. For the future improvements, an intelligent grouping function, or sound design style before the compliance is necessary to be developed, to reduce the time spent on the linking process between rule variables and model component properties. Furthermore, the ontologies of design concepts regarding building components can be considered and become an essential help to solve the issue mentioned above. Nevertheless, the current results regarding the violations and design defects in the case can be identified efficiently and effectively, judged by the entity dependency generated in Table 6.
3. For a part of articles of Building Technical Regulations for Design and Construction in Taiwan, such as General Design Principle, Height of Buildings, Building Coverage Ratio as well as the five rules used in the case, these rule patterns are easy to be defined by users through the

interface, due to the intuitive mapping relationships (Class 1 or 2) between article phrases and geometrical components in the BIM model. While the authors did observe that, based on the presented model, certain regulations like Sunshine, Lighting, Ventilation, Energy Saving or Noise Proofing, may not be easy to formulate, given the direct mapping of the related properties in the model did not exist or the article phrases are hard to be transferred. These could be improved by adding the related properties explicitly or re-designing the model components, such as putting energy analysis results and serviceability information to the building components or further creation of component families. Further rule considerations from different perspectives, such as environmental impact or safety, can be discussed through further investigation.

4. The proposed system provides flexibility in adding, adjusting, and modifying rules, thought the mechanisms to examine user's authorization or the correctness of the rules defined are still missing. A suggestion would be further development effort in a user account control with different roles and permissions, such as rule creators, examiners, and supervisors, and a standardized rule approval process for the proposed system.
5. The rule patterns discussed in this case are only standardized building regulations instead of user-defined rules from the design collaboration perspective. Further combinations between standardized and user-defined rule patterns may come out with more opportunities for design intention validations. Such flexibility can be realized and implemented through the proposed interface. For example, a road component was customized in this case for being linked with one rule phrase of Article 14. A capability evaluation for the typical civil, building engineers and architects to translate their own needs (design intentions) or related local rules to predicate logic sentences can be a future direction. It can be further investigated by using the proposed interface to achieve the potential design conflict resolutions.

552 **Table 6. Applied Rules and Derived Entity Dependency in the Case**

Article No.	Description	Involved Entity	Dependency Roads Floors Ceilings Stairs Beams Prop
14 (Building height restriction against the width of adjacent road)	The height of the building shall not exceed 1.5 times the width of the adjacent road plus 6 meters.	Roads, Floors, Ceilings	
32 (Ceiling)	The clear height of the ceiling shall not be lower than 2.1 meters. While there are ceilings with different heights, at least more than half of them shall be higher than 2.1 meters. The lowest one shall not be lower than 1.7 meters.	Floors, Ceilings	
33 (Structure of stair)	As the building has more than 200 square meters in the area of each story above ground story or more than 200 square meters in the area of underground story, the width of stair and landing shall be bigger than 1.2 meters. The size of each riser shall be lower than 20 centimeters, and the size of each tread shall be bigger than 24 centimeters.	Floors, Stairs	
35 (Horizontal clearance of stair)	The distance between the outermost stair tread and the bottom of the ceiling (or beam) shall not be less than 190 centimeters.	Ceilings, Beams, Stairs	
161 (Floor Area Ratio, FAR)	Floor Area Ratio (FAR) is determined by dividing the total or gross floor area of the building by the gross area of the lot or lot size. In this case, the ratio shall be bigger than 2.	PropertyLines, Floors	

553

7. Conclusions

The complexity of the building design and construction is increasing, and the need for automatic model checking has become urgent. However, current interfaces of BIM-based rule checking systems are implemented without flexibility. They do not allow users to add, adjust or modify the rules in the system; neither do reserve design intentions for higher rule classes compliance checking.

The effort of this study is put on reviewing the current BIM-based rule checking systems, classifying the rules and further developing a user-oriented interface to enable users to establish the rules to meet their needs. Two-layered logic-based rule checking framework has been implemented to facilitate a flexible rule manipulation as well as to see how the applied rules affect the dependency among BIM model elements topologically. A prototype of the rule checking interface, based on the proposed framework, has been developed, and a case study on an example project in Taiwan has been carried out and discussed. The results show that the influence of rule compliance can be identified based on the BIM elements' dependency in different models, further enabling the tracking of design change consequences.

The contribution of the study includes the design of a rule checking interface which is helpful for capturing the design dependency from BIM models caused by rules, and further realizing flexible rule checking and design changes through the identification of rule influences on the BIM components. Given that little research has been focused on the user input and potential issues are hindering the development of automated rule checking systems for general purposes. A user interface development for a general-purpose rule checking system with comprehensive considerations (cost, environment, and so forth) can be a further step toward an efficient and effective BIM model design. The development of the proposed interface and related discussions are concluded for the improvement and validation in the future.

577 ## 8. References

578 AEC3 2015. Test Drive On-line Code Compliance Checking, AEC3 News. Available from
579 http://www.aec3.com/en/5/5_010_XABIO.html [Cited 10 October 2017].

580 Borrmann, A., and E. Rank. 2009. Topological Analysis of 3D Building Models using a Spatial Query
581 Language. Advanced Engineering Informatics, **23**(4): 370-385. doi: 10.1016/j.aei.2009.06.001.

582 Conover, D. 2007. Development and Implementation of Automated Code Compliance Checking,
583 (PowerPoint). International Code Council. Available from
584 http://www.slideserve.com/osanna/development-and-implementation-of-automated-code-
585 compliance-checking [Cited 10 October 2017].

586 Building Technical Regulations 2018. Laws and Regulations Database of the Republic of China.
587 Available from http://laws.mywoo.com/4/17/1016/2.html [Cited 8 May 2018] (in Chinese)

588 CRC 2016. Code Checking, CRC Construction Innovation Research Library. Available from
589 http://www.construction-innovation.info/indexf5c2.html?id=772 [Cited 10 October 2017].

590 Digital Alchemy, 2015. SMARTcodes, Digital Alchemy – IFC BIM Validation Service. Available
591 from http://www.digitalalchemypro.com/html/services/SmartCodes.html [Cited 10 October 2017].

592 Dimyadi, J., and R. Amor. 2013. Automated Building Code Compliance Checking - Where is it at?
593 In Proceedings of the 19th CIB World Building Congress, Brisbane, Australia, May 6-8. doi:
594 10.13140/2.1.4920.4161.

595 Ding, L., R. Drogemuller, M. Rosenman, D. Marchant, and J. Gero. 2006. Automating Code Checking
596 for Building Designs – Design Check, Clients Driving Construction Innovation: Moving Ideas into

Practice, CRC for Construction Innovation, pp. 113-126, Brisbane, Australia. ISBN: 1-7410712-8-3.

dRofus, 2015. dRofus – The Leading Planning and Data Management Solution for the Global Building Industry, dRofus. Available from http://www.drofus.no/en/ [Cited 10 October 2017].

Eastman, C. M., J. M. Lee, Y. S. Jeong, and J. K. Lee, 2009. Automatic Rule-based Checking of Building Designs, Automation in Construction, **18**: 1011-1033. doi: 10.1016/j.autcon.2009.07.002.

Eastman, C. M., P. Teicholz, R. Sacks, and K. Liston, 2011. BIM Handbook: A Guide to Building Information Modeling for Owners, Managers, Architects, Engineers, Contractors, and Fabricators, 2nd Ed., John Wiley and Sons, Hoboken, NJ. ISBN: 978-0-470-54137-1.

Fenves, S. J., 1979. Recent Developments in the Methodology for the Formulation and Organization of Design Specifications, Engineering Structure, **1**(5): 223-229. doi: 10.1016/0141-0296(79)90002-6.

Garrett, J. H., and M. M. Hakim, 1992. Object-Oriented Model of Engineering Design Standards, Journal of Computing in Civil Engineering, **6**(3): 323-347. doi: 10.1061/(ASCE)0887-3801(1992)6:3(323).

GSA, 2009. U.S. Courts Design Guide, Technical Report. Available from http://www.gsa.gov/graphics/pbs/Courts_Design_Guide_07.pdf [Cited 10 October 2017].

Hakim, M. M., and J. H. Garrett, 1993. A Description Logic Approach for Representing Engineering Design Standards, Engineering with Computers, **9**: 108-124. doi: 10.1007/BF01199049.

Han, C., J. Kunz, and K. Law, 1998. Client/server Framework for On-line Building Code Checking, Journal on Computing in Civil Engineering, **12**(4): 181-194. doi: 10.1061/(ASCE)0887-

3801(1998)12:4(181).

Heikkila, E. J., and E. J. Blewett, 1992. Using Expert Systems to Check Compliance with Municipal Building Codes, Journal of the American Planning Association, **58**(1): 72-80. doi: 10.1080/019443692089755.

ICC, 2015. International Code Council, ICC | International Code Council. Available from http://www.iccsafe.org/ [Cited 10 October 2017].

IFC, 2016. Open Source Projects Supporting IFC, Open Source – IFC. Available from http://www.ifcwiki.org/index.php/Open_Source [Cited 10 October 2017].

Ismail, A. S., Ali, K. N., and Iahad, N. A., 2017. A Review on BIM-Based Automated Code Compliance Checking System. In International Conference on Research and Innovation in Information Systems (ICRIIS 2017), Langkawi Island, Malaysia, July 16-17. doi: 10.1109/ICRIIS.2017.8002486.

ISO 10303-11:2004, 2004. Industrial Automation Systems and Integration -- Product Data Representation and Exchange -- Part 11: Description Methods: The EXPRESS Language Reference Manual, International Organization for Standard. Available from http://www.iso.org/standard/38047.html [Cited 20 December 2018].

Jiang, L., and R. M. Leicht, 2015. Automated Rule-Based Constructability Checking: Case Study of Formwork, Journal of Management in Engineering, **31**(1): A4014004. doi: 10.1061/(ASCE)ME.1943-5479.0000304.

Jotne, 2016. BIM / VDC, Jotne IT. Available from http://www.epmtech.jotne.com/solutions/bim

637 [Cited 10 October 2017].

638 Khemlani, K. 2005. CORENET e-PlanCheck: Singapore's Automated Code Checking System,
639 AECBytes. Available from http://www.novacitynets.com/pdf/aecbytes_20052610.pdf [Cited 10
640 October 2017].

641 Lê, M. A. T., F. Mohus, O. K. Kvarsvik, and M. Lie, 2006. The HITOS Project - A Full Scale IFC
642 Test, In Proceedings of the European Conference of Product and Process Modelling (ECPPM),
643 Valencia, Spain, Sept. 13-15. ISBN: 9780415416221.

644 Lee, J. K., C. M. Eastman, Y. C. Lee, 2015a. Implementation of a BIM Domain-specific Language
645 for the Building Environment Rule and Analysis, Journal of Intelligent and Robotic Systems, **79**(3):
646 507-522. doi: 10.1007/s10846-014-0117-7.

647 Lee, J. K., J. Lee, Y. S. Jeong, H. Sheward, P. Sanguinetti, S. Abdelmohsen, and C. M. Eastman, 2012.
648 Development of Space Database for Automated Building Design Review Systems, Automation in
649 Construction, **24**: 203-212. doi: 10.1016/j.autcon.2012.03.002.

650 Lee, J. M. 2010. Automated Checking of Building Requirements on Circulation Over a Range of
651 Design Phases, Ph.D. Dissertation, Georgia Institute of Technology. Available from
652 https://smartech.gatech.edu/handle/1853/34802 [Cited 10 October 2017].

653 Lee, Y., C. M. Eastman, and J. Lee, 2015b. Automated Rule-Based Checking for the Validation of
654 Accessibility and Visibility of a Building Information Model, In Proceedings of the International
655 Workshop on Computing in Civil Engineering, Austin, Texas, June 21-23. doi:
656 10.1061/9780784479247.071.

657 Li, L., B. Qin, and T. Liu, 2017. Contradiction Detection with Contradiction-Specific Word
658 Embedding, Algorithms, **10**(2): 59. doi: 10.3390/a10020059.
659 Li, X., G. P. Shen, P. Wu, and Y. Teng, 2019. Integrating Building Information Modeling and
660 Prefabrication Housing Production, Automation in Construction, **100**: 46-60. doi:
661 10.1016/j.autcon.2018.12.024.
662 Luo, H., and P. Gong, 2015. A BIM-based Code Compliance Checking Process of Deep Foundation
663 Construction Plans, Journal of Intelligent and Robotic Systems, **79**(3): 549-576. doi: 10.1007/s10846-
664 014-0120-z.
665 Martins, J. P., and A. Monteiro, 2013. LicA: A BIM Based Automated Code-checking Application for
666 Water Distribution Systems, Automation in Construction, **29**: 12-23. doi:
667 10.1016/j.autcon.2012.08.008.
668 Nawari, N. O. 2011. A Framework for Automating Codes Conformance in Structural Domain, Journal
669 of Computer and Information Technology, **1**(1): 34-48. doi: 10.1061/41182(416)70.
670 NovaCITYNETS, 2015. About FORNAX™ - PlanCheck Expert, FORNAX. Available from
671 http://www.novacitynets.com/fornax/about.htm [Cited 10 October 2017].
672 Pahl, G., and W. Beitz, 1996. Engineering Design - A Systematic Approach, 2nd Ed., Springer, London.
673 ISBN: 978-1-4471-6025-0.
674 Preidel, C., and A. Borrmann, 2015. Automated Code Compliance Checking Based on a Visual
675 Language and Building Information Modeling, In Proceedings of the International Symposium on
676 Automation and Robotics in Construction and Mining (ISARC 2015), Oulu, Finland, June 15-18. doi:

677 10.22260/ISARC2015/0033.

678 Preidel, C., S. Daum, and A. Borrmann, 2017. Data Retrieval from Building Information Models
679 based on Visual Programming, Visualization in Engineering, **5**:18. doi: 10.1186/s40327-017-0055-0.

680 Rasdorf, W. J., S. Lakmazaheri, 1990. Logic-based Approach for Modeling Organization of Design
681 Standards, Journal of Computing in Civil Engineering, **4**(2): 102-123. doi: 10.1061/(ASCE)0887-
682 3801(1990)4:2(102).

683 Sharpe, R., 1991. BCAider, CSIROpedia. Available from http://csiropedia.csiro.au/bcaider/ [Cited 20
684 December 2018].

685 Slaney, J., and B. W. Paleo, 2018. Conflict Resolution: A First-Order Resolution Calculus with
686 Decision Literals and Conflict-Driven Clause Learning, Journal of Automated Reasoning, **60**(2): 133-
687 156. doi: 10.1007/s10817-017-9434-4.

688 SMC, 2015. Solibri Model Checker, Solibri | Solibri Model Checker. Available from
689 http://www.solibri.com/products/solibri-model-checker/ [Cited 10 October 2017].

690 Solihin, W., and C. M. Eastman, 2015. Classification of Rules for Automated BIM Rule Checking
691 Development, Automation in Construction, **53**: 69-82. doi: 10.1016/j.autcon.2015.03.003.

692 Tan, X., A. Hammad, and P. Fazio, 2010. Automated Code Compliance Checking for Building
693 Envelope Design, Journal of Computing in Civil Engineering, **24**(2): 203-211. doi:
694 10.1061/(ASCE)0887-3801(2010)24:2(203).

695 Teng, Y., X. Li, P. Wu, and X. Wang, 2019. Using Cooperative Game Theory to Determine Profit
696 Distribution in IPD Projects, International Journal of Construction Management, 19(1): 32-45. doi:

697 10.1080/15623599.2017.1358075.

698 Zhang, S., J. Teizer, J. K. Lee, C. M. Eastman, and M. Venugopal, 2013. Building Information

699 Modeling (BIM) and Safety: Automatic Safety Checking of Construction Models and Schedules,

700 Automation in Construction, **29**: 183-195. doi: 10.1016/j.autcon.2012.05.006.