Understanding Computable Building Codes

Nawari. O. Nawari^{1,*}, Adel Alsaffar²

¹School of Architecture, College of Design, Construction and Planning, University of Florida, USA

²College of Architecture, Kuwait University, Kuwait

Copyright©2015 by authors, all rights reserved. Authors agree that this article remains permanently open access under the terms of the Creative Commons Attribution License 4.0 International License

Abstract Building codes and regulatory standards are becoming increasingly complex with the acquisition of new knowledge in the design and construction domain. The necessity for computable representation of the building codes and regulations for automating the code checking process is becoming ever more critical. Within the framework of Building Information Modeling (BIM) work process, model checking against building regulations is generally needed to be an automatic or semi-automatic process. These checking mechanisms generally do not modify a building design, but rather evaluates a design on the basis of the configuration of objects, their relations and attributes. This paper presents a summary of the core existing methods for computerizing building codes and standards. It reviews main concepts for these methods including knowledge representation, reasoning procedures, and knowledge acquisition. Moreover, this study evaluates the capabilities of the reviewed methods in reference to their practical applications and applied limitations for modeling computable building codes and standards. Additionally, the paper proposes methods with practical flexibility of encoding building codes knowledge domain and at the same time possess transparent and verifiable syntax and semantic features. The suggested approaches rely on evidently identifying objective and subjective data of the regulatory text before formalizing building codes. The methodology realizes the limitations of the formalization systems by clearly identifying which components of the building codes and standards can be transformed into computable model and which parts can't be encoded and requires manual compliance checking.

Keywords Computable Building Code Representations, Building Information Modeling, BIM, Automated Code Checking

1. Introduction

In the Architecture, Engineering, and Construction (AEC) industry, specifications and regulations developed by experts and read and implemented by professionals. Since the cognitive and analytic ability of the human brain is dissimilar

to anything implemented in computer systems, the automation of this process poses a real challenge to the AEC industry [1]. Providentially, new developments in the Artificial Intelligence (AI) research and Building Information Modeling (BIM) can offer practical solutions to resolve these problems.

AEC building standards and regulations commonly strive for to organize, categorise, label, and define the rules, actions, and patterns of the build environment to attain safety against any kind of failure, efficiency and overall economy. However, their best-laid plans are overwhelmed by the inevitable change, growth, innovation, progress, evolution, diversity, and entropy [1]. Quiet often regulations can amend provisions and interpretive manuals which normally lead to massive volumes of semi-structured documents that amend, complement and potentially conflict with one another. These issues, which represent complications for both young engineers as well as experienced professionals, are also far more disorderly for the fragile traditional knowledge bases in computer systems. Even though precise definitions and specifications are essential for solving encoding building regulations, many code provisions aren't well defined and highly subjective. Furthermore, some code provisions are characterized by continuous progressions and open-ended range of exceptions that make it difficult to give complete, exact definitions for any concepts that are learned through experience.

The introduction of computable building code will greatly improve the current design practice by simplifying the access to code provisions and complaints checks. Representing building codes and standards in a computable digital model that accommodate and make sense of the specific characteristics of the knowledge domain, plays a vital role in automating building regulation compliances. The computable digital schema of the building regulations enables automated rules verification without changing a building design, but rather evaluates a design on the principle of the conformation of parametric objects, their relations or attributes. It does engage rule-based systems to a proposed design, and yield outcomes in format such as "SUCCESS", "FAILURE" or "WARNING", or "UNIDENTIFIED" for circumstances where the obligatory data is inadequate or missing.

Computable building codes could be defined through two key modules:

(a)- Schema and Dictionary:

This denotes the computer exemplification of the rules and regulations of the building codes and the dictionary needed for that schema. To maintain consistency of properties (meaning and unit of measurement) within the computable format of the Codes a dictionary of the properties found within the building codes is needed. The dictionary is being developed as part of the International Framework for Dictionaries effort and, in the US, is being managed by the Construction Specifications Institute (CSI) in cooperation with ICC. This approach is also allowing the parameters within the building regulations to be identified against appropriate tables within the Omniclass classification system that has been developed by CSI and being recently accepted by the US National BIM Standard.

(b)- Code Conformance Evaluation:

This part addresses development and implementation of standards rule checking and reporting systems. It includes building model preparation, where the essential information required verification is prepared; code provisions interpretation and logical structuring of rules for their application; and the checking result representation through reporting and visualization systems.

It is clear that computable building codes and regulations hinge on Information readiness and rules development. Each of these components has some limitations aspects. Chief constellations of difficulties are linked to the nature of building codes and standards. For instance, building Codes are not self-contained documents. This denotes that most of the provisions of a design standard refer to knowledge that all professionals are expected to be aware of. Unfortunately such kind of information is not formally expressed anywhere. Furthermore, understanding a design standard necessitates some knowledge about the field of the design standard. In many professional fields, basic scientific knowledge (the knowledge that engineers and architects attain in their education) is expected from the users of an engineering design standard. Moreover, knowledge and heuristics are needed to differentiate when to examine another, referenced standard and when to proceed based on assumed compliance [6].

2. Literature Review

The issue of modelling rules and regulations checking of building codes and standard has interested many researchers and practitioners since the mid-sixties. For instance, Fenv [2] examined the application of decision tables to model AISC standard specifications. He made the remark that decision tables, an If-Then programming and program documentation technique, could be utilized to signify design standard provisions in an exact and unambiguous form. This model

was put to use when the 1969 AISC Specification [3] was represented as a set of interrelated decision tables.

Consequently, Lopez et al.[4] implemented the SICAD (Standards Interface for Computer Aided Design) system ([4], [5], and [6]). The SICAD system was a software model created to establish the compliance verification of designed components as described in application package databases for conformance with building codes. The SICAD models were in actual application use in the AASHTO Bridge Design System [6]. Garrett proposed the Standards Processing Expert (SPEX) system [24], using a standard-independent methods for sizing and proportioning structural member cross-sections. The system reasoned with the model of a design standard, denoted using SICAD system representation, to generate a set of rules on a set of basic information items that characterize the attributes of a design to be determined.

Then further research work was conducted by Singapore building officials, who began investigating code compliance verification on 2D drawings in 1995. In its next research effort, it changed and initiated the CORENET System working with IFC (Industry Foundation Classes) building models [7]. Additional focused research works on frameworks for the representation and processing of design standards for automated code conformance started three decades ago. Throughout that period, building models and the approaches for regulations checking had been investigated, however, effective computable building code systems are just beginning to emerge. In the 1990s, the development of the Industry Foundation Classes (IFC) steered to start investigation on utilizing this building model format for building regulations compliance checking. Han and others laid out schema for a client-server approach [8]. They later established a simulation approach of the American Disability Act (ADA) wheelchair accessibility checking [9], and [10].

In summary, there are over 300 relevant research studies focusing on automating building codes, traversing over 40 years, can be identified [11]. Some of the important major investigations recognized and their impacts have been depicted in a timeline (Figure 1). A more comprehensive survey of previous developments for digital representation of design codes and automated rule checking can be found in the work by Fenves et al. [12]; Eastman et al. [13]; Nawari [1]; and Dimiyati et al. [11]).

Most of the codes automated compliance checking systems shown in figure 1 are generally associated with a specific domain, such as spatial assessment, structural integrity, safety, energy usage and so on. Some of them offer certain degree of customization to modify the parameters of each rule to match specific local regulations. Once the rule structure has been encoded, it is available for multiple projects. In general, these systems can be classified into three widely used types of platforms for automated code compliance checking systems:

- As a software application integrated with a specific design tool, such as a plug-in. It is accessible to verify

- As a stand-alone software application detached from the modeling tools. An example of this platform would be Solibri Model Checker (SMC), which has its own rule engine that can work on multiple models;
- As a web-based application which can be available to verify designs from various sources.

Recent efforts on computerizing building code rules are focused more on the concept of marking-up regulatory texts to create a computable representation [14]. Other research investigations are centered chiefly on the investigations of automated or semi-automated extraction of information from regulatory texts into rules and other computable objects ([15], [16], [17], [18], [19], and [20]). In the next sections an introduction of these methods along with their strengths and practical limitations will be provided.

3. Building Codes Formats

AEC codes and regulations are legal documents written and authorized by experts and read and implemented by professionals. These documents are generally in human language formats, typically in written text, tables and often equations with legal status. They are hardly precise as formal

logic. That elasticity of expression is essential for a system of knowledge acquisition. Yet professionals can read those documents and translate them into formal scientific notations and software applications. They can excerpt any type of information they need, reason about it, and apply it at various levels of precision. The means by which these extraction and application are carried out is a field that academics and professional investigators have tried to automated or semi-automated for many decades.

Until recently, most of the previous efforts on modeling language representation are only focused on the syntax and grammar of rules. However, comprehending the meaning of the regulation rules is also critical, which requires experts' knowledge and experience to interpret the semantics of the regulation rules. For instance, CORENET ePlanCheck (Singapore Civil Defense, 2002), which utilizes a logic-based interpretation approach, the interpretation in transforming the rules from uncertain into more precise definition. During the process of interpretation, implicit assumptions and expectations are discovered, which could help to complete the understanding of what needs to be checked. However, this approach may lead to underestimations of the complexity of the rules involved in the execution of the checking process.

Figure 1. Timeline of related research to Automated Code Checking (modified from [11]).

It is evident that human rule compliance checking process generates large inconsistencies, because human judgments always fill in ambiguities, incorporating experience and unwritten local adaption of rules. Ontology and BIM standardization are key factors in solving these problems and help to move from human efforts to more automatic process, to generate consistent, precise and quantifiable conditions and constraints for each rule compliance.

The key approach to attaining computerized execution of regulations and specification in building informatics is to utilize modeling languages that can generate computer-interpretable regulation and provisions, i.e. provisions that a computer can examine automatically (see Figure 2). Currently, researchers have studied different modeling techniques for the generation computer-executable rules from natural language regulations. The following is a summary of the two main modeling approaches for building codes and regulations:

Figure 2. Computer-interpretable regulation and provisions.

(i) Artificial Intelligence Methods

Methods based on AI (Artificial Intelligence) utilize different Natural Language Processing (NLP) algorithms. They are focused on the area of human-computer interaction. Many challenges in NLP approaches involve natural language understanding and extracting rules from the building codes text. They aim to enable computers to derive meaning from textual format of building regulations and to generate logical rules for further processing.

Most of the NLP methods extract data and metadata creating an array of technical critical tasks. These include for instance: (a) Text analytics; (b) Content monetization; (c) automatic classification of content for regulatory compliance checking; and (d) text mining. Content extraction is an important area of NLP. These methods can be based on the syntactic and/or semantic characteristics of the main content. They generally utilize the concept of “Entity”. An Entity can be defined as the result of the extraction of proper nouns from text, such as people, places, building elements such as beams, columns, walls, windows, doors, ... etc. Each extracted named entity is classified, tagged and assigned a sentiment score, which gives meaning and context to each entity.

The general approach using NLP algorithms starts with developing ontology to capture domain knowledge to enhance interpretability and understanding of domain particular text content [19]. The NLP methods can be rule-based approach or machine learning-based approach. Rule-based NLP uses manually-coded rules for meaning extraction and processing. These rules are then iteratively built and refined to enhance the accuracy of semantics processing. The machine learning-based NLP utilizes algorithms for training text processing models based on the content of a given training text. Rule-based NLP methods tend to yield better text processing performance (in terms of precision and recall), but requires more human intervention.

Recent research efforts used intermediate processing step before the final extraction of rules from the source text (e.g. [19]). This step generally results in 3-tuple semantic element in the form of: <Subject, Attribute, Value>. Such semantic element is characterized by: (i) an ontology concept; (ii) an ontology relation; and (iii) a deontic operator indicator.

The process often has many steps before the final encoding results of the regulatory text. These include, preprocessing phase, feature generation, analysis of the extracted information, extraction rules, and the final execution of the encoding instances. In the preprocessing phase of the original content under tokenization, sentence splitting, de-hyphenation, and morphological analysis. In the second step a set of semantic and syntactic features that describe the content are derived based on domain specific ontology [19]. An example illustrating these steps is depicted in table 1.

Table 1. Encoding of part of the International Building Code (IBC 2006) using NLP method (modified from [19])

Information tuple extracted from Text Sentences		Horn Clause Logic Representation
Subject	Airspace	$\forall (a,i,r,s) ((\text{airspace}(a) \wedge \text{insulation}(i) \wedge \text{roof_sheathing}(r) \wedge \text{between}(a, i, r) \text{ has } (a, s) \rightarrow O(\text{Greater_Than_Or_Equal}(1, \text{inch}))))$
Subject Restriction	Relation (between, insulation, roof sheathing	
Compliance Checking Attribute	NA	
Deontic Operator Indicator	Obligation	
Quantitative Relation	Provide	
Comparative Relation	Greater_Than_Or_Equal	
Quantity Value	1	
Quantity Unit/Reference	Inch	
Quantity Restriction	NA	

Table 2. Example of using FOL method [1]

Building Code	Provision	Encoding
ASCE 7-10 Standard for minimum design loads for buildings and other structure has the following provision (3.2.1) for computing lateral pressure.	<i>"In the design of structures below grade, provision shall be made for the lateral pressure of adjacent soil. If soil loads are not given in a soil investigation report approved by the authority having jurisdiction, then the soil loads specified in Table 3.2-1 shall be used as the minimum design lateral loads. Due allowance shall be made for possible surcharge from fixed or moving loads. When a portion or the whole of the adjacent soil is below a free-water surface, computations shall be based upon the weight of the soil diminished by buoyancy, plus full hydrostatic pressure. The lateral pressure shall be increased if soils with expansion potential are present at the site as determined by a geotechnical investigation."</i>	$\forall x \text{ Structure}(x) \wedge \text{BelowGrade}(x) \rightarrow \text{Required}(x, \text{lateral pressure})$ $\forall x \neg \text{SoilReport}(x) \rightarrow \text{Required}(x, \text{Table 3.2.1})$ $\forall x \text{ Structure}(x) \wedge \text{BelowGrade}(x) \wedge \text{BelowWaterSurface}(x) \rightarrow (\text{Required}(x, \text{lateral pressure submerged}) \wedge \text{Required}(x, \text{hydrostatic pressure}))$ $\forall x \text{ Structure}(x) \wedge \text{BelowGrade}(x) \wedge \text{ExpansiveSoil}(x) \rightarrow \text{Required}(x, \text{increase lateral pressure})$

Some of the weaknesses of these modeling representations include the needs for human interpretation, in order to make them: (i) accessible electronically; (ii) structured and understandable by machines; (iii) represented in a standard format; (iv) interoperable; (v) explicit; and (vi) transparent. Furthermore, considering the fact that code regulations are complex and often highly subjective in nature, thus these modeling approaches do require building regulation officials to be involved in the process to ensure the correct interpretations of the encoding.

Nawari [1] proposed First-Order Logic (FOL) as a modeling language for encoding regulatory text. The method is dedicated to the knowledge extraction approach that aims to directly transform text regulations into formal languages. The approach has the advantage of simplicity but has the limitation of manual work as well as the technical knowledge to convert the regulatory texts into a set of rules.

First-order logic is a formal logical scheme which represents areas of discourses over which quantifiers range. Similar to natural language, it accepts that the universe comprises of: (i) Objects: Slabs, beams, columns, trusses, colors, ...etc.; (ii) Relations: deep, round, square, bigger than, part of, comes between, ...etc.; (iii) Functions: extreme deformation, maximum bending moments, ...etc. There are two major components of FOL: The syntax governs which groups of symbols are permissible expressions, while the semantics control the meanings and sense behind these

expressions [1]. Dissimilar to human languages, such as English, the language of first-order logic is entirely formal, so that it can be routinely determined whether a certain formulation is correct. There are two main kinds of permissible terminologies: terms, which signify objects, and formulas, which denote predicates that can be true or false. The terms and formulas of first-order logic are strings of symbols which together constitute the script of the language. These symbols of the alphabet are grouped into logical symbols, which constantly have the same connotation, and non-logical symbols, whose meaning differs by clarification. An example of the application of these approach is depicted in table 2.

The FLO method would work perfectly in case of condition clauses as depicted in Table 2. Often condition clauses and content clauses are encapsulated in one sentence in a building code. In such cases, the conditional clauses should be checked before checking content clauses. Under these conditions, one clause is generally composed of sentences having Subject, Predicate (verb) and Object, which is referred to as S + P + O pattern. These declarative sentences can be readily parsed into FLO rule-based schema (see Table 2).

(ii) Mark-Up Language Methods

Currently, most of the building codes and standard documents are available in Hypertext Markup Language

(HTML), Portable Document Format (PDF) or hardcopy. To ease the knowledge representation for further processing, a number of researchers have recommended different variations of Mark-up languages to formalize the building regulatory rules and guidelines.

For example, [20] proposed the extensible Markup Language (XML) as a unified format to exemplify regulations since XML's ability to deal with semi-structured data such as legal documents. To partially automate the translational process, they did develop a shallow parser as the first step to merge different formats of regulations into XML. The schema of the regulatory information, namely its

hierarchical and referential structures, is reassembled in XML schema. For non-structural characteristics of regulations, a feature extraction mechanism had been proposed (see Figure 3).

They further developed parse trees utilizing a context-free grammar and a semantic modeling approach that is capable of tagging regulation provisions with the list of references they encompass. For instance, an XML reference tag is shown in Figure 4, where Section 4.7.4 of ADA regulations cites Section 4.5 once. When properly extracted and linked, references offer users supplementary but vital information to better comprehend the regulations [20].

Figure 3. Development of Regulations in XML (Modified from [20]).

```

<regulation id="adaag" name="ADA Accessibility Guidelines" type="Federal"> ...
<regElement id="adaag.4" name="Accessible Elements and Spaces..."> ...
<regElement id="adaag.4.7" name="Curb Ramps"> ...
<regElement id="adaag.4.7.4" name="Surface">
<regText> Surfaces of curb ramps shall comply with 4.5. </regText>
<reference id="adaag.4.5" num="1" />
</regElement> ...
</regElement> ...
</regElement> ...
</regulation>
  
```

Figure 4. An example of XML format for building regulations [20]

Further example of research work on the conversion of regulation texts into formal languages is the automated mark-up of the Italian legislative texts in XML [21]. The automatic transformation of legal documents is attained by a content-based clustering of regulations and labelling of Italian law texts. The significance of various expressions that can recognize a regulation text is highlighted such as terms appearing in head notes, heading section, case name, etc. These terms are then taken into consideration when dealing with information extraction methods for legal case retrieval and are formalized as mark-up values in the XML representation of a legal text.

Hjelseth et al. [16] suggested a method that can model building regulations rules from direct semantic understanding of text by usage of four semantic mark-up operators: *Requirement*, *Applies*, *Select* and *Exception*. The semantic based approach is able to handle a wide range of purposes as validating, guiding systems, adaptive and content based model checking [16].

Table 3. Mark-up specification ICC IECC 2006 502.5 Moisture control [16].

Code Clause source	ICC IECC 2006 502.5 Moisture control
Rule Description	All $\{A/\}$ framed $\{A/\}$ $\{S/\}$ walls, floors $\{S/\}$ and $\{S/\}$ ceilings $\{S/\}$ $\{E/\}$ not ventilated $\{E/\}$ to allow moisture to escape shall be provided with an $\{R/\}$ approved vapor retarder $\{R/\}$ having $\{R/\}$ a permanence rating of 1 perm $\{R/\}$ (5.7×10^{-11} kg/Pa s m^2) or less, when tested in accordance with the desiccant method using Procedure A of ASTM E 96. The vapor retarder shall be $\{R/\}$ installed on the warm-in-winter side $\{R/\}$ of the insulation. Exceptions: $\{E/\}$ Buildings located in Climate Zones 1 through 3 $\{E/\}$. In construction where $\{E/\}$ moisture $\{E/\}$ or its $\{E/\}$ freezing $\{E/\}$ will not damage the materials. Where other approved means to avoid $\{E/\}$ condensation $\{E/\}$ in unventilated framed wall, floor, roof and ceiling cavities.

This approach still requires a manual identification of the computable rules by a person with AEC domain skills and implementation of applicable software code based on common predefined measures. The proposed mark-up language model is based on few operators, namely: *Select* (*S*), *Applies* (*A*), *Requirement* (*R*) and *Exception* (*E*). Applied to provisions text, the user may highlight any phrase that means: more scope as a '*Select* (*S*)' less scope as an '*Applies* (*A*)'; 'shall'/'must' etc. as a '*Requirement* (*R*)', (including alternative requirements); 'unless' etc. as an '*Exception* (*E*)', (including composite exceptions). (*R*), (*A*), (*S*) and (*E*) constructs can be generally attributed to have a topic, a property, a comparator and a target value. The topic and property will ideally be drawn from a restricted dictionary composed of terms defined within the code provisions and normal practice. The outcome (with any unit) may be numeric, whereupon the comparators will contain 'greater', 'lesser', 'equal' and their opposites. If the value is expressive, then only the 'equal' or 'not equal' comparators are relevant.

If the value signifies a set of objects, then the comparator may be any of the set comparison operators such as 'includes', 'excludes' [16].

The approached is further suggested to be linked to IFC constraint sub-schema. This constraint can be instantiated as an objective and qualified as a specification constraint. An example illustrating the application of this approach to the International Building Code ICC IECC 2006 moisture control specification is shown in table 3.

This approach can be applied successfully in the case of fully normative documents that are expected to result in a pass or fail results. However, the approach has limitations to handle other type of code regulations text specifically the deep hierarchies and heavy cross referencing among provisions in code regulations.

4. Suggested Approaches

This study aims at introducing practical approaches for computerizing building codes. One of the principle objectives of these practical methods of encoding building regulations is to provide a computable model with clear syntax and semantics that can be used to represent and reason about building codes regulations and provisions. Furthermore, the model should be well-suited to the requirements of digital content providers. It is essential that computable building rules model is attained before one starts the development of automatic code checking systems. An object-based representation of building regulations should define the minimum extent of data required to enable automatic checking of compliance and thus aids achieving integrated practice goals when utilizing BIM workflow in the industry.

In this research, clauses of any given building code are divided into four main categories as illustrated in figure 4. Namely, these are conditional clauses, content clauses, ambiguous clauses, and dependent clauses.

Figure 4. Major categories of building regulations clauses.

Conditional clauses are the ones that it is valid to interpret directly from the textual format into set of rules. Examples of

these are very common and typical features include rules with specific values such as those given in table 2 and 3.

Contents clauses are the ones cannot be translated into TRUE or FALSE. Instead of advising, these clauses are usually used for definitions, such as the definition of firewall, fire rate, smoke evacuation, high-rise building etc.

Ambiguous clauses are the subjective provisions. They normally include words such as: approximately, about, relatively, close to, far from, maybe, etc. An example of these is the footnote of the design lateral soil pressure for the clause given in Table 2: “For *relatively* rigid walls, as when braced by floors, the design lateral soil load shall be increased for sand and gravel type soils to 60 psf (9.43 kN/m²) per foot (meter) of depth. Basement walls extending not more than 8 ft (2.44 m) below grade and supporting light floor systems are not considered as being *relatively* rigid walls.”

Dependent clauses indicate that one clause is dependent upon one or more other clauses. This means some provisions are only suitable for a particular condition when other clauses are met. These are often difficult to convert to sets of rules and may require manual verification for compliance.

The research proposes that knowledge representations of the building codes and standards can be established by considering a number of development levels [22]:

- (i) High-Order Level: Requires the Development of Model View Definition (MVD), which will lead to IFC schema.
- (ii) Higher-order level: Requires feature extraction of all

specific data objective concepts leading to full encoding.

- (iii) Lower-order level: Necessitates feature extraction of all ambiguous information and uncertain data, then employing partial encoding using fuzzy logic or full manual checking.

Figure 5 exemplifies the suggested approach model for AEC regulations and standard provisions digital encoding. The method depends primarily upon the XML standard as it offers many benefits [23]:

- In-memory XML programming interface that facilitates communicating with XML from within the various software programming languages.
- The powerful extensibility of the query architecture can provide implementations that work over both XML and traditional SQL data stores.
- The query operators over XML use an efficient, easy-to-use, in-memory XML facility to provide XPath/XQuery functionality in the host programming language. This integration provides strong typing over relational data objects while retaining the expressive power of the relational database model and the performance of query evaluation directly in the underlying data store.
- All standard query operators can be replaced with user-defined implementations that provide additional services such as remote evaluation, query translation, and optimization.

Figure 5. Suggested method for computable building regulation model (modified from [22]).

The feature extraction is usually generated with the assistance of ontological representation of all the knowledge taking part in the building regulation text along with the Model View Definition (MVD) and the resulting IFC schema. According to the proposed encoding methods, features from building regulations can ultimately be embodied according to the ifcXML tag-based context: possible values for certain tags are provision numbers, rules, concepts, and properties of the features being extracted (see Figure 6). The ifcXML schema is in effect a standardized computable rule format, consumable by any computer software application capable of reading an XML building model file.

```
<ifcASCE7-2010>
  <ifcProvision id="3.2.1">
    ...
  </ifcProvision>
</ifcASCE7-2010>
```

Figure 6. An example of ifcXML schema for ASCE 7-2010.

The proposed approach aims to formalize encoding of building regulation in a way that is close to natural language and easily verifiable with several syntactic and semantic features that are capable of handling objective as well as subjective regulation provisions. The transformation into rules of the uncertain data can be handled partially by utilizing fuzzy logic. Fuzzy logic has been applied in many areas successfully to assist in processing information that may not be completely defined. In general, there is always the possibility that certain part of building regulations do still requires manual checks and review for compliance.

5. Conclusions

Building codes and standards are getting increasingly voluminous and experts' knowledge are advancing and updating the knowledge domain continually. Thus, the necessity for computable representation of the building regulations for automating the code compliance process is becoming ever more imperative. Specifically, in the case of Building Information Modeling (BIM) workflow, model verification against building codes and standards is critically needed to be an automatic or semi-automatic process.

The study strives for providing an overview and introducing practical methods of encoding building rules and regulations. It recommends methods with practical flexibility of encoding building codes knowledge domain and at the same time have transparent and verifiable syntax and semantic features. The proposed approaches rely on evidently identifying objective and subjective data of the regulatory text before formalizing building codes. The proposed methods require development of MVD and IFC schema along with FOL, Fuzzy logic, and ifcXML for encoding building provisions and guidelines. The methodology realizes the limitations of the formalization system by clearly identifying which components of the building codes and

standards can be transformed into computable model and which parts can't be encoded and necessitates manual compliance checking.

Acknowledgements

Authors are sincerely grateful to Fulbright U.S. Scholar Program for funding and supporting this research. Also, authors would like to express their gratitude to Collage of Design, Construction and Planning at the University of Florida as well as the School of Architecture at the University of Florida for promoting the international exchange activities and facilitating various supportive information and tools.

REFERENCES

- [1] N. O. Nawari. "Automated Code Checking in BIM Environment". In the Proceedings of the 14th International Conference on Computing in Civil and Building Engineering, Moscow, Russia, 27-29 June, 2012.
- [2] S. J. Fenves. "Tabular decision logic for structural design", Journal of Structural Engineering, ASCE, Vol 9-92, pp. 473-490., 1966.
- [3] American Institute of Steel Construction (1969). "Specification for the Design, Fabrication & Erection of Structural Steel for Building," 1969.
- [4] Lopez, L. A., and R. N. Wright (1985). "Mapping Principles for the Standards interface for Computer Aided Design", NBSIR 85-3115, National Bureau of Standards, Gaithersburg, MD.
- [5] Lopez, L. A., S. Elam and K. Reed (1989). "Software concept for checking engineering designs for conformance with codes and standards". Engineering with Computers, 5, pp.63-78.
- [6] S. J. Fenves, J. H. Garrett, H. Kiliccote, K. H. Law, and K. A. Reed (1995). Computer representations of design standards and building codes: U.S. perspective, The Int. J. of Constr. Information Technol., 3(1), pp. 13-34., 1995.
- [7] Khemlani, K. (2005). "CORENET e-PlanCheck: Singapore's automated code checking system", AECBytes, <http://www.aecbytes.com/buildingthefuture/2005/CORENETePlanCheck.html>.
- [8] Han, C., Kunz, J. and Law, K., Law (1999). "Building design services in a distributed architecture", Journal of Computing in Civil Engineering, ASCE 13 (1), 1999 12-22.
- [9] Han, C., Kunz, J., Law, K. (1997). "Making automated building code checking a reality", Facility Management Journal (September/October), pp. 22-28.
- [10] Han, C., Kunz, J. and Law, K., Law (2002). "Compliance Analysis for Disabled Access, Advances in Digital Government Technology, Human Factors, and Policy", in: WilliamJ. McIver
- [11] J. Dimyadi, and R. Amor. "Automated Building Code

- Compliance Checking – Where is it at?” Proceeding of the 19th International CIB World Building Congress, 5th-9th May 2013, Australia.
- [12] Fenves, S. J. and Garrett Jr, J. H. (1986). “Knowledge-based standards processing”, *Int. J. Artificial Intelligence Engn 1*, pp. 3-13.
- [13] EASTMAN, C. M., JAE-MIN LEE, YEON-SUK JEONG, JIN-KOOK LEE. “Review Automatic rule-based checking of building designs “, *Journal of Automation in Construction* (18), pp. 1011–1033, Elsevier, 2009.
- [14] N. Kiyavitskaya, N. Zeni, L. Mich, J. R. Cordy, and J. Mylopoulos. Text mining through semi automatic semantic annotation. In *Proc. of PAKM'06*, volume 4333 of *LNCS*, pages 143–154. Springer-Verlag, 2006.
- [15] E. Hjelseth, and N. Nisbet. “Overview of concepts for model checking“. In the Proceedings of the 2010 27th CIB W78 International Conference, Cairo, Egypt, 2010.
- [16] E. Hjelseth, and N. Nisbet. Exploring semantic based model checking. In the Proceedings of the 2010 27th CIB W78 International Conference, Cairo, Egypt, 2010.
- [17] J. Zhang, N. El-Gohary. AUTOMATED INFORMATION EXTRACTION FROM CONSTRUCTION-RELATED REGULATORY DOCUMENTS FOR AUTOMATED COMPLIANCE CHECKING, Proceedings of the 28th International Conference of CIB W78, Sophia Antipolis, France, 2011.
- [18] J. Zhang, N. El-Gohary. Extraction of Construction Regulatory Requirements from Textual Documents Using Natural Language Processing Techniques. *Journal of Computing in Civil Engineering*, ASCE (2012): pp. 453-460.
- [19] J. Zhang, and N. El-Gohary. "Semantic NLP-Based Information Extraction from Construction Regulatory Documents for Automated Compliance Checking." *Journal of Computing in Civil Engineering*, ASCE, 1943-5487, 2013.
- [20] G. T. Lau, and K. Law “An Information Infrastructure for Comparing Accessibility Regulations and Related Information from Multiple Sources”. In the Proceedings of 10th International Conference on Computing in Civil and Building Engineering, Weimar, Germany, 2-4 June, 2004.
- [21] Bolioli, A., Dini, L., Mercatali, P., and Romano, F. ‘For the Automated Mark-Up of Italian Legislative Texts in XML’ in T.J.M. Bench-Capon, A. Daskalopulu and R.G.F. Winkels (eds.), *Legal Knowledge and Information Systems. Jurix 2002: The Fifteenth Annual Conference*. Amsterdam: IOS Press, 2002, pp. 21-30.
- [22] Nawari O. Nawari and. Adel Alsaffar (2015). “Practical Approaches for Computable Building Codes”. *Proc. of the 32nd CIB W78 Conference 2015, 27th-29th 2015, Eindhoven*, 569-576.
- [23] N. Nawari ”Automating Codes Conformance.” *Journal of Architectural Engineering*, ASCE, 18(4), 315–323, 2012.
- [24] Garrett, J. H., Jr., and S. J. Fenves, (1987). “A Knowledge-based standard processor for structural component design”, *Engineering with Computers*, 2(4), pp 219-238.