Regulatory Room Working Group

Report on Open Standards for Regulations, Requirements and Recommendations Content

Foreword

This is the final report of the buildingSMART Regulatory Room working group, after a nine months' study on Open Standards for Regulations, Requirements and Recommendations. We warmly express our thanks to the Contributors to the Regulatory Room project (who are listed in Annex 2: Contributor) and to participants in the Regulatory Room group and friends (who are listed in Annex 3: 'Regulatory Room' Group and Friends.)

The introduction in section 1 and the conclusions in section 5 give an overview, with the intervening sections go into the detailed intentions and discoveries. We commend this report for your attention and in particular section 5 and the five issues (5.3), the two recommendations (5.4) and sixteen actions (5.5) are detailed.

Øivind Rooth   Inhan Kim.

Joint Chairs, buildingSMART International Regulatory Room

April 2017

Page no.	Author
1	bSI Regulatory Room

Contents

Introduction

This is a review the currently available and proposed exchange formats for supporting interoperability between Regulatory, Requirements and Recommendatory (RRR) content.

This report aims to:

- present a case for investigating such issues
- identify solutions
- propose recommendations that will be implemented by the buildingSmart community, by the research community and by regulatory bodies.

In response to clients and the many other factors which influence the design and procurement of a building, the construction sector analyses RRR and then delegates them downwards in an iterative and cyclic process. For example; designers and engineers develop proposals, which are then aggregated together, analysed and approved: and are returned if issues are detected.

BuildingSMART defines the representation of the proposals (by using IFC) and 'approvals and issues' can be conveyed (by using buildingSMART BCF). However, there is currently no common representation of the requirements.

This can reduce the credibility and the efficiency of the digital aspect of the industry and its attractiveness to its customers.

1.1 Scope

RRR interoperability is a necessary prerequisite for obtaining support from Government and near-Government regulatory bodies for process improvement and for the support of buildingSMART. It will help create a market for authoring, capture, evaluation, and other related tools. Such tools can thereby help meet national and regional expectations.

- Issues and disapprovals
  - BCF
  - IFC Object metadata
- RRR
  - Requirements
  - Regulations
  - Recommendations
- Model(s)
  - IFC Objects
- Approvals
  - IFC Object metadata

© AEC3 Ltd 2015

Whilst it is conceivable that the construction sector could devise a specific solution, it is unlikely to achieve the critical mass of multiple authoring tools and multiple checking tools. However, RRR requirements in the facility sector are not substantially different from those in any other legal, engineering or service sector.

Therefore, in order to feasibly (and within realistic time frames) resolve such issues it is critical to look outside the facility/construction domain towards other industries that have already resolved such problems. It is noted that there are significant developments in financial and legal regulation and in engineering and service requirements.

Interaction and automation are both key desirable features. Managerial and technical criteria will impact this. Interoperability for rules will help ensure that multiple tools and approaches become possible and economic to deliver. However, any interoperable solution will need to handle the worst and best of cases.

1.2   Outline

This report consists of this introduction, an outline business case, and then an examination of the processes underlying compliance checking. The criteria for the possible technical solutions, the example regulation and some associated issues – vocabulary and expected

properties - are summarised. The bulk of the report   examines a number of technical solutions as 'candidates'. Finally, there are the conclusions.

1.3   Out of scope

This study will not directly answer the question as to what a Digital-Friendly Code might be, in terms of style, presentation, and content. However, it will define key principles and objectives which will be documented in the form of 'recommendations and guidance'.

The maintenance and approval of a 'universal' dictionary is not in scope. For example:

Regulations may be implemented in a different geographic region than the 'designer' (who has their own regulations). The 'designer' may not speak the same language, the design may be presented in a third language, schema or vocabulary such as IFC.

The technical solutions may have a need for a dictionary, but the source, maintenance and approval of such a dictionary is not in scope. The content may be in scope, and each technical solution may include the comments it needs.

1.4 Terms and definitions

RRR   Regulations, Requirements and Recommendations: a collective term for the of the demand side for the facilities sector

IFC   Industry Foundation Classes, the BuildingSMART schema for the facilities sector

BCF   BIM Collaboration Format: the BuildingSMART messaging schema for the facilities sector

Contributing authors

Name	Affiliation	Email
Nicholas Nisbet	buildingSMART UKI	nn@aec3.com
	AEC3	
	Leeds Beckett University	
Dr Nicholas Humes	InForm Architecture Ltd.	nick@InFormArch.co.uk

1 Business case and need

There is a need to consider the current and proposed interoperability formats for supporting Regulatory, Requirements and Recommendatory (RRR) interoperability.

The role of this business case is to support the work of buildingSMART in relation to Government and near-Government regulatory bodies in obtaining their support and approval.

The business case aims to explain the benefits and procedures proposed by buildingSMART in order to improve regulatory processes. If successful, the findings of this report will help strengthen the 'BIM' market as well as helping to establish a market for the authoring, capture, evaluation and other associated tools. Such a market would facilitate specific national and regional expectations through the use of such generic tools.

Currently, within BIM processes, there is a growing problem relating to conformity, specifically the standardisation of schemas, formats, and their relationship to IFC Project models. In order to resolve this a full review of current practices is required; it is important to investigate other industries outside of the facility/construction domain as it is widely acknowledged that there are significant developments i.e. in financial and legal regulation, in engineering and service requirements.

Annex: Contributing authors

Initials	Name	Affiliation or Company	Email
NN	Nicholas Nisbet	buildingSMART UKI and AEC3 and Leeds Beckett University	nn@aec3.com
NH	Dr Nicholas Humes	InForm Architecture Ltd.	nick@InFormArch.co.uk

2 Processes

2.1 Authoring and sourcing content

Typically, regulatory, requirement and recommendatory (RRR) content is authored and published outside of a VDC (virtual design and construction) environment. There is a considerable legacy of such documents. Legislation and secondary documents (approved documents, amendments, statutory instruments etc.) are typically revised in a cycle. It may take a number of years for RRR content to be written, checked, approved and published but the final versions may only be known perhaps six months prior to coming into effect. In certain situations, new requirements may be developed even more rapidly.

2.2 Regulatory content

Legislation is often prepared in committee and open forums; where initial structure or vision may be diluted by the processes of discussion and compromise resulting in the loss of the original intent and logical structure.

Most primary legislation defines the 'certifying' authority, the processes and protocols involved, and the points at which the authority can 'approve' or 'reject' proposals. Legislation is often delegated down to alternative means of 'approval' some of which may invoke technical analysis or 'deemed-to-satisfy' solutions.

2.3 Requirement content

RRR often develop progressively over time with many evolving iterations, sometimes these relate to the function of the building or the type of infrastructure.

Of course, RRR can be entirely novel and new to the industry. The creation and subsequent changes to RRR is often due to a new understanding of the topic, to increased industry experience, to events that impact the industry or improvements in technology or products. Changes may even be released in time to correspond with the calendar, industry publications or to pre-empt certain circumstances or to correct 'loopholes'.

Some requirements may be focussed on operational outcomes rather than constraining the facility solution itself.

RRR can often be viewed as 'open' or 'closed'. Open RRR allow the applicant to choose how the RRR is met, the emphasis being that the clause is met by a specific performance. Closed RRR prescribes how the RRR should be met, this approach allows little or no variation as to how the clause is met.

In the UK construction industry there is a distinct emphasis on the performance and 'outcomes' of the building rather than prescriptions dictating how the building should be constructed or how the RRR are to be met. In 2016 a new Part L Approved Document 'Conservation of Heat and Power' was introduced which placed a stronger emphasis on the minimum requirement of the fabric of the building i.e. walls, floors, roofs and windows. This could have been in response to an 'overall' performance rating being affected by the 'over performance' of other building elements e.g. a highly efficient boiler in a poorly insulated house.

Where requirements are provided by the client or generated by the design process itself, it may be possible to 'negotiate' or 'balance' the satisfaction of requirements. RRR can often be met according to a 'grade' based system. In some scenarios, such as LEED and BREEAM environmental assessments, a point based system is used to attain 'approval'. Such schemes have minimum pass rates for each topic. Topics are also 'weighted' and a grade based on the cumulative total determines the result, such as a platinum, gold, silver or bronze award.

2.4   Recommendatory content

Recommendations may be found in reports, case studies and assessments of past projects or may be found in statistical and research findings.

The content may be suggestive, offering different priorities or preferences for alternative solutions or for requirements. In some cases, the link between conditions (high-rise, concrete panel construction) and risks and precautions (falls and safety equipment) may be speculative, probabilistic or tentative.

Recommendations may also be found in 'technical guidance'. Such documents aim to explain the RRR, often using 'real world' experience, before demonstrating how the RRR can be successfully met, and hazards or pitfalls avoided.

2.5   Publication

The majority of RRR content is published formally as documents. Some content may have legal status from a certain date or may be withdrawn at a certain date. However, most content remains significant and valid for decisions made at the time. Hence the time-stamping of both the document and the target building model may be critical.

This is particularly important in the construction industry as the RRR may change during the approval or construction phase and before the project is even completed.

2.6   Dissemination and characterisation

RRR content may be filtered, amplified or presented in alternative formats and media. A structured web-page may serve to guide and educate. An automated or attended telephone hotline may guide the caller through aspects.

New RRR content may be compared against old versions or against other content for changes, conflicts or duplications. This assessment may be of great commercial importance, if a scheme becomes invalid (or valid) at a transition.

2.7   Assessment

Most RRR content is used to assess proposals, in-process, or as built facilities. Proposers may be interested in all issues detected. Where human engagement is required it is easy for assessors to focus on new or recurring issues and 'miss' more obscure issues. Assessors with a duty to inspect may be interested in only the existence of any one failure, or in the top (ranked) issues. However, such human engagement is crucial for judgements to be made. In some cases, RRR content is interpreted as higher level checklists, which guide assessors to manual, detailed procedures for individual requirements.

2.8   Incorporation

RRR content may be used to influence a decision. As such the RRR content can be interpreted to support or effect design decision making parametric tools within the design/engineering environment. This may be as simple as a circle shown within a design,

limiting the location of desks, or it may be used to generate solutions or eliminate invalid shapes, forms, or configurations.

These tools can even be used to progress and evolve the design process resulting in an improved design e.g. thanks to a more efficient design, improved sustainability etc.

2.9   Response

The response to the outcomes of assessment may be through a combination of formal notifications, informal advice, or changes to the status of a proposal.

The response may detail one, many or all points of failure or concern. The response may be illustrated with the current points of weakness in the proposal, or with recommendations for improvement. It is common for such recommendations or refusals to be backed up with references to the RRR content document, section, paragraph or phrasing.

Unfortunately, the systems within which we work often focus on where the design has failed, missing where the scheme has excelled or surpassed the RRR. This type of assessment has the potential to perpetuate a 'tick box' or 'minimum standards' culture. It has been suggested that tools which alleviate the pressure on applicants to meet such 'minimum standards' would allow applicants to focus on improving their design and thereby the built environment.

Annex: Contributing authors

Initials	Name	Affiliation or Company	Email
NN	Nicholas Nisbet	buildingSMART UKI and AEC3 and Leeds Beckett University	nn@aec3.com
NH	Dr Nicholas Humes	InForm Architecture Ltd.	nick@InFormArch.co.uk

3a Criteria

The intention of this review is consider several different RRR candidates for the purposes so as to make recommendations that are justified and based on a fair and open methodology.

To enable comparison each candidate technology is described using a consistent approach so that it can be assessed strategically, commercially and technically. Each assessment is reviewed as follows:

1. Introduction
2. Names and synonyms
3. Description
4. Strategic assessment of format
5. Control, ownership, availability
6. Age, and stability
7. Suitability for multiple purposes
8. Authoring
9. Computer-assisted and Automated code compliance checking
10. Analysis and code comparisons

11. Filtering and structured dialogues
12. Controlling parametric objects

3a.1 Introduction

The introduction covers the name of the candidate and any synonyms, along with a description of the format in general terms.

3a.2 Strategic assessment of format

The startegic assessmment of the format should document the control of the candidate in terms of its ownership, licencing and availability. Candidates of the most relevance will fall broadly into the 'open' philosphy espoused by buildingSMART.

The assessment may also consider the age and relative stability of the format. It can also consider its capability to address the multiple use-cases discussed in chapter 2 including

- Authoring
- Computer-assisted and Automated code compliance checking
- Analysis and code comparisons
- Filtering and structured dialogues
- Controlling parametric objects

3a.3 Commercial base

The assessmsent of the commercial base may consider the availability (singular, multiple) of tools that support the candidate and their costs and licensing models. This may include the stability, maturity of the available authoring and capture tools, assessment and execution engines, and the scalability of the solution in creating a market place for RRR applications.

3a.4 Technical base

The techncial assessment may examine the critical features and opportunities offered by a candidate:

- Expressivity and generality
  - What RRR content can be expressed
  - Ability to handle easy and technical requirements through to hard and management requirements .
  - Target (project) model schemas and formats
  - Performance measured against an independent RRR performance benchmark and support for heuristics and optimisation
  - Ability to work with external dictionaries
  - Openness, interoperability and convertablity
- Acceptability and provenance
  - Linkages back to (requirements) source
  - Linkages forward to (project) model
  - Depth of results that can be reported

- Authoring/capture tools (singular, multiple)
  - Assessment engines, software and toolkits for (singular, multiple)
  - Scaleability and role in creating a market place for RRR applications.
- Technical Expressivity and generality
  - What RRR content can be expressed
  - Ability to handle easy and technical requirements through to hard and management requirements.
  - Target (project) model schemas and formats
  - Performance measured against an independent RRR performance benchmark and support for heuristics and optimisation
  - Ability to work with external dictionaries
  - Openness, interoperability and convertablity.
- Acceptability and provenance
  - Linkages back to (requirements) source
  - Linkages forward to (project) model
  - Depth of results that can be reported.
- Definitive references and contacts
- Background references and publications, authors and contacts for further discussions

Annex: Contributing authors

Initials	Name	Affiliation or Company	Email
NN	Nicholas Nisbet	buildingSMART UKI and AEC3 and Leeds Beckett University	nn@aec3.com
NH	Dr Nicholas Humes	InForm Architecture Ltd.	nick@InFormArch.co.uk

3b The test case

3b.1   Test Example of RRR

For the purposes of consistency and to demonstrate the characteristics of each RRR candidate a section of the Korean Building Code has been chosen. The example has been provided by the buildingSmart Korean Chapter [JLK] and is felt to be sufficiently representative of the nature of that document and regulatory content in general. Furthermore, the purposes of the code are similar to those found in other countries and represent a common function of many buildings; vertical circulation. It has a number of features of interest:

- Some geometric measurement
- a separate permit from the MLIT (Ministry of Land, Infrastructure and Transport)
- exceptions which provide an alternative means of meeting the code

- potential for ambiguity through contentious terms such as "living room".

3b.2   Example open text file

The clause was originally provided by JLK for the purposes of reviewing approaches. As a benchmark the following text is used as it is believed to be an accurate and complete translation of the original Korean document (Article 34 Clause 1).

Korean Building Act 34-1

> *On each floor of a building, direct stairs leading to the shelter floor or the ground other than the shelter floor shall be installed in the way that the walker distance from each part of the living room to the stairs is not more than 30 meters: Provided, that in the cases of a building of which the main structural part is made of a fireproof structure or non-combustible materials, the walking distance of not more than 50 meters may be established, and in cases of a factory prescribed by Ordinance of the Ministry of Land, Infrastructure and Transport, which is equipped with automatic fire extinguishers, such as sprinklers, in an automated production facility, the walking distance of not more than 75 meters may be established.*

However, the following text was later provided as being a more accurate and complete translation of the original Korean document (Article 34 Clause 1). This text was not used in the trials.

> *On each floor of a building, direct stairs leading to the shelter floor or the ground (including slope ways; hereinafter the same shall apply) other than the shelter floor (referring to a floor having a doorway leading directly to the ground and the shelter safety zone of a skyscraper under paragraphs (3) and (4); hereinafter the same shall apply) shall be installed in the way that the walking distance from each part of the living room to the stairs (referring to the stair nearest to the living room) is not more than 30 meters: Provided, That in cases of a building of which main structural part (excluding a performance hall, assembly hall, auditorium and exhibition hall which are installed on underground floors and which have a total floor area of not less than 300 square meters) is made of a fireproof structure or non-combustible materials, the*

walking distance of not more than 50 meters (in cases of multi-unit dwellings higher than 16 storeys, not more than 40 meters) is permitted, and in cases of a factory prescribed by Ordinance of the Ministry of Land, Infrastructure and Transport, which is equipped with automatic fire suppression systems such as sprinklers, in an automated production facility, the walking distance of not more than 75 meters (in cases of unmanned factories, 100 meters) is permitted.

For the purposes of clarity, the term 'living room' refers to the common terminology of 'habitable space'. Further details on "main structural parts" are explained in the Korean Building Acts full document:

> *The term "living room" means a room of a building used for dwelling, business, working, meeting, recreation and other similar purposes;*
>
> *The term "main structural parts" means bearing walls, pillars, floors, beams, roof frameworks and main stairways: Provided, that studs, the lowest floor, small beams, sunshades, outdoor stairways and other similar parts not essential to the structure of a building shall be excluded therefrom.*

3b.3   Commentary

An issue with 'verbose' RRR relates to the wording and the semantics. In this case the issue relates to the 'focus' and intent of the text i.e. what is being regulated? In the example above there are several key elements: stairs, shelter, habitable rooms and travel distances.

It could be argued that all elements are required in order to test and therefore satisfy the RRR, but how can this code be processed if some of the elements do not exist? Should the scheme fail if there is no 'living room'/ habitable space?

It can be expected that a proposal as a whole, the project or facility, is liable to be failed, but more specifically, is it the 'stair', the 'stairway' or most practically, the 'living room' that is being regulated? Put simply, the absence of a stair could still leave the building in violation, where as the absence of a 'living room' would not. So contrary to the drafting, it may be that it is a regulation of 'living rooms' that they should have a means of escape.

A second issue may be the clarification or confirmation of the definition of 'living room', as it probably does not mean the 'living room' as in the lounge of a residential dwelling, but rather

an "occupied space" or "habitable space". For the current exercise it may be taken as the factory space, (and not stairways).

3b.5   Sample IFC file

A simple two storey factory building with a staircase has been provided in IFC, IFCXML and RDF/OWL formats. It was also provided as a COBie file.

Name	CreatedBy	CreatedOn	Category	FloorName	Description	ExtSystem	ExtObject	ExtIdentifier	TagNumber	UsableHeight	GrossArea	NetArea
1-001	nn@aec3.c	2016-07-27T13:13:54	n/a	Level 1	512.4	Quarter1	IfcSpace	2EgIl3ohznjcabtJYWDDdN	n/a	0	0	0
1-002	nn@aec3.c	2016-07-27T13:13:54	n/a	Level 1	20.0	Quarter1	IfcSpace	0W1W0lub2FJj__7_TvHhqR	n/a	0	0	0
2-001	nn@aec3.c	2016-07-27T13:13:54	n/a	Level 2	512.4	Quarter1	IfcSpace	3L_TlqLLeSAftF04LSEcVk	n/a	0	0	0
2-002	nn@aec3.c	2016-07-27T13:13:54	n/a	Level 2	20.0	Quarter1	IfcSpace	0UjLnSJkgRYeGUcs1WOBUb	n/a	0	0	0

Instruction | Contact | Facility | Floor | Space | Zone | Type | Component | System | Assembly | Connection ...

Figure 1: Entities such as Spaces will be categorised and additional attributes added as required.

Two variants are provided, one of which is intended to pass and one is intended to fail. The model referred to as KR-Factory-V1002 is expected to fail due to inadequate travel distances. KR-Factory-V1001 is expected to pass.
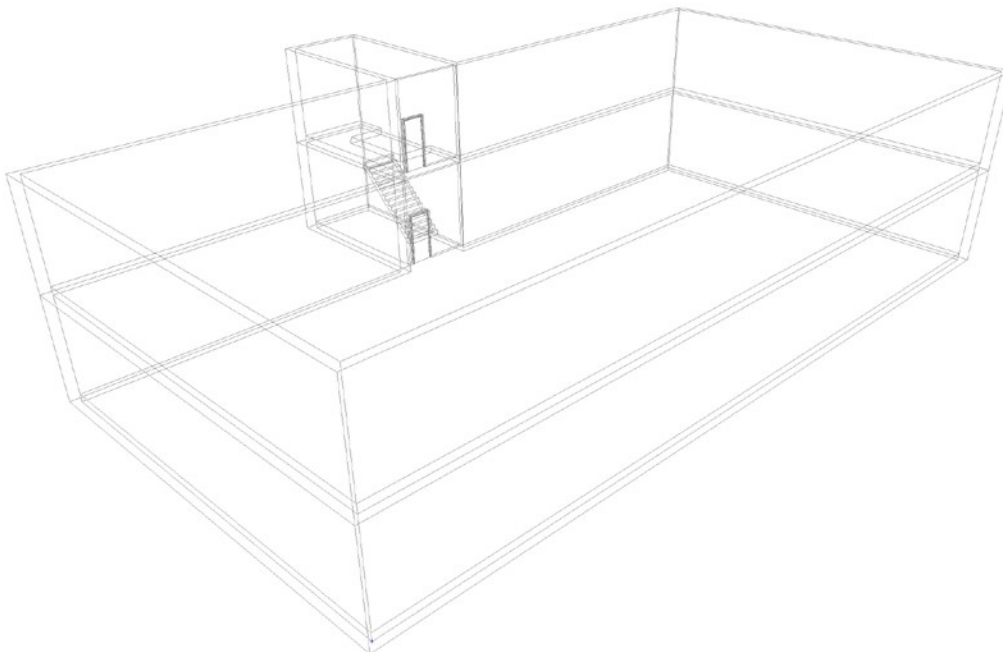

Figure 2: KR-Factory-V1001 is expected to pass the chosen regulation. In the KR-Factory-V1001 the longest escape distance has been estimated as 25.936m.
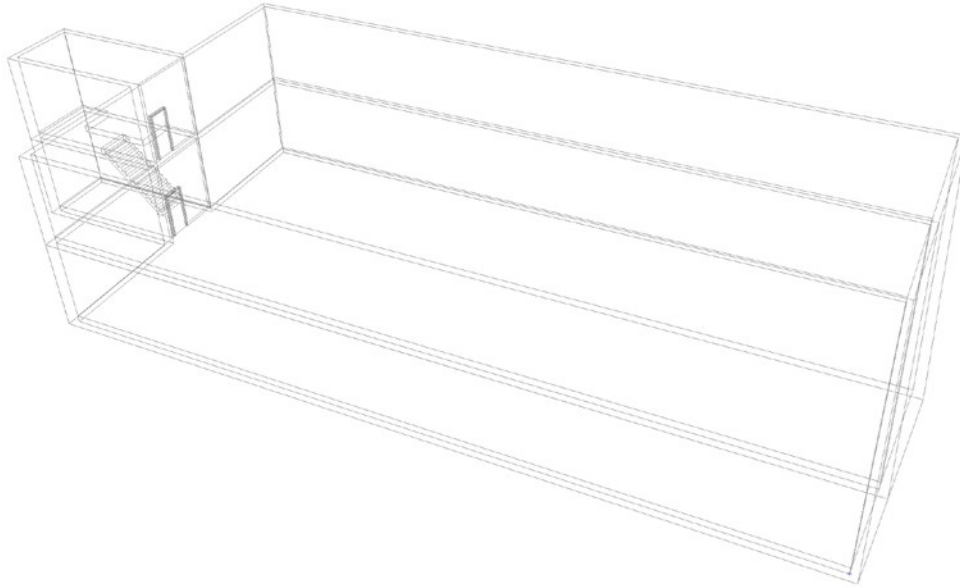
Figure 3: KR-Factory-V1002 is expected to fail.

In the KR-Factory-V1002 the longest escape distance has been estimated as 36.296m

3b.6   Revisions to the base model

The example models were developed further in response to queries and corrections detected by the contributors. These were named KR-Factory-V1001-Rev2 and KR-Factory-V1002-Rev2.

Revision 2 added approximate EscapeDistance to the factory 'living space' in each building. This was changed to accommodate approaches/ engines that could make the geometric estimate and those that could not. This addition was felt to be reasonable as any values entered could be challenged and secondary evidence produced if necessary. Secondary evidence in this case would be either the name of the application or algorithm used to calculate it, or a marked up diagram.

Rev2-Modifed corrected some errors from the IFC export process and also included:
   (a) Three components had been assigned the wrong spatial containment.
        a.   Two doors was re-assigned to 'level 1' and 'level 2' instead of 'building parts'.
        b.   The stair was re-assigned to 'level 1' instead of 'building parts'.
   (b) Duplicate guids had been introduced on the site object.

This update also corrected some errors in the information contained within the model.

(a) The description of the second project was corrected to be "Escape Distance B"
(b) Classification and properties in the second project were reassigned to the building from the site.

Revision 3 models were named as follows KR-Factory-A-V003. KR-Factory-B-V003. In addition, the names of the project, site and building were rationalised, along with the file naming policy. Several contributors noted the difficulty in differentiating the factory space from the stair space. It was noted that all the factory spaces were named *-001 and all stair space *-002.

One contributor noted that a lower floor space contained a stair. However, the example cases are multi-storey buildings, each has stairways but the roof is inaccessible. A space on the top-most floor is not considered to be part of a stairway. On the top floor the space contains no stair, stair flight or stair landing objects, as these are all contained in the space below. So there is no 'containment of stair parts'. As the floor slab touching the space contains a virtual opening it is not sufficient, as there may not be a slab, so the answer is not 'containment of virtual openings'. 'Space boundaries' do not improve the situation as a virtual space boundary below the space could also be a light-well.

It was recognised that there was a missing relationship, which was simulated in one exercise. Its correct representation in IFC is 'Ifc Rel Referenced in Spatial Structure' which is intended for secondary relationships between products and spaces other than containment and bounding.

The alternative solution, of ensuring that classification information was attached to the spaces was also adopted. "SL_30_50 : Manufacturing spaces" and "SL_90_10_87 : Stairways". This was documented as the Object Type and as a classification reference.

See also: http://www.iaarc.org/publications/fulltext/FFACE-ISARC15-3001458.pdf

Annex: Contributing authors

Initials	Name	Affiliation or Company	Email
NN	Nicholas Nisbet	buildingSMART UKI and AEC3 and Leeds Beckett University	nn@aec3.com
JKL	Jin Kook Lee	Hanyang University	designit@hanyang.ac.kr

JD	Johannes Dimyadi	University of Auckland	j.dimyadi@auckland.ac.nz
NH	Dr Nicholas Humes	InForm Architecture Ltd.	nick@InFormArch.co.uk

3c Dictionary

3c.1 Introduction

To facilitate widespread implementation of RRR checking for the AEC industry, it is anticipated that a number of resources will be needed, irrespective of the chosen candidate.

Beyond the rule formats and product models discussed elsewhere in Chapter 3 and 4, there may be other resources that can be used to relate the operable rules to the subject project models, such as dictionaries. The preparation of the interoperable formats discussed in Section 4 may yield schedules of required information, such as the required classification of building types, identification of location and dates etc. Such requirements may be mapped to a pre-checking tool or informal requirements (typically a digital Plan of Works, a schedule of objects and expected attributes for a given purpose). However, because of the open-ended nature of regulations, it may not be practical to schedule and map all the properties that may be relevant.

3c.2 Dictionaries

The preparation of the interoperable formats discussed in Section 4 may yield schedules of information which may need to be interpreted following extraction or export from any resource, either an interoperable model (such as IFC) or from a specific expert with a preferred language. It both cases a dictionary may intermediate the terms.

The most obvious resource is a multi-lingual dictionary. BuildingSMART has produced an interoperability format for language dictionaries in ISO 12006 Part 3 and a public implementation can be found on the bsDD server. The scope of dictionary required may be wider than the bsDD as it may need to include:

a. Individual objects, relationships, attributes, and properties.
b. Terms and definitions pertaining to syntactic, semantic, and geometrical requirements.
c. Alternaive synonyms within legal, spoken and written styles.
d. Language alternatives for multiple locales and countries
e. Classifcation codes that are indicative of sematic meaning.
f. Alternative usages from multiple vendor's implementations.
g. Representation of formulae relating terms together
h. Referencing of local or remote services able to provide procedural support.

3c.3 Information requirements

As previously discussed, the preparation of the interoperable approaches may produce schedules of information, such as the required classification of building types, identification of location and dates etc. Such requirements may be mapped to a pre-checking tool or informal requirements (typically a digital Plan of Works, a schedule of objects and expected attributes for a given purpose.

However, because of the open-ended nature of regulations, it may not be practical to schedule and specify all the data that may be necessary. BuildingSMART is proposing mvdXML as the preferred format for information requisition and constraints on specifications and requirements of Model View Definitions (MVDs) defined as a subset of the IFC schema.

MVDs consist of Exchange Requirements (ERs) representing each data exchange processes at the particular phases. Since ERs encompass the same requirements of the same objects or relationships, a modularized unit referred to as a concept is used to define the requirements of ERs using the smallest set of units available. The concepts are generated by the 'concept template' which has the capability to consist of editable attributes and entities of an object, a relationship, or a property. However, since an MVD mainly entails specifications and requirements with regard to semantic binding information necessary for the IFC translation of BIM native data to BIM authoring tools, mvdXML provides limited scope for requirements regarding syntactical and semantical specifications instead of geometrical or topological rules.

MvdXML can be written, imported and exported on the platform of IfcDoc. The IfcDoc tool is supposed to be provided for automated MVD documentation by buildingSMART. Specifications and constraints information of an MVD can be written and shared as a format of mvdXML. To represent constraints and attribute definitions, the mvdXML schema contains the section of Rules. The Rule sections contain EntityRule that refer to AttributeRule items. RuleID of AttriuteRule defines a particular ID used to indicate specific usage of relevant entities within an MVD.

```xml
<Rules>
  <AttributeRule AttributeName="IsNestedBy">
   <EntityRules>
    <EntityRule EntityName="IfcRelNests">
       <AttributeRules>
       <AttributeRule AttributeName="RelatedObjects">
        <EntityRules>
         <EntityRule EntityName="IfcDistributionPort">
          <AttributeRules>
            <AttributeRule AttributeName="Name" RuleID="Name"/>
            <AttributeRule AttributeName="PredefinedType" RuleID="Type"/>
            <AttributeRule AttributeName="FlowDirection" RuleID="Flow"/>
          </AttributeRules>
         </EntityRule>
        </EntityRules>
       </AttributeRule>
      </AttributeRules>
    </EntityRule>
```

Figure 4 The example of a concept template about IfcDistributionElement (Tim C., Thomas L., Matthias W. 2016)

3c.4 Feedback and Reporting

The result of any RRR checking may be a single result (pass/fail or unknown) or it may be a number of subsidiary results (approvals for some sections, not for others). Where an outcome is unknown there may be additional information required. In some cases a document may be generated as a formal or informal notification. In others the results may best be consumed as a message requesting or suggesting changes. In such situations buildingSMART advocates BCF. This format has the ability to convey a description of the issue to a server or individual by email. The description of the issue may include the specific objects (components or spaces or activities) and suggested actions, conveying a number of suggestions or remedial actions.

It is possible that the suggestions or remedial actions should not be applied to one specific object, but that it should be applied to the system, specification or other grouping. In some

instances, it is viable that the suggestions or remedial actions may affect the whole project, for example relocating to another climate zone (such actions are appropriate for resource planning and can be useful for multi-national strategies).

- Tim C., Thomas L., Matthias W. 2016. **"mvdXML Verion 1.1"**, Model Support Group (MSG) of buildingSMART International Ltd.

Annex: Contributing authors

Initials	Name	Affiliation or Company	Email
NN	Nicholas Nisbet	buildingSMART UKI and AEC3 and Leeds Beckett University	nn@aec3.com
YL	Yong-Cheol Lee	Louisiana State University	yclee.it@gmail.com
NH	Dr Nicholas Humes	InForm Architecture Ltd.	nick@InFormArch.co.uk

4 Overview of interoperable candidates

This section is intended to act as an overview and guide to the candidates that are documented in sections 4a-4m. It is not intended that any conclusions are to be found here. However, the information in this document can be used for assessment, where the conclusion is related to the weighting of the criteria.

This section is intended to act as an overview and guide to the approaches that are documented in section 4a-4x. It is not intended that any conclusions are to be found here. However, the information in this document can be used for assessment, where the conclusion is related to the weighting of the criteria.

The following approaches are reviewed:

4n.1   Candidates (Concepts / Methods)

a)   *RASE*
RASE stands for 'Requirements, Applicability, Selection and Exception'. It uses additional mark-up of RRR content to allow it to be computationally processed. By contrast, many mark-up processes focus on search terms or grammatical analysis, however RASE mark-up seeks only to classify the functional role of key noun phrases and paragraphs in terms of the four roles.

b)   *Ifc Constraint Model*
The Ifc Constraint model is one of the core sub-schemas identified in the IFC development diagram. It was introduced in IFC2x and rationalised for IFC4.

The IfcConstraintResource schema provides the specification of constraints (IfcConstraint) that may be applied to any object that is a sub-type of IfcObjectDefinition or IfcPropertyDefinition (through the provision of the relationship class IfcRelAssociatesConstraint).

c)   *Procedural code and BERA*
Procedural code is intended to be a generic term to cover most conventional programming languages such as Java and JavaScript, Dotnet, C / C++

It may also include procedural languages that are closely bound to the target domain. For example, the Singapore ePlanCheck system was developed using ExpressX, the procedural language associated with the Express schema used for buildingSMART

IFC development and standardised in the ISO-10303 product data series of standards. In some areas this coding was supported by DLL libraries for geometric calculations.

d) *SWRL*
Semantic Web Rule Language

e) *SPARQL*
SPARQL and SPIN are implemented technologies for querying and checking RDF data. SPARQL was standardized by the RDF Data Access Working Group (DAWG) of the World Wide Web Consortium (W3C), and is one of the fundamental technologies in the Semantic Web and Linked Data world (Prud'hommeaux, Seaborne 2008).

SPIN is a framework developed and maintained by TopQuadrant to utilize SPARQL as a rule language (Knublauch et al 2011). It is a W3C Member Submission and has received much attention within the Semantic Web community in recent years.

f) *DROOLS*
Drools is a Business Rules Management System (BRMS) solution. It provides a core Business Rules Engine (BRE), a web authoring and rules management application (Drools Workbench) and an Eclipse IDE plugin for core development.

g) *LegalDocML and LegalRuleML*
LegalDocML and LegalRuleML are two emerging open standards that have been under development since 2012 in the legal domain. They are intended to represent any kind of legal documents but can be extended to include recommendatory documents.

There are four aspects to any document (including parliamentary, legislative and judiciary documents):

- Content: a set of words and punctuation that form sentences of the text.
- Presentation: how the information looks, e.g. the colour of the text used in the document, the font used in the headings and other such formatting matters.
- Structure: how the information is organised, e.g., the identification of some parts of text as headings, some parts as clauses, etc.
- Semantics: what the information represents or means.

The literal content, structure and presentation aspects of a document are important to the "human reader" as they maintain the human familiarity with the document but they are not particularly relevant to a machine. The logical content of a document, however, is both human and machine-readable.

LegalDocML and LegalRuleML are intended to operate together to represent all aspects of a document as described above. The metadata contained in LegalDocML

provides a means of maintaining the LegalRuleML counterpart to ensure it is up-to-date.

h) *BIMRL (BIM Rule Language)*
    BIMRL is a research work done by the author as part of his PhD thesis which aims to investigate 'rule checking' using a scientific methodology. Initial findings suggest that a solution to such a problem is to address all parts simultaneously, which includes: an efficient query system, integrated geometry and spatial operators, standardized rule definition language and suitable computable forms for the rules. BIMRL tries to address all of these aspects. A paper published by the author highlights the challenges and the broad categorizations of an automated rule checking system.

i) *Rule Table*
    With the growing number of requirements for client-driven projects, the volume of regulatory and required data has grown quickly. The diverse types of requirements should be categorised and managed consistently so that an automated checking process can be firmly and successfully developed. This section includes the types of rules identified from the IFC interoperability checking project. The project has an objective to figure out the types of syntactic and semantic requirements pertaining to the evaluation of an IFC instance file according to a Model View Definition, the subset of the Industry Foundation Classes (IFC) schema. Considered another way, an MVD consists of criteria to be used for evaluating an IFC instance file according to the specifications of data exchange (Lee, 2015). The types of rules in this section are subject to be updated accordingly.

j) *Semantic web and the Jena Rule Language*
    The increase in the number of documents and data sets described through the languages of the Semantic Web is leading to the development of more and more applications designed to facilitate the subsequent processing. Many of these applications implement inference engines to support the automated processing of these data.

    Within this group, rule-based inference engines are able to reuse rules described in standardized rule languages. This way, rules described in these languages can be applied to infer new knowledge from information described through the Semantic Web languages. Typically, the capabilities of these languages are analysed according to

their expressiveness, syntax, and built-in functions. One of these rule languages is Jena rules. The open source Jena inference engine is used to process Jena rules.

k) *NLP*
NLP stands for Natural Language Processing, it aims to enable computers to understand and process natural language text and speech in a human-like manner (Cherpas 1992). It has been successfully utilized in processing information in many domains such as medical, business, and legal domains. Innovative methods built on NLP have been developed in recent years to support the automated extraction and transformation of building code requirements from textual documents into computable rule formats (Zhang and El-Gohary 2013; 2015).

l) *Prolog*
Prolog is the classic platform for the implementation of the logic programming paradigm. Logic programming is different from other programming paradigms in that logic programs only need to define a problem and the 'solving' process is then automated, thanks to the built-in reasoning mechanism supported by search strategies and backtracking. Prolog was utilized to represent code requirement rules to support automated code compliance checking (Zhang and El-Gohary 2016a).

m) *Commercial rule engines / Business Rule Management Systems*
Solutions for rule based checking is implemented in industries like finance, automobile, aero, oil and gas and others. Other formats presented in RRR can interact with commercial solutions in different ways. No commercial solutions cover the entire process from interpreting code to presentation of results. Commercial solutions are not an alternative, but a supplement for formats presented in this project.

Use Level / Layer Tier to distinguish different perspectives.

In reviewing the candidates, it became apparent that the candidates did not all serve the same purpose, but that they could be organised into layers. So far four layers have been identified, a source layer for the regulations documentation, a mark-up layer which annotates the original documents with referencing and grammatical roles; an interpreted layer, where formal logical rules are documented; and an operational layer where the rules are presented in a form that universal rule engines can process.

O) Source layer – the original document - Source layer: PDF, HTML and word formats, XML.

The source layer for this study is any RRR document. We are excluding already existing applications, methods or coding that may be embedded in applications. We are also excluding mental knowledge or professional opinions.

A) Interpretation of regulations - Mark-up layer: RASE and LegalDocML

The mark-up layer annotates the original documents with referencing and grammatical roles. Mark-up, for example embedded in XHTML, can perform many roles, including hosting metadata, identification and grammatical or semantic analysis. Texts marked-up with RASE can be transformed into interpretative and operational layers.

LegalDocML is a marked-up document containing the literal content of the document it represents. It has a unique feature to capture the entire life-cycle of any document and provides a robust version control when used in conjunction with LegalRuleML, which is designed to represent the logical content of the the document.

B) Processing for rules

Interpretive layer: LegalRuleML, MDL, OWL, SWRL, Procedural coding, Ifc constraint model

The interpretative layer uses a number of formats and syntaxes, but offers a structured form for documenting manual interpretation. Specific engines or compilers may be able to operate on this layer.

C) Generation of results Operational layer: MDL, OWL, SWRL, DROOLS

The operational layer documents formats that can be fed to generic rule engines along with any project model.

RRR as an example of N-Tier Architecture

Previous text has illustrated how layers are used to enable different candidates to be interoperable therefore contributing to the overall solution. As previous clauses about layer

illustrate has different formats unique scope which enable use of multiple formats to be used, and to contribute to a complete solution.

This method is widely in use in software (systems) development. A key characteristic of n-tier architecture is that is that each layer can be developed independently; therefore the interface between each layer is crucial and has to standardise in a precise and accurate way. However, this is demanding and requires – unless the process will be much back-and forth – or iterative.

Figure 1. Presents a general template for N-Tier Architecture. This template will be use d to illustrate the approaches which apply to each tier.
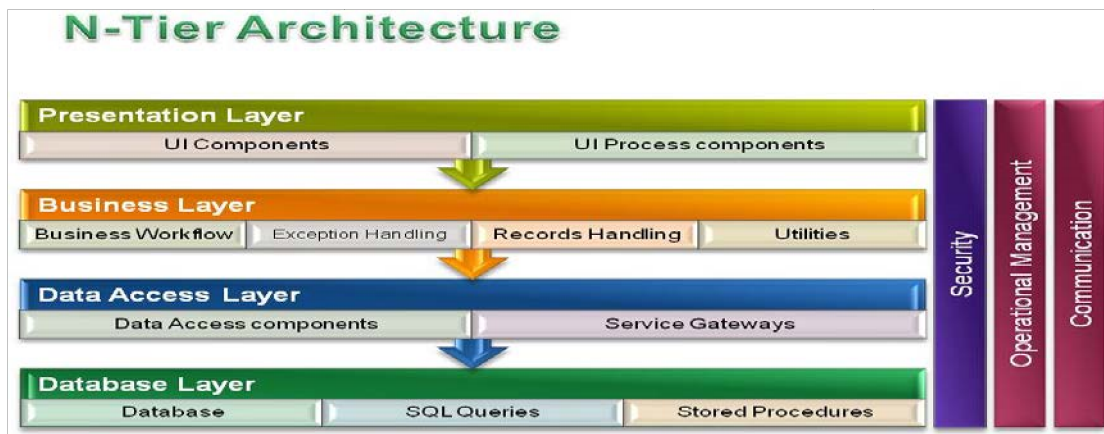
Figure 1. General template for N-Tier Architecture

Annex: Contributing authors

Initials	Name	Affiliation or Company	Email
EH	Eilif Hjelseth	Oslo and Akershus University College of Applied Sciences	Eilif.Hjelseth@hioa.no
JD	Johannes Dimyadi	University of Auckland	j.dimyadi@auckland.ac.nz
NN	Nicholas Nisbet	buildingSMART UKI and AEC3 and Leeds Beckett University	nn@aec3.com

| NH | Dr Nicholas Humes | InForm Architecture Ltd. | nick@InFormArch.co.uk |

4a RASE

4a.1 Introduction

4a.1.1 Names and synonyms

Requirement, Applicability, Selection and Exception (RASE) is an approach to RRR checking developed by AEC3 following the Singapore ePlanCheck project. Whilst this project was succcessul and well in advance of any other system around 2000, it was recognised that for the approach to become widespread there were two strategic challenges. In 2004 AEC3 made an outline proposal to the EU for a development project, but this was not supported and work focussed on the ICC Automated Code Checking Project, later branded 'SmartCodes'.

4a.1.2 Description

RASE uses additional mark-up of RRR content to render it computable. Many mark-up processes focus on search terms or gramatical analysis while RASE mark-up seeks only to classify the functional role of key noun phrases and paragraphs in terms of the four logical roles: Requirement, Applicability, Selection and Exception .

4a.2 Strategic assessment of format

4a.2.1 Control, ownership, availability

RASE was developed by AEC3 and has been extensively published and demonstrated. It has been used in US Army ERDC funded projects "Facility Capacity Analysis" and "Model Compliance Checking". It has also been used in the UK InnovateUK RegBIM project where three UK Building Regulations Approved Documents, the BREEAM environmental assessment method and an environmental housing standards called the 'Code for Sustainable Homes' were automated.

4a.2.2 Suitability

RASE is suitable for the following functions:

- Authoring
- Computer-assisted and Automated code compliance checking
- Analysis and code comparisons
- Filtering and structured dialogues
- Controlling parametric objects.

The purpose of RASE is two-fold, firstly to render the text computable and secondly to facilitate the mapping of RRR content to other formats.

1. For ICC SmartCodes, and US Army ERDC, the RASE-marked-up content was processed to generate an IfcConstraint model (see section 4b). This was then consumed along with the target project models also in IFC. Three software implementations were demonstrated: AEC3-XABIO with Octaga viewer, Solibri Model Checker with add-on, and Singapore ePlanCheck with Fornax Viewer.
2. For the Innovate UK RegBIM project, the mark-up was mapped into the Java Drools DRL format (see section 4c).
3. From 2015 onwards AEC3 has been demonstrating a checking engine which consumes the RASE marked-up content directly, without use of any intermediate format.

4a.3 Commercial base

4a.3.1 authoring/capture tools (singular, multiple)

Initially, markup was performed using a standard XML editor, Altova XMLSPY which was applied to ICC's proprietary internal document XML format. Latterly, the ICC SmartCodes 'editor', which was a controlled mark-up tool, was developed but that is now not commercially supported.

All three of the commercial demonstrations still exist, though the Solibri add-on is not available commercially.

For the US Army ERDC projects, mark-up was acheived using Altova XMPLSPY for the textual content. Tabular content was transformed automatically to insert markup systematically using standard XSLT transformation tools.

All three of the commercial demonstrations still exist, though the Solibri add-on is not available commercially.

A range of tools are available including:

- AEC3-XABIO with Octaga viewer and Singapore ePlanCheck with Fornax Viewer both used a core engine written in ExpressX by AEC3. The Solibri Model Checker add-on was Codes by Solibri.
- The US Army ERDC projects used a command line utility written by AEC3 called Compliance1 which was released as part of the AEC3 BimServices toolkit and has been freely available since 2010.
- For the Innovate UK RegBIM project, the rule engine was Java Drools. The University of Cardiff developed and tested the wrapper around this.
- From 2015 onwards AEC3 has been demonstrating a checking engine which consumes the RASE marked-up content directly.

4a.3.3 Scalability and role in creating a market place for RRR applications

Because RASE acts as an efficient method of capturing RRR content it facilitates an efficient process, perhaps 15x more efficient than procedural coding, based on an analysis conducted a commercial client with AEC3. By being embedded in the source test, it makes the version and revision control process implicit, and allows all subsequent analysis able to refer to the specific noun phrases in the specific document, adding credibility and transparency.

4a.4 Technical

4a.4.1 Expressivity and generality

RASE has the following attributes:

- Any RRR content can be marked up. With RASE there is no need to anticipate later issues of computability or whether data will be available. Given the speed with which text can be marked up it is preferable that all requirements be tagged. Prioritisation of which clauses need automation emerges naturally from the execution process, where some clauses are rarely explored, whilst others emerge as critical.
- The four RASE roles are treated equally and symmetrically so that the regulation can be driven in any direction: simple heuristics are easily included, such as if a pipe is already known to be red, there is no need to further check a regulation that requires hot water pipes to be red.
- Regulations that are predominantly performance based can be handled equally well as the engine can track any requirements for evidence (such as acceptable calculation sheets).

- Regulations that are predominantly procedural can also be handled if the engine has access to the state of the process or can query the applicant for the additional information.
- Issues surrounding heuristics can be delegated to the rule engine: for example, some engines may track the difficulty of getting a particular metric answered and weigh that against its importance in the overall outcome.
- Target (project) model schemas and formats. RASE makes no assumptions about the target domain, nor about the target schema. RASE documents are XHTML to ensure that they are directly addressable by any XML technology.
- No performance characteristics have been measured. Because of its support for applicability, the engine must begin by applying every RRR against every object. However, in most cases this is a momentary test. (This pencil passes the Building Act because the regulation is applicable only to habitable structures).
- Ability to work with external dictionaries. The Smartcode implementation worked with a local RDF/OWL dictionary. Compliance1 used a suite of Microsoft DotNet DLLs to provide the dictionary functionality. For Innovate UK Regbim and subsequently AEC3 Require1 both use a local XML dictionary.

4a.4.2 Acceptability and provenance

Linkages back to the source of the requirements are created by noting the URI to the specific ID attached to the mark-up. This link can be passed through to any resulting report or user interface.

Linkages forward to the project model. Linkages forward to the ID or GUID of the project model objects are captured and can be used to generate BCF messages or to create a render or 3D interactive view with the objects in question highlighted in red.

Depth of results that can be reported. All results are entered first into an XML log that can be transformed and rendered to provide as much or as little feedback as desired.

4a.5 Definitive references and contacts

- Wix, J., Nisbet, N. and Liebich, T. 2008. "**Using Constraints to Validate and Check Building Information Models"** 7th European Conference on Product and Process Modeling (ECPPM) Sophia Antipolis. 10.-12 Sept. 2008 p. 467-476 Edited by Raimar Scherer and Alain Zarli Taylor & Francis 2008. Print ISBN: 978-0-415-48245-5. eBook ISBN: 978-0-203-88332-7
- http://www.aec3.com/en/5/5_013_ICC.htm
- Nisbet N, Wix J and Conover D. 2008. "**The future of virtual construction and regulation checking**", in Brandon, P., Kocaturk, T. (Eds), Virtual Futures for Design, Construction and Procurement, Blackwell, Oxfordshire. doi: 10.1002/9781444302349.ch17
- Eilif Hjelseth, Nick Nisbet (2010) "**Overview of Concepts for Model Checking**" in CIB W78 2010 27th International Conference - Applications of IT in the AEC Industry.
- Eilif Hjelseth, Nick Nisbet (2010) "**Exploring Semantic Based Model Checking**" in CIB W78 2010 27th International Conference - Applications of IT in the AEC Industry.

- Eilif Hjelseth, Nick Nisbet (2011) "**Capturing normative constraints by use of the semantic mark-up (RASE) methodology**" in CIB W78 2011 28th International Conference - Applications of IT in the AEC Industry.
- (see also Eilif Hjelseth "**Converting performance based regulations into computable rules in BIM based model checking software**").
- E. William East, Nicholas Nisbet (2010) "**Facility Capacity Analysis**" in CIB W78 2011 28th International Conference - Applications of IT in the AEC Industry.
- Nisbet, N., and S. Lockley. (2012). "**Automated enhancement of BIM models**." ECPPM 2012
- Beach TH, Kasim T, Li H, Nisbet N, Rezgui Y, "**Towards automated compliance checking in the construction industry**", *Lecture Notes in Computer Science* , 8055 (2013) 366-380 ISSN 0302-9743 10.1007/978-3-642-40285-2_32

Applications

- Nisbet, N. (2010) "**BimServices – Command-line and Interface utilities for BIM**" http://www.aec3.com/en/6/6_04.htm (cited 01-Jun-11), including
  - Transform1 for data schema interoperability
  - Compliance1 for model validations
- Nisbet, N. (2014) "**BimServices – Requirements Workbench**", including
  - Require1 (2014) for document mark-up and rule extraction
  - Require1 (2015) with dictionary maintenance interface
  - Require1 (planned 2017) with integrated compliance checking.

4a.6 Background references and publications, authors and contacts for further discussions.

Example 1: Korean Building Act 34-1 marked up using RASE.

Example 2: Korean Building Act 34-1 reflected back as structured prose.

Page no.	Author
39	bSI Regulatory Room

Korean Building Act 34-1 expects ... ... Objective kr-ba-34-01.

~

Objective kr-ba-34-01 expects either first that *...* Objective kr-ba-34-01a or second that *not* stair exists or third that *neither* stair leading to shelter floor *nor* stair leading to ground or last that *...* stair leading to shelter floor .

~

Objective kr-ba-34-01a expects either *...* space walking distance is less than or equal to 30 m or *either* Objective kr-ba-34-01-e1 *or* Objective kr-ba-34-01-e2.

~

Objective kr-ba-34-01-e1 expects either *...* space walking distance is less than or equal to 50 m or *neither* structure isFireProof *nor* structure isNonCombustible.

~

Objective kr-ba-34-01-e2 expects either *...* space walking distance is less than or equal to 75 m or *either not* building usage factory *or not* building proscribedUse *or not* building usage factory or *neither* building hasSystem automatic fire extinguishers *nor* building hasSystem sprinklers .

Example 3: Korean Building Act 34-1 reflected back as a list of metrics

	Phrase	Object	Property	Comparison	Target	Unit
Selection	automatic fire extinguishers	building	hasSystem		automatic fire extinguishers	
Selection	sprinklers	building	hasSystem		sprinklers	
Application	prescribed by Ordinance of the Ministry of Land, Infrastructu	building	proscribedUse			

	re and Transport					
Application	factory	building	usage		factory	
Application	automated production facility	building	usage		factory	
Requirement	walking distance from each part of the living room to the stairs is not more than 30 meters	space	walking distance	<=	30	m
Requirement	walking distance of not more than 50 meters	space	walking distance	<=	50	m
Requirement	the walking distance of not more than 75 meters	space	walking distance	<=	75	m
Application	stairs	stair	exists			
Selection	leading to the shelter floor	stair	leading to		shelter floor	
Selection	the ground	stair	leading to		ground	
Exception	the shelter floor	stair	leading to		shelter floor	
Selection	fireproof structure	structure	isFireProof			

Selection	non-combustible materials	structure	isNonCombustible			

Example 4: Korean Building Act 34-1 reflected back as logical tree

- Korean Building Act 34-1 expects
  - ...
    - ...
      - *either first*
        - ...
          - *either*
            - ... space walking distance <= 30 m
          - *or*
            - *either*
              - *either*
                - ... space walking distance <= 50 m
              - *or*
                - *neither* structure isFireProof
                - *nor* structure isNonCombustible
            - *or*
              - *either*
                - ... space walking distance <= 75 m
              - *or*
                - *either not* building usage factory
                - *or not* building proscribedUse
                - *or not* building usage factory
              - *or*
                - *neither* building hasSystem automatic fire extinguishers
                - *nor* building hasSystem sprinklers
      - *or second*
        - *not* stair exists
      - *or third*
        - *neither* stair leading to shelter floor
        - *nor* stair leading to ground
      - *or last*
        - ... stair leading to shelter floor

Annex: Contributing authors

Initials	Name	Affiliation or Company	Email
NN	Nicholas Nisbet	buildingSMART UKI and AEC3 and Leeds Beckett University	nn@aec3.com

4b. Ifc Constraint Model

4b.1 Introduction

4b.1.1 The Ifc Constraint model is one of the core sub-schemas identified in the IFC development diagram. It was introduced in IFC2x and rationalised for IFC4.

4b.1.2 Description

The IfcConstraintResource schema provides for the specification of constraints (IfcConstraint) that may be applied to any object that is a subtype of IfcObjectDefinition or IfcPropertyDefinition (through the provision of the relationship class IfcRelAssociatesConstraint). Also, constraints may be applied to specific resource objects such as an IfcProperty (through the provision of the relationship class IfcResourceConstraintRelationship).

A grade may be set for the constraint that establishes whether it is a hard constraint (must be satisfied), a soft constraint (should be satisfied) or simply advisory.

A constraint must be named and may have one or more sources within which it is defined or from which it is taken. Additionally, a constraint may optionally be assigned a creating actor, creation date and a description.

Constraints may be either qualitative (an objective constraint) or quantitative (a measured constraint or metric). A qualifier can be applied to an objective constraint that determines the purpose for which it is applied. It may be applied to define the constraining values beyond which building codes may be violated or to limit the resulting range of values (for example, value of X must be greater than A, but less than B). Several possible purposes are provided through an enumeration.

A measured constraint or metric defines the actual value or values to be compared. Values can be defined in terms of a benchmark requirement which sets the intent of the constraint, for example, whether the benchmark is greater than or less than. The value of a constraint may be defined according to a number of data types that are available through a select mechanism.

EXAMPLE:

Where a manufactured specifies that their pump should be serviced every six months, 'a 6 month countdown' would be the 'TriggerCondition', named 'PumpMaintenanceCondition', have as its source 'ManufacturerData' and be graded as 'Advisory'. It could have as a single value 10^-2 / sec as the frequency of vibration and have a benchmark of 'GreaterThanOrEqualTo'.
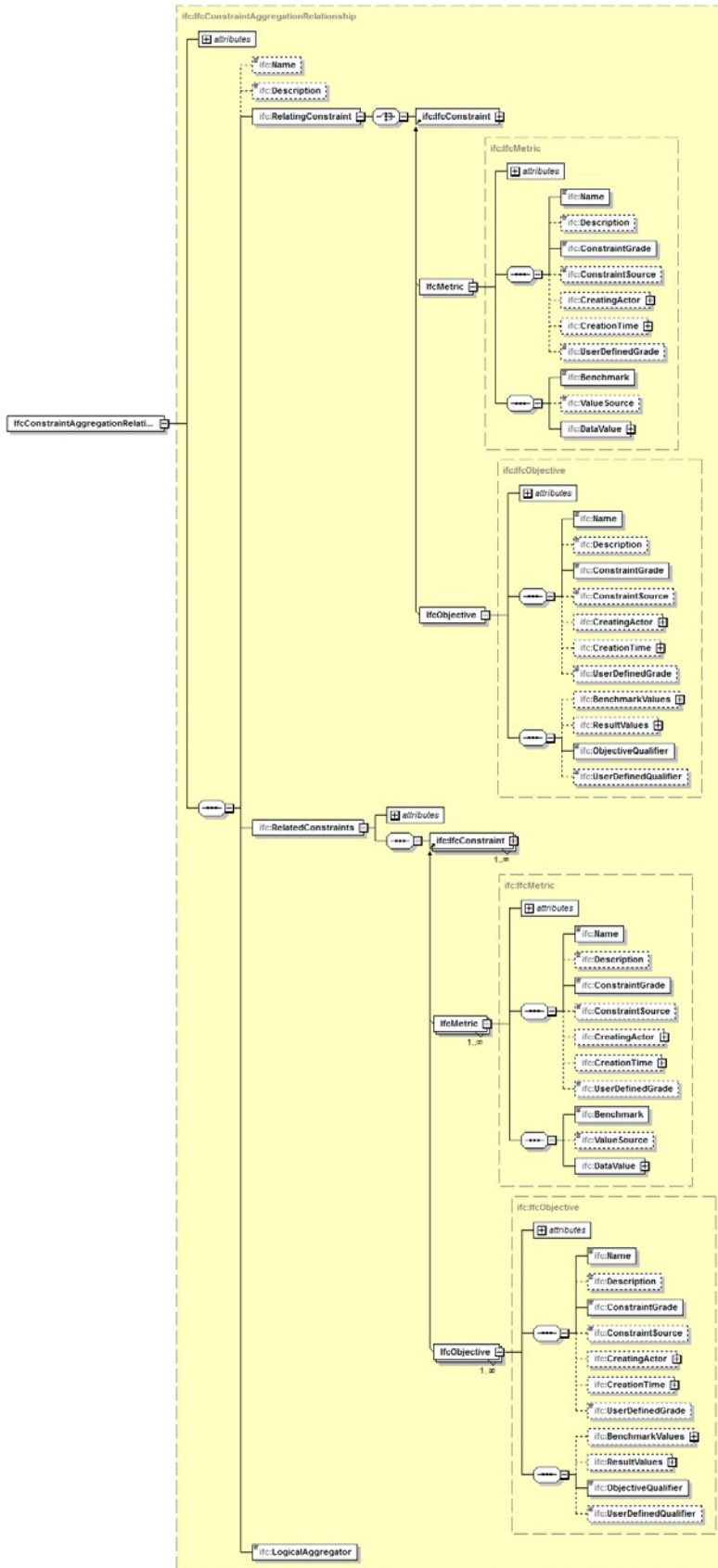
Figure 5: Ifc Constraint Resource (Ifc2x3)

Figure 6: Ifc Constraint Resource (IFC4)

Page no.	Author
49	bSI Regulatory Room

4b.2 Strategic assessment of format

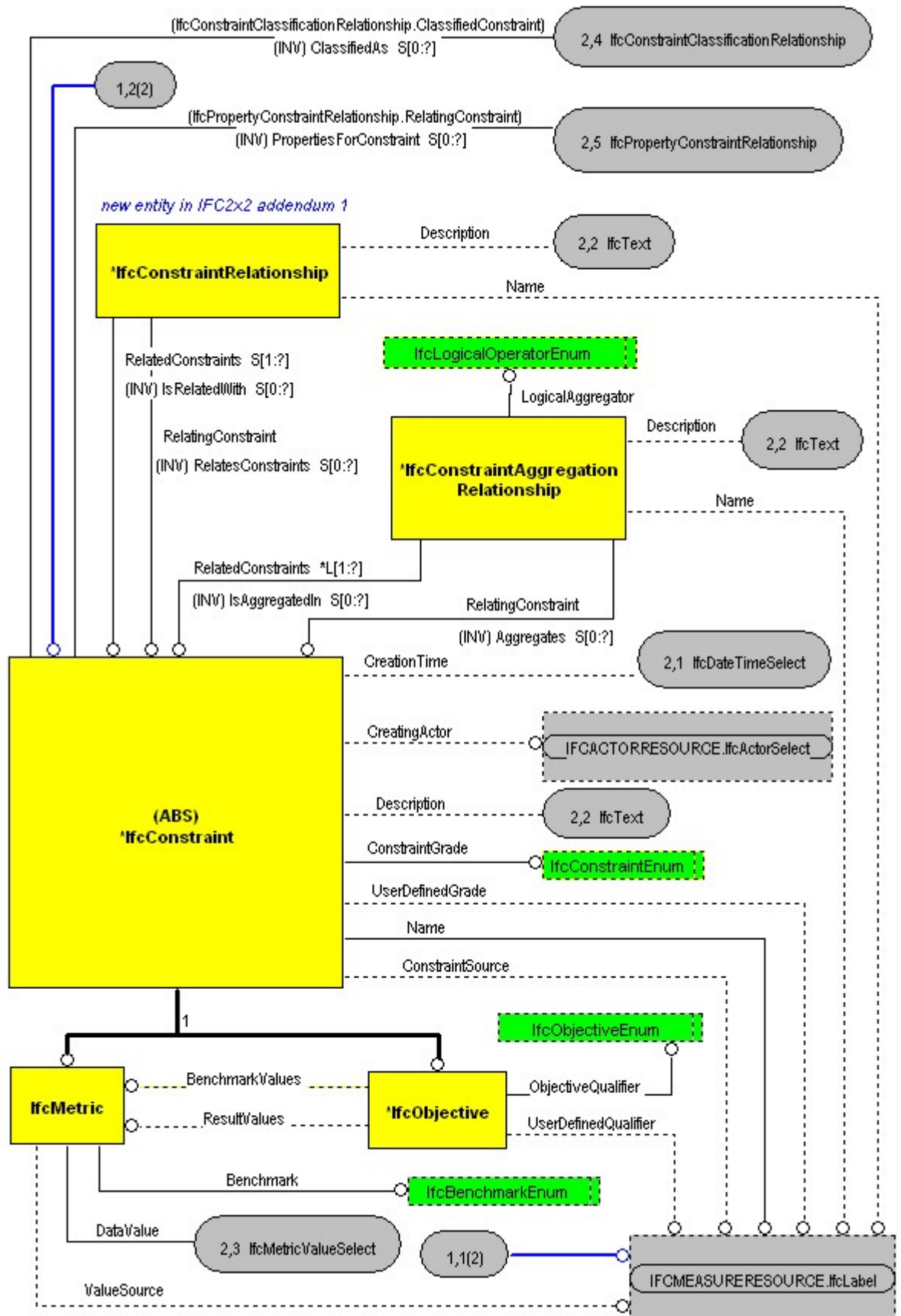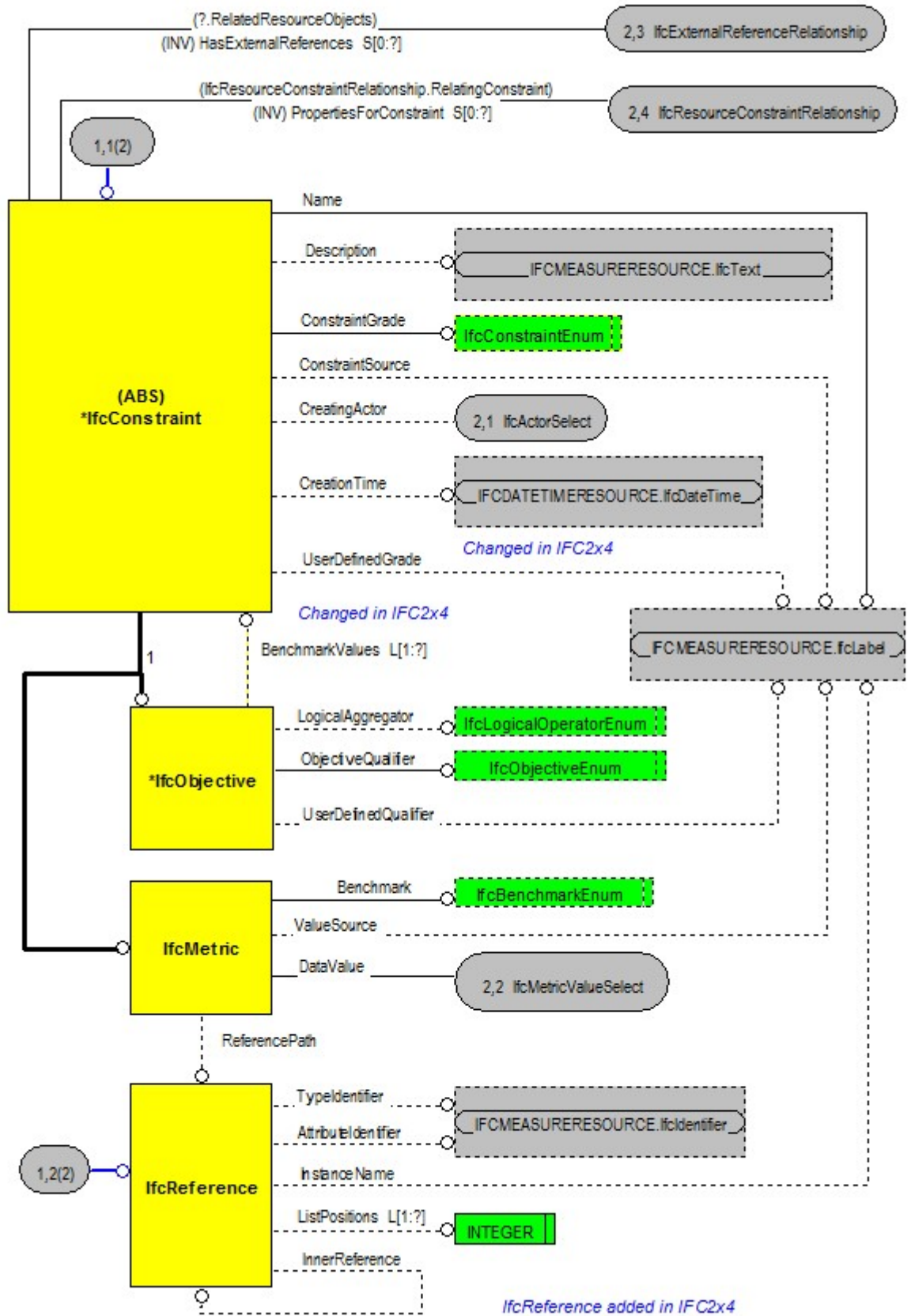4b.2.1 The IfcConstraint Resource is part of buildingSMART's ISO 16859 IFC schema. It is built on two kinds of IfcConstraint. An IfcObjective is described but is defined by its logical relationship to other IfcObjectives and IfcMetrics. An IfcMetric is described but defined by the examination of another object. That object can be in the same model, in a remote model, or defined in terms of a search term. The evaluation depends on a comparison against a second value.

4b.2.2 The changes made to IFC4 have replaced a separate objectified relationship with a more hierarchical representation directly from IfcObjective to its deciding IfcConstraints. IFC4 also introduced two missing logical operators. The majority of demonstrations of the use of the IfcConstraint model have been made using Ifc2x3.

The IFC Constraint approach is suitable for multiple purposes as:

- The simplicity makes authoring or generation relatively straight-forward.
- Computer-assisted and automated code compliance checking, and Filtering and structured dialogues has been demonstrated in the ICC SmartCodes and US Army MCA and FCA projects.
- The schema representation is neutral as to how it is explored and so analysis and comparison is possible.
- It can be used to control parametric objects. This has been demonstrated on an IKEA configurable kitchen unit (available for review) by Tim Chipman of Constructivity.

4b.3 Commercial base

Toolkits for manipulating IFC are built into most BIM authoring tools and are available separately:

Free / Open source tools include:

- XBIM
- Bim Server
- IfcEngine (RDF)

Commercial tools include:

- EPM
- SiteTools

See further lists on the buildingSMART website.

4b.3.1 Authoring/capture tools (singular, multiple).

Authoring has been achieved through:

- RASE markup leading to the generation of IfcConstraint models.
- XSLT mappings from large data tables in HTML and in Spreadsheets.

4b.3.2 assessment engines, software and toolkits for (singular, multiple).

To date, the AEC3 XABIO for ICC, Solibri for ICC and ePlanCheck for ICC, Compliance1 for US Army ERDC and UK Innovate UK RegBIM projects have used the IfcConstraint format.

4b.3.3 Scalability and role in creating a market place for RRR applications.

Given the dominance of IFC in the building and infrastructure sector, it may make market sense to deliver AEC RRR content as IfcConstraint models.

4b.4 Technical

4b.4.1 Expressivity and generality

The IFC Constraint approach is characterised by the following:

- To date, all RRR content has been expressed with the use of several conventions for the search/filter terms.
- The IfcConstraint has an attribute ConstraintSource which can be used to hold a hyperlink or other reference to the original RRR content.
- Ability to handle 'easy' technical requirements through to 'hard' management requirements by representing the logical structure
- Target (project) model naturally includes IFC as STEP, ifcXML and ifcOWL.
- Performance measured against an independent RRR performance benchmark and support for heuristics and optimisation.
- It has the capability to work with external dictionaries. The search/filter terms can be put through the bsDD or an equivalent local dictionary to obtain a refined query against an IFC project model.
- Currently the IfcConstraint schema cannot represent uncertainty, as may be required for Recommendatory content. There is no uncertainty in the logical relationships between constraints.
- Within an individual metric, any uncertainty in the comparison between a property and a target value, nor any uncertainty in the result of evaluating a metric or objective.
-

4b.4.2 Acceptability and provenance

- Linkages back to RRR source are held in the ConstraintSource attribute. This can be a hyperlink to a specific marker within the text.

- Linkages forward to (project) model are typically remote and soft, based on the search/filter criteria.
- The logical structure allows all contributing decisions to be identified and expressed as possible remedies. For example, using a candidate regulation for Dubai, a room that was failed was supported with the following suggestions:

1. Remeasure the space, an example of evidential support being sometimes required
2. Enlarge the space, an example of a simple response to the last cause of failure
3. Reclassify the space as not habitable, an example of a response to an intermediate pass and a potential trigger for an evidential dispute
4. Reclassify the structure as not a habitable building, an example of a response to top level scope pass and a potential trigger for an evidential dispute
5. Relocate the facility to a different jurisdiction.

These suggestions were logically correct, and were felt to be acceptable as plausible responses to a particular problem.

Example 1: Korean Building Act 34-1 as Ifc Constraints automatically generated from RASE

The example represents Korean Building Act 34-1 as a number of Ifc Metrics joined to aggregate into Ifc Objectives.

BuildingSMART IFC Constraint model of Regulation, Requirements and Recommendations

Entity	Name	Detail
IfcObjective i100000	Korean Building Act 34-1_kr-ba-34-01 kr-ba-34-01a kr-ba-34-01-e1 kr-ba-34-01-e2_x0	user defined i100001 i100002 document codecompliance
IfcOrganization i100001	AEC3 UK Ltd	AEC3 Korean Building Act 34-1 i100038
IfcCalendarDate i100002		25 3 2008
IfcConstraintAggregationRelationship i100003		i100000 i100004 logicalor

IfcObjective i100004	Korean Building Act 34-1_kr-ba-34-01 kr-ba-34-01a kr-ba-34-01-e1 kr-ba-34-01-e2_x0_asrequired	as required notdefined codecompliance
IfcConstraintAggregationRelationship i100005		i100004 i100006 logicaland
IfcObjective i100006	kr-ba-34-01	userdefined i100001 i100002 document code compliance
IfcConstraintAggregationRelationship i100007		i100006 (i100008 i100009 i100010 i100011) logicalor
IfcObjective i100008	kr-ba-34-01_notselected	not selected notdefined codecompliance
IfcObjective i100009	kr-ba-34-01_notapplicable	not applicable notdefined codecompliance
IfcObjective i100010	kr-ba-34-01_excepted	excepted notdefined codecompliance
IfcObjective i100011	kr-ba-34-01_asrequired	as required notdefined codecompliance
IfcConstraintAggregationRelationship i100012		i100008 (i100013 i100014) logicalnotor
IfcMetric i100013	Selection leading to eq shelter floor	leading to the shelter floor userdefined i100001 i100002 Selection equalto shelter floor
IfcMetric i100014	Selection leading to eq ground	the ground userdefined i100001 i100002 Selection equalto ground
IfcConstraintAggregationRelationship i100015		i100009 i100016 logicalnotand
IfcMetric i100016	Application exists eq true	stairs userdefined i100001 i100002 Application equalto true

|		
IfcConstraintAggregationRelationship i100017		i100010 i100018 logicalor
IfcMetric i100018	Exception leading to eq shelter floor	the shelter floor userdefined i100001 i100002 Exception equalto shelter floor
IfcConstraintAggregationRelationship i100019		i100011 i100020 logicaland
IfcObjective i100020	kr-ba-34-01a	userdefined i100001 i100002 document codecompliance
IfcConstraintAggregationRelationship i100021		i100020 (i100022 i100023 i100024) logicalor
IfcObjective i100022	kr-ba-34-01a_notselected	not selected notdefined codecompliance
IfcObjective i100023	kr-ba-34-01a_notapplicable	not applicable notdefined codecompliance
IfcObjective i100024	kr-ba-34-01a_asrequired	as required notdefined codecompliance
IfcConstraintAggregationRelationship i100025		i100022 (i100026 i100027 i100028 i100029) logicalnotor
IfcMetric i100026	Selection isFireProof eq true	fireproof structure userdefined i100001 i100002 Selection equalto true
IfcMetric i100027	Selection isNonCombustible eq true	non-combustible materials userdefined i100001 i100002 Selection equalto true
IfcMetric i100028	Selection hasSystem eq automatic fire extinguishers	automatic fire extinguishers userdefined i100001 i100002 Selection equalto automatic fire extinguishers
IfcMetric i100029	Selection hasSystem eq sprinklers	sprinklers userdefined i100001 i100002 Selection equalto sprinklers

IfcConstraintAggregationRelationship i100030		i100023 ([i100031](#) [i100032](#) [i100033](#)) logicalnotand
IfcMetric i100031	Application usage eq factory	factory userdefined [i100001](#) [i100002](#) Application equalto factory
IfcMetric i100032	Application proscribedUse eq true	userdefined [i100001](#) [i100002](#) Application equalto true
IfcMetric i100033	Application usage eq factory	automated production facility userdefined [i100001](#) [i100002](#) Application equalto factory
IfcConstraintAggregationRelationship i100034		i100024 ([i100035](#) [i100036](#) [i100037](#)) logicaland
IfcMetric i100035	Requirement walking distance <= 30 m	walking distance from each part of the living room to the stairs is not more than 30 meters userdefined [i100001](#) [i100002](#) Requirement lessthanorequalto 30 : m
IfcMetric i100036	Requirement walking distance <= 50 m	walking distance of not more than 50 meters userdefined [i100001](#) [i100002](#) Requirement lessthanorequalto 50 : m
IfcMetric i100037	Requirement walking distance <= 75 m	the walking distance of not more than 75 meters userdefined [i100001](#) [i100002](#) Requirement lessthanorequalto 75 : m
IfcActorRole i100038		userdefined codecompliance regulation, requirements and recommendations management

Example 2: Korean Building Act 34-1 IFC file automatically generated from RASE


```
#100000 = IFCOBJECTIVE('Korean Building Act 34-1_kr-ba-34-01 kr-ba-34-01a kr-ba-34-01-e1 kr-ba-34-01-e2_x0', '', .USERDEFINED., '', #100001, #100002, 'document', $, $, .CODECOMPLIANCE., $);
#100003 = IFCCONSTRAINTAGGREGATIONRELATIONSHIP($, $, #100000, (#100004), .LOGICALOR.);
#100004 = IFCOBJECTIVE('Korean Building Act 34-1_kr-ba-34-01 kr-ba-34-01a kr-ba-34-01-e1 kr-ba-34-01-e2_x0_asrequired', 'as required', .NOTDEFINED., $, $, $, $, $, .CODECOMPLIANCE., $);
#100005 = IFCCONSTRAINTAGGREGATIONRELATIONSHIP($, $, #100004, (#100006), .LOGICALAND.);
#100006 = IFCOBJECTIVE('kr-ba-34-01', '', .USERDEFINED., '', #100001, #100002, 'document', $, $, .CODECOMPLIANCE., $);
#100007 = IFCCONSTRAINTAGGREGATIONRELATIONSHIP($, $, #100006, (#100008, #100009, #100010, #100011), .LOGICALOR.);
#100008 = IFCOBJECTIVE('kr-ba-34-01_notselected', 'not selected', .NOTDEFINED., $, $, $, $, $, .CODECOMPLIANCE., $);
#100009 = IFCOBJECTIVE('kr-ba-34-01_notapplicable', 'not applicable', .NOTDEFINED., $, $, $, $, $, .CODECOMPLIANCE., $);
#100010 = IFCOBJECTIVE('kr-ba-34-01_excepted', 'excepted', .NOTDEFINED., $, $, $, $, $, .CODECOMPLIANCE., $);
#100011 = IFCOBJECTIVE('kr-ba-34-01_asrequired', 'as required', .NOTDEFINED., $, $, $, $, $, .CODECOMPLIANCE., $);
#100012 = IFCCONSTRAINTAGGREGATIONRELATIONSHIP($, $, #100008, (#100013, #100014), .LOGICALNOTOR.);
#100013 = IFCMETRIC('Selection  leading to eq shelter floor', 'leading to the shelter floor', .USERDEFINED., '', #100001, #100002, 'Selection', .EQUALTO., '', IFCTEXT('shelter floor'));
#100014 = IFCMETRIC('Selection  leading to eq ground', 'the ground', .USERDEFINED., '', #100001, #100002, 'Selection', .EQUALTO., '', IFCTEXT('ground'));
#100015 = IFCCONSTRAINTAGGREGATIONRELATIONSHIP($, $, #100009, (#100016), .LOGICALNOTAND.);
#100016 = IFCMETRIC('Application  exists eq true', 'stairs', .USERDEFINED., '', #100001, #100002, 'Application', .EQUALTO., '', IFCTEXT('true'));
#100017 = IFCCONSTRAINTAGGREGATIONRELATIONSHIP($, $, #100010, (#100018), .LOGICALOR.);
#100018 = IFCMETRIC('Exception  leading to eq shelter floor', 'the shelter floor', .USERDEFINED., '', #100001, #100002, 'Exception', .EQUALTO., '', IFCTEXT('shelter floor'));
#100019 = IFCCONSTRAINTAGGREGATIONRELATIONSHIP($, $, #100011, (#100020), .LOGICALAND.);
#100020 = IFCOBJECTIVE('kr-ba-34-01a', '', .USERDEFINED., '', #100001, #100002, 'document', $, $, .CODECOMPLIANCE., $);
#100021 = IFCCONSTRAINTAGGREGATIONRELATIONSHIP($, $, #100020, (#100022, #100023, #100024), .LOGICALOR.);
#100022 = IFCOBJECTIVE('kr-ba-34-01a_notselected', 'not selected', .NOTDEFINED., $, $, $, $, $, .CODECOMPLIANCE., $);
#100023 = IFCOBJECTIVE('kr-ba-34-01a_notapplicable', 'not applicable', .NOTDEFINED., $, $, $, $, $, .CODECOMPLIANCE., $);
#100024 = IFCOBJECTIVE('kr-ba-34-01a_asrequired', 'as required', .NOTDEFINED., $, $, $, $, $, .CODECOMPLIANCE., $);
#100025 = IFCCONSTRAINTAGGREGATIONRELATIONSHIP($, $, #100022, (#100026, #100027, #100028, #100029), .LOGICALNOTOR.);
#100026 = IFCMETRIC('Selection  isFireProof eq true', 'fireproof structure', .USERDEFINED., '', #100001, #100002, 'Selection', .EQUALTO., '', IFCTEXT('true'));
#100027 = IFCMETRIC('Selection  isNonCombustible eq true', 'non-combustible materials', .USERDEFINED., '', #100001, #100002, 'Selection', .EQUALTO., '', IFCTEXT('true'));
#100028 = IFCMETRIC('Selection  hasSystem eq automatic fire extinguishers', 'automatic fire extinguishers', .USERDEFINED., '', #100001, #100002, 'Selection', .EQUALTO., '', IFCTEXT('automatic fire extinguishers'));
#100029 = IFCMETRIC('Selection  hasSystem eq sprinklers', 'sprinklers', .USERDEFINED., '', #100001, #100002, 'Selection', .EQUALTO., '', IFCTEXT('sprinklers'));
#100030 = IFCCONSTRAINTAGGREGATIONRELATIONSHIP($, $, #100023, (#100031, #100032, #100033), .LOGICALNOTAND.);
#100031 = IFCMETRIC('Application  usage eq factory', 'factory', .USERDEFINED., '', #100001, #100002, 'Application', .EQUALTO., '', IFCTEXT('factory'));
#100032 = IFCMETRIC('Application  proscribedUse eq true', '', .USERDEFINED., '', #100001, #100002, 'Application', .EQUALTO., '', IFCTEXT('true'));
#100033 = IFCMETRIC('Application  usage eq factory', 'automated production facility', .USERDEFINED., '', #100001, #100002, 'Application', .EQUALTO., '', IFCTEXT('factory'));
#100034 = IFCCONSTRAINTAGGREGATIONRELATIONSHIP($, $, #100024, (#100035, #100036, #100037), .LOGICALAND.);
#100035 = IFCMETRIC('Requirement  walking distance <= 30 m', 'walking distance from each part of the living room to the stairs is not more than 30 meters', .USERDEFINED., '', #100001, #100002, 'Requirement',
#100036 = IFCMETRIC('Requirement  walking distance <= 50 m', 'walking distance of not more than 50 meters', .USERDEFINED., '', #100001, #100002, 'Requirement', .LESSTHANOREQUALTO., '', IFCTEXT('50 :
#100037 = IFCMETRIC('Requirement  walking distance <= 75 m', 'the walking distance of not more than 75 meters', .USERDEFINED., '', #100001, #100002, 'Requirement', .LESSTHANOREQUALTO., '', IFCTEXT('75 : m'));
```
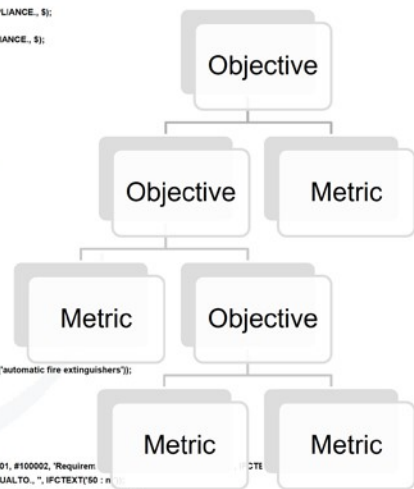
Annex: Contributing authors

Initials	Name	Affiliation or Company	Email
NN	Nicholas Nisbet	buildingSMART UKI and AEC3 and Leeds Beckett University	nn@aec3.com
NH	Dr Nicholas Humes	InForm Architecture Ltd.	nick@InFormArch.co.uk

4c Procedural code generation

4c.1 Introduction

Procedural code is intended to be a generic term to cover most conventional programming languages such as

- Java and javascript
- Dotnet
- C / C++
- Visual basic

It may also include procedural languages that are closely bound to the target domain. For example, the Singapore ePlanCheck system was developed using ExpressX, the procedural language associated with the Express schema used for buildingSMART IFC development and standardised in the ISO-10303 product data series of standards. In some areas this coding was supported by DLL libraries for geometric calculations.

It is characterised by a fixed direction of flow, based on a fixed set of inputs and a fixed set of outputs. In this, it is easily inserted into a specific business process.

4c.2 Interpreted and Generated coding

Procedural code can be written by hand (interpretive) or it may be generated from other resources.

Interpreted code depends on the understanding of the source RRR content, the capabilities of the chosen language and on the target domain model. For ePlanCheck this frequently required discussions between Singapore code experts, ExpressX authors and the core IFC development team. This interpretive step was considered a weak link in the connection between the RRR content and the results. This is because the similarity between the source RRR content and the code cannot be verified except by joint expertise or by repeated and extensive public scrutiny and testing.

[Enforcement of the Building Act Article 34 Clause 1]

"On each floor of a building, direct stairs leading to the shelter floor or the ground other than the shelter floor shall be installed in the way that the walking distance from each part of the living room to the stairs is not more than 30 meters: Provided, That in cases of a building of which main structural part is made of a fireproof structure or non-combustible materials, the walking distance of not more than 50 meters may be established, and in cases of a factory prescribed by Ordinance of the Ministry of Land, Infrastructure and Transport, which is equipped with automatic fire extinguishers such as sprinklers, in an automated production facility, the walking distance of not more than 75 meters may be established." [31]

KBimCode

```
IF (getFloorProperty() != "egressFloor")
    ED = 30m   //Egress Distance
    IF (getBuildingUsage() = "unmanned factory"
        AND isExist(automaticFireExtinguishers) = TRUE)
            ED = 100m
    ELSE IF (getBuildingUsage() = "factory"
        AND isExist(automaticFireExtinguishers) = TRUE)
            ED = 75m
    ELSE IF
        (getObjectMaterialType(mainStructuralPart")
        = "non-combustible materials"
        AND isFireResistant(mainStructuralPart) =
        TRUE)
            ED = 50m
    END IF
    getObjectDistance(livingRoom, stair, MRP) <= ED
END IF
```
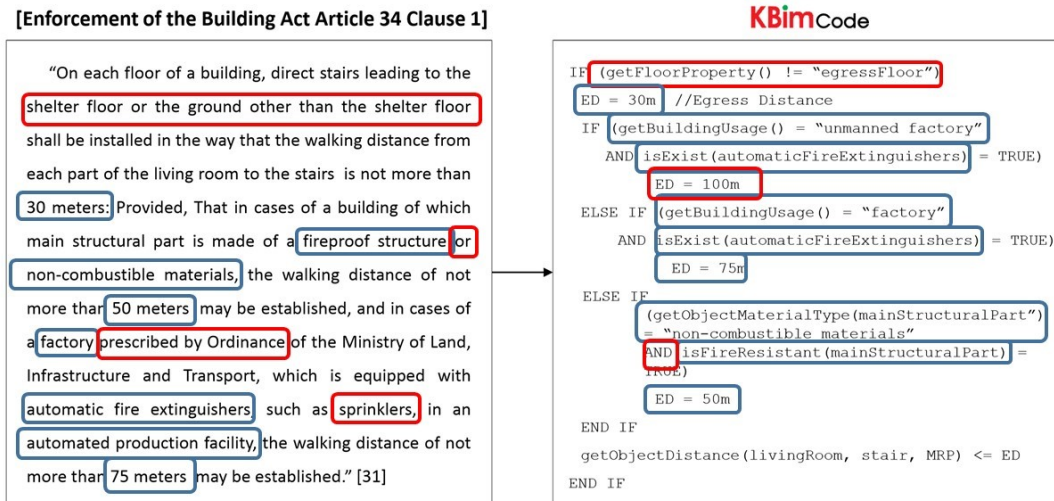
Figure 7: Example showing concepts that are matched in the RRR source and coding (blue) and those for which there is no match (red)

Errors and concerns that can be introduced include:

- Logical errors where the intent of the RRR content is mis-represented ("AND" instead of "OR")
- The omission of qualifications or exceptions that are perceived as not significant ("prescribed by Ordinance" and "sprinklers")
- The introduction of novel concepts which may simplify coding but are not directly related to the source. ("egressFloor")
- The mis-representation of key parameters ("50m", "75m" and "100m)" instead of ("30m", "50m" and "75m").
- Depending on the style of coding, large numbers of specific sub-functions or object classes may be required, whose efficacy may be hard to verify ("isFireresistant()" or "mainStructuralPart")

4c.2.1 Control, ownership, availability

Procedural languages and tools are almost universally available, and have forwards compatability, with the occasional exception of graphical user interfaces. Tools exist to ensure that a codebase can be used on multiple devices and operating systems.

- 10.2.2 suitability for multiple purposes
  - Procedual coding offers both interpretative coding and the opportunity for automated code generation, particulalry if the logical structure of the

objectives can be separated from the evaluation of individual metrics (see 4b IfcConstraint). Different languages may differ in their ability to create novel classes and object types.

- Computer-assisted and Automated code compliance checking. These two types of checking differ only in their source of information about the target model. Computer-assisted checking will depend primarily on an active user-interface to collect information, whereas automated code checking will depend primarily on the building model.

- Analysis and code comparisons are generally not possible using procedural code, as individual styles of coding may preclude textual comparison. The generated code can only be driven in one direction, so it is not possible to identify if two versions of the code come into conflict or create incentives or disincentives around a switch-over.

- Filtering and structured dialogues are not easily supported because the code has no direct relationship to the original RRR content.

- Controlling parametric objects is not possible, as the code necessarily goes from a fixed product model to an outcome, rather than configuring the product model to conform to the RRR content.

4c.4 Technical

- Expressivity and generality
    - Not all procedural languages can handle triple-value truth operators (true/unknown/false). This is probably essential for regulatory assessment, and when recommendatory content is considered, even more subtle representations of veractiy, such as probability and fuzzy logic, may be required.
    - Any implementation will need to use Procedural langauges that do not have an in-built ability to handle trade-offs or uncertainty. This suggest they are best suited to prescriptive regulations and may have difficulty with requirements and recommendations.
    - Their ability to handle easy and technical requirements is well established but, because of their delivering a result, rather than tracking an overall status, they are less able to handle 'hard' and management requirements .
    - Procedural languages have extensive connections to conventonal data resources, such as databases and spreadsheets. Most have additional libraries that can access STEP, XML and RDF/OWL resources, which may include dictionaries and other support files, such as locale and language modules.
    - It is widely reported that the Singapore ePlanCheck generated results within 20-30 minutes for a substantial commercial mixed development. Procedural langauges cannot introduce heuristics and optimisation beyond that which the compiler/executor supports. For example most

standard languages will truncate the execution of boolean operands if the 1st operator is decisive, leaving the second operator un-examined.
- Procedural code is rarely re-usable as it stands and rarely convertible into any other format.

4c.4.2 Acceptability and provenance

- Automatic procedural code generation may be able to include hyperlinks back to the source RRR content. The iteration over the object in the target building model may also generate pointers or retain identifiers that can be used to point to the objects concerned.
- A procedural language does not naturally keep a trace of the decisions and paths taken within the code, nor of any requirements for evidence encountered, so this is additional functionality that is needed to be coded alongside the core RRR content based code.

Example 1: Korean Building Act 34-1 procedural code automatically generated from RASE

Subroutine Korean-Building-Act-34-1()

Exception_1 = (not (test ("main structure", "fire proof")) or not (test ("main structure", "non-combustible"))) or test ("walking distance", "not more than", "50m")

Exception_2 = (not (test ("prescribed factory") or not (test ("automatic fire extinguishers") or test ("sprinklers"))) or test ("walking distance", "not more than","75m")

Requirement_1 = not (test ("stairs")) or (test ("leading to", "includes", "shelter floor") or (test ("leading to", "includes", "ground floor")) and not test ("leading to", "includes", "shelter floor")) and (Exception_1 or Exception_2 or test ("walking distance", "not more than","30m"))

Korean-Building-Act-34-1 = Requirement_1

End Subroutine

Annex: Contributing authors

Initials	Name	Affiliation or Company	Email
NN	Nicholas Nisbet	buildingSMART UKI and AEC3 and Leeds Beckett University	nn@aec3.com

4d SWRL

4d.1 Introduction

This section describes the use of SWRL [1] (Semantic Web Rule Language) for the purposes of regulatory compliance checking in the construction industry. SWRL is a rule language for the semantic web and is able to perform rule checking against data stored in ontologies.

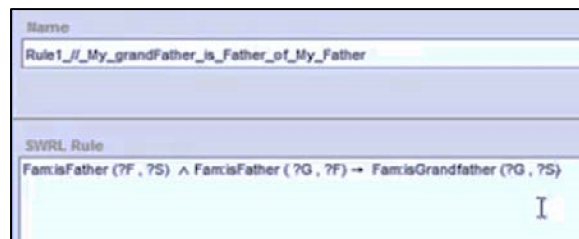An example of an SWRL rule is shown in Figure 1 below.

Figure 1- A rule written in SWRL. Figure reproduced from Slama et al. 2015.

4d.2 Strategic Assessment of Format

4d.2.1 Control, Ownership, Availability

The SWRL language is a freely available open standard managed by the W3C (World Wide Web Consortium). This means that any individual or organisation is free to develop either software utilising the language, or engines that execute rules using this language.

4d.2.2 Age, and Stability,

The SWRL standard was originally defined in 2004 and has since seen widespread adoption. It should be noted however, that SWRL itself is simply a standard for a rule language The following additional tools are required in order to make use of it:

- A rule engine that can execute this language
- An authoring environment to create rules.

These will be discussed in the following section.

4d.3 Commercial base

There are a variety of tool sets available for executing SWRL rules. These include:

Execution Engines: There is only one rule engine that is currently in common use for SWRL. This is Pellet [2], which is by far the most widely used currently. A second tool called Bossam [3] is also mentioned online – but no updates have been made to this software since 2007.

Authoring Tools: There are a variety of tools available to support the authoring of SWRL rules. These include:

- Protégé [4] – from Stanford University – is one of the most widely used ontology editors. It also supports the authoring of SWRL rules.
- Topbraid Composer [5] – is a commercial ontology editor that supports authoring of SWRL rules.

4d.4 Technical

The technical notes in this section adopt the approach utilised in the RegBIM project [6]. The RegBIM project took an approach utilising several components:

- Rule Specification is performed using RASE (as described previously).
- The rules specified in RASE are converted into an executable format.
- A rule engine executes the rules
- A dictionary is utilised in order to perform the translation between the terminology used in the regulations and that used within a BIM model
- A series of procedures are used to perform calculations that cannot be performed explicitly by the rule engine.

4d.4.1 Expressivity and generality

While SWRL was found to be able to represent all the rules in the test case the following limitations were identified:

- There is no way in which an execution order between SWRL rules can be specified. This creates limitations when certain rules require the results of another rule in order to be executed.
- It is not possible to express mathematical calculations. This means even the simplest calculations i.e. calculating an area must be embedded in procedures.
- SWRL used the open world paradigm of reasoning. This is beneficial when considering that a BIM model may often have missing data, however, it adds complexity when debugging and understanding rule execution, as many developers are used to operating using a closed world assumption.

4d.4.2 Acceptability and provenance

Page no.	Author
63	bSI Regulatory Room

SWRL is a low level rule language designed for executing rules over an ontology data structure. On its own it cannot be the requirements of linking to a building model or relating back to the regulatory text. However, a properly developed application (in this case using the RegBIM approach described previously) using SWRL can achieve these requirements as has been shown in [6].

4d.4.3 Examples of SWRL from Korean Regulations

Figures 1-3 show some example of SWRL rules illustrating the Korean regulations example. These rule work, by examining individuals within an ontology, and also storing results within the ontology, in terms of individuals of a class names *Pass*. Likewise, a class named *Fail* with equivalent individuals also exists. This means that SWRL rules can also be created that use the results of these rules, i.e. when a regulation is true if 1 or more sub-elements of the regulation are met. These is shown in Figures 4 and 5.

leadingTo(Stairs, Ground) ∧ walkingDistance(Stairs, 30) -> Pass(34-11)

Figure 8 SWRL Example 1

leadingTo(Stairs, ShelterFloor) ∧ mainStructuralPartMadeOf(Building, FireProofStructure) ∧ walkingDistance(Stairs, 50) -> Pass(34-12)

Figure 9 SWRL Example 2

leadingTo(Stairs, ShelterFloor) ∧ Usage(Building, Factory) ∧ walkingDistance(Stairs, 75) ∧ prescribedUse(Building, OrdinanceOfMinistryOfLand) ∧ hasSystem(Building, Sprinklers) -> Pass(34-13)

Figure 10 SWRL Example 3

Pass(34-11) ∧ Pass(34-12) ∧ Pass(34-13) -> Pass(34)

Figure 11 SWRL Example 4

Pass(34-11) -> Pass(34)
Pass(34-12) -> Pass(34)
Pass(34-13) -> Pass(34)

Figure 12 SWRL Example 5

4d.6 References

[1] W3C SWRL: A Semantic Web Rule Language Combining OWL and RuleML, [Website Accessed Aug 2016] https://www.w3.org/Submission/SWRL/#1

[2] Pellet OWL Reasoner [Website Accessed Aug 2016]

https://github.com/Complexible/pellet

[3] Bossam Rule Engine [Website Accessed Aug 2016]
http://projects.semwebcentral.org/projects/bossam/

[4] Protégé [Website Accessed Aug 2016] http://protege.stanford.edu/

[5] Topbraid Composer, Topquadrat, [Website Accessed Aug 2016]
http://www.topquadrant.com/

[6] Beach, Thomas H., et al. "A rule-based semantic approach for automated regulatory compliance in the construction sector." *Expert Systems with Applications* 42.12 (2015): 5219-5231.

Annex: Contributing authors

Initials	Name	Affiliation or Company	Email
TB	Thomas Beach	Cardiff University	beachth@cf.ac.uk

4e SPARQL and geoSPARQL

4e.1 Introduction

SPARQL and **SPIN** are implemented technologies for querying and checking RDF data. **SPARQL** was standardized by the RDF Data Access Working Group (DAWG) of the World Wide Web Consortium (W3C), and is one of the fundamental technologies in the Semantic Web and Linked Data world (Prud'hommeaux, Seaborne 2008).

http://www.slideshare.net/HolgerKnublauch/spin-in-five-slides

SPIN is a framework developed and maintained by TopQuadrant to utilize SPARQL as a rule language (Knublauch et al 2011). It is a W3C Member Submission and receives more and more attention within Semantic Web community in recent years.

4e.1.1 Names and synonyms

Page no.	Author
65	bSI Regulatory Room

SPARQL refers to SPARQL Protocol and RDF Query Language, which is an established standard recommended by W3C.

SPIN refers to SPARQL Inferencing Notation. It is developed by Topquadrant and is a W3C Member Submission.

W3C (World Wide Web Consortium) is an international community that develops open standards to ensure the long-term growth of the Web.

GeoSPARQL is an RDF vocabulary and a SPARQL extension for processing geospatial data. It is a standard in Open Geospatial Consortium (OGC).

RDF (Resource Description Framework) is a standard model for data interchange on the Web. It is the basis of the Semantic Web architecture.

4e.1.2 Description

SPARQL is a W3C standard for querying RDF triples. As one of the key technologies in the Semantic Web architecture, SPARQL is widely implemented by almost all RDF APIs and databases. SPARQL offers full full CRUD (create, read, update and delete) functionalities by different kinds of query forms.

SPARQL Inferencing Notation (SPIN) is a W3C Member Submission initiated by Topbraid (Knublauch et al 2011). Its architecture is shown in Fig. 1. The SPIN SPARQL syntax provides an RDF syntax for SPARQL, enabling SPARQL queries can be stored and maintained with RDF data. The SPIN Modeling Vocabulary provides a meta-vocabulary to organize and manipute SPARQL queries in order to facilitate rule-based reasoning and data constraint checking. SPIN has been used in many research prototypes in other domains e.g. geospatial and biology (Bue and Machì 2015, Callahan and Dumontier 2012, Fürber and Hepp 2010). One example of using SPIN in the building industry is described in (Zhang and Beetz 2015).

SPARQL is extendable for domain usage. An example is the GeoSPARQL standard, which is standardized by OGC to query geospatial data based on qualitative spatial reasoning and computations (Perry and Herring 2012). Many RDF databases has provided mechnisms to extend SPARQL functions.
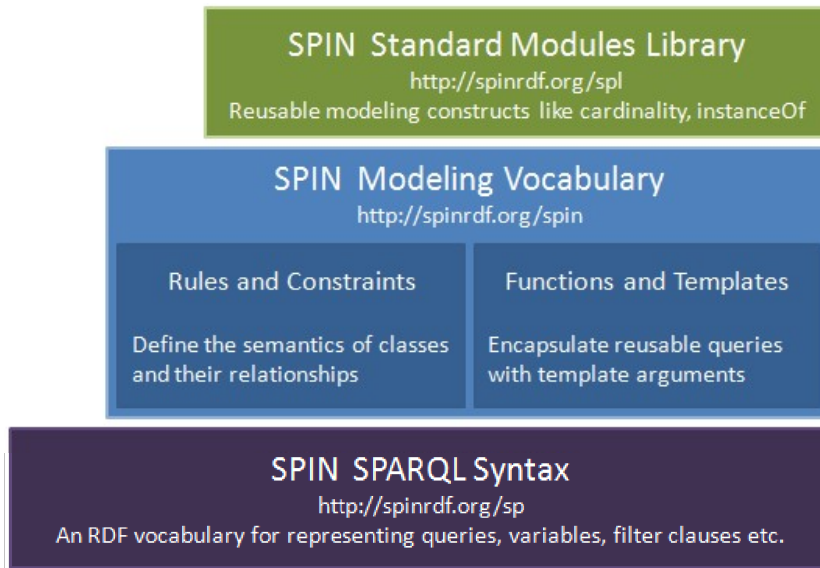
Figure 1. SPIN architecture (SPIN 2011)

4e.2 Strategic assessment of format

4e.2.1 Control, ownership, availability

4e.2.2 age and stability

The current status and development of SPARQL is reported in https://www.w3.org/standards/techs/sparql#w3c_all. SPARQL specification is maintained by W3C. On 15 January 2008, SPARQL 1.0 became an official W3C Recommendation, and SPARQL 1.1 in March, 2013.

SPIN has became a W3C Member Submission in 2011 and is maintained by TopQuadrant. Its specification evolves with SPARQL.

4e.3 Commercial base

There are many commercial and open source tools supporting editing and processing of SPARQL. A detailed implementation list is provided by W3C https://www.w3.org/wiki/SparqlImplementations. Here only lists implementations of SPIN.

4e.3.1 authoring/capture tools

SPIN can be edited in Topbraid Composer, which has a free version. Topbraid also provides an open source API based on Apache Jena.

4e.3.2 assessment engines, software and toolkits

SPIN rules can be processed by following toolkits:

- All versions of Topbraid (Free version, Standard version, Maetro version etc)
- Open source SPIN API, which is based on Apache Jena. Its reasoning engine is developed based on Jena ARQ query engine. It is compatible with all Jena components such as OWL and Jena rule reasoner, ARQ extension for SPARQL and Jena TDB database.
- Sesame, also known as RDF4J, which is a popular open source RDF database.
- Allegrograph, which is a commercial RDF database supports some SPIN features.

4e.4 Technical

4e.4.1 Expressivity and generality

The Expressive power of SPARQL is analyzed by Angles and Gutierrez, who concludes that it equals with non-recursive safe Datalog with negation and Relational Algebra. A recent analysis SPARQL allows for a query to consist of triple patterns, conjunctions, disjunctions, negations and optional patterns.

The target model of SPARQL and SPIN is RDF based models. In our domain, they can work with ifcOWL. They can also process all kinds of data represented in RDF triples. There are many exiting tools to convert conventional data models captured in e.g. relational data base, CSV etc. to RDF based data. These kinds of data can all be integrated with ifcOWL data and processed by SPARQL and SPIN.

Performance of SPARQL depends on its implementations. There are existing performace benckmark projects to test SPARQL queries in different RDF databases. Berlin SPARQL Benchmark (BSBM) and Lehigh University Benchmark (LUBM) are two examples of them.

Performance testing for SPIN and Jena TDB database has been reported in Pauwels et al 2016. This performace benchmark compares one implementation of SPIN and other rule languages.

External dictionaries can be converted to RDF vocabularies and work with SPARQL and SPIN. They can be remotely served as SPARQL endpoint.

4e.4.2 Acceptability and provenance

SPIN provides a RDF representation for SPARQL, which is compatible with Linked Data and URI-based Web environment. All rules can be identified by URIs and linked

with any RDF resources or web pages, so as original rule documents and ifcOWL based building models.

4e.5 Definitive references and contacts

Bue, A.L. and Machì, A., 2015, September. Open Data Integration Using SPARQL and SPIN: A Case Study for the Tourism Domain. In Congress of the Italian Association for Artificial Intelligence (pp. 316-326). Springer International Publishing.

Callahan, A. and Dumontier, M., 2012, May. Evaluating scientific hypotheses using the SPARQL inferencing notation. In Extended Semantic Web Conference (pp. 647-658). Springer Berlin Heidelberg.

Fürber, C. and Hepp, M., 2010, May. Using SPARQL and SPIN for data quality management on the semantic web. In International Conference on Business Information Systems (pp. 35-46). Springer Berlin Heidelberg.

Knublauch, H., Hendler, J.A. and Idehen, K., 2011. SPIN-overview and motivation. W3C Member Submission, 22.

Pauwels, P., de Farias, T.M., Zhang, C., Roxin, A., Beetz, J. and De Roo, J., 2016, June. Querying and reasoning over large scale building data sets: an outline of a performance benchmark. In Proceedings of the International Workshop on Semantic Big Data (p. 11). ACM.

Perry, M. and Herring, J., 2012. OGC GeoSPARQL-A geographic query language for RDF data. OGC Implementation Standard. Sept.

Prud'hommeaux E., Seaborne A., SPARQL query language for RDF - W3C Recommendation 15 January 2008, W3C Recommendation, available online: http://www.w3.org/TR/rdf-sparql-query/. Last accessed on 14 January 2015, 2008

SPIN. (2011). SPARQL Inferencing Notation. http://spinrdf.org/.

Zhang, C., Beetz, J., 2015. Model checking on the Semantic Web: IFC validation using modularized and distributed constraints. In Proceedings of 32nd CIB W78, Eindhoven, Netherlands.

Please see section 3 for the suggested report scope and structure

Annex: Contributing authors

Initials	Name	Affiliation or Company	Email
NN	Nicholas Nisbet	buildingSMART UKI and AEC3 and Leeds Beckett University	nn@aec3.com
CZ	Chi Zhang	Eindhoven University of Technology	c.zhang@tue.nl

4f Java DROOLS

4f.1 Introduction

This section describes the use of DRL [1] (DROOLS Rule Language) for the purposes of regulatory compliance checking in the construction industry. DRL is a rule language based on the java programming language designed specifically to work with the DROOLS [2] rule engine.

4f.2 Strategic Assessment of Format

4f.2.1 Control, Ownership, Availability

The DRL language was originally designed to be used with the DROOLS rule engine. However, the specification for the language is freely available enabling any developers to developing applications using the language. To simply use DRL, most users will opt to use DROOLs itself which is freely available open source software, made available under the ASL2.0 license. More details on the license can be found here: (http://www.apache.org/foundation/license-faq.html#WhatDoesItMEAN)

4f.2.2 Age, and Stability,

DROOLS is a mature and well supported rule engine. It has been around for many years and is regularly updated, meaning it is very stable and possesses a rich set of features. A commercially supported version of DROOLS is also available if required.

4f.3 Commercial base

Being that the DRL rule language is traditionally only used by the DROOLS rule engine, then this is the only viable product that can be utilised for executing of rules specified in this format. Currently, DROOLS comes in two forms, and open source (free) product and a supported commercial version known as JBOSS [3].

4f.4 Technical

The technical notes in this section adopt the approach utilised in the RegBIM project[6]. The RegBIM project took an approach utilising several components:

- Rule Specification is performed using RASE (as described previously).
- The rules specified in RASE are converted into an executable format.
- A rule engine executes the rules
- A dictionary is utilised in order to perform the translation between the terminology used in the regulations and that used within a BIM model
- A series of procedures are used to perform calculations that cannot be performed explicitly by the rule engine.

4f.4.1 Expressivity and generality

DRL is a highly expressive language allowing for the specification of a series of rules in the form of RULE <When> <Then>. The language of DRL is based on the java language.

Within this rule the <When> element of the rule is a Boolean expression, where the <Then> element is a command to execute if the <When> element is found to be true.

The flexibility of this approach is expanded by allowing <When> and <Then> elements of each rule to call external java functions and access java data structures.

Finally, the order of execution of rules are able to be controlled via a salience parameter

4f.4.2 Acceptability and provenance

While DRL is a low level rule language that, in itself, cannot meet all of requirements from within the construction industry. However, a well developing application utilising DRL and DROOLS can.

The RegBIM project acted as a proof of concept for this. The RegBIM software implementation using DRL and DROOLS was able to [6]:

- Fully execute rulesets from BREAM (2010), UK Building Regulations and Code for Sustainable Homes.
- Maintain linkage between the source regulations and the execution of DROOLS (as shown in Figure 1)
- Provide provenance as to the results of each individual rules execution. (as shown in Figure 1)

- Extract data from a BIM model
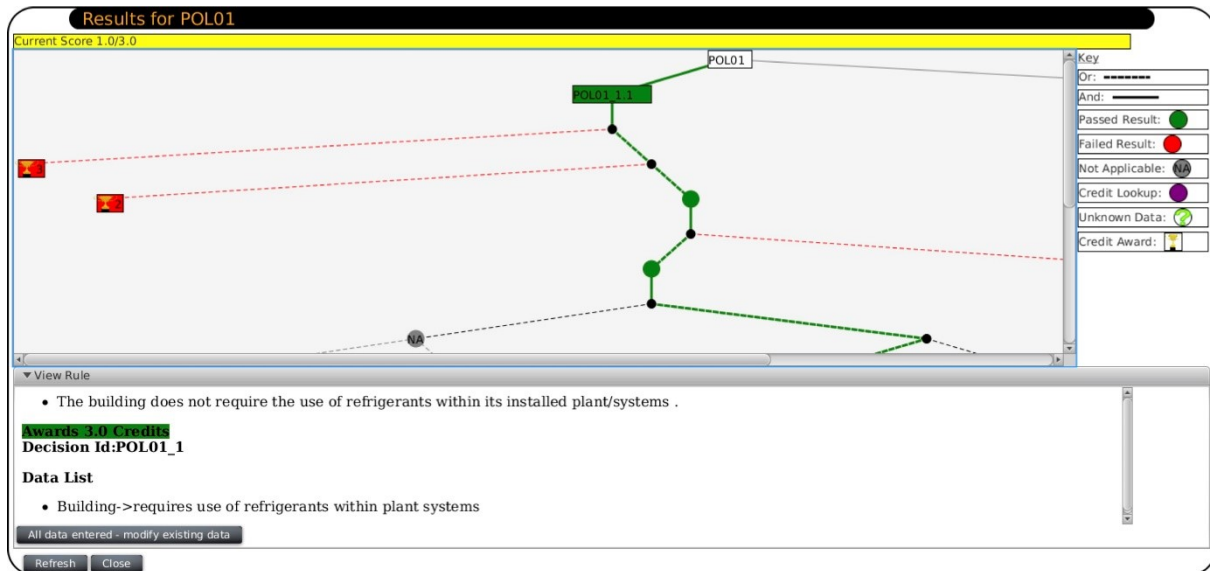- Execute stored procedures and perform mathematics when necessary.

Figure 13 Tracking Rule Execution and Maintaining linkage between rules and regulatory text

4f.4.3 Examples of DRL from Korean Regulations

Two examples of DRL rules generated automatically from RASE tags are shown in this section. Figure 2 shows a rule that checks if the walking distance from living space to the stairs is < 30 meters. For this example it is important to note that two rules are generated, one to test if the rule is passed, and one to test if the rule is failed.

```
rule "KOREAN-0-0-FAIL" salience 0 when
Applicability(id=="KOREAN-0",applicable==true)
not Result(id=="KOREAN-0-0")
exists Floor(walking_distance_from_living_room_to_stairs>30) from bim.fetch("Floor")
not Floor(walking_distance_from_living_room_to_stairs==null) from bim.fetch("Floor")
then results.fail("KOREAN-0-0"); end

rule "KOREAN-0-0" salience 0 when
Applicability(id=="KOREAN-0",applicable==true)
not Result(id=="KOREAN-0-0")
forall ( $temp: Floor() from bim.fetch("Floor")      Floor( this==$temp,walking_distance_from_living_room_to_stairs<=30) from bim.fetch("Floor"))
then results.pass("KOREAN-0-0"); end
```

Figure 14 Checking the Walking Distance

Figure 3 shows a more complete examine consisting of a pass and fail rule, along with a rule to determine where or not these rules are required for the type of building being checked.

Page no.	Author
73	bSI Regulatory Room

```
rule "KOREAN-0-1-0-OUTSCOPE" salience 75 when
exists Applicability(id=="KOREAN-0-1",applicable==true)
not Building(fireproof==true||made_of_fireproof_materials==true) from bim.fetch("Building")
not Building(fireproof==null || made_of_fireproof_materials==null) from bim.fetch("Building")
then results.na("KOREAN-0-1-0"); end

rule "KOREAN-0-1-0-FAIL" salience 0 when
Applicability(id=="KOREAN-0-1",applicable==true)
not Result(id=="KOREAN-0-1-0")
exists Floor(walking_distance_from_living_room_to_stairs>50) from bim.fetch("Floor")
not Floor(walking_distance_from_living_room_to_stairs==null) from bim.fetch("Floor")
then results.fail("KOREAN-0-1-0"); end

rule "KOREAN-0-1-0" salience 0 when
Applicability(id=="KOREAN-0-1",applicable==true)
not Result(id=="KOREAN-0-1-0")
forall ( $temp: Floor() from bim.fetch("Floor")        Floor( this==$temp,walking_distance_from_living_room_to_stairs<=50) from bim.fetch("Floor"))
then results.pass("KOREAN-0-1-0"); end
```

Figure 15 Rules with Scoping

References

[1] DRL Rule Language Reference [Website Accessed Aug 2016]

http://docs.jboss.org/drools/release/6.4.0.Final/drools-docs/html/ch08.html

[2] DROOLS [Website Accessed Aug 2016] http://www.drools.org/

[3] JBOSS Software Tools [Website Accessed Aug 2016]

http://developers.redhat.com/products/?referrer=jbd

[4] Beach, Thomas H., et al. "A rule-based semantic approach for automated regulatory compliance in the construction sector." *Expert Systems with Applications* 42.12 (2015): 5219-5231.

Annex: Contributing authors

Initials	Name	Affiliation or Company	Email
TB	Thomas Beach	Cardiff University	beachth@cf.ac.uk

4g LegalDocML and LegalRuleML

4g.1   INTRODUCTION

LegalDocML and **LegalRuleML** are two emerging open standards that have been under development since 2012 in the legal domain. They are intended to represent any kind of legal documents and can be extended to include non-legal documents.

There are four aspects to any document (including parliamentary, legislative and judiciary documents):

- **Content:** a set of words and punctuation that form sentences of the text
- **Presentation:** how the information looks, e.g. the colour of the text used in the document, the font used in the headings and other such formatting matters
- **Structure:** how the information is organised, e.g., the identification of some parts of text as headings, some parts as clauses, etc.
- **Semantics:** what the information represents or means

The literal content, structure and presentation aspects of a document are important to the "human reader" as they maintain the user familiarity with the document but they are not particularly relevant to a machine. The logical content of a document, however, is both human and machine-readable.

LegalDocML and **LegalRuleML** are intended to operate together to represent all aspects of a document as described above. The metadata contained in LegalDocML provides a means of maintaining the LegalRuleML counterpart to ensure it is up-to-date.

4g.1.1 Names and synonyms

Akoma Ntoso (Architecture for Knowledge-Oriented Management of African Normative Texts using Open Standards and Ontologies) is a United Nation's initiative developed in 2002. It is currently being standardised into **LegalDocML**.

RuleML is an open standard mark-up language developed to express rules in XML for use on the web, developed by RuleML Inc. in 2002. It is intended for deduction, rewriting, and further inferential-transformational tasks.

MetaLex is an open standard interchange format for legal and legislative resources in XML developed by the European Committee for Standardisation (CEN) in 2002.

LegalDocML is intended to represent the literal content and structure of a document for the entire life cycle of the document. The presentation aspect is handled by rendering a LegalDocML document using a stylesheet into HTML or other formats.

LegalRuleML is intended to represent the logical content and semantics of the document. It incorporates RuleML as the base markup language.

OASIS (Organisation for the Advancement of Structured Information Standards), an international consortium that drives the development, convergence and adoption of open standards for the global information society, set up two separate technical committees in 2012 to develop LegalDocML and LegalRuleML.

4g.1.2 Description

LegalDocML is a standardisation (work in progress) of **Akoma Ntoso** (Architecture for Knowledge-Oriented Management of African Normative Texts using Open Standards and Ontologies), which was initiated by UN/DESA (United Nations Department of Economic and Social Affairs) in 2004/2005. The term Akoma Ntoso also means "linked hearts" in the Akan language of West Africa (Vitali, 2007). **Akoma Ntoso** was partly inspired by **CEN Metalex** (developed at the University of Amsterdam in 2002) and has been designed to be compatible with it. The entire set of Dutch regulations have been encoded in **CEN Metalex**. The UK Statute Law Database has also used **CEN Metalex**.

LegalRuleML is an extension of **RuleML** being developed by OASIS. **RuleML** has been designed for the interchange of all kinds of Web rules in an XML format that is uniform across various rule languages and platforms. It has broad coverage and is defined as an extensible family of languages, whose modular system of schemas permit rule interchange with high precision.

4g.2 Strategic assessment of format

LegalDocML and LegalRuleML have been developed by OASIS and have been published extensively (see references) and started to be used in the following projects:

- LegalDocML and LegalRuleML are currently being applied to the Australian Building Code as a pilot project undertaken by Data61 of CSIRO.
- LegalDocML (Akoma Ntoso) was successfully applied to Japanese Legislation (Kawachi et al, 2015)

- LegalDocML has been used as the basis for USLM (United States Legislative Markup) schema for the US codified laws.
- LegalDocML is explicitly designed to be compliant with CEN Metalex, another de facto standard used in the legal domain in UK.

OASIS has defined 5 levels of compliance to LegalDocML, as follows:

1. Level 1: structure of the document
2. Level 2: structure and naming convention of URI/IRI (FRBR metadata)
3. Level 3: structure, naming convention, basic metadata (e.g. normative references)
4. Level 4: structure, naming convention, basic metadata, advanced metadata (e.g. events, modifications, qualifications of the document, etc)
5. Level 5: structure, naming convention, basic and advanced metadata, enriched semantic elements (e.g. references, location, quantity, term, person, etc)

4g.2.2 Age and stability

LegalDocML and **LegalRuleML** are emerging open standards and have been developed since 2012, so it is timely to be considered as a candidate rule interoperability standard for the AEC/FM domain.

They can be used to represent the following types of document:

1. **collectionDocs**: amendment List, official gazette, document collection
2. **legislativeDocs**: act and bill, which includes secondary legislations such as building regulations and building code
3. **debateDocs**: report and debate records
4. **amendmentDocs**
5. **judgementDocs**
6. **doc**: any informative general documents such as requirement specifications

There are several important international events supporting the development of these standards, as follows:

1. Akoma Ntoso Summer School
2. IANC (International Akoma Ntoso Conference)
3. European LEX Summer School

The latest schemas (Akoma Ntoso 3.0) were made available for public review in 2015.

4g.3 Commercial base

4g.3.1 Authoring/capture tools (singular, multiple)

There are a few open source and commercial tools available to author LegalDocML and LegalRuleML, as follows:

1. LegisPro by Xcential, a web-based authoring tool to mark up a text document into LegalDocML
2. LIME (Language Independent Markup Editor) Editor by University of Bologna, Italy
3. AT4AM Akoma Ntoso Editor and Parser
4. HTML5-based editor

4g.3.2 Assessment engines, software and toolkits for (singular, multiple)

There are a few tools and projects, as follows:

1. SPINdle reasoner
2. Regorous process engine

4g.4 Technical

4g.4.1 Expressivity and generality

- LegalDocML and LegalRuleML can represent any legal document as well as recommendatory document.
- Compatible with establish CEN Metalex standard
- LegalRuleML incorporates RuleML
- LegalRuleML can be mapped to RDF triples for Linked Data reuse
- LegalRuleML can be reasoned via Modal Defeasible Logic (MDL) with modal operators.
- LegalDocML uses the open standard FRBR (Functional Requirements for Bibliographic Records) model to capture document metadata, which uses a four level entity-relationship model of metadata for information objects.
- LegalDocML is able to capture the entire life-cycle of a document and provides a version control mechanism for its LegalRuleML counterpart.

The schema of LegalDocML is shown in Figure 2 below.
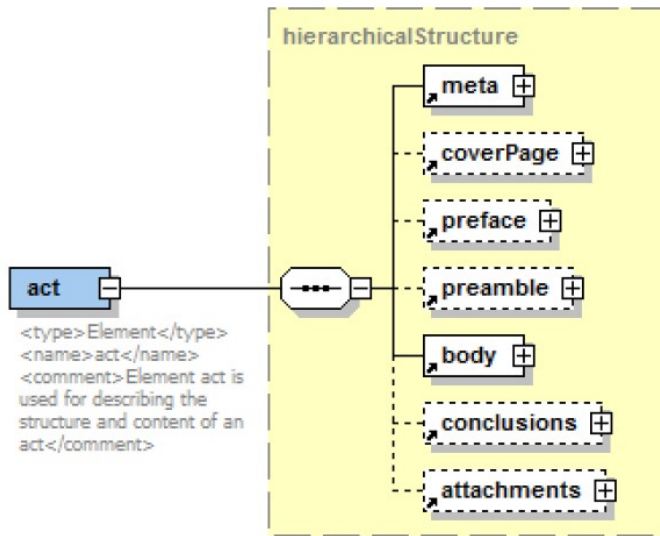
Figure 2: High-level schema of Akoma Ntoso

Example of LegalDocML

(From US Bill H.R.1190 – Protecting Seniors' Access to Medicare Act of 2015)

```xml
<?xml version="1.0" encoding="UTF-8"?>
<akomaNtoso
   xmlns="http://docs.oasis-open.org/legaldocml/ns/akn/3.0/WD17"
   xmlns:html="http://www.w3.org/1999/xhtml">
   <bill name="bill">
     <meta>
       <identification source="#source">
         <FRBRWork>
           <FRBRthis value="/akn/us/bill/2015-10-26/!main"/>
           <FRBRuri value="/akn/us/bill/2015-10-26"/>
           <FRBRdate date="2015-10-26" name=""/>
           <FRBRauthor href="#" as="#"/>
           <FRBRcountry value="us"/>
         </FRBRWork>
         <FRBRExpression>
           <FRBRthis value="/akn/us/bill/2015-10-26/eng@/!main"/>
           <FRBRuri value="/akn/us/bill/2015-10-26/eng@"/>
           <FRBRdate date="2015-10-26" name=""/>
           <FRBRauthor href="#limeEditor" as="#limeEditor"/>
           <FRBRlanguage language="eng"/>
         </FRBRExpression>
         <FRBRManifestation>
           <FRBRthis value="/akn/us/bill/2015-10-26/eng@/main.xml/!main"/>
           <FRBRuri value="/akn/us/bill/2015-10-26/eng@.xml"/>
           <FRBRdate date="2016-10-19" name=""/>
           <FRBRauthor href="#limeEditor" as="#limeEditor"/>
         </FRBRManifestation>
       </identification>
       <publication date="2015-10-26" name="" showAs="" number=""/>
       <references source="#source">
         <TLCPerson eId="limeEditor" href="/lime.cirsfid.unibo.it" showAs="LIME editor"/>
         <TLCPerson eId="source" href="/ontology/person/somebody" showAs="Somebody"/>
       </references>
     </meta>
     <preface>
```

Page no.	Author
80	bSI Regulatory Room

<p>H.R.1190 -- Protecting Seniors' Access to Medicare Act of 2015 (Referred in Senate - RFS)

<eol/>HR 1190 RFS

<eol/>114th CONGRESS

<eol/>1st Session

<eol/>H. R. 1190

<eol/>IN THE SENATE OF THE UNITED STATES

<eol/>

<docDate date="2015-06-24">June 24, 2015</docDate>

<eol/>Received; read twice and referred to the Committee on Finance

<eol/>

</p>

</preface>

<preamble>

<p>AN ACT

<eol/>To repeal the provisions of the Patient Protection and Affordable Care Act providing for the Independent Payment Advisory Board.

<eol/>

</p>

<formula name="formulaIniziale" eId="formula_1">

<p>Be it enacted by the Senate and House of Representatives of the United States of America in Congress assembled,</p>

</formula>

</preamble>

<body>

<section eId="sec_1">

<num>SECTION 1.</num>

<content eId="sec_1__content">

<p>SHORT TITLE.This Act may be cited as the `Protecting Seniors' Access to Medicare Act of 2015'.</p>

</content>

</section>

<section eId="sec_2">

<num>SEC. 2. </num>

<content eId="sec_2__content">

<p>REPEAL OF THE INDEPENDENT PAYMENT ADVISORY BOARD. Effective as of the enactment of the Patient Protection and Affordable Care Act (Public Law 111-148), sections 3403 and 10320 of such Act (including the amendments made by such sections) are repealed, and any

provision of law amended by such sections is hereby restored as if such sections had not been enacted into law.</p>

```
        </content>
    </section>
    <section eId="sec_3">
        <num>SEC. 3.</num>
        <content eId="sec_3__content">
            <p>RESCINDING FUNDING AMOUNTS FOR PREVENTION AND PUBLIC HEALTH
FUND.</p>
        </content>
    </section>
    <section eId="sec_4002">
        <num>Section 4002</num>
        <heading eId="sec_4002__heading">(b) of the Patient Protection and Affordable Care Act
(42 U.S.C. 300u-11(b)) is amended--</heading>
        <paragraph eId="sec_4002__para_1">
            <num>(1)</num>
            <content eId="sec_4002__para_1__content">
                <p>in paragraph (2), by striking `2017' and inserting `2016';</p>
            </content>
        </paragraph>
        <paragraph eId="sec_4002__para_2">
            <num>(2)</num>
            <content eId="sec_4002__para_2__content">
                <p>in paragraph (5)--
</p>
                <blockList eId="sec_4002__para_2__content__list_1">
                    <item eId="sec_4002__para_2__content__list_1__item_a">
                        <num>(A)</num>
                        <p>by striking `2022' and inserting `2026'; and</p>
                    </item>
                    <item eId="sec_4002__para_2__content__list_1__item_b">
                        <num>(B)</num>
                        <p>by redesignating such paragraph as paragraph (7); and</p>
                    </item>
                </blockList>
            </content>
        </paragraph>
```

```
<paragraph eId="sec_4002__para_3">
    <num>(3)</num>
    <content eId="sec_4002__para_3__content">
        <p>by striking paragraphs (3) and (4) and inserting the following :</p>
    </content>
</paragraph>
<paragraph eId="sec_4002__para_3">
    <num>(3)</num>
    <content eId="sec_4002__para_3__content">
        <p>` for fiscal year 2017, $390,000,000 ;</p>
    </content>
</paragraph>
<paragraph eId="sec_4002__para_4">
    <num>(4)</num>
    <content eId="sec_4002__para_4__content">
        <p>` for each of fiscal years 2018 and 2019, $487,000,000 ;</p>
    </content>
</paragraph>
<paragraph eId="sec_4002__para_5">
    <num>(5)</num>
    <content eId="sec_4002__para_5__content">
        <p>` for each of fiscal years 2020 and 2021, $585,000,000 ;</p>
    </content>
</paragraph>
<paragraph eId="sec_4002__para_6">
    <num>(6)</num>
    <content eId="sec_4002__para_6__content">
        <p>` for each of fiscal years 2022 through 2025, $780,000,000; and'. Passed the
House of Representatives June 23, 2015. Attest: KAREN L. HAAS, Clerk.</p>
    </content>
</paragraph>
</section>
</body>
</bill>
</akomaNtoso>
```

Example of LegalRuleML

Page no.	Author
83	bSI Regulatory Room

Example text of Article 34 Clause 1 of the Korean Building Regulation:

"On each floor of a building, direct stairs leading to the shelter floor or the ground (including slope ways; hereinafter the same shall apply) other than the shelter floor (referring to a floor having a doorway leading directly to the ground and the shelter safety zone of a skyscraper under paragraphs (3) and (4); hereinafter the same shall apply) shall be installed in the way that the walking distance from each part of the living room to the stairs (referring to the stair nearest to the living room) is not more than 30 meters: Provided, that in cases of a building of which main structural part (excluding a performance hall, assembly hall, auditorium and exhibition hall which are installed on underground floors and which have a total floor area of not less than 300 square meters) is made of a fireproof structure or non-combustible materials, the walking distance of not more than 50 meters (in cases of multi-unit dwellings higher than 16 storeys, not more than 40 meters) is permitted, and in cases of a factory prescribed by Ordinance of the Ministry of Land, Infrastructure and Transport, which is equipped with automatic fire suppression systems such as sprinklers, in an automated production facility, the walking distance of not more than 75 meters (in cases of unmanned factories, 100 meters) is permitted".

Formalism of the above Article 34 Clause 1 of Korean Building Regulation:

building has floors, ~ground floor => [O] floor has emergency stairs

ground floor ~> [P] ~floor has emergency stairs

[O] floor has emergency stairs => [O] stairs are within distance living space

building is factory, protected fire suppression system, fireproof structure, max distance stairs less 40m => stairs are within distance living space

building is factory, automated production, protected fire suppression system, max distance stars less 75m => stairs are within distance living space

unmanned factory, automated production, protected fire suppression system, max distance stairs less 100m => stairs are within distance living space

building is multi-dwelling, floor number greater 16, fireproof structure, max distance stairs less 40m => stairs are within distance living space

max distance stairs less 30m => stairs are within distance living space

LegalRuleML fragment from representation of the above Article 34 Clause 1 of the Korean Building Regulation:

```xml
<lrml:PrescriptiveStatement key="r1">
    <ruleml:Rule strength="defeasible">
        <ruleml:if>
            <ruleml:And>
                <ruleml:Atom>
                    <ruleml:Rel>building has floors</ruleml:Rel>
                </ruleml:Atom>
                <ruleml:Negation>
                    <ruleml:Rel>ground floor</ruleml:Rel>
                </ruleml:Negation>
            </ruleml:And>
        </ruleml:if>
        <ruleml:then>
            <lrml:Obligation>
                <ruleml:Atom>
                    <ruleml:Rel>floor has emergency stairs</ruleml:Rel>
                </ruleml:Atom>
            </lrml:Obligation>
        </ruleml:then>
    </ruleml:Rule>
</lrml:PrescriptiveStatement>
```

4g.4.2 Acceptability and provenance

- Each rule in LegalRuleML has an associated key that links it to the source in LegalDocML
- These emerging standards have gained a lot of publicity the last few years from both research and industry (legal and business domain).
- The Commonwealth government of Australia has recently commissioned a pilot project for Data61 of CSIRO to model the Australian Building Code into LegalDocML and LegalRuleML.
- As mentioned in the introduction, the entire set of Dutch Regulations encoded in MetaLex is compatible with LegalDocML

4g.5 Definitive references and contacts

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

Akoma Ntoso 3.0 schema is available for access at the following link: http://sinatra.cirsfid.unibo.it/XSDocViewer/

OASIS. (2016). OASIS LegalDocumentML (LegalDocML) TC. *OASIS*. Retrieved April 10, 2016, from https://www.oasis-open.org/committees/tc_home.php?wg_abbrev=legaldocml

Athan, T., Governatori, G., Palmirani, M., Paschke, A., & Wyner, A. (2015). LegalRuleML: Design Principles and Foundations. In *The 11th Reasoning Web Summer School*. pp. 1–37. Berlin, Germany.

TRASYS SA. (2011). *LEOS - Legislative Editing Open Software*.

McGibbney, L. J., & Kumar, B. (2013). A comparative study to determine a suitable representational data model for UK building regulations. *Journal of Information Technology in Construction*, *18*(February), pp. 20–38. Retrieved from http://itcon.org/cgi-bin/works/show&_id=2011elearning/Show?2013_2

Athan, T., Governatori, G., Palmirani, M., Paschke, A., & Wyner, A. (2015). LegalRuleML: Design Principles and Foundations. In *The 11th Reasoning Web Summer School*. pp. 1–37. Berlin, Germany.

Palmirani, M., Governatori, G., & Rotolo, A. (2011). LegalRuleML: XML-based rules and norms. *Rule-Based Modeling and Computing on the Semantic Web*, pp. 298–312.

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

Athan, T., Governatori, G., Palmirani, M., Paschke, A., & Wyner, A. (2014). Legal Interpretations in LegalRuleML. In *Semantic Web for the Law Workshop*. pp. 11. Krakow, Poland.

Governatori, G., & Hashmi, M. (2015). No Time for Compliance. In *Proceedings of the 2015 IEEE 19th International Enterprise Distributed Object Computing Conference*. pp. 9–18. Adelaide, Australia: IEEE. doi:10.1109/EDOC.2015.12

Lam, H.-P., Hashmi, M., & Scofield, B. (2016). Enabling Reasoning with LegalRuleML. In N. Bassiliades, G. Gottlob, F. Sadri, A. Paschke, & D. Roman (Eds.), *10th International Web Rule Symposium (RuleML 2016)*. pp. 241–257. New York, USA: Springer International Publishing. doi:10.1007/978-3-319-42019-6_16

Palmirani, M., Cervone, L., Bujor, O., & Chiappetta, M. (n.d.). RAWE: A Web Editor for Rule Mark-up in LegalRuleML.

Athan, T., Boley, H., Governatori, G., Palmirani, M., Paschke, A., & Wyner, A. (2013). LegalRuleML: from Metamodel to Use Cases - A Tutorial. In *The 7th International Web Rule Symposium (RuleML) 2013*. Seattle, WA, USA: Springer.

Ceci, M. (2012). Combining Ontology and Rules to Model Judicial Interpretation. In *Proceedings of the RuleML*. Vol. 874, pp. 25–32. Montpellier, France.

Wyner, A., & Peters, W. (2011). On rule extraction from regulations. In *Proceedings of JURIX*. pp. 1–10. Retrieved from http://wyner.info/research/Papers/WynerPetersJURIX2011.pdf

OASIS. (2015). LegalRuleML Technical Committee - Charter. Retrieved July 31, 2016, from http://www.oasis-open.org/committees/legalruleml/charter.php

Symposium, I., & Hutchison, D. (2013). *Theory, Practice, and Applications of Rules on the Web*. (L. Morgenstern, P. Stefaneas, F. Levy, A. Wyner, & A. Paschke, Eds.). Seattle, WA, USA: Springer.

Hashmi, M., Governatori, G., & Wynn, M. T. (2013). Normative Requirements for Business Processes Compliance. *International Conference on Business Process Management*, (August), pp. 26–30.

Taddeo, M., Trombetta, A., Montesi, D., & Pierantozzi, S. (2013). Querying data across different legal domains. In *Proceedings of IDEAS '13*. Barcelona, Spain. doi:10.1145/2513591.2513642

McGibbney, L. J., & Kumar, B. (2013). A comparative study to determine a suitable representational data model for UK building regulations. *Journal of Information Technology in Construction*, *18*(February), pp. 20–38. Retrieved from http://itcon.org/cgi-bin/works/show&_id=2011elearning/Show?2013_2

Governatori, G., & Governatori, G. (2015). The Regorous Approach to Process Compliance e Regorous Approach to Process Compliance, (October).

Distinto, I. (2012). Checking Compliance in European Tender Documents through Ontologies and Rules. In *Proceedings of the RuleML*. pp. 33–41. Montpellier, France.

Wyner, A., & Governatori, G. (2013). A Study on Translating Regulatory Rules from Natural Language to Defeasible Logic. In *Proceedings of the RuleML 2013*. Seattle, WA, USA. Retrieved from http://wyner.info/LanguageLogicLawSoftware/index.php/category/legal-knowledge-engineering/

LegalRuleML representation of the above Article 34 Clause 1 of the Korean Building Regulation:

```
<lrml:PrescriptiveStatement key="r1">
  <ruleml:Rule strength="defeasible">
    <ruleml:if>
      <ruleml:And>
        <ruleml:Atom>
          <ruleml:Rel>building has floors</ruleml:Rel>
        </ruleml:Atom>
        <ruleml:Negation>
          <ruleml:Rel>ground floor</ruleml:Rel>
        </ruleml:Negation>
      </ruleml:And>
    </ruleml:if>
    <ruleml:then>
      <lrml:Obligation>
        <ruleml:Atom>
          <ruleml:Rel>
            floor has emergency stairs
          </ruleml:Rel>
        </ruleml:Atom>
      </lrml:Obligation>
    </ruleml:then>
  </ruleml:Rule>
```

```
</lrml:PrescriptiveStatement>

<lrml:PrescriptiveStatement key="r2">
  <ruleml:Rule strength="defeater">
    <ruleml:if>
      <ruleml:And>
        <ruleml:Atom>
          <ruleml:Rel>ground floor</ruleml:Rel>
        </ruleml:Atom>
      </ruleml:And>
    </ruleml:if>
    <ruleml:then>
      <lrml:Permitted>
        <ruleml:Negation>
          <ruleml:Atom>
            <ruleml:Rel>
              floor has emergency stairs
            </ruleml:Rel>
          </ruleml:Atom>
        </ruleml:Negation>
      </lrml:Permitted>
    </ruleml:then>
  </ruleml:Rule>
</lrml:PrescriptiveStatement>

<lrml:PrescriptiveStatement key="r3">
  <ruleml:Rule strength="defeasible">
    <ruleml:if>
      <ruleml:And>
        <lrml:Obligation>
          <ruleml:Atom>
            <ruleml:Rel>
              floor has emergency stairs
            </ruleml:Rel>
          </ruleml:Atom>
        </lrml:Obligation>
      </ruleml:And>
    </ruleml:if>
```

```
<ruleml:then>
  <lrml:Obligation>
    <ruleml:Atom>
      <ruleml:Rel>
        stairs are within distance living space
      </ruleml:Rel>
    </ruleml:Atom>
  </lrml:Obligation>
</ruleml:then>
  </ruleml:Rule>
</lrml:PrescriptiveStatement>

<lrml:ConstitutiveStatement key="r4">
  <ruleml:Rule strength="defeasible">
    <ruleml:if>
      <ruleml:And>
        <ruleml:Atom>
          <ruleml:Rel></ruleml:Rel>
        </ruleml:Atom>
        <ruleml:Atom>
          <ruleml:Rel></ruleml:Rel>
        </ruleml:Atom>
        <ruleml:Atom>
          <ruleml:Rel></ruleml:Rel>
        </ruleml:Atom>
        <ruleml:Atom>
          <ruleml:Rel>less than</ruleml:Rel>
          <ruleml:Var>distance living room</ruleml:Var>
          <ruleml:Data>40meters</ruleml:Data>
        </ruleml:Atom>
      </ruleml:And>
    </ruleml:if>
    <ruleml:then>
      <ruleml:Atom>
        <ruleml:Rel>stairs are within distance living space</ruleml:Rel>
      </ruleml:Atom>
    </ruleml:then>
  </ruleml:Rule>
```

</lrml:ConstitutiveStatement>

<lrml:ConstitutiveStatement key="r4">
  <ruleml:Rule strength="defeasible">
    <ruleml:if>
      <ruleml:And>
        <ruleml:Atom>
          <ruleml:Rel>building is factory</ruleml:Rel>
        </ruleml:Atom>
        <ruleml:Atom>
          <ruleml:Rel>protected fire suppression system</ruleml:Rel>
        </ruleml:Atom>
        <ruleml:Atom>
          <ruleml:Rel>fireproof structure</ruleml:Rel>
        </ruleml:Atom>
        <ruleml:Atom>
          <ruleml:Rel>less than</ruleml:Rel>
          <ruleml:Var>distance living space</ruleml:Var>
          <ruleml:Data>40meters</ruleml:Data>
        </ruleml:Atom>
      </ruleml:And>
    </ruleml:if>
    <ruleml:then>
      <ruleml:Atom>
        <ruleml:Rel>stairs are within distance living space</ruleml:Rel>
      </ruleml:Atom>
    </ruleml:then>
  </ruleml:Rule>
</lrml:ConstitutiveStatement>

<lrml:ConstituiveStatement key="r5">
  <ruleml:Rule strength="defeasible">
    <ruleml:if>
      <ruleml:And>
        <ruleml:Atom>
          <ruleml:Rel>building is factory</ruleml:Rel>
        </ruleml:Atom>
        <ruleml:Atom>

```xml
      <ruleml:Rel>automated production</ruleml:Rel>
    </ruleml:Atom>
    <ruleml:Atom>
      <ruleml:Rel>protected fire suppression system</ruleml:Rel>
    </ruleml:Atom>
    <ruleml:Atom>
      <ruleml:Rel>less than</ruleml:Rel>
      <ruleml:Var>distance living space</ruleml:Var>
      <ruleml:Data>75meters</ruleml:Data>
    </ruleml:Atom>
    </ruleml:And>
  </ruleml:if>
  <ruleml:then>
    <ruleml:Atom>
      <ruleml:Rel>stairs are within distance living space</ruleml:Rel>
    </ruleml:Atom>
  </ruleml:then>
  </ruleml:Rule>
</lrml:ConstitutiveStatement>

<lrml:ConstitutiveStatement key="r6">
  <ruleml:Rule strength="defeasible">
    <ruleml:if>
      <ruleml:And>
        <ruleml:Atom>
          <ruleml:Rel>building is factory</ruleml:Rel>
        </ruleml:Atom>
        <ruleml:Atom>
          <ruleml:Rel>unmanned factory</ruleml:Rel>
        </ruleml:Atom>
        <ruleml:Atom>
          <ruleml:Rel>automated production</ruleml:Rel>
        </ruleml:Atom>
        <ruleml:Atom>
          <ruleml:Rel>protected fire suppression system</ruleml:Rel>
        </ruleml:Atom>
        <ruleml:Atom>
          <ruleml:Rel>less than</ruleml:Rel>
```

```
      <ruleml:Var>distance living space</ruleml:Var>
      <ruleml:Data>100meters</ruleml:Data>
    </ruleml:Atom>
   </ruleml:And>
  </ruleml:if>
  <ruleml:then>
   <ruleml:Atom>
    <ruleml:Rel>stairs are within distance living space</ruleml:Rel>
   </ruleml:Atom>
  </ruleml:then>
 </ruleml:Rule>
</lrml:ConstitutiveStatement>


<lrml:ConstitutiveStatement key="r7">
 <ruleml:Rule strength="defeasible">
  <ruleml:if>
   <ruleml:And>
    <ruleml:Atom>
     <ruleml:Rel>building is multi-dwelling</ruleml:Rel>
    </ruleml:Atom>
    <ruleml:Atom>
     <ruleml:Rel>greater than</ruleml:Rel>
     <ruleml:Var>number of floors</ruleml:Var>
     <ruleml:Con>16</ruleml:Con>
    </ruleml:Atom>
    <ruleml:Atom>
     <ruleml:Rel>fireproof structure</ruleml:Rel>
    </ruleml:Atom>
    <ruleml:Atom>
     <ruleml:Rel>less than</ruleml:Rel>
     <ruleml:Var>distance living space</ruleml:Var>
     <ruleml:Data>40meters</ruleml:Data>
    </ruleml:Atom>
   </ruleml:And>
  </ruleml:if>
  <ruleml:then>
   <ruleml:Atom>
    <ruleml:Rel>stairs are within distance living space</ruleml:Rel>
```

```
      </ruleml:Atom>
    </ruleml:then>
  </ruleml:Rule>
</lrml:ConstitutiveStatement>


<lrml:ConstituiveStatement key="r8">
  <ruleml:Rule strength="defeasible">
    <ruleml:if>
      <ruleml:And>
        <ruleml:Rel>less than</ruleml:Rel>
        <ruleml:Var>distance living space</ruleml:Var>
        <ruleml:Data>30meters</ruleml:Data>
      </ruleml:Atom>
    </ruleml:And>
    </ruleml:if>
    <ruleml:then>
      <ruleml:Atom>
        <ruleml:Rel>stairs are within distance living space</ruleml:Rel>
      </ruleml:Atom>
    </ruleml:then>
  </ruleml:Rule>
</lrml:ConstitutiveStatement>
```

Annex: Contributing authors

Initials	Name	Affiliation or Company	Email
JD	Johannes Dimyadi	University of Auckland	j.dimyadi@auckland.ac.nz
GG	Guido Governatori	Data61 CSIRO	guido.governatori@data61.csiro.au

4h BIMRL (BIM Rule Language)

Automated rule checking is a complex and challenging subject that involves multiple parts (Figure 16). BIMRL is a research work done by the author as part of his PhD dissertation aiming to look at the rule checking problem from the comprehensive vantage point [1]. It is believed that to really overcome such complex problem, all parts must be addressed at the same time that includes: an efficient query system, integrated geometry and spatial operators, standardized rule definition language, and a suitable computable forms for the rules. BIMRL tries to address all of them (see Figure 16). A paper published by the author highlights the challenges and the broad categorizations of an automated rule checking system [4].
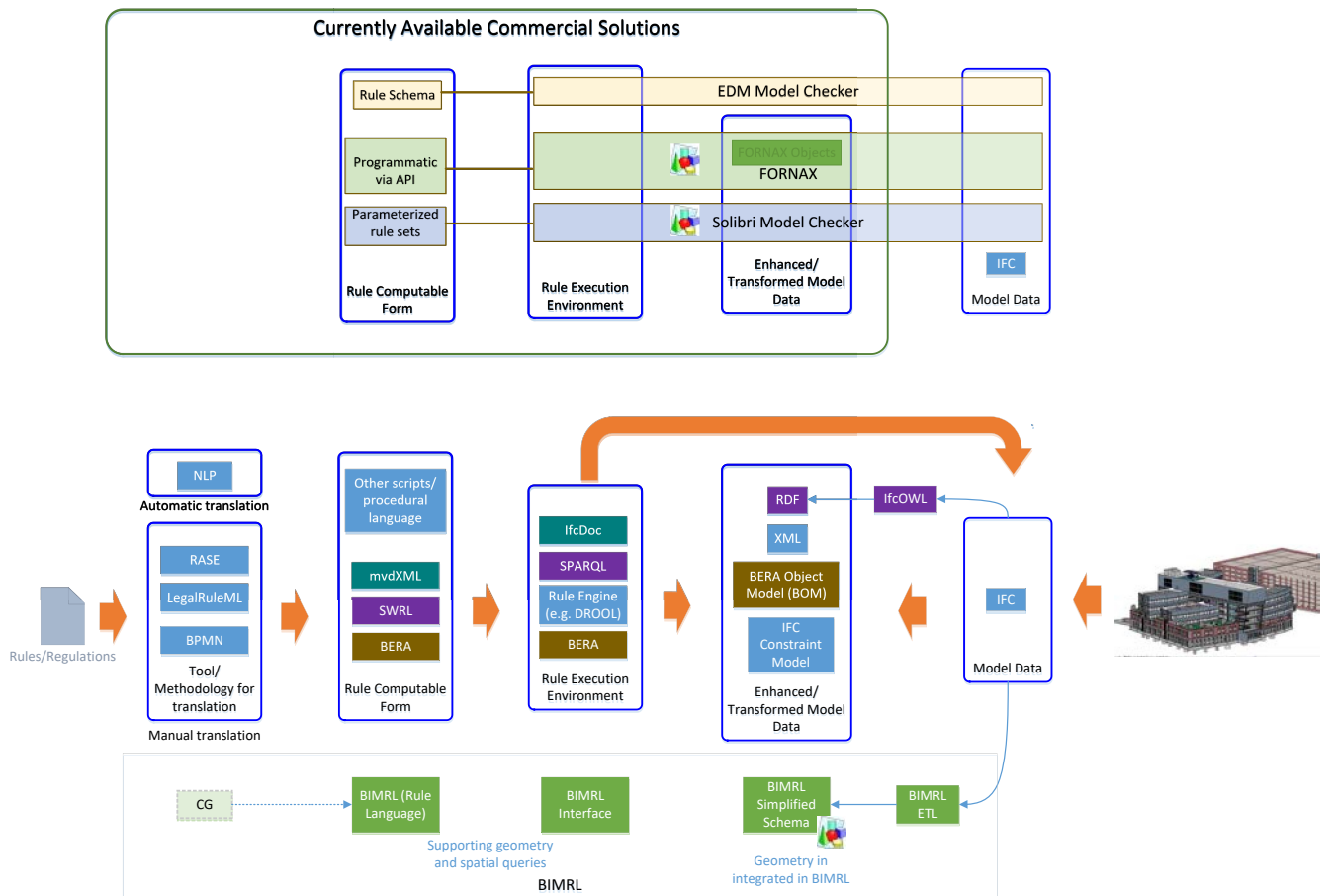
Figure 16 - Parts in the Automated Rule Checking Systems

4h.1 Introduction

4h.1.1 Names and synonyms

BIMRL is a short form for BIM Rule Language. It was initially intended to provide a standardized language to represent complex rules. It has then expanded to include the other parts of the automated rule checking system that provide an efficient query system to BIM data (BIMRL Simplified Schema), rule execution environment (BIMRL Interface), standardized rule definition language (BIMRL), and an integrated support for the geometry and spatial operations.

4h.1.2 Description

BIMRL addresses most of the parts in the automated rule checking domain (figure 4-1). It does not provide an automated tool to convert an existing rules into a computable form, but it proposes the process to capture the requirements in a more precise and detailed approach using CG as a documentation tool [2, 3].

BIMRL consists of the folowing components:

- A simplified schema, currently an RDBMS based that simplifies the IFC structure into a star-like schema used widely in the data warehouse domain. The main intention to use such as simplified schema is to provide flexible and optimized search/queries to the building models.
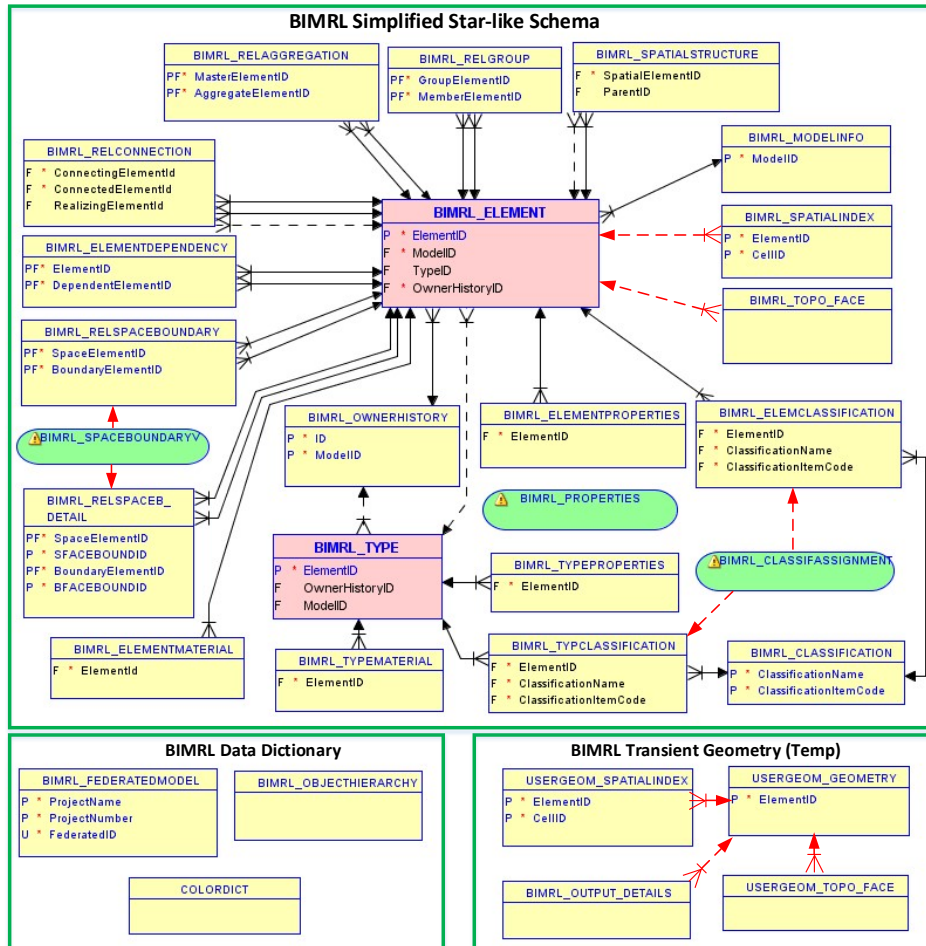
- An ETL (Extract, Transform, Load) module that reads in IFC file and tranform the data into the BIMRL simplified schema lossless.
- BIM Rule Language (BIMRL) definition. It is an SQL like language definition that defines 3 sections:
-
    o CHECK section that defines filter conditions to select objects of interests. The section supports multiple query sets for unrelated queries that need to be joined later on

    Example:

CHECK

{Ifcspace s, ifcdoor d

      WHERE property(s, HabitableSpace) ='TRUE'

          and classificationof (d).classificationitemcode='2.6.2'

          and classificationof (d).classificationname='BCIS'

          and s.container=d.container

      COLLECT s.elementid spaceid, d.elementid doorid, s.name spacenumber;

} AS SET1;

{IfcSpace SC

      Where CONTAINS (SC, USEGEOMETRY).ElementType='IFCFLOWTERMINAL'

          and CONTAINS (SC,USEGEOMETRY).Name like 'Sprinkler%'

      COLLECT SC.ElementId SPACEID, SC.Name SPACENUMBER,

          SC.LongName SPACELNAME, COUNT (unique CO.ElementID) SPRINKLERNO

      GROUP BY SC.ELementId, SC.Name, SC.LongName

} AS SET2;

- o EVALUATE section. It defines the evaluation function that implements the checking logic that will operate on the query sets obtained in CHECK section. EVALUATE section allows the following functionalities:
    - ▪ Evaluation function chaining, i.e. result of a function execution to flow into the subsequent function within the same section.
    - ▪ Joining of the query sets from CHECK section
    - ▪ Creation of transient geometry for use in the evaluation function. Supported geometry types currently are: line, face, and Brep solid
    - ▪ Designed for exensibility allowing a plug-in mechanism to extend the provided standard evaluation functions. It is very important and useful feature to deal with variety of complex and specialized rules exist in the domain.
- o Example:
    - ▪ A single evaluation function with Join

    EVALUATE computePathAndDistance (spaceid, doorid)

        output?traveldistance

        From SET1 LEFT OUTER JOIN SET2

        USING (SPACEID, SPACENUMBER);

    - ▪ A single evaluation function with transient geometry construction

    EVALUATE

      COMPUTEINTERSECTION (EB, SET2.MEPOBJEID) OUTPUT? clash

        FROM SET1

      CONSTRUCT EB (EXTRUSION (DefFace (FACEGEOM), +ZAXIS,

STELEVATIONHT-ELEMHEIGHT));

- A Chained evaluation functions

EVALUATE

```
{NOTHING () Output? Occupancy
   From SET1 FULL OUTER JOIN SET2 USING (SPACEID, SPACENAME,
     SPACELNAME)
};
{NOTHING () Output? SprinklProt
     From (select * from Occupancy where occupancyno>=? occNo or
   seatingcapacity>=? occNo or spacearea>=?occNoArea)
       LEFT JOIN SET3 USING (SPACEID, SPACENAME, SPACELNAME)
};
```

- o ACTION section that defines series of decision to perform on the outcome of the EVALUATE section. Each of the decision starts with WHEN clause and it supports the full logic expressions such as AND, OR, NOT and their combinations including nested conditions. The ACTION section allows:
  - Saving the results into a table
  - Saving the geometry information (part of the model and the transient geometry) into an X3D file for viewing. It has ability to control the color, highlight and transparency of the objects to be exported to improve readability of the result.

  Example:

ACTION

```
   WHEN? traveldistance>30000 and
       NOT (? BuildingClassification='En_30_50_50'
           and (? SprinklerProtectedAuto ='yes'
                OR? ProtBySprinklerSystem>0
                OR SPRINKLERNO>100))
       OR NOT ((? NonCombustibleCnt/? MainStructCnt)>0.8
           OR? FireProtectionClass='FireProof')
{print result SAVE INTO TABLE FAIL30 DRAW COLOR RED With BACKGROUND
       (IfcSpace, ifcDoor) HighLight (spaceid, doorid) COLOR CYAN
       TRANSPARENCY 0.75
       Save Into X3D 'c:\temp\drawpath30.x3d'
};
```

```
WHEN? traveldistance>40000 and (? BuildingClassification='En_45_10'
        and? NoStorey<=16) AND ((? NonCombustibleCnt/? MainStructCnt)>0.8
                OR? FireProtectionClass='FireProof')
{print result SAVE INTO TABLE FAIL40 DRAW COLOR RED With BACKGROUND
        (IfcSpace, ifcDoor) HighLight (spaceid, doorid) COLOR CYAN
        TRANSPARENCY 0.75
        Save Into X3D 'c:\temp\drawpath40.x3d'
};
```

- Execution environment
  BIMRL provides also an interface where the CHECK-EVALUATE-ACTION triplets can be defined and executed. The flexibility and extensibility of the language allows extremely wide-range rules to be written without much programming, except for the evaluation function extensions.
  BIMRL execution environment also allows definition of variables that can be used to assign values for variables used in the rule definition. This allows parameterization of rules. In addition the value itself can be assigned through another query allowing a flexible combination of many queries that contribute to the rules, especially in the ACTION section to allow complex expression to meet the rule criteria. The full example of this can be seen in the example that accompanies this document using BIMRL to define the rule example.
  The interface also allows chaining of CHECK-EVALUATE-ACTION triplets when dealing with overly complex rules that is not feasible to be done in just one rule. Example:

// define a variable on the classification code of the building

```
SETVAR? BuildingClassification SELECT a.classificationItemCode
        from bimrl_elemClassification a, bimrl_element b
        where a.elementid=b.elementid and b.elementtype='IFCBUILDING';
```

- Integrated geometry support
  BIMRL simplified schema supports integration of geometry into the schema and therefore allows integrated queries to be performed on the building models including their geometries and spatial operations.

4h.2 Strategic assessment of format

4h.2.1 Control, ownership, availability

BIMRL is developed as part of the PhD dissertation and therefore that concepts and details are available for public access. The author maintains their copyright and any intellectual property right that the author may develop out of them. The software that

defines the language, the execution environment, and the ETL (Extract, Transform, Load) module for the simplified schema are in the process to be made an open source project that will be freely available to public for research and development efforts.

4h.2.2 age, and stability,

- o suitability for multiple purposes
  - Authoring
  - Computer-assisted and Automated code compliance checking
  - Analysis and code comparisons
  - Filtering and structured dialogues
  - Controlling parametric objects

BIMRL is a relatively recent development (the dissertation was published in Dec 2015). However, the concepts and the scope of the solutions have been significantly influenced by the development of the Singapore's CORENET ePlanCheck system. It represents a much more streamlined and standardized approach, and yet still capable of solving complex rules in a relatively simple language (see [1, 6] for examples).

BIMRL is designed for use by the rule experts in mind. It does not need one to write a complex program to perform rather complex rules with the integrated geometry and spatial support. It also is designed for extendibility with the concept of plug-in evaluation functions that are often required to deal with overly complex logic or tedious concepts. It also allows parameterized rule by defining variables that can have the values assigned right before the rule execution.

An example of BIMRL output to X3D is shown below. The extrusion geometries in magenta are the transient geometry created in the EVALUATE section to allow a spatial query finding sanitary pipes that may pass right above the transformer or a water tank, while the rest of the objects are selected elements from the models defined in the ACTION section including highlight color and transparency.
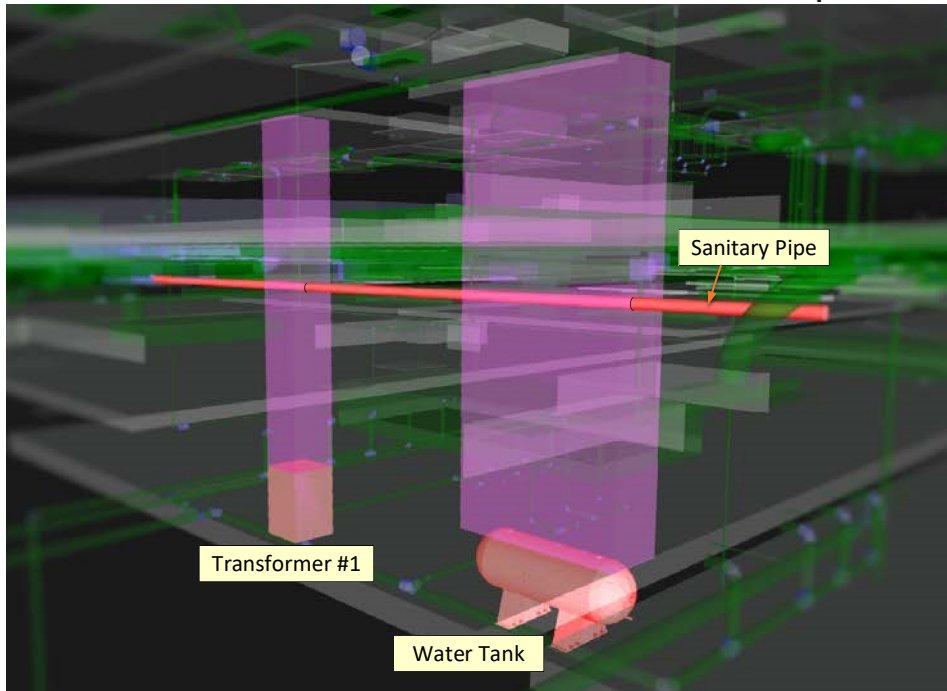
4h.3 Commercial base

> Availability (singular, multiple) of tools (costs, licensing)

> > ▪ Stability, maturity and available execution engines

BIMRL will be licensed under the open source license. The details for the license is not yet fully determined at this juncture.

> > 4h.3.1 authoring/capture tools (singular, multiple)

> > 4h.3.2 assessment engines, software and toolkits for (singular, multiple)

The tool has been demonstrated to work for relatively large and complex models and variety of complex rules as part of the dissertation work. Parts of the concepts are also making inroad into the commercial application.

> > 4h.3.3 Scaleability and role in creating a market place for RRR applications

What BIMRL provides is about the concept on dealing with complex rule checking system. The tool developed as part of the research is not the only tool that can be used or developed. The very minimum part that will be very useful is the simplified schema. This can

be accessed directly using the standard SQL, transformed it into a much more efficient RDF based data, or even adapted into the big data platform.

4h.4 Technical

- 4h.4.1 Expressivity and generality
  BIMRL is very expressive. It is able to support geometric and spatial operations as part of the integrated language and schema. It can handle very wide range of rules from simple rules all the way to a complex rules.
  Various approaches can be used to addressed really complex rules by breaking rules into multiple smaller rules, chain them into multiple rules, chain them into multiple evaluation functions within one rule, and by extending the evaluation functionality using the plugin mechaninsm.

  - What RRR content can be expressed

    BIMRL is not specifically tied to any RRR format. Currently, it will require manual steps to encode the RRR into BIMRL script, but it can be done by rule experts with minimum knowledge of IFC, BIMRL language and the simplified schema

  - Ability to handle easy and technical requirements through to hard and management requirements .

    As mentioned earlier, it is able to handle simple rules to complex rules including the integrated geometry and spatial operations.

  - Target (project) model schemas and formats

    It works with IFC. Currently with IFC2x3, but IFC4 support will be coming soon.

  - Performance measured against an independent RRR performance benchmark and support for heuristics and optimisation

    BIMRL uses RDBMS and therefore can be optimized alongside with the RDBMS optimization.

  - Ability to work with external dictionaries

    BIMRL is agnostic to the dictionaries. User will determine what disctionary to use and the use it consistently in the rule definitions. The dictionaries are very important piece though to ensure that the rules can be more generic and reliable dealing with the models.

- Openness, interoperability and convertablity

  It is based on IFC and SQL. It is also accessible openly for public. With the tools to be made open source, the entire things will be very open and easy to use.

  - 4h.4.2 Acceptability and provenance
    - Linkages back to (requirements) source

      BIMRL does not directly deal with this linkages. However, users have the flexibility to link that through various way including direct use of the RDBMS table that is suppurted directly, or through othes means, e.g. JSON [5].

    - Linkages forward to (project) model

      Linkages to the model is directly through the IFC GUID. All are accessible easily through the SQL interface.

    - Depth of results that can be reported

      BIMRL offers flexibility of reporting results. The basic information can be provided by text report, database tables, JSON. More interestingly, it supports output of the model (or part of thereof) and the additional geometries that may be used in the rules, currently into a standard x3d format.

- 4h.5 Definitive references and contacts
  1. Wawan, S., *(Ph.D. Thesis) A Simplified BIM Data Representation Using a Relational Database Schema for an Efficient Rule Checking System and Its Associated Rule Checking Language*, in *School of Architecture*. 2015-12-29, Georgia Institute of Technology.
  2. Solihin, W. and C. Eastman, *A Knowledge Representation Approach to Capturing BIM Based Rule Checking Requirements Using Conceptual Graph*, in *CIB W78 Conference 2015*. 2015: Eindhoven, the Netherland.
  3. Solihin, W. and C. Eastman, *A Knowledge Representation Approach to Capturing BIM Based Rule Checking Requirements Using Conceptual Graph*, accepted paper for the special issue on the "Assessment and QA, rule and code compliance checking, and compliance checking in general", ITCon.
  4. Solihin, W. and C. Eastman, *Classification of rules for automated BIM rule checking development.* Automation in Construction, 2015. **53**(0): p. 69-82.
  5. Dimyadi, J., Solihin, W., C. Eastman, R. Amor, *Integrating the BIM Rule Language into Compliant Design Audit Processes*, submitted paper for the CIB W78 2016 conference
  6. Solihin, W. and C. Eastman, *A Simplified BIM Model Server on a Big Data Platform*, submitted paper for the CIB W78 2016 Conference.

- 4h.6 Background references and publications, authors and contacts for further discussions.

Wawan Solihin

wsolihin@outlook.com

This document shows how the example rule can be expressed using BIMRL and the expected result that BIMRL should produce.

/* Common information for the Building to be obtained into variables first */

// define a variable that determines whether building is protected by an automatic sprinkler protection system using information from Pset_BuildingCommon

SETVAR? SprinklerProtectedAuto SELECT a.propertyvalue

    from bimrl_properties a, bimrl_element b

    Where a.elementid=b.elementid

      and a.propertyname='SprinklerProtectionAutomatic'

      and b.elementtype='IFCBUILDING';

// define a variable that determines whether building is protected by an automatic sprinkler protection system using information from the existence of the actual Sprinkler System (IfcSystem, and ServicesBuildings attribute)

SETVAR? ProtBySprinklerSystem select count (*) from

    bimrl_element a, bimrl_type b, bimrl_relconnection c,

      bimrl_classifassignment d

    where a.elementtype='IFCSYSTEM' and b.elementid=a.typeid

      and b.name like '%Automatic%Sprinkler%'

      and ((c.connectedelementid=a.elementid)

          or (c.connectingelementid=a.elementid))

      and c.relationshiptype='IFCRELSERVICESBUILDINGS'

      and d.elementid=a.elementid

      and d.classificationitemcode='Ss_55_30_96_85';

// define a variable on the classification code of the building

SETVAR? BuildingClassification SELECT a.classificationItemCode

      from bimrl_elemClassification a, bimrl_element b

      where a.elementid=b.elementid and b.elementtype='IFCBUILDING';

// define a variable to determine whether the main structural part (Slab and Column) are made of non-combustible materials. This number is a percentage of total (e.g. 0.8 means 80% of the main structural elements are non-combustible)

SETVAR? NonCombustibleCnt select count (*)

      from bimrl_element a, bimrl_properties b

      where a.elementid=b.elementid and b.propertyname='Combustible'

         and b.propertyvalue='False' and a.elementtype

         in ('IFCSLAB','IFCCOLUMN');

SETVAR? MainStructCnt select count (*) from bimrl_element_0006

      where elementtype in ('IFCSLAB','IFCCOLUMN');

// define a variable to determine a FireProtectionClass of a building

SETVAR ?**FireProtectionClass** select b.propertyvalue from bimrl_element a, bimrl_properties b

      WHERE b.elementid=a.elementid and a.elementtype='IFCBUILDING'

         and b.propertyname='FireProtectionClass';

// define a variable that counts the number of building storey excluding building stories that are underground

SETVAR? NoStorey SELECT count (*) from bimrl_element

      where elementtype='IFCBUILDINGSTOREY'

      and (name NOT LIKE '%Basement%'

         and name NOT LIKE '-%' and name NOT LIKE 'UG%');

// define a variable that check a special property whether a factory is unmanned (Assuming there a property named "UnmannedFactoryOpertation")

SETVAR? UnmannedFactory SELECT b.propertyvalue from bimrl_element a, bimrl_properties b

      WHERE b.elementid=a.elementid and a.elementtype='IFCBUILDING'

         and b.propertyname='UnmannedFactoryOperation';

/* BIMRL rule checking

- The rule is defined using 2 sets of CHECK statements that collect 2 sets of information:

Page no.	Author
106	bSI Regulatory Room

- SET1 collects the pair of space and exit door that the distance is to be calculated
  - - SET2 collects the existence of the sprinkler heads in the space concerned and count them
- The EVALUATE performs:
  - Join the SET1 and SET2 table using Left Join
  - Find the shortest path from a remote location in the space to the exit door and compute the distance inside the *ComputePathAndDistance()* function
- The ACTION defines data to output for various non-compliance conditions

```
*/

CHECK
{IfcSpace s, ifcdoor d
        WHERE property(s, HabitableSpace) ='TRUE'
                and classificationof (d).classificationitemcode='2.6.2'
                and classificationof (d).classificationname='BCIS'
                and s.container=d.container
        COLLECT s.elementid spaceid, d.elementid doorid, s.name spacenumber;
} AS SET1;
{IfcSpace SC
        Where CONTAINS (SC, USEGEOMETRY).ElementType='IFCFLOWTERMINAL'
                and CONTAINS (SC, USEGEOMETRY).Name like 'Sprinkler%'
        COLLECT SC.ElementId SPACEID, SC.Name SPACENUMBER,
                SC.LongName SPACELNAME, COUNT (unique CO.ElementID) SPRINKLERNO
        GROUP BY SC.ELementId, SC.Name, SC.LongName
} AS SET2;
EVALUATE computePathAndDistance (spaceid, doorid) output? traveldistance
        From SET1 LEFT OUTER JOIN SET2 USING (SPACEID, SPACENUMBER);
ACTION
        WHEN? traveldistance>30000 and
                NOT (? BuildingClassification='En_30_50_50'
                        and (? SprinklerProtectedAuto ='yes'
                                OR? ProtBySprinklerSystem>0
                                OR SPRINKLERNO>100))
                OR NOT ((? NonCombustibleCnt/? MainStructCnt)>0.8
                        OR? FireProtectionClass='FireProof')
        {print result SAVE INTO TABLE FAIL30 DRAW COLOR RED With BACKGROUND
                (IfcSpace, ifcDoor) HighLight (spaceid, doorid) COLOR CYAN
                TRANSPARENCY 0.75
                Save Into X3D 'c:\temp\drawpath30.x3d'
        };
```

```
WHEN? traveldistance>40000 and (? BuildingClassification='En_45_10'
       and? NoStorey<=16) AND ((? NonCombustibleCnt/? MainStructCnt)>0.8
               OR? FireProtectionClass='FireProof')
{print result SAVE INTO TABLE FAIL40 DRAW COLOR RED With BACKGROUND
       (IfcSpace, ifcDoor) HighLight (spaceid, doorid) COLOR CYAN
       TRANSPARENCY 0.75
       Save Into X3D 'c:\temp\drawpath40.x3d'
};


WHEN? traveldistance>50000 and (? BuildingClassification='En_45_10'
       and? NoStorey>16) AND ((? NonCombustibleCnt/? MainStructCnt)>0.8
               OR? FireProtectionClass='FireProof')
{print result SAVE INTO TABLE FAIL50 DRAW COLOR RED With BACKGROUND
       (IfcSpace, ifcDoor) HighLight (spaceid, doorid) COLOR CYAN
       TRANSPARENCY 0.75
       Save Into X3D 'c:\temp\drawpath50.x3d'
};


WHEN? traveldistance>75000 and? BuildingClassification ='En_30_50_50'
       and (? SprinklerProtectedAuto ='yes' OR? ProtBySprinklerSystem>0
               OR SPRINKLERNO>100)
       AND? UnmannedFactory != 'TRUE'
{print result SAVE INTO TABLE FAIL75 DRAW COLOR RED With BACKGROUND
       (IfcSpace, ifcDoor) HighLight (spaceid, doorid) COLOR CYAN
       TRANSPARENCY 0.75
       Save Into X3D 'c:\temp\drawpath75.x3d'
};


WHEN ?traveldistance>100000 and? BuildingClassification ='En_30_50_50'
       and (? SprinklerProtectedAuto ='yes' OR? ProtBySprinklerSystem>0
               OR SPRINKLERNO>100)
       AND? UnmannedFactory = 'TRUE'
{print result SAVE INTO TABLE FAIL100 DRAW COLOR RED With BACKGROUND
       (IfcSpace, ifcDoor) HighLight (spaceid, doorid) COLOR CYAN
       TRANSPARENCY 0.75
       Save Into X3D 'c:\temp\drawpath100.x3d'
};
```

Figure 17 - The checking results from the above rule applied to both of the example models

Explanation:

- Pre-assigned variables for common information for the building is shown here for two information, i.e. building type (from classification), and sprinkler protection. The list can be added as necessary for example or the number of stories, etc. The variables are then used in the ACTION sub-clause to identify the condition the rule is to be complied.
- CHECK subclause define selection criteria of objects of interests. In the example, 3 objects are collected:
  - Spaces (only limited to the habitable space. In this example just using a name because the example model does not contain the necessary information)
  - Doors (only a specific door based on the classification code is used to identify the exit doors)
  - Building (to collect the type of building obtained from the classification assigned to it)

We will use the ids of the space and door to pair them later on.

The Check subclause allows a lot more conditional statements, which is not used in this example due to the limited availability of the data in the example models. BIMRL allows a powerful query to filter the data down to the objects of interests. It supports enhanced relationship information provided in the IFC model, and currently circulation graph (connection only).

- EVALUATE subclause calls a function to compute the shortest path from a remote location in a space to the exit door (computePathAndDistance()). The start and end are defined by space id (start) and doored (end)
- ACTION subclause tell BIMRL what to do when the distance information and the building type information are evaluated to be true. This allows different reports to be created depending on the data and the noncompliance. The above example outputs the result of query and path/distance into a table and at the same time output graphically part of the building and the actual path into an X3D file (see below)
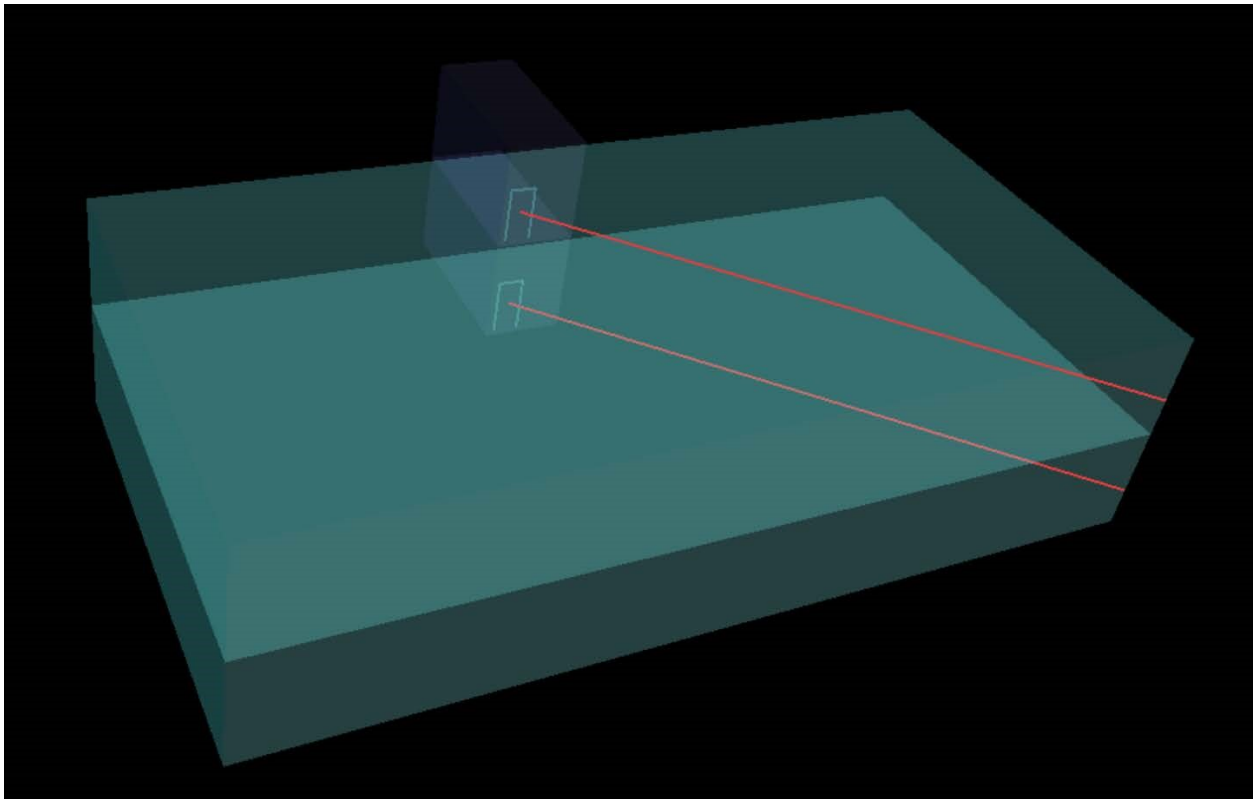
Figure 18 - Checking result generated from BIMRL for example model-A
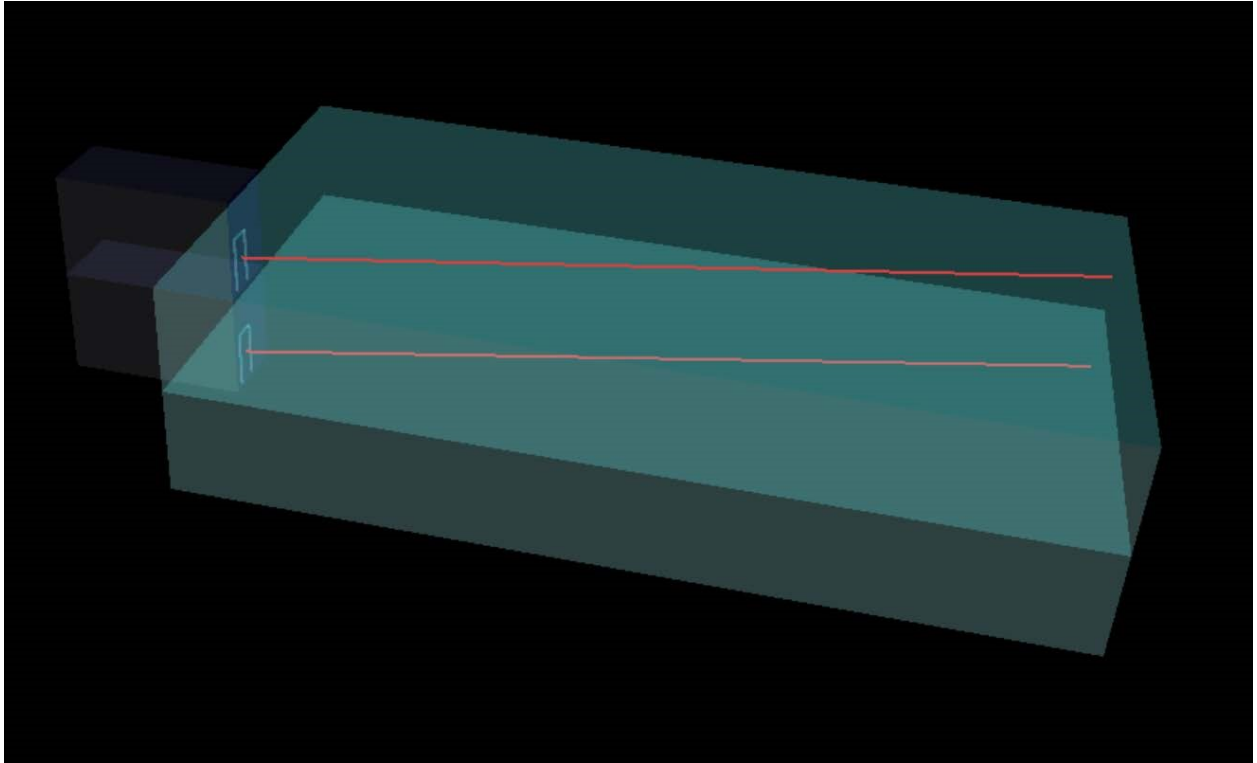
Figure 19 - Checking result generated from BIMRL for example model-B

<u>Note</u>:

The distance calculated is not the final number as the function computePathAndDIstance () is not yet fully implemented in this example. Only the graph that connects all related objects to form a circulation graph in the entire building has been implemented (not the actual walking path and distance yet). The extension can be added easily and the rule will be unchanged.

References

Annex: Contributing authors

Initials	Name	Affiliation or Company	Email
WS	Wawan Solihin		wsolihin@outlook.com

Page no.	Author
111	bSI Regulatory Room

4i Rule Table

4i.1 Introduction

Rule tables are widely used for the informal capture of knowledge and expectations. In particular spreadsheets have been used to make such rules accessible and editable by domain participants.

Four examples:

- The Georgia Tech/GSA Court Design project tabulated the adjaceny and circulation rules found in the GSA Court design guide. Eventually 14 different spreadsheets were used, reflecting the number of logical structures found. A similar templating approach has been attempted following the application of NLP analysis. No upper limit has been set on the numbe rof templates that may be needed.
- The ealry stages of the ICC SmartCodes project used a spreadsheet to summarise both rules and the editorial commentary. The commentary included whether the rules were felt to be too abstract or complex to be considered. As these were re-examined, it was realised that no single spreadsheet or family of spreadsheets was going to be able to span the variety of logical structures encountered, which led to the adoption of the RASE approach.
- The OpenRule (OpenRule.com) has adopted ordinary spreadsheets to capture both data and rules. The Rule format can be either row or column based. Most entries are human readable, but three entries are reserved for technical implementation details.

	B	C	D
3	Rules void **defineGreeting**(int hour, Response response)		
4	C1	C2	A1
5	min <= hour	hour <= max	response.map.put("greeting", greeting);
6	int min	int max	String greeting
7	**Hour From**	**Hour To**	**Set Greeting**
8	0	11	Good Morning
9	12	17	Good Afternoon
10	18	22	Good Evening
11	23	24	Good Night

Figure 20: User (black and white) and technical (cyan and blue) entries to define a Ruleset in OpenRules

- buildingSMART adopted a systematic use of the spreadsheet format to develop and document MVD rule sets. MVD rules define
  - sub-sets of the full IFC shecma for implementation and software testing, for eample the scope of the 'FM Design to Construction' view.

- o specific implementation requirements for particular contexts for example the use of speciifc attrbutes and classiifcation codes in Norway.
- The example shows the atomic exchnage requirements, the use of the COBie spreadsheet format, whether export and.or import is madnatory and links to the IFC schema and to a repository of detailed named concepts (previously called functional parts)

Figure 21: Tables used to define an MVD on the buildingSMART IFC schema

- Recently this approach has been superceded by the use of the ifcDoc tool which automates the selection of standard concepts and the specificaiton of detailed requirements, so as to generate a specific mvdXML1.1 .

4i.2.1 Strengths

- It is very accessible, as spreadhseets and tables are universally supported.
- It introduces a first level of rigoor, over the use of free text, to define RRR.

4i.2.2 Weaknesses

- It tends to feed into the procedural approach if(x)then(y)
- It may not constrain the content or syntax of fields, leading to unsupported outcomes.
- It may not be able to represent more complex logical structures, or may become sparse and unreadable in its attempts to cover complex and simple cases.
- It may require a mixture of domain and techncial syntaxes

Tabular approach: complexity always explodes

4i.2.3 Opportunities

- Leaving aside the logical structure, the tabular approach can be used to summarise the metrics (lowest level checks) found in RRR documents,

Korean Building Act 34-1

Colour	Phrase	Id	Title	Object	Property	Comparison	Target	Unit
Selection	automatic fire extinguishers			building	hasSystem		automatic fire extinguishers	
Selection	sprinklers			building	hasSystem		sprinklers	
Application	prescribed by Ordinance of the Ministry of Land, Infrastructure and Transport			building	proscribedUse			
Application	factory			building	usage		factory	
Application	automated production facility			building	usage		factory	
Requirement	walking distance from each part of the living room to the stairs is not more than 30 meters			space	walking distance	<=	30	m
Requirement	walking distance of not more than 50 meters			space	walking distance	<=	50	m
Requirement	the walking distance of not more than 75 meters			space	walking distance	<=	75	m
Application	stairs			stair	exists			
Selection	leading to the shelter floor			stair	leading to		shelter floor	
Selection	the ground			stair	leading to		ground	
Exception	the shelter floor			stair	leading to		shelter floor	
Selection	fireproof structure			structure	isFireProof			
Selection	non-combustible materials			structure	isNonCombustible			

-

Figure 22: Table of metrics found in KBA 34-1 (colours reflect their RASE role)

- Some client requirements, such as the US UFC 4-510-01 DESIGN: MEDICAL MILITARY FACILITIES Appendix A:requirements for hospital rooms, have been captured in tables. These tables have a simple structure, as in the example where the space type code defines the conditonal applicability and all other columns represent discrete requirements. Most requirements are from defined pick lists. In these cases in may be possible to map the table to a more general syntax, such as the IfcConstriant model, or RASE.

UFC 4-510-01, 1 August 2011 - Change 4
ARCHITECTURAL AND ENGINEERING DESIGN REQUIREMENTS

Room Code	Room Name	ARCHITECTURAL								STRUCTURAL	ELECTRICAL				MEDICAL GASES											HVAC							
		MATRL + FIN			CLG	DOOR		NOISE		FLR	LIGHT		ES PWR			M C	D C		L	P	M	H	L D	O	N	W A N G			TEMP		R	FL		E
		FL	WL	CLG	HT	SIZE	RM	SC		LOAD	LEV	N	PWR	LT	N	A	A	A	A	V	V	V	E X	Q	I	D	N	BL	AC	OA	SM	WN	H	IN	EN	N	X
AMB01	AMBULANCE SHELTER	CS	SSP	SSP	VAR	SP	.	.		A	50	.	.	LS	.	.	.	.	.	.	.	.	. .	.	.	.		.	.	.	.	.	.	.	.
		.	.	.	VAR	SP	.			A	.	.	.	.		.	.	.	.	.	.	.	. .	.	.	.		.	.	.	.	.	.	.
AMB02	AMBULANCE GARAGE	CS	BPT	SSP	VAR	SP	35	50		A	50	.	.	LS	.	.	.	.	.	.	.	.	. .	.	.	.		0	6	4	.	20	.	.	.	H	Y
		.	.	.	VAR	SP	40			A	.	.	.	.		.	.	.	.	.	.	.	. .	.	.	.		.	.	68	.	.	.	.
ANCW1	ANESTHESIA CLEAN WORKROOM	SV	GLG	AT2	2400	900	35	40		3	700	.	R	L		1	.	.	1	.	.	.	1 1	.	1	D		.	4	15	24	.	14	.	.	Y
		ET	.	.	8-0	3-0	40			60	.	.	.			.	.	.	.	.	.	.	. .	.	.	.		.	.	75	.	.	.	.

Figure 23: Tabular requirements.

4i.2.4 Threats

- The tabular approach can be initially seductive. In some cases, the problems emerge only when the specific first examples are completed and the approach is applied more generally.
- The following section examines a format 'Decision Modell and Notation' which combines a hierarchical approach with a tabular approach.
- When used to tabulate simple metrics, without the logical structure, it may conceal that some data is optional. This can lead to an explosion of apparently required data, leading to impractical and overwhelming demands on the authors of project models. This can be a serious challenge to the MVD approach discussed below.

4i.3 DMN 'Decision Model and Notation'

4i.3.1 Introduction

<<Decision Model and Notation (DMN) It is a standard approach for describing and modeling repeatable decisions within organizations to ensure that decision models are interchangeable across organizations.

The DMN standard provides the industry with a modeling notation for decisions that will support decision management and business rules. The notation is designed to be readable by business and IT users alike. >> Wikipedia

4i.3.2 Strategic assessment of format

Release 1.0 appeared in late 2013 and the latest revision was published in 2015. Release 1.1 appeared in 2016, but is not fully adopted. DMN is seen as complementary to the

evolving BPMN standards which are already used by BuildingSMART for process modelling – both are published by the Object Management Group.

Authoring is in two parts, the creation of decision tables and formulae, and the creation of decision graphs which join together such parts with outputs linking to inputs. A variety of engines exist which must be embedded in an application which provides the required inputs and uses the resulting output. This may result in over-demand of inputs which may prove unnecessary.

All though all the examples and test cases are based on business processes, Computer-assisted and Automated code compliance checking is directly analogous and feasible.

Analysis and code comparisons, Filtering and structured dialogues and Controlling parametric objects are not possible, due to the directed flow of execution implied by the decision graphs, tables and formulae.

One application makes a feature of the ability to explain and report the detailed execution of a ruleset. .

4i.3.3 Commercial base

The following applications have been identified:

- Camunda (www.camunda.com)
- OpenRules (www.openrules.com)
- Redhat
- ...

One website claims that there are at least 16 vendors with DMN implementations.

4i.3.4 Technical base

Expressivity and generality is high in that the combination of graph, table and formulae should allow most actual cases in RRR to be represented. For example, any situation where the logical operators AND, OR, NOTAND and NOTOR arise can be represented as conventional truth-tables with the appropriate number of operands. Since the graphs can be of any size, very complex regulations can be represented. Any metric is defined as an input which the wrapping application must provide, accessing any database or user interface required. DMN has no need to represent any specific vocabulary, as this must be handed by the wrapping application.

The main artefact of DMN is an XML schema. This is a relatively immature schema, with technical errors, and the published examples also have errors. In 2017 there was an active challenge to test and eradicate the non-interoperability of the format: there are many stylistic and formulae languages supported which mitigates against portability. As the format and its engine operate as a black box, there are few opportunities for reporting, or for linking back to the requirements source content or for linking forward to the entities in the project model.

4i.3.5 Definitive references and contacts

4i.3.6 Background references and publications, authors and contacts for further discussions.

4i.4 MVD

MVD is an example where data collection in tables has been used. It is also an example of the pre-requisites for RRR checking, and so is also covered in the section 3b on Resources.

With the growing number of requirements for client-driven projects, the regulatory and required data that should be satisfied has grown quickly. Diverse types of requirements should be categozied and managed consistently so that an automated checking process can be firmly and successfully developed. This section includes the types of rules identified from the IFC interoperability checking project. The project has an objective to figure out the types of syntactic and semantic requirements pertaining to the evaluation of an IFC instance file according to a Model View Definition, the subset of the Industry Foundation Classes (IFC) schema. Considered another way, an MVD consists of criteria to be used for

evaluating an IFC instance file according to the specifications of data exchange (Lee, 2015). The types of rules in this section are subject to be updated accordingly.

4i.4.1 Scope of work

To identify the scope of MVD rule checking and the types of rules, this section uses the Precast Concrete National BIM Standard including the model view of Precast/Pre-stressed Concrete Industry (PCI) (Lee, 2015). The PCI model view, which has 46 exchanges and 96 concept descriptions reused by each exchange (Eastman et al 2010), is complete specifications for implementation of data exchanges of a precast concrete domain. In addition, the PCI model view includes details about how precast concrete objects, their attributes, and references should be represented, translated, and referred when being exchanged using the IFC format. This section contains rule types extracted and categorized from its model view implemented on the modularized validation platform of the IfcDoc tool (Lee, 2015).

4i.4.2 Types of rules

Table 1 represents several types of rules extracted from 96 PCI concepts. The generalized types of implementable specifications in concept descriptions are executed by diverse parameters and checking features developed on the validation framework of the IfcDoc tool.

Table 1 Rule types classified by the concepts of the PCI MVD (Lee, 2016)

Rule Set Type		Note
Data value	Check accuracy of a data value	Checking a data value
		Checking a value with a substring
		Checking an arithmetic constraint
		Checking an enumeration value

	Checking a value of an attribute to be of a defined type
Check cardinality	Checking existence or Null of a value
	Evaluating lower and upper bound: Setting a limit on the number of attributes
Check uniqueness of a value	Checking global uniqueness within a file
	Checking local uniqueness in aggregation
Data type	Checking the correct type of an entity
	Checking subtypes
References and relationships	Checking a referencing relationship
	Checking an inverse relationship
Conditional checking	Checking an instance only if a given condition is satisfied
Syntactic checking	Checking the fundamental syntax of a model view

4i.4.3 Data accuracy

This checking type primarily addresses the semantics of a building information model required for data exchange for a scoped domain. For such exchanges, a model view allows users to declare mandatory and optional values for the attributes of entities such as a name, a description, an object type, a representation type, a connection, and a tag. The values of such attributes, determined by a specific purpose, become criteria for validation of an IFC instance file pertaining to fulfillment and accuracy of requisite values for data exchange. This accuracy checking is the most fundamental and explicit rule type that is indispensable for

constructing diverse rule sets. This type of checking includes a simple comparison between defined values from model views and an IFC instance file.

4i.4.4 Data existence and null

A model view requires the inclusion or exclusion of exchange building data: For example, a structural engineer has to receive values associated with a structural load, stiffness, precise dimensions, materials, connections, and others required for a structural analysis from other experts. Domain professionals need to evaluate the existence and the null status of a corresponding value according to three levels of definitions: an attribute, an entity, and a relationship.

4i.4.5 Data uniqueness

The IFC schema defines that all object instances require a globally unique identifier (GUID) attribute: A unique 22 character length string must be fulfilled by all IFC instances. In addition, a model view can declare that an attribute such as Tag must have a unique value within an IFC instance file. The uniqueness checking is also needed at a local syntactic level where attribute values exist in a SET data type because such type disallows duplicate elements. Even though the uniqueness regarding GUID and a data type can be validated in the level of syntax, such requirements are also defined in specifications of a model view that must be fulfilled in a data exchange process.

4i.4.6 Entity data type

The IFC specifications define distinct entity data types for attributes and thus an IFC instance file must comply with the predefined types. Within the allowable range of such regulations, domain professionals and model view developers can define entity data types on model views for their purposes, narrowing down the scope of acceptable data types for a targeted attribute. Thus, the entity data types of instances should be evaluated to ensure the accuracy and interoperability of data models. In particular, a user-defined entity data type must be restricted and validated by multiple inheritances such as SUPERTYPE OF or SUBTYPE OF because the IFC schema has a strictly layered hierarchy. For instance, if an aggregation concept description defines that the RelatingObject attribute of IfcRelAggregate

is IfcWall and that the RelatedObjects attribute is IfcBuildingElement, the subtype entities of IfcBuildingElement such as IfcColumn can satisfy the RelatedObjects attribute.

4i.4.7 Conditional checking

The implementation of validation typically consists of diverse types of rule sets that should be launched only if instances meet the conditions of another rule logic such as data accuracy or a rule parameter. Such correlated rules are dependent on the validation outcome of a precedent rule such as TRUE or FALSE. Based on the checking result of a precedent condition, subordinate rules and their executions must be controlled and managed, potentially with multiple levels of nesting.

[NN] MVDs and MVDXML are not able currently to support anything beyond the minimal if(x)then(y) structure. More complex structures may not be representable.

- if(s1 or s2) then (r1 or r2)
- if(not (a1 and a2) or not(s1 or s2) or (e1 or e2) or (r1 and r2)) then pass

4i.4.8 Cardinality

The cardinality checking evaluates the lower and upper bounds of values of an attribute. The IFC schema declares cardinality for all attributes as a syntactic specification. Similar to entity data type checking, model view definers can set appropriate lower and upper bounds for an attribute within the available range of cardinality defined in the IFC schema.

4i.4.9 Reference/Inverse relationship

An IFC instance file has a complex structure that consists of various references and inverse relationships, allowing multiple inheritance. Within the limited range of the IFC schema specifications pertaining to an allowable relational structure, each data exchange requires diverse entity relationships and their different configurations. Thus, an attribute must refer to correct entities and be referred by acceptable inverse relationships as defined in a model view.

4i.4.10 Fundamental syntactic checking

Because a model view is a subset of the IFC schema, an IFC instance file must follow specifications not only defined in a model view but also the IFC schema. Thus, if an IFC instance file has an entity, an attribute, and a reference that are out of scope of a model view definition for a specific domain, such validation should be reported as a syntactic error.

4i.4.11 Geometry and topology checking

Diverse rules associated geometry and topology exist such as circulation relations, fire code exits, clash detections, path distance checking, and space program checking. These rule sets and guidelines are used to ensure the required design information is available and trusted. This checking type addresses BIM quality assurance, quality control, compliance control, design analysis, and building code checking,

4i.4.12 Definitive references and contacts

A building design will encompass a significant number of requirements and data demanded by diverse domain professionals. As a result, they will become more keenly aware of the integrity of building data. To identify and formalize all possible types of rules required for interoperable requirements and rules, various rule checking tools and examples should be studied and referred. In addition, since diverse domains need a wide range of distinct domain knowledge, the thoughtful categorization of all identified rule types will be required to verify building model data pertaining to the conformity of various requirements and guarantee the accuracy of data.

References

Lee, Y., Eastman, C. (2015). Formalized Validation Rules and Framework of a Building Information Model Pertaining to a Model View Definition. In *Proceeding of the 32nd CIB W78 Conference 2015*. pp. 450-459.

Lee, Y., Eastman, C. (2016). Logic for Ensuring the Integrity of Building Information Model Data, *Journal of Automation in Construction*. Under review.

Eastman C., Sacks R., Panushev I., Aram S., Yagmur E., (2010). Precast Concrete
BIM Standard Documents: Model View Definitions for Precast Concrete.

Annex: Contributing authors

Initials	Name	Affiliation or Company	Email
YL	Yong-Cheol Lee	Louisiana State University	yclee.it@gmail.com

4j Semantic web and the Jena Rule Language

Introduction

The increase in the number of documents and data sets described in the languages of the Semantic Web is leading to develop more and more applications designed for their processing. Many of these applications implement inference engines to support automated processing of these data. Within this group, rule-based inference engines are able to reuse rules described in standardized rule languages. This way, rules described in these languages can be applied to infer new knowledge from information described in Semantic Web languages. Typically, the capabilities of these languages are analysed according to their expressiveness, syntax, and built-in functions. One of these rule languages is Jena rules. A free rule-based reasoner to process jena rules is Jena inference engine.

A comparative analysis between Jena rule language (and the Jena inference engine) and other standard rule languages and reasoners is provided in [1]. This comparative is performed based on the expressiveness of the rule languages; supported operations and syntax; built-in functions available; reasoning strategies, algorithms, and operations; and Semantic Web languages supported by the inference engines.

According to the documentation, "*Jena is a Java framework for building Semantic Web applications. It provides a extensive Java libraries for helping developers develop code that handles RDF, RDFS, RDFa, OWL and SPARQL in line with published W3C recommendations. Jena includes a rule-based inference engine to perform reasoning based on OWL and RDFS ontologies, and a variety of storage strategies to store RDF triples in memory or on disk*".

Support for inference within Jena is provided in [2].

Exemplar Projects

- Currently, Jena rules application cases for the construction industry can be found in some proof of concept implementations and individual experiments, as for example in [4].

References

[1] Rattanasawad, T., Saikaew, K.R., Buranarach, M., & Supnithi, T. (2013). A Review and Comparison of Rule Languages and Rule-Based Inference Engines for the

Semantic Web. In: International Computer Science and Engineering Conference, pp. 1–6. IEEE Press, New York. Available online: http://www.easybi.in.th/icsec2013/papers/1569794919.pdf

[2] The Apache Software Foundation. 2009. Apache Jena - Reasoners and rule engines Jena inference support. Available online: http://jena.apache.org/documentation/inference

[3] Fuseki, http://jena.apache.org/documentation/serving_data/

[4] Costa, G., & Pauwels, P. (2015). Building product suggestions for a BIM model based on rule sets and a semantic reasoning engine. In 32rd international CIB W78 conference, pp. 98-107. Available online: https://biblio.ugent.be/publication/6890587/file/-6974113

Annex: Contributing authors

Initials	Name	Affiliation or Company	Email
GC	Gonçal Costa	La Salle Engineering and Architecture - Ramon Llull University	gcosta@salleurl.edu

4k NLP

4k.1 Introduction

4k.1.1 Names and synonyms

NLP stands for Natural Language Processing, it aims at enabling computers to understand and process natural lanaguge text and speech in a human-like manner (Cherpas 1992). It has been successfully utilized in processing information in many domains such as medical, business, and legal domains. Innovative methods built on NLP was developed in recent years to support the automated extraction and tranformation of building code requirements from textual documents into computable rule formats (Zhang and El-Gohary 2013; 2015).

4k.1.2 Description

The automated code requirement extraction and transformation methods built on NLP techniques have the potential to be integrated with many different types of code requirement rules representations, such as RASE, SWRL, and N3Logic. The effectiveness of the methods have been demonstrated in the creation of a semantic natural language processing (NLP)-based automated compliance checking (SNACC) system that integrates the methods with Prolog logic representation of code requirement rules, as well as automated IFC information processing algorithms (Zhang and El-Gohary 2016b).

4k.2 Strategic assessment of format

4k.2.1 Control, ownership, availability

The automated code requirement extraction and transformation methods built on NLP were developed by Zhang and El-Gohary, but has been extensively published and demonstrated.

4k.2.2 age, and stability,

- o suitability for multiple purposes
    - ▪ Computer-assisted and Automated code compliance checking
    - ▪ Automated/semi-automated analysis and code comparisons

The purpose of automated code requirement extraction and transformation methods is supporting an automated conversion of regulatory requirement rules from text to any computable format.

4k.3 Commercial base.

4k.3.1 text processing tools (singular, multiple)

The text processing in the methods leveraged a group of NLP tools provided in general architecture for text engineering (GATE) <https://gate.ac.uk/>, including tokenizer, sentence splitter, gazetteer lists, morphological analyzer, ontology editor, and Java Annotation Patterns Engine (JAPE) transducer. Features are generated using these tools and then used by meta rules for extracting single semantic information elements and resolving conflicts in the extraction (Zhang and El-Gohary 2013).

The general architecture for text engineering (GATE) tools are both commercially available.

4k.3.2 assessment engines, software and toolkits for (singular, multiple)

4. General architecture for text engineering (GATE) <https://gate.ac.uk/>

4k.3.3 Scalability and role in creating a market place for RRR applications

The code requirement extraction and transformation methods was intended for supporting an automated analysis and processing of regulatory requirement information into any computable rule representation, therefore it has the ability to be integrated with most of the rule representation and reasoning methods mentioned in this report.

4k.4 Technical

- o 4k.4.1 Expressivity and generality
  - Any RRR content can be automatically analyzed, extracted, and transformed into a computable format, There is no need of manual mark-up but rather an available domain ontology with the common concepts encoded.
- o 4k.4.2 Acceptability and provenance
  - Linkages back to (requirements) source are created by the creation of an arbitrary ID number to each computable rule. There is a one-to-one mapping between English regulatory requirements and computable rules. This ID can be passed through to any resulting report or user interface.

- 4k.5 Definitive references and contacts

- Cherpas, C. (1992). "Natural language processing, pragmatics, and verbal behaviour." Anal. Verbal Behav, 10(1992), 135–147.

- Zhang, J., and El-Gohary, N.M. (2013). "Semantic NLP-based information extraction from construction regulatory documents for automated compliance checking." J. Comput. in Civ. Eng., 30(2), 04015014.

- Zhang, J., and El-Gohary, N.M. (2015). "Automated information transformation for automated regulatory compliance checking in construction." J. Comput. in Civ. Eng., 29(SP2013IWCCE), B4015001.

- Zhang, J. and El-Gohary, N. (2016a). "Semantic-Based Logic Representation and Reasoning for Automated Regulatory Compliance Checking." J. Comput. Civ. Eng., 10.1061/ (ASCE) CP.1943-5487.0000583, 04016037.
- Zhang, J., and El-Gohary, N.M. (2016b). "Integrating semantic NLP and logic reasoning into a unified system for fully-automated code checking." Automation in Construction, accepted.
- Zhou, N. (2014). "B-Prolog user's manual (version 8.1): Prolog, agent, and constraint programming." 〈 http://www.picat-lang.org/bprolog/download/manual.pdf 〉 (Sept. 05, 2016).

Applications

- Zhang, J., and El-Gohary, N.M. (2015). "Automated extraction of information from building information models into a semantic logic-based representation." Proc., 2015 ASCE Intl. Workshop on Comput. in Civ. Eng., ASCE, Reston, VA, 173-180.
  ZE_BIM_FOL_Converter.zip

  A java-based tool to convert IFC files to Prolog facts that can be used together with

  Prolog represented code requirement rules.

4k.6 Background references and publications, authors and contacts for further

discussions.

Annex: Contributing authors

Initials	Name	Affiliation or Company	Email
JZ	Jiansong Zhang	Assistant Professor at Western Michigan University	jiansong.zhang@wmich.edu

4I Prolog

4I.1 Introduction

4I.1.1 Names and synonyms

Prolog is the classic platform implementing the logic programming paradigm. Logic programming is different from other programming paradigms in that logic programs only need to define a problem and the solving process is automated, thanks to the built-in reasoning mechanism supported by search strategies and backtracking. Prolog was utilized to represent code requirement rules and design facts to support automated code compliance checking (Zhang and El-Gohary 2016a).

4I.1.2 Description

The effectiveness of the information representation and compliance reasoning method using Prolog have been demonstrated in checking the quantitative rules in Chapter 19 of International Building Code 2009 and a building test case (Zhang and El-Gohary 2016a), it was further integrated into a semantic natural language processing (NLP)-based automated compliance checking (SNACC) system that integrates the NLP methods with Prolog logic representation of code requirement rules, as well as automated IFC information processing algorithms (Zhang and El-Gohary 2016b).

4I.2 Strategic assessment of format

4I.2.1 Control, ownership, availability

The representation and reasoning methods built on Prolog were developed by Zhang and El-Gohary, but has been extensively published and demonstrated.

19.2.2 age, and stability,

- o suitability for multiple purposes
  - Computer-assisted and Automated code compliance checking
  - Other automated reasoning

The purpose of the representation and reasoning methods using Prolog is supporting the automated reasoning with code requirement rules, as well as providing a more human-readable data that can be checked more easily (comparing to less human-readable data) when necessary.

4l.3 Commercial base.

4l.3.1 Prolog tools (singular, multiple)

The Prolog logic representation was implemented in B-Prolog (Zhou 2014). There are many other Prolog systems commercially available.

4l.3.2 assessment engines, software and toolkits for (singular, multiple)

5. B-Prolog (Zhou 2014)

4l.3.3 Scalability and role in creating a market place for RRR applications

The Prolog-based logic representation and reasoning methods for code requirement rules are efficient thanks to the well matured and optimized reasoning mechanism in Prolog systems.

4l.4 Technical

- o 4l.4.1 Expressivity and generality
  - Prolog is built upon first order logic which is very expressive. It can be used to represent almost all code checking information if carefully designed.
- o 4l.4.2 Acceptability and provenance
  - The use of original names for predicates and arguments in the representation makes the data easily human-readable, therefore can be manually checked when necessary.

4l.5 Definitive references and contacts

- Cherpas, C. (1992). "Natural language processing, pragmatics, and verbal behaviour." Anal. Verbal Behav, 10(1992), 135–147.

- Zhang, J., and El-Gohary, N.M. (2013). "Semantic NLP-based information extraction from construction regulatory documents for automated compliance checking." J. Comput. in Civ. Eng., 30(2), 04015014.

- Zhang, J., and El-Gohary, N.M. (2015). "Automated information transformation for automated regulatory compliance checking in construction." J. Comput. in Civ. Eng., 29(SP2013IWCCE), B4015001.

- Zhang, J. and El-Gohary, N. (2016a). "Semantic-Based Logic Representation and Reasoning for Automated Regulatory Compliance Checking." J. Comput. Civ. Eng., 10.1061/(ASCE)CP.1943-5487.0000583, 04016037.
- Zhang, J., and El-Gohary, N.M. (2016b). "Integrating semantic NLP and logic reasoning into a unified system for fully-automated code checking." Automation in Construction, accepted.
- Zhou, N. (2014). "B-Prolog user's manual (version 8.1): Prolog, agent, and constraint programming." 〈 http://www.picat-lang.org/bprolog/download/manual.pdf 〉 (Sept. 05, 2016).

Applications

- **Zhang, J., and El-Gohary, N.M. (2015). "Automated extracti**on of information from building information models into a semantic logic-based representation." Proc., 2015 ASCE Intl. Workshop on Comput. in Civ. Eng., ASCE, Rest**on, VA, 173-180.
  ZE_BIM_FOL_Converter.zip

  A java-based tool to convert IFC files to Prolog facts that can be used together with

  Prolog represented code requirement rules.

4I.6 Background references and publications, authors and contacts for further discussions.

Annex: Contributing authors

Initials	Name	Affiliation or Company	Email
JZ	Jiansong Zhang	Assistant Professor at Western Michigan University	jiansong.zhang@wmich.edu

4m   Commercial rule engines / Business Rule Management Systems

1 Introduction

Solution for rule based checking is well established in industries like finance, automobile, aero, oil & gas and others industries. This type of solution is also implemented in public services like tax and health services (examples from Norway).

This document give a short overview of commercial software and their standard version. There is customisation of software to clients need (and economy).

Commercial rule engines are often banded under a wide variation of labels, where Business Rule Management System (BRMS) is general description.

Other formats presented in RRR can interact with commercial solutions in different ways. No commercial solutions cover the entire process from interpreting code to presentation of results. Commercial solutions is not an alternative, but a supplement for formats presented in this project.

Blaze Advisor BRMS

http://www.fico.com/en/products/fico-blaze-advisor-decision-rules-management-system

SMARTS Decision Manager / PENCIL Decision Modeller

http://www.sparklinglogic.com

Annex: Contributing authors

Initials	Name	Affiliation or Company	Email
EH	Eilif Hjelseth	Oslo and Akershus University College of Applied Sciences	Eilif.Hjelseth@hioa.no

Annex: Contributing authors

Initials	Name	Affiliation or Company	Email
EH	Eilif Hjelseth	Oslo and Akershus University College of Applied Sciences	Eilif.Hjelseth@hioa.no
JD	Johannes Dimyadi	University of Auckland	j.dimyadi@auckland.ac.nz
NN	Nicholas Nisbet	buildingSMART UKI and AEC3 and Leeds Beckett University	nn@aec3.com
NH	Dr Nicholas Humes	InForm Architecture Ltd.	nick@InFormArch.co.uk

5 Executive summary and conclusions

Automation of Regulation, Requirements and Recommendations (RRR)

This chapter contains both an executive summary and the conclusions reached following the preparation of the main sections of the report. Section 5.3 gives an overview of the current landscape and Section 5.4 makes specific recommendations.

5.1 Background

The BuildingSMART Regulatory Room is the open forum for discussions set up by BuildingSMART International. It hosts two series of meetings a year, and numerous teleconferences in between. The Room established the Working Group on Interoperability in March 2016, to look at interoperability for Regulations, Requirements and Recommendations (RRR). This report is the outcome of the series of discussions and teleconferences held between June 2016 and March 2017.

Many individuals have contributed their knowledge, time and goodwill in the RRR interoperability group which has been meeting twice weekly for almost twelve months. The contributions have included the preparation of the test models, their translation into a variety of formats, detailed implementation of the test case rules and the preparation of presentations and contributions to the online web-conferences.

5.2 Scope

The intention of the investigation has been to identify the prerequisites for a thriving commercial and technical market, making RRR checking scalable and trustable. The scope of the inquiry spanned from RRR as a research topic, through to the discovery of what the current best capability and practicality may be. Chapters of the report cover the business case for RRR interoperability, and the resources needed to back up a RRR market, where the applications are generic and international but configured by national and regional and project-specific RRR resources.

The separate sections of chapter 4 document the various 'candidates' considered. Each 'candidate' may be variously comprised of:

- an approach to encoding,
- a format or representation,
- one or more interpretative engines

We were able to go into some depth of the technical capabilities and to challenge some of the candidate formats with a realistic example.

Some questions, such as performance of individual rule engines, could not be fairly investigated. Since the manual alternatives to RRR typically take days and weeks, this was not taken as significant factor.

We addressed the option of procedural coding (the manual authoring of conventional code) as an alternative to rule interoperability (though we did not examine specific hard coded solutions such as Solibri or ePlanCheck. We also examined configurable commercial solutions such as used in some aspects of the financial sector, which ultimately came down to representing RRR interoperability as simple tables. Chapter 4 has sections on 'procedural coding' and on 'simple tables' which discuss the risks associated with these approaches.

5.3 Commentary

5.3.1 Working method

The investigation started with an open agenda, seeking nomination of tools and formats that seemed relevant: experts were sought out and invited to contribute. Initial documents were prepared including chapters 1, 2, and 3 of the report. Chapter 3 evolved to document the common criteria, and the choice of an example regulation and the two test buildings.

A shared working environment allowed all participants to read and update the report sections. Each chapter and candidate section was also given a sub-directory for examples and supporting material.

Web-conferences were held twice weekly to engage both hemispheres, and over 30 contributors attended at various stages.

5.3.2 Capability

Towards the end of the project, each candidate was assessed systematically on 15 criteria. Each criterion was answered on a five-point scale. The intention was primarily to validate the criteria and check the groups understanding of the material. No attempt was made to create any overall weighting of the criteria or scoring of the answers or ranking of the candidates. However, reviewing these criteria and tables, there were five that proved particularly problematic.

Technical issues:

1. Whilst there are some interoperability conversion routes between candidates, the external representation of rules has generated a wide variety of languages and representations, many with very subtle use of structure and punctuation. In the time availbale, it was only possible in a few cases to show that conversions should be possible. The remaining uncertainty should be reduced by more systematic research and development.

2. Not all candidate tools tolerate 'unknowns', technically called the 'open world' assumption. In most realistic scenarian, RRR rules will be used in an environment were the scope of relevant data is dynamic, depending on other data, and some aspects of the relevant data may stay irrelevant or unknown. Rules systems that require total knowledge at the outset ('closed' world') are unlikely to support realistic commercial implementations.

3. Some academic and technical implementations had taken regulations as a sequence of direct requirements, each of which was to be met. In such cases each requirement is therefore implicitly linked by 'and', or individual requirements are tested separately. Earlier work, such as the Singapore ePlanCheck and the US ICC SmartCodes had shown that actual codes will contain complex logical operators, with 'and' being used alongside 'or' for alternative methods of satisfaction, and 'not-and' and 'not-or' arising where differing applicabilities and selections are involved. Candidates may need to show that they can represent these operations without recourse to duplication or repetition.

4. Amongst the candidate formats and tools, there was very little capability to represent or track uncertainty. Recomondendations, such as found in guidance and advisory material) and to a lesser extent requirements are rarely as absolute as regulations. The uncertainty may arise from incomplete or approximate information in the project proposals, or in the expression of uncertain knowledge. Schemes like BREEAM and SEQUEL feed a point scheme which introduces a rating and ranking scheme, and the regBIM project has shown how these can be incorporated. More work may be needed to extend or expand the interpretation of uncertainty.

5. Some checks depend on complex geometric calculations. For example the test regulation focussed on escape distances within a factory complex. Rather than only admit candidates that had such geoemtric capability built-in, an assumption was made that the results of such analysis would be found within a proposal model, which might be challenged, so that the detailed calculations would be treated as secondary, supporting evidence. This is similar to how other considerations, such as the possession of an exemption certificate, would be handled. This supports the view that RRR compliance is a first-and-formost a management process.

5.3.3 Layering

As the candidates were discussed it became clear that both BIM and RRR interoperability can be best seen as a layered or sandwich problem. These layers mediate the original RRR document, with, at the centre, an engine capable of consuming a RRR representation and a BIM representation, which is derived from the original BIM model.

RRR Candidates can be layered into

- Inspection, Mark-up and NLP
- Authored languages
- Executable languages and engines

Note: NLP is included because of its tools to mark-up up the grammatical role of words and relate them to external dictionaries. NLP has been less effective in deducing the meaning of RRR material.

Few candidates spanned all the layers.

Similarly, access to BIM information can be layered:

- BIM authoring tools
- Export formats
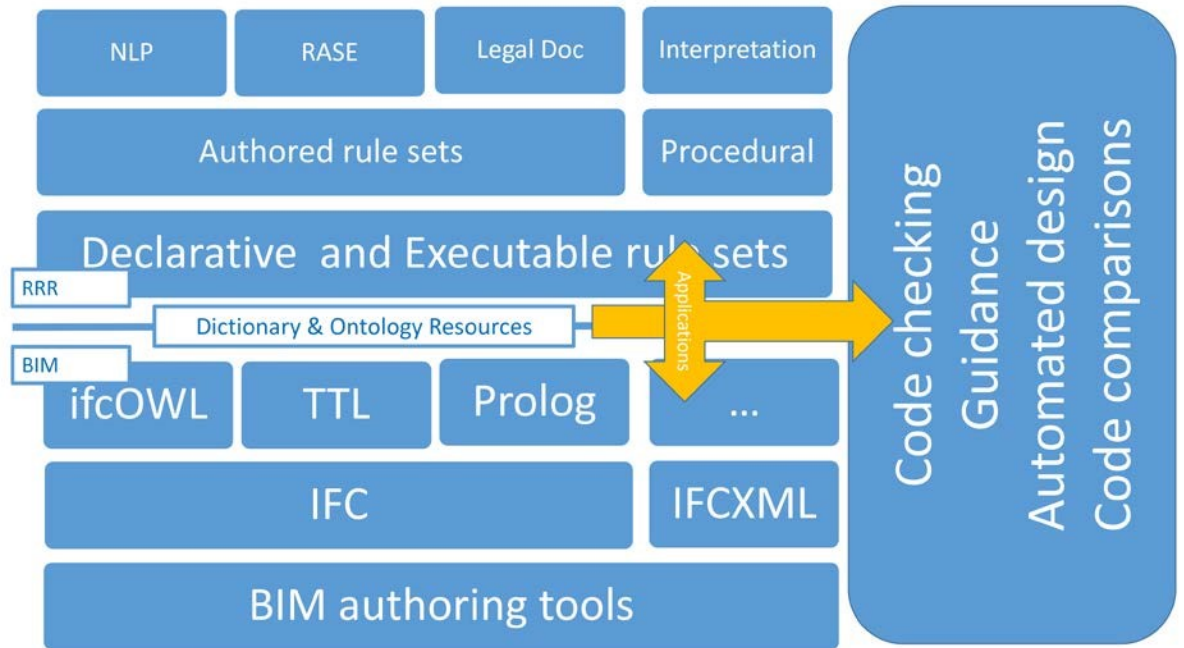- Information representations

Figure 24: RRR and BIM layers meet for compliance applications

5.4 Recommendations

The recommendations here reflect the lessons learned in preparing the main report and particularly the development of chapter 4 on the candidates. They are addressed to the BuildingSMART community, the broader facilities services sector, and government, clients, insurers and other authors of RRR material.

In summary, to create an environment to support interoperable RRR compliance requires two key activities:

No single candidate spanned all three layers completely, nor met all the technical expectations.

A. The candidate resources need to be extended, and researchers are invited to address the technical issues 1-5 above.

The report found that beyond the mark-up layer, it was not possible to identify or rank the candidates, although several showed considerable capabilities in representing the test example and several had been used successfully in academic and pilot implementations.

B. The Mark-up layer offers the best opportunity for common standards that will create credibility and scalabliity for a full market for solutions. Attention shoud focus on the necessary enablers which are in the hands of the standards community and RRR authors.

The following Actions are required to implement the recommendations.

5.5.1 Affecting the buildingSMART community

buildingSMART has a role in promoting standards for both the demand and supply sides of the facilities industry.

5.5.1.1 The BuildingSMART Regulatory room should:

1. Record appreciation for the efforts of the worldwide community for their input into this study.

2. Respond to and update this report on an annual basis, and invite developments to be showcased at the BuildingSMART bi-annual Summits, *based on a repository of example challenges such as the factory. so as to document increased progress and understanding.*

3. Publish a functional requirements document for a generic RRR management tool, *based on the business case and processes documented in sections 1 & 2 to guide the commercial developments*

4. Prepare a concise specification as buildingSMART model view definition and property sets for the initial attributing of facility models (application date, jurisdiction, application type, and facility type) *so as to provide a common basis for the initial checks in RRR compliance checking against building and infrastructure projects*

5.5.1.2 The BuildingSMART Standards Committee should:

5. Adopt as a 'Technical Report' of this report and in particular the recommendations and actions recorded here (A, B, and 1-17), *so as to give formal status within buildingSMART to the outcomes documented here.*

6. Formalise an integration of legalDocML, RASE and other Natural Language Processing results *as a necessary prerequisite for the extension of existing and new candidates and applications.*

5.5.1.3 BuildingSMART International should:

7. Include RRR as a key digitisation objective for the facilities sector <u>demand side</u>, alongside the facilities sector <u>supply side</u>, *so as to promote a complete 'systems engineering' view of the facilities sector including building and infrastructure projects*

8. Encourage the adoption of the buildingSMART Regulatory Room Memorandum by RRR bodies alongside the buildingSMART Building Room Memorandum by Facility clients.

5.5.2 Affecting the AEC RRR research and commercial communities

5.5.2.1 The AEC RRR research community should:

9. Investigate the technical issues 1-5 above and propose solutions based on extending the mark-up layer, the rule representations and engines able to process them, *so as to address the actual and perceived gaps in capability*

5.5.2.2 The AEC RRR commercial community should:

10. Note the growing commercial potential for rules-based checking and rule-driven design in the facilities sector and prepare solutions to meet this need.

11. Note the importance of interoperability in both the model and rule aspects and support the emerging standards.

5.5.3 Affecting RRR authoring community, including Government and near-Government regulators and client and advisory bodies

Some jurisdictions make access to the RRR source text difficult. *This is counter-productive and is rated a 0-star approach.*

5.5.3.1 The AEC RRR authoring community should:

12. Ensure the source text of RRR documents and supporting material is available in digital (not digitised, scanned or frozen) formats. *This warrants 1-star, ensuring research and development can engage with the RRR documents.*

13. Provide, as a minimum, RRR documents with mark-up using legalDocML of Date, Jurisdiction and Para numbering. *This warrants 2-stars, ensuring that solutions an establish relevance quickly and accurately.*

14. Publish their documents with mark-up (legalDocML, RASE and NLP), or adopt licencing policies that allow third parties to include mark-up, within publishing formats and exploit the resulting efficiencies. *This warrants 3-stars, giving a common base for development and commercial solutions.*

15. Provide resources that document national and regional Building Type, Application Type, and Jurisdiction Type allowed values and definitions. *This warrants 4-stars, ensuring that project proposals can be matched to RRR documents quickly and accurately.*

16. Adopt style guidelines for future RRR documents that clearly and explicitly

      a. separates normative material from definitions and illustrations

      b. distinguishes distinct testable metrics and objectives as Requirements, Applicability, Selection or Exceptions.

      c. clarifies acceptable primary evidence and typical examples of secondary evidence

This warrants 5-stars, as it promotes well-structured RRR material that can be reviewed and implemented fully and transparently.

Annex: Contributing authors

Initials	Name	Affiliation or Company	Email
NN	Nicholas Nisbet	buildingSMART UKI and AEC3 and Leeds Beckett University	contact@buildingsmart.org

Annex 1: Tabulation of candidates and some key metrics

Strategic assessment of format	Guidance and explanation	Ifc Constraint Schema	Procedural Code Generation	Rule Table	BIM RL	RASE	Legal DocML	LegalRuleML	DROOLS	SWRL	SPARQL	Jena RL	NLP	Prolog	Commercial Rule Engine
1 Control, ownership, availability	How available is it for use now?	Open public standard	Published	Published	Published	Published	Open public standard	Published	Well published	Published	Open public standard	Published	Published	Published	Published
2 age, and stability,	How stable is it for use now?	Well established	Evolving	Evolving	Evolving	Evolving	Established	Established	Established	Established	Well established	Evolving	Evolving	Evolving	Evolving
3 suitability for multiple purposes	How multi-purpose is it (see list below)?	Few purposes	Many purposes	Many purposes	Many purposes	Many purposes	All purposes	Most purposes	Many purposes	Many purposes	Many purposes	Many purposes	Many purposes	Many purposes	Many purposes
	•Authoring or mark-up of codes	Not supported	Not supported	Not supported	Not supported	Not supported	Very well supported	Very well supported	Not supported	Not supported	Supported	Not supported	Not supported	Not supported	Not supported
	•Computer-assisted and Automated code compliance checking	Very well supported	Very well supported	Very well supported	Very well supported	Very well supported	Very well supported	Very well supported	Very well supported	Very well supported	Well supported	Very well supported	Very well supported	Very well supported	Very well supported
	•Analysis and code comparisons	Poorly supported	Poorly supported	Poorly supported	Poorly supported	Poorly supported	Very well supported	Very well supported	Poorly supported	Poorly supported	Poorly supported	Poorly supported	Poorly supported	Poorly supported	Poorly supported
	•Filtering and structured dialogues	Poorly supported	Supported	Supported	Supported	Supported	Well supported	Well supported	Supported	Supported	Supported	Supported	Supported	Supported	Supported

		Ifc Constraint Schema	Procedural Code Generation	Rule Table	BIM RL	RASE	LegalDocML	LegalRuleML	DROOLS	SWRL	SPARQL	Jena RL	NLP	Prolog	Commercial Rule Engine
	•Controlling parametric objects	*Supported*	*Poorly supported*	*Poorly supported*	*Supported*	*Poorly supported*	*Very well supported*	*Supported*	*Supported*	*Supported*	*Well supported*	*Poorly supported*	*Poorly supported*	*Poorly supported*	*Poorly supported*

Commercial base	Guidance and explanation	Ifc Constraint Schema	Procedural Code Generation	Rule Table	BIM RL	RASE	LegalDocML	LegalRuleML	DROOLS	SWRL	SPARQL	Jena RL	NLP	Prolog	Commercial Rule Engine
1 Authoring /capture tools (singular, multiple)	How many creative tools are there?	*Two tools*	*Two tools*	*Two tools*	*Two tools*	*Two tools*	*Few tools*	*Two tools*	*Multiple authoring and capture tools*	*Multiple authoring and capture tools*	*Two tools*	*Two tools*	*Two tools*	*Two tools*	*Two tools*
2 Assessment engines, software and toolkits for (singular, multiple)	How many execution engines are there?	*Few engines*	*Two engines*	*Two engines*	*Two engines*	*Two engines*	*Few engines*	*Few engines*	*One engine*	*One engine*	*Two engines*	*Two engines*	*Two engines*	*Two engines*	*Two engines*
3 Scalability and role in creating a market place for RRR applications	How much overhead in setting up?	*Very low overhead*	*Moderate overhead*	*Moderate overhead*	*Moderate overhead*	*Moderate overhead*	*Moderate overhead*	*Moderate overhead*	*Moderate overhead*	*Moderate overhead*	*Moderate overhead*	*Moderate overhead*	*Moderate overhead*	*Moderate overhead*	*Moderate overhead*

Layering	Guidance and explanation	Ifc Constraint Schema	Procedural Code Generation	Rule Table	BIM RL	RASE	LegalDocML	LegalRuleML	DROOLS	SWRL	SPARQL	Jena RL	NLP	Prolog	Commercial Rule Engine
Supports mark-up layer	Does it help in the mark-up layer?	*Very good mark-up*	*Mark-up*	*Mark-up*	*Mark-up*	*Mark-up*	*Very good mark-up*	*Mark-up*	*No mark-up*	*No mark-up*	*No mark-up*	*Mark-up*	*Mark-up*	*Mark-up*	*Mark-up*

		Ifc Constraint Schema	Procedural Code Generation	Rule Table	BIM RL	RASE	Legal DocML	Legal RuleML	DROOLS	SWRL	SPARQL	Jena RL	NLP	Prolog	Commercial Rule Engine
Support authoring layer	Does it help in the authoring layer?	*Authoring*	*Authoring*	*Authoring*	*Authoring*	*Authoring*	*Very good authoring*	*Authoring*	*No authoring*	*No authoring*	*Authoring*	*Authoring*	*Authoring*	*Authoring*	*Authoring*
Support execution layer	Does it help in the execution layer?	*Execution*	*Execution*	*Execution*	*Execution*	*Execution*	*Good execution*	*Execution*	*Very good execution*	*Very good execution*	*Good execution*	*Execution*	*Execution*	*Execution*	*Execution*

Technical	Guidance and explanation	Ifc Constraint Schema	Procedural Code Generation	Rule Table	BIM RL	RASE	Legal DocML	Legal RuleML	DROOLS	SWRL	SPARQL	Jena RL	NLP	Prolog	Commercial Rule Engine
1 Expressivity and generality	How expressive is it?	*Very complex logical structures*	*Complex logical structures*	*Complex logical structures*	*Complex logical structures*	*Complex logical structures*	*Some complex structures*	*Complex logical structures*	*Complex logical structures*	*Complex logical structures*	*Complex logical structures*	*Complex logical structures*	*Complex logical structures*	*Complex logical structures*	*Complex logical structures*
Support for AND/OR/NOT AND/NOT OR	For example, does it efficiently support these?	*All*	*Some*	*Some*	*All*	*Some*	*All*	*Most*	*All*	*Some*	*All*	*Some*	*Some*	*Some*	*Some*
Support for TRUE/UNKNOWN/FALSE	For example, does it efficiently support these?	*Three*	*Two-ish*	*Two-ish*	*Two-ish*	*Two-ish*	*Two-three*	*Two-ish*	*Three*	*Nearly three*	*Nearly three*	*Two-ish*	*Two-ish*	*Two-ish*	*Two-ish*
2 Acceptability and provenance	How clearly linked to the source	*Moderately close and revie*	*Moderately close and reviewable*	*Moderately close and revie*	*Moderately close and revie*	*Moderately close and revie*	*Close to text, easily reviewed*	*Moderately close and reviewable*	*Moderately close and revie*	*Far from text, hard to review*	*Fairly close to text and revie*	*Moderately close and revie*	*Moderately close and revie*	*Moderately close and revie*	*Moderately close and reviewable*

		Ifc Constraint Schema	Procedural Code Generation	Rule Table	BIMRL	RASE	LegalDocML	LegalRuleML	DROOLS	SWRL	SPARQL	JenaRL	NLP	Prolog	Commercial Rule Engine
	document is it?	wable		wable	wable	wable			wable		wable	wable	wable	wable	
Reasoning trail and evidence	How well does it evidence its reasoning?	Moderate links to reasoning or source	Moderate links to reasoning or source	Moderate links to reasoning or source	Moderate links to reasoning or source	Moderate links to reasoning or source	Good links to reasoning an source	Moderate links to reasoning or source	Moderate links to reasoning or source	Moderate links to reasoning or source	Moderate links to reasoning or source	Moderate links to reasoning or source	Moderate links to reasoning or source	Moderate links to reasoning or source	Moderate links to reasoning or source
Model trail and evidence	How well does it evidence its data?	Moderate links to project and evidence	Moderate links to project and evidence	Moderate links to project and evidence	Moderate links to project and evidence	Moderate links to project and evidence	Good links to project and evidence	Moderate links to project and evidence	Moderate links to project and evidence	Moderate links to project and evidence	Good to moderate links to project and evidence	Moderate links to project and evidence	Moderate links to project and evidence	Moderate links to project and evidence	Moderate links to project and evidence

Recommendations and point schemes	Guidance and explanation	Ifc Constraint Schema	Procedural Code Generation	Rule Table	BIMRL	RASE	LegalDocML	LegalRuleML	DROOLS	SWRL	SPARQL	JenaRL	NLP	Prolog	Commercial Rule Engine
Uncertainly in values	Could it handle uncertainty in values?	Feasible	Perhaps	Perhaps	Feasible	Perhaps	Perhaps	Feasible	Very feasible	Perhaps	Perhaps	Perhaps	Perhaps	Perhaps	Perhaps
Uncertainty in comparisons	Could it handle uncertainty in comparisons?	Perhaps	Perhaps	Perhaps	Feasible	Perhaps	Perhaps	Feasible	Perhaps	Perhaps	Perhaps	Perhaps	Perhaps	Perhaps	Perhaps
Uncertainty in logical connections	Could it handle uncertainty in logical connections?	Infeasible	Perhaps	Perhaps	Feasible	Perhaps	Perhaps	Feasible	Perhaps	Perhaps	Perhaps	Perhaps	Perhaps	Perhaps	Perhaps

Interoperability	Guidance and expla	Ifc Constraint Schema	Procedural Code Generation	Rule Table	BIMRL	RASE	Legal DocML	Legal RuleML	DROOLS	SWRL	SPARQL	Jena RL	NLP	Prolog	Commercial Rule

	natio n														Engin e
Convertible to other formats	Is it in a regular or xml format?	*Easily converted from*	*Perhaps convertible from*	*Perhaps convertible from*	*Perhaps convertible from*	*Perhaps convertible from*	*Easily converted from*	*Easily converted from*	*Convertible from*	*Perhaps convertible from*	*Perhaps convertible from*	*Perhaps convertible from*	*Perhaps convertible from*	*Perhaps convertible from*	*Perhaps convertible from*
Convertible from other formats	Is it in a well-structured layout?	*Perhaps convertible into*	*Perhaps convertible into*	*Perhaps convertible into*	*Perhaps convertible into*	*Perhaps convertible into*	*Easily converted into*	*Convertible into*	*Convertible into*	*Perhaps convertible into*	*Perhaps convertible into*	*Perhaps convertible into*	*Perhaps convertible into*	*Perhaps convertible into*	*Perhaps convertible into*

Annex 2: Contributors

Tom Beach

Jakob Beetz

Lars Bjorkhaug

Peter Bonsma

Jungskik Choi

Goncal Costa

Tan Dan

Johannes Dimyadi

Chuck Eastman

Nora El-Gohary

Guido Governatori

Eilif Hjelseth

Nick Humes

Karam Kim

Henning Kongsgard

Yongcheol Lee

Sibel Macit

Leonardo Manzione

Sylvain Marie

Lewis John Mcgibbney

Peter Muigg

Nick Nisbet

Pieter Pauwels

Farzad Pour Rahimian

Ana-Maria Roxin

Odilo Schoch

Wawan Solihin

Rasso Steinmann

Walter Terkaj

Bob Wakelam

Chi Zhang

Jiansong Zhang