2022 EC3
2022 European Conference on Computing in Construction
Ixia, Rhodes, Greece
July 24-26, 2022

PRESENTATIONS OF RASE KNOWLEDGE MARK-UP

.

Nicholas Nisbet[1 2]
Dr Ling Ma[1]
Dr Gulnaz Aksenova[1]
[1]UCL London n.nisbet@ucl.ac.uk, [2]AEC3 nn@aec3.com

Abstract

RASE is a mark-up method allowing subject matter experts to expose the implied logic within the documents. RASE is now a formally defined mark-up of documents including text and tables compatible with W3C HTML standards. The mark-up renders normative, definitive and descriptive text documents machine-interpretable. This paper extends the understanding of RASE and classifies the reasons and means to translate RASE into various derived presentations. A generic iterative method responds to events encountered within the mark-up structure to generate presentations used in formal logic, diagramming and other technical representations, so as to expose and explain the logical and operational structure.

Introduction

Normative knowledge, describing how a domain is required or expected to be manifested, is a significant category of knowledge alongside definitive knowledge and descriptive/narrative knowledge. These three kinds of knowledge can be structured and analysed, either separately or they can be brought together to effect automated compliance checking. RASE embeds in the source material sufficient extra content to render the document machine-readable whilst maintaining human-readability. The addition of RASE to a document is ideally carried out by the subject matter experts, or the original authors, who are best placed to resolve any ambiguity of language. Domain experts find the process natural and informative, allowing them to capture once-off their tacit knowledge and potentially to improve their writing and interpretation skills. Projects such as SMARTcodes [Nisbet et al, 2008] and regBIM [Beach et al, 2013] have been completed successfully. Projects under the DCOM initiative [DCOM, 2021] are reviewing various UK guidance documents relating to fire, energy and accessibility using RASE. The outcomes should be reported next year.

Ontology, schema, representation and presentation

The RASE approach (figure 1) addresses knowledge capture by exposing within the source text a simple 'tree' ontology of objectives containing other objectives and/or metrics [Nisbet et al, 2008]. If a metric or objective occurs more than once, then the structure may transition from being a strict tree to being a directed graph. At a schema level, RASE adds an optional identifier on the metrics and objectives, which can be invaluable in linking code compliance checking results or detailed trace messages back to the exact source text. The RASE schema [AEC3, 2021] identifies four types of objectives and metrics, 'Requirement', 'Applicability', 'Selection' and 'Exception', identified by the four-letter acronym 'RASE'. Objectives and Applicability, Selection and Exceptions metrics are common in all three kinds of knowledge, but Requirement metrics may be normative ('shall be') , definitive ('is defined as') or descriptive/narrative ('is reported as'). A metric is a simple property expression, but it may also have comparator, target and unit attributes identified separately. RASE is usually represented as a mark-up added to the text but can be used as metadata added to the fields of structured tables.

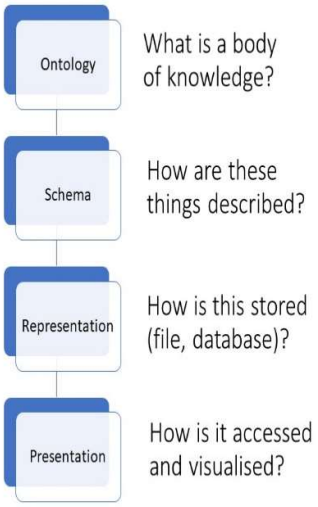

Figure 1: Presentations in context

Documents with RASE mark-up can be interpreted directly by an algorithm, consuming the source knowledge directly, as implemented in AEC3 Require1 [AEC3, 2022]. All rules in each normative body of

knowledge are applied to all spatial, physical and process objects and facts in the descriptive/narrative body of knowledge. Metrics are evaluated by reference to the target description knowledge. Objectives are evaluated by reference to their logical operator and the logical value of their constituent objectives and metrics. In most cases, the rule-object pair is quickly resolved by considerations of subject relevance, before the predicate expectation are considered. A document with RASE mark-up can be consumed directly by an application or 'rule-engine' that supports three constructs:

- Tracking the state of any metric or objective using 3-value (true, unknown, false) logic. This is in preference to 2-value Boolean logic (true, false) which is less effective at handling the uncertainties and missing information that characterizes regulatory compliance checking and real-world applications in general. Metrics that are evaluated as 'unknown' should not be defaulted to either Boolean value as this can cause incorrect outcomes.
- Using a suite of four logical operators, (AND, OR, NOTAND, NOTOR), associated with the four RASE roles, in preference to the AND/OR/NOT paradigm which is less compatible with normal language and grammar. In some cases, NOTAND and NOTOR can be implemented internally using the traditional AND/OR/NOT paradigm, but this complication is best not made visible to the user.
- Using variadic (open list) implementations of these logical operators, as any number of metrics and/or objectives may be encountered within the text. This is in preference to using any in-line binary operators. In some programming languages, variadic logical operators can be overloaded to implement optimization heuristics, such as early resolution of a logical operator without necessarily evaluating all the terms.

For these reasons and for efficiency, it will often be preferable to use this direct consumption of documents with RASE mark-up. However, it may be necessary to present the knowledge embedded in a RASE document in other forms. This paper explores the relationship between RASE and other knowledge representation by reviewing some of these as alternative presentations of RASE mark-up. This challenge is potentially open-ended, so this paper will categorize the alternative presentations by purpose and show that there is substantial coverage within each. If successful, then other knowledge representations can be expected to be derivable presentations of RASE.

Method

The method used for these exercises was to explore the solution space by iterative development building on the first presentations of RASE. The earliest exercises were undertaken in 2007-2008 and these have subsequently been redeveloped and extended as the RASE mark-up was formalized and simplified. Each presentation was validated against the published syntax and examples of the target representation. The visual presentations were further developed to organise the diagrams with the most compact layout, shortest connectors and the least crossovers. Operable representations were validated against existing 'rule-engines'.

Algorithm development

The RASE mark-up uses 'boxing' of text content to represent the definition of objectives which can contain other objectives, and metrics, represented by underlining of text content. This definition implies a recursive tree (branch-and-leaf) structure, embedded in the conventional HTML mark-up of document structure and text-decoration. An algorithm to expose this ontology therefore follows one of the mathematically well-defined tree-traversal algorithms. Since only metrics can be evaluated and their evaluated logical value then affects the objectives above, a depth-first search (DFS) approach was selected. Using a 'left-right-node' (LRN) sequence effectively generates an expression tree. The transformation uses a single tree-iteration algorithm, with specific 'events' detected and the appropriate output specific to the desired presentation is generated. The output in response to each event may be an indentation, a textual marker, a symbol or an arrow. The following events are detected:

a. Start: generate any header material and start the recursive exploration of the mark-up
b. Before or after the first, 'i'-th, or last
    a. objective or metric: document the appropriate output including any position-dependant syntax
    b. objective or metric of one of the four types: document the appropriate logical operator and any position-dependant output
c. Drop down from an objective: start a recursive exploration
d. Lift up to an objective: end a recursive exploration
e. End: end the recursive exploration of the mark-up and generate any footer material

A number of different programming approaches could have been used. RASE content is embedded in HTML [W3C, 2019] and all the presentations in the following experiments have a text-based representation, using HTML, XML, SVG or plain text. HTML is compatible

with XML [W3C, 2008] conventions. This allows XSLT [W3C, 1999] transformations to be used to effect the generation of the desired presentations. Any necessary translation of the output into the binary executable is left to the relevant programming compiler.

In many of these presentations, the child nodes of each objective have been sorted in the order A/S/E/R. The use of sorting and the selection of A/S/E/R as the preferred order is arbitrary but reflects a common sequence found in documents, though some legal styles of writing may place Exceptions first. Applicability, Selections and Exceptions together define the grammatical subject and are sorted to appear first within their objective so as to highlight them as pre-requisites. Requirements as the grammatical predicate are sorted to appear last. A second reason for sorting Requirements to appear last is that definitive Requirements, such as dictionary, thesaurus and classification knowledge, are required to be accepted as true so if any are present then they should be sorted to be considered last so as to create new knowledge entries only if all other alternatives such as Not Applicable, Not Selected, or Excepted have been explored.

Experiments

Knowledge representations have been developed to serve a number of different purposes. Five such purposes are discussed in the following sections.

1. Direct inspection.
2. Review and use as checklists.
3. Consumption by other rule engines.
4. Export to interoperable formats.
5. Formal knowledge representations.

Purpose: Direct inspection

Presenting the RASE mark-up as visual decoration on the original text and tables supports review and development. This avoids the executable ruleset being a separately authored artefact. The generation of presentations to support review by domain experts is intended to re-enforce the credibility of the RASE methodology by offering other readable forms. Credibility may be generated if the original text, the RASE mark-up and the domain experts' view are recognized as being aligned. This supports a validation and verification process that aims to recognize the metrics identified in the text as being accurately captured and relatable to the domain. It also aims to recognize the four RASE roles identified for the metrics and objectives.

The simplest example is the application of an embedded or external CSS presentation stylesheet to make the RASE mark-up visible. Figure 2 shows an artificial example of a single check used in the subsequent presentations. A set of presentation conventions have been embedded in a 'cascading stylesheet' CSS [W3C, 2007] 'rase.css' [AEC3, 2021] so that any web page editor or browser can be used to review the RASE document. This stylesheet contains rules for annotating the document with boxing and underlining in four colours and responding to 'mouse-over' events to show any identifier and metric parameters. Requirements are shown in solid blue, Applicability in dotted green, Selection in purple dashed and Exceptions in double underlined orange. Most of the following presentations use the same stylesheet conventions to build familiarity and legibility. SVG [W3C, 2011] graphics are used for graphic output to maintain legibility in different contexts.

Figure 2: Example RASE mark-up

Purpose: Review and use as checklists

Presenting normative content as checklists shows the content in a familiar form either as lists, decision trees or tables and shows that the RAE mark-up can support conventional manual checking processes using worksheets. Since the RASE ontology maps knowledge to a tree structure, any RASE document can be presented using the HTML unordered list <ul> tag. Language annotations can clarify the logical relationships at each branch in the tree, making the tree readable.

The tree structure can be summarized in a table giving the parent-child identifiers and any parameters on the metrics. This exposes the metrics that need to be responded to either by the target domain or by an intermediate dictionary knowledge resource.

Since any tree can be traversed depth-first or width-first, the RASE mark-up in the source document can be re-constructed into a continuous text. Such text will often be stilted, lacking fluency or variety, but can be used to engage domain experts who may be over-familiar with the original text.

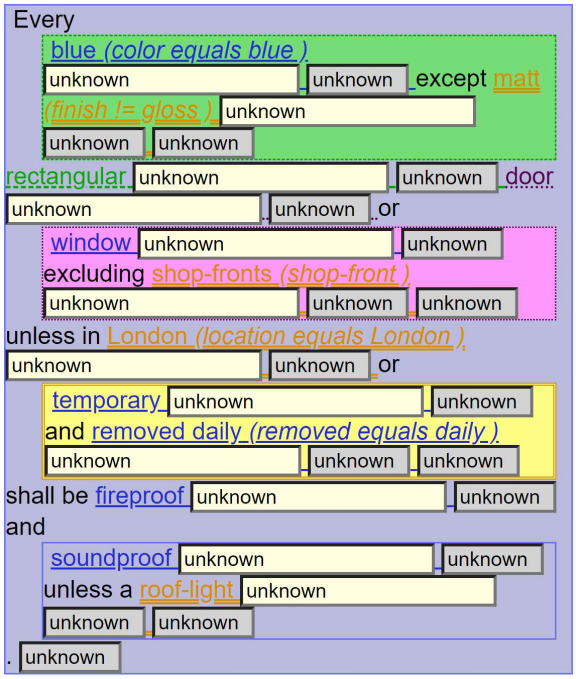

Figure 3: Checkable text awaiting values

Each of these presentations can be enhanced by annotating any metric with an interactive HTML text input widget, and each metric and objective with a passive HTML read-only text widget (figure 3). This converts an informative document into a dynamic checklist. Checklists are an important tool in manual checking practice. These presentations ensure that the checklist is accurate and complete. The user can provide any known values for a single situation, for example, a room, door or accident. Each additional piece of information can cause some of the logical tree to be re-evaluated, filling in the logical (true/unknown/false) state of each metric and objective until an overall outcome is obtained. Additional coding embedded in the document can hide any sub-tree that is fully resolved. This feature fulfils the ambition to have self-filtering regulations that are automatically tailored to the user's context. For example, given the specific building type, large swathes of the building control documents can be suppressed from view.

Purpose: Consumption by other rule engines

Presenting normative content in forms that can be consumed by existing rule engines serves to confirm that RASE is compatible with existing rule infrastructure,

Presentations may be prepared for consumption by existing rule engines. This can contribute to research and exploitation efforts that are already using other rule methodologies, rule engines or geometric engines. Existing rule engines may come with fully developed support structures such as user interfaces and reporting. Existing rule engines may be taken as robust and so may not need further validation. However, these may not support the three algorithm requirements and so workarounds may need to be introduced. By accepting an off-the-shelf rule engine, the opportunity to operate some heuristics may be not available.

The regBIM project [Beach et al, 2013] has used transformations of the RASE markup into the Java DROOLS [DROOLS, 2021] syntax. Since DROOLS operates within a conventional Java programming environment, additional methods can be included, for example, the methods for accessing the target domain such as IFC [ISO, 2018] BIM models. Side-effects, such as actual and maximum BREEAM points can be tracked by additional coding. DROOLS by default uses two-value logic and so guard clauses must be generated to protect the rule statements from metrics or objectives that are as yet unknown. In the current UK DCOM project [DCOM, 2021] asynchronous processing is handled so that calls to analysis and simulation servers can be used.

Datalex [Datalex, 2021] is a simple text-based rule language developed for legal support that can join simple metrics together in a logical hierarchy (figure 4). It has constructs to allow definitional knowledge and intermediate results to be accumulated. There is an online interface that offers a controlled dialogue with the user, with explanations and conclusions being offered. Being interactive, it is suitable for the gathering and the analysis an individual set of facts, rather than a large body of cases, each having a set of facts associated.

```
DEFAULT REGULATION Training
GOAL RULE Training PROVIDES  Training ONLY IF

            BEGIN
                        BEGIN
            BEGIN
                        NOT BEGIN
            BEGIN
                        BEGIN finish  gloss END
            AND/OR
                        BEGIN color  blue END
            END
            AND         rectangular   END
            AND/OR
                              NOT BEGIN door   AND/OR
            BEGIN
                        BEGIN shop-front   END
            AND/OR
                        BEGIN window   END
            END
            END
            AND/OR
                        BEGIN location  London AND/OR
            BEGIN
                        BEGIN temporary   AND removed  daily END
            END
            END
            AND/OR
                        BEGIN fireproof   AND
            BEGIN
                        BEGIN roof-light   END
            AND/OR
                        BEGIN soundproof   END
            END END END END END
```

Figure 4: Datalex presentation

DMN [DMN 2021] is a business rule execution language that can be integrated into business process engines. As such it should be of interest to the business management of code compliance services. However, DMN does not directly support the four logic operators nor three-value logic, or variadic functions, so this functionality has to be simulated by providing extensive truth tables using textual representations of ‘true’, ‘unknown’ and ‘false’.

These can be generated automatically but the resulting DMN is often very lengthy.

Many automated code compliance projects rely on the development of procedural code to cover both the overall logic and the execution of queries against the target domain body of knowledge. Whilst there are a wide variety of general-purpose and domain-specific programming environments, it has been possible to generate single procedural functions representing a normative body of knowledge, leaving the implementers to provide supporting functions for querying the target domain and developing the tracking and reporting capabilities. As a subset of this methodology, there are a variety of 'semi-procedural, semi-declarative' coding methods, such as Prolog, List and ASP These accept simple, two-part IFTTT (if-this-then-that) statements. Each metric or objective found in a normative body of knowledge can be expressed in this form, but only if the three language requirements can be met. The directional implication means that the embodied knowledge can only be exploited in one direction, limiting the use of logical differentiation and other heuristics

Purpose: Export to interoperable formats

Presentations can be used to export knowledge into existing interoperability formats. Each of the following can represent the tree (or directed graph).

The first implementation of RASE used the 'constraint' sub-schema of IFC 2x3 [ISO, 2018]. 'IfcMetric' can hold testable constraints either associated with other entities or as an open (un-associated) query. As found, the schema supported AND, OR, XOR and NOT. Subsequent revisions to IFC have added NOTAND and NOTOR. The benefit of presenting RASE knowledge as IFC is that existing engines such as Solibri and EPMJotne ModelServer (used in Singapore ePLanCheck and AEC3 XABIO) can already read IFC as a Step Physical File or as XML. This simplified the development of the demonstration implementations for the built environment by providing the import interfaces. Only the execution engine needed to be written, as a special add-in for Solibri and as a concise EXPRESS_X module for EPMJotne ModelServer.

KIF is an XML based knowledge representation that captures the structure of objectives and metrics, and the parameters for the individual metrics. This format has been generated but not validated.

Logic circuits using any of the four gates can be simplified because three of the four logical operators can be written in terms of NOTAND (or less commonly in terms of NOTOR). This simplification of the number of kinds of logical gates is useful in that it allows the logical structure to be presented as a wiring diagram, with the emphasis on the terminal sensors implementing the identified metrics (Figure 5). It is also of theoretical benefit, because it shows that any development of the NOTAND function, for example, to implement fuzzy logic or general uncertainty, automatically defines the expected behaviour of the other three kinds of gates. This can be used to link creative processes and uncertainty [Nisbet, 2021].

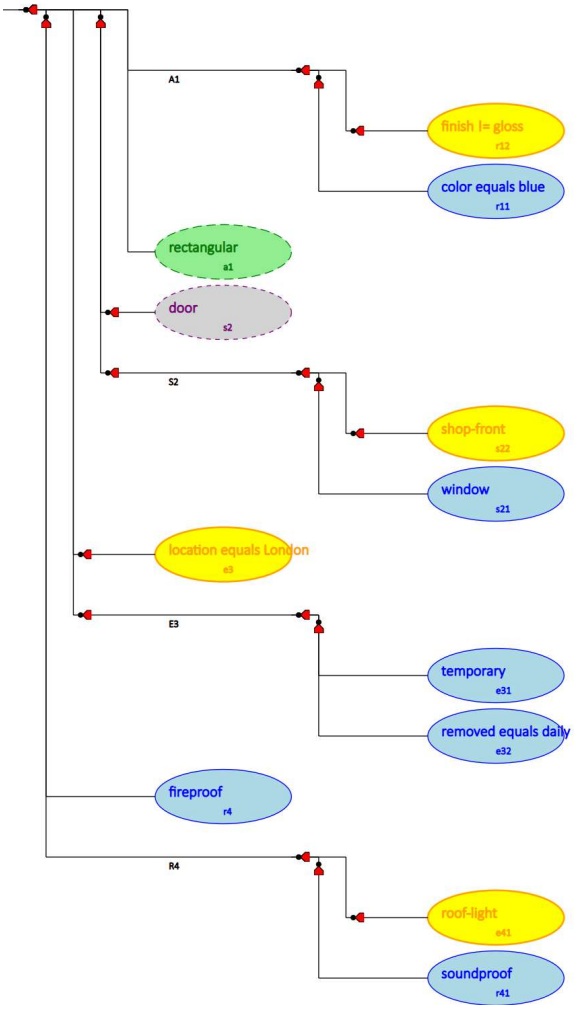

Figure 5: NOTAND logic gate presentation

Many BIM authoring platforms have introduced flavours of 'visual programming' which provide an extensible library of functional nodes that can be connected through ports representing arguments and outputs. Any RASE document can be presented and potentially implemented as a visual programming diagram using an objective node and a metric node.

Since the diagram (see figure 6) is generated automatically it is not subject to mistakes or obsolescence. Alternatively, it may be possible to create a RASE node that consumes a RASE document and interrogates the model entities without the intermediate step of developing a specific visual program.

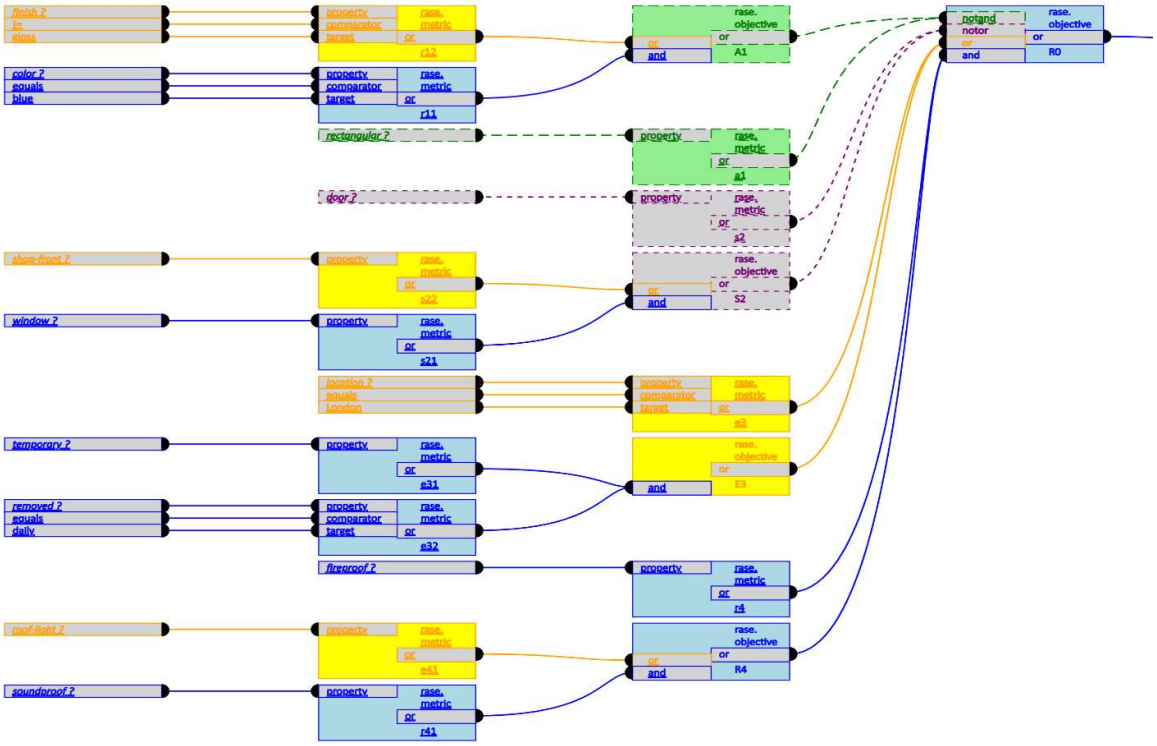

Figure 6: Visual programming presentation

Purpose: Formal knowledge representations

It has been possible to generate presentations to expose the relationship of RASE to existing knowledge representations. These do not necessarily have a direct application, but they may be important in positioning RASE as a competent approach.

An early review of RASE [Hjelseth et al, 2010] noted the analogy between the four RASE annotations with their four logical operators to Aristotle's 2x2 'Square of oppositions' quadrature. One difference is that Aristotle focuses on abstract generalities, whereas RASE focuses on the specific metrics of a piece of knowledge. The 'square' is one of three possible projections of a tetrahedral structure which each concept strongly to the other three. RASE takes the results of the four distinct operators and then accepts any of them as valid ways of passing, so a nested series of diagrams can be generated. Developing a readable presentation of this 'fractal' structure is currently in progress. In ISO 12911 [ISO, 2012] a recursive 1x4 box approach is used in the example annexes.

A RASE mark-up can be expressed as either Propositional or Predicate logic sentences. The exact form of this presentation may depend on whether logical functions are admitted. Such sentences should be acceptable to mathematical work benches or to theorem-proving tools.

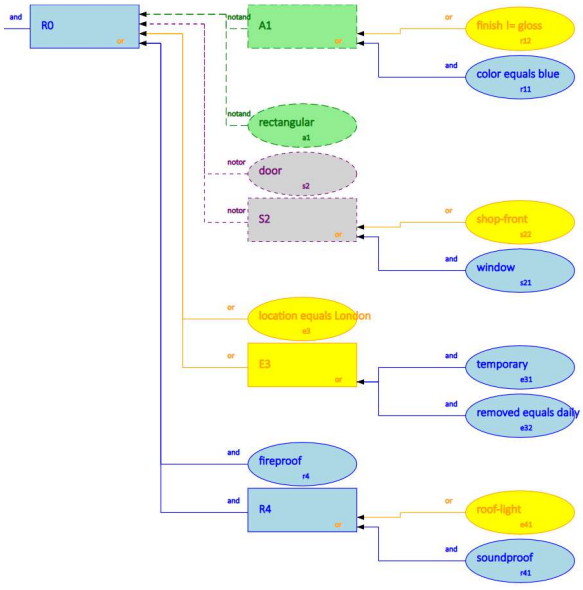

Figure 7: Concept graph presentation

The same early review also noted the relationship between RASE structure and 'concept graphs'. Solihan [Solihan et al, 2015] presents some manually generated concept graphs for the example regulatory clauses. The paper also introduced the need for logical annotation of 'OR' on some of the connectors, rather than the default assumption

of ‘AND’. The presentation of RASE takes this further and suggests that all four RASE annotations may be needed for the connectors (Figure 7).

Discussion and results analysis

The five different purposes for developing presentations are indicative of the challenges that a novel approach must overcome to generate credibility and acceptance. It must be subject to inspection, review, be compatible with existing tools, be interoperable and match formal definitions. These could have proved insurmountable but for the use of a common tree-iteration algorithm which has allowed several presentations to be generated from a consistent base. The identification of the small number of distinct ‘events’ during the iteration allows a variety of well presented, tabulated or verbalized presentations to be offered. The presentation as ‘conceptual graph’ has extended that state of the art by showing that logical operators are needed. New presentations can be developed rapidly if required. Each of the presentations discussed goes some way towards meeting these challenges.

Conclusions

RASE is a relatively novel conceptualization of knowledge and so has been necessary to demonstrate and illustrate its relationship to the variety of existing conventional knowledge representations. The examples discussed show that many of these knowledge representations are presentations of RASE that can be generated automatically. It also enhances the other representations by making them available automatically and without subjective interpretation or error. Different communities can review the knowledge embedded in documents using the presentations that they are most comfortable with, and code presentations can be bound into existing business frameworks. This strengthens the suggestion that RASE is a necessary and sufficient starting point for automated code compliance and other applications.

This review has shown that there are few difficulties in using RASE for a range of knowledge exploitation techniques. The availability of several alternative presentations can ease the burden of quality assurance and control in the development of automated code compliance applications. This suggests that RASE can be taken as the starting point for assurance, research and implementation,

However, this does not prove the converse, that any of these knowledge presentations can be mapped to RASE. This would need to be demonstrated by reconstructing a readable and marked-up text from these other knowledge resources. In particular, RASE mark-up may need further development and extension to match some constructs found in the predicate logic and conceptual graph syntax. This is the subject of continuing research.

References

AEC3 RASE standard 2021: http://www.aec3.eu/require1/Help_en-GB/help_en-GB_200.html .

AEC3 Require1 2022: http://www.aec3.eu/require1/Help_en-GB/help_en-GB_300.html .

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

Datalex Interface 2021: http://www.datalex.org/dev/import/.

DCOM Network 2021: https://www.dcom.org.uk/ .

DMN Standard 2021: https://www.omg.org/spec/DMN/ .

DROOLS toolkit 2021: https://drools.org/ .

E Hjelseth, N Nisbet 2010 “Exploring semantic based model checking” Proceedings of the 2010 27th CIB W78 international …, 2010

ISO/TS 12911:2012 Framework for building information modelling (BIM) guidance https://www.iso.org/standard/52155.html .

ISO 16739-1:2018 Industry Foundation Classes (IFC) for data sharing in the construction and facility management industries — Part 1: Data schema https://www.iso.org/standard/70303.html .

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

Nisbet, N. 2021 "Using uncertainty to link compliance and creativity" ECPPM 2021–eWork and eBusiness in Architecture, 2021

Solihan et al, 2015 "A Knowledge Representation Approach to Capturing BIM Based Rule Checking Requirements Using Conceptual Graph" CIB W78 2015 proceedings.

W3C CSS 2007: https://www.w3.org/Style/CSS/ .

W3C HTML 2019: https://www.w3.org/html/ .

W3C SVG 2011: https://www.w3.org/TR/SVG11/ .

W3C XML 2008 https://www.w3.org/XML/

W3C XSLT 1999 https://www.w3.org/Style/XSL/

(All web references checked on 2022-01-30.