Foundation for development of computable rules

E. Hjelseth
Norwegian University of Life Sciences (UMB), Dept. of Mathematical Sciences and Technology, Norway

ABSTRACT: This paper gives an overview of the foundation for development of computable rules that can be implemented into commercial rule-checking software. The foundation for development starts with an overview over knowledge engineering and its roots in logic, mathematic, linguistic and knowledge philosophy. This gives motivation investing in a semantic based knowledge system. Otherwise is likely that the BIM based rule-checker systems ends up as building information mess. The link from the theoretical foundation to tools and methods can be connected by use of semantic software tools. Preliminary results and experiences from an ongoing Norwegian project are presented.

1 OVERVIEW

1.1 *Introduction*

Rule checker software has achieved increased interest and is often regarded as one of the big benefits by using BIM / IFC based software in the design process (SmartMarket Report, 2009). Clash detection and check of doublets is one of the most used examples. This is of course good examples on the power of these software tools, but the rules are in principle very easy on purely based on Boolean expressions. These rules can also be implemented parametric, allowing the user to change the "rule" by changing the min / max values the components are checked against. Borrmann and Rank (2008) have used 3D Spatial Query Language on BIM models to check out relationships. The operator comprise of; *metric* (distance, closerThan, fartherThan etc.), *directional* (above, below, northOf etc.) and *topological* (touch, within, contains etc.) operators. Despite this can be complicated to perform, the rules itself are simple.

Other appropriate areas for rule checking is fulfillments of clients' demands from space program or performance (energy), fulfillments of public demands, industry dependent demands and design according to given standards etc. In this paper we choose to use standards as reference.

The AEC-industry – design and construction – is regulated by a large numbers of codes (laws, regulations, commands, recommendations, standards) form different departments and organizations, and with different goals. This indicates that "something" that can make this easier and more valid will be of great benefit.

So far has the main focus been put on the new technical possibilities and limitation, which can be solved in next version of software or IFC-schema. This innovation approach is not unusual, but the question is if the time also is mature to look at the fundamentals for developing rules.

The scope of this paper is to give an overview of the foundation for development of computable rules that can be implemented into commercial rule-checking software. This domain is illustrated by the "Scope" in figure 1. In addition we present some experiences and preliminary results from an ongoing Norwegian project about methods for development of computable rules from standard code.

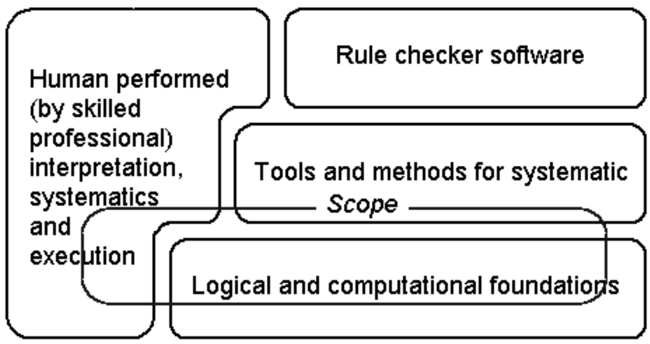

Figure 1 The 3-tier framework for development of computable rules

The list of commercial rule checker (or model checker) software from Solibri, NavisWorks, AEC3/SmartCode, Selvaag Bluethink, CRC, Jotne EPM Technology and others indicate that this has gone from research to commercial software.

1.2 *Improvement of the design process*

It is important to be reminded on that a large share of defects in constructions originates in early stages of the construction process. Findings in Norway, included other European countries, suggest that as much as 40 % of building defects in Norway can be related to mistakes or omissions in the design process. (Ingvaldsen, 1994, 2001).

The potential of support by digital rule checkers in the design process is illustrated by the doted line for “Knowledge based design process” in figure 2 below.

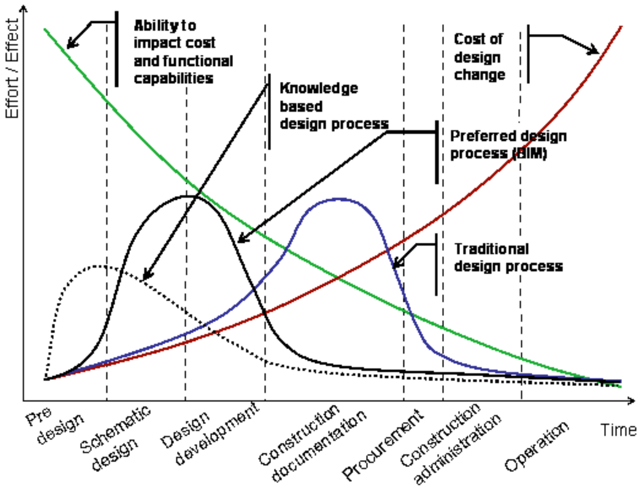

Figure 2. Early decision making with knowledge based design process

Use of rule-checking software in the design process will in addition to earlier decisions, also have other effects. The decision process are according to findings by Andersen (2000) characterized by;

- Design involves a lot of subjective value judgments, and decisions are often based on experience, “gut feeling”, or intuition. Design options are evaluated based on quantitative and qualitative performance measures. There exists no objective optimal design solution.
- Decision-making in design happens mainly through evaluation of proposed design solutions.

Other benefits by use of automatic rule checkers software in the design process can be:

- Quality improvement; fields where you are not expert can be checked.
- Creativity; can verify if a solution is possible, instead of selecting the safe one.
- Learning; Study of feedback of rule violations gives feedback for learning

It is also important to remember that an error free model is not the same as the best design solutions or optimal solutions.

1.3 *History of IA and rule checking*

Rule checking is a part of the AI (artificial intelligence) computer science. In addition to ICT also include disciplines as logic part of mathematic, linguistics and philosophy. The history IA is far older than the computers and a start can be the Greek philosopher Aristotle (384 BC – 322 BC) which invented syllogistic logic, the first formal deductive reasoning system. In the 13th century the Spanish genius, Ramon Lull (1232- 1316) wrote De Nova Logica and invented Figura Universalis, the first device for logical combinatorics (Sowa, 2000 and Buchanan, B.G., 2008).

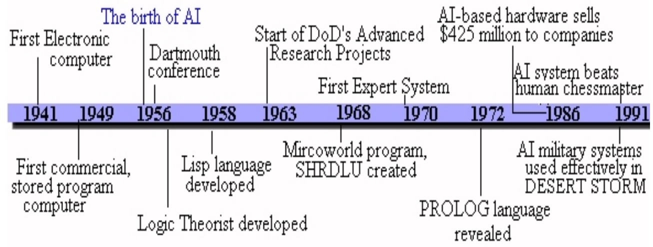

Figure 3. History of AI, artificial intelligence (IA, 1997)

Figure 3 take a more narrow scope. 1956 is often regarded as start of IA, when John McCarthy coined the term "artificial intelligence" as the topic of the Dartmouth Conference. First International Joint Conference on Artificial Intelligence (IJCAI) held in Washington, D.C. in 1969. Herb Simon wins in 1978 the Nobel Prize in Economics for his theory of bounded rationality, one of the cornerstones of AI. In 1980's was Lisp used in commercial applications. The first autonomous drawing program, Aaron was created by Harold Cohen in 1985 and was demonstrated at the AAAI National Conference (based on more than a decade of work, and with subsequent work showing major developments). (Buchanan, B.G., 2008). The expectation to AI has been full of enormous possibilities. Just code all the information from a textbook – and you get all the answers you need – after just one punch on the button. But the reality has not been that easy. John F. Sowa (2006) refer to the Halo project where on tried to representing the knowledge in a chemistry book into an AI system. The results were a score from 40% to 47 % correct and a cost of about $ 10.000 per page textbook. One explanation was the heterogeneity for the chemistry text leading into the “knowledge soup”. On the other hand, in 1997 the “computer” managed to beat the world chess champion, illustrating possibilities of IA in a structuralized domain.

In the AEC-industry has automated code-checking or standards analysis and compliance has been an active area of research since the 1960s. At CIFE at Stanford University they managed in 1996 to develop a proof-of-concept prototype demonstrating the feasibility of an online code-checking methodology. (Han et.al. 2009).

It was first with use of BIM / IFC based software on could achieve practical benefits by rule-checking software. The AEC related software industry took in 1995 initiative to found the IAI, International Alliance for Interoperability, for implanting of the IFC-format for improving of the interoperability and content of information exchanged between software (Wikforss, 2003). IAI has now changed name to buildingSMART.

1.4 *BIM – focus on information modeling process*

Giving a general definition of BIM is today very hard due to a large variation of the interpretation of the letters in the acronym. Is M for model (file) or modeling (process). In this paper BIM is used for as an acronym for Building Information Modelling:
– The process of defining relevant information and relations between the information and its purpose.

2 PROBLEMS ADDRESSED

2.1 *General challenges in development of computer interpretable rule sets*

Experience shows that the process of interpretation of code developed for humans (skilled professional) in natural language into formalized rule-sets who are computational for ICT systems / software way is not straight forward (Sowa 2000).

Use of software for performing digital rule checking is “Black-box” solution because most of the software for performing digital rule checking is done by the software developer. The end-user does not have complete control, overview or documentations on how the codes are interpreted and implemented. According to Gross (1996) this approach will lead to lack trust to the results. Syvertsen (2009) point it that limited control and insight in use of BIM lead to Building Information Mess. Simplicity and transiency will have a revealing effect on the information exchange process.

The information needed in the BIM (building information model - represented as a file is in some cases absent or of poor quality / relevance. Theoretic this implies that rule checks can not be performed. But in practice thee are done. This is possible because in building in some assumptions and algorithms to find or use information that “normally fits”. An example is energy calculations where the walls in the design model from the architect do not contain any information about thermal properties. When imported into the energy calculation program, it has built in some algorithms that assume the open space in the wall must be insulating with a given value, so the calculation can be performed.

Another side is that only 30% of client requirement can be / are expressed and values. (Kiviniemi 2005) The remaining 70% must be covered in other ways, e.g. as presence or not presence of defined qualities or reasoning about defined qualities. For technical implementation and interoperability Treldal (2008) found 70 % of the input and output data can be properly defined in IFC. This implies that there are several aspects that can not be performed or performed with high precision.

2.2 *Validity of the rule - Translation or transformation*

Often only the simple geometrical rules are implemented into the model checker software, leaving some codes / regulation “unsaved”, or done by software developer (Solibri 2009). This can make it hard to know how much of the design validation that can be done by software, and what must be done manually. The use of rule checker is relative immature in the AEC industry and methods has to be developed. (Eastman 2009).

Figure 4 illustrate that for a source of rules, (code, standard, law, regulations) there will be some part that are well fitted for implantation – “Rulish”, but other parts still have to be done by skilled professionals – “English”.

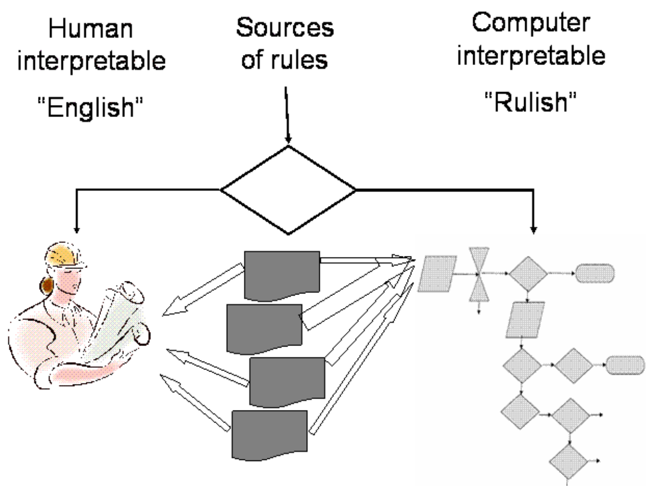

Figure 4. Division between the “Person-“ and the “Rulish” interpretation of different part of standards (codes).

A system / awareness for what rules can be computable (green zone), what is “tricky” (yellow zone), and what is best for human interpretation (red zone). The last item should be supported with check-lists or other QA systems. After our opinion is the crucial to know exactly what still has to be done in the old way. Even if many implementations of codes are done, we have hard to find documentation of what is not done. One consequence of this is that on the final stage all parts of the model has to be checked manually, because one do not exactly know what the automated rule checking software has performed.

This means that executing of a rule must be included in a knowledge system. Figure 5 is representing a knowledge system and illustrates the relations between a “Knowledge model”, who is represented by the codes (source of rules), “Ontology" who is representing the AEC-industry specific and precise definitions (includes systems as taxonomy and classification), and the "Meta-model" representing the expression of the computer interpretable rules

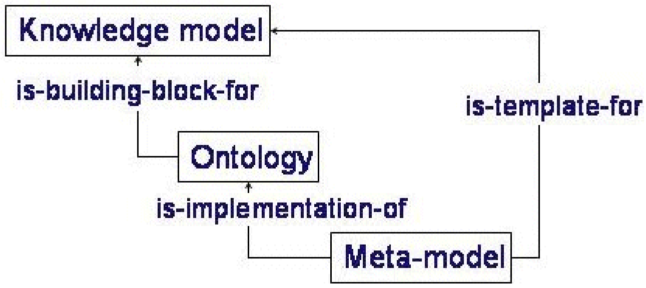

Figure5. Knowledge model – Ontology – Meta-models (Shakeri et.al. 2001).

There are today no standards or defined methods for development and documentation of rules for implementation into software. In addition to the logic rule, this must include information about the necessary content of relevant information in the model (e.g. represented as an IFC file).

2.3 *Knowledge soup*

The development of rules involves a checking of the "constraints" in the documents. An example for this is a standard for measurements of areas, where thy use the terms; floor, covering and level, often in variation combinations with space or area. Within this standard, and with an attitude that this must be the same – causes of course no problems. But when using this terms in combinations with other standards one can get into "uncertainty" and answer that this is "normally" the same (but this can / will depend on the circumstances (construction). John F. Sowa (2000) calls this situation for the "Knowledge soup" and points out these four reasons for their occurrence: a) Overgeneralizations, b) Incomplete definitions, c) Conflicting defaults and d) Unanticipated applications. Sowa further notice that experience shows that these exceptions and borderline cases result from the nature of the world, not from language or logic. We think one need a system (method, tool) for pre-checking the validity and reliability rules before they are implemented into software.

Without a consistent logical system as foundation knowing its limitations, every system will fail while scaling. Again, du to the complexity in the codes is possible to define rules that overrun other rule without being aware of the consequences. Use of BIM based rule checkers can in worst cases end up with Building Information Mess.

An early example of scaling failure is from Ramon Lull (1232-1316) who development of a logic based system for determine combinations. This worked well with two and three circles. Figura Universalis with 14 concentric circles with 16 sectors each giving the first combinational explosions in the history with 16^{14}, over 16 quadrillion combinations. This approach demonstrates the most basic and inefficient AI algorithm: *Generate and Test.*

3 THEORY AND METHODS ADAPTABLE FOR USE IN THE AEC-INDUSTRY

3.1 *Theoretical foundation for development of rules*

Based on the multi discipline foundation of AI on logic, mathematics, linguistic, philosophy and informatics there are a large number of theories for defining and develop reasoning systems for logic of rules. There should be no need for starting with "empty sheets"

Methods and theories are in nature deductive, and aiming to re-use the same "principles" in many different situations. This approach can be used to develop modules – topology of rule – same structure of rule applied on different construction parts – reuse and modular assembling of rules into a rule set. To discover this pattern of modules, use of metalanguage will be of great help. John F. Sowa (2007) therefore contradicts people who say that metalevel representations are complex and inefficient. For many applications, metalanguage can significantly reduce the complexity, as in the following sentences in controlled English and their translations to an algebraic notation:

"Every house is a construction"
$\Rightarrow$ $(\forall x)(\text{house}(x) \supset \text{construction}(x))$.

"House is a subtype of Construction"
$\Rightarrow$ House < Construction

Every operator of any version of logic is a specialization of some word or phrase in natural language: $\exists$ for there exists, $\forall$ for every, $\wedge$ for and, $\vee$ for or, $\supset$ for if-then, ~ for not, $\Diamond$ for possibly, and $\Box$ for necessarily. The metalevel words for talking about logic and deduction are the same words used for the corresponding concepts in natural languages: truth, falsity, reasoning, assumption, conclusion, and proof. This notation makes it computable and suitable for support by ICT systems – and for defining re-usable modules of rules and logic.

Another interesting thing is to be able to describe relationships between topics, and for this the topic map standard provides a construct called the topic association (Pepper, 2000). Topic maps is well defined in ISO 13250 series of standards. This semantic network builds on the concept of conceptual graphs (CGs). They express meaning in a form that is logically precise, humanly readable, and computationally tractable. Conceptual graphs can be translated to predicate calculus and to the Knowledge Interchange Format (KIF) (Delugach, 2006). Rule Interchange Format (RIF) developed by the W3C consortium is as a Resource Description Framework (RDF) schema.

The RDF triples indicates it connetction to OWL ontologies, whit all its possiblities and frameworks for further development. (Bruijn, 2005). According to Beetz et. al. (2005) is an OWL notation of IFCs advantages over generic XML schema representation.

3.2 *Classification in the AEC industry*

One of the benefits by classification is to find the right information base, and its relation to other information. Amor and Xu (2005) state that the amount of useful information available on the web for A/E/C professionals increases inexorably. They have tested numerous search engines allow users to identify potentially useful information in this vast resource, though the majority of these systems work purely on the search terms entered by the user. This means that the web pages which are found are often not as relevant to the user's needs as would be expected. What is returned is certainly far from the promise of the semantic web where the properties of the content can be readily ascertained.

There will therefore still be need for the traditional developed classification system. In a report developed for Standards Norway, Bakkmoen (2009) point out that the complexity and number of different systems clearly illustrates the need for harmonization or mapping between the systems if structure and classification at any time needs to be transferred across borders between nations, organizations, or classification systems. With references to preliminary study he finds that IFD Library appears to be the most obvious alternative available to the building and construction industry.

Borup (2008) points out following fundamentals for classification;

– Existing classification systems like OmniClass, BSAB, Uniclass are Building Information Models of the classified and defined objects in the construction domain. All objects in classification BIMs are corresponding to concepts. They all have names based on the language used as a tool in our thinking

– Classification systems need solid theoretical foundations and standardized methods for their creations -e.g. use of the international standardized concepts and methods in the terminology domain including standardized definitions of definition

– Models muddle in the construction industry can be an increasing problem.

On the other side, it can now be a "Window of opportunity" to establish a sustainable foundation for rule development. The technical side of BIM / IFC is in a large degree solved, and the remaining parts can be solved by extension of entities and/or property sets in the IFC-schema. (Ding et. al. 2004).The knowledge theories and methods itself are well enough developed. Common use of the standardized 3-tier framework in table 1. can be used for specifying deliverables (from different suppliers) within each layer. Instead of the software depended situation of today. Deliverables in second layer could be re-used in a number of implementations in third layer.

3.3 *Concepts for mapping ontology's - IFD*

IFD – Interntioanl Framework for Dictionaries can be rgardes as a concet for mapping different calssifications. As illustrated in figure 6, IFD is a mapping between different parts of different classifications tables. This will lead to an increasing numbers of relations, and a method for presentation of these relationships is by use of the *Hyperbolic tree* concept. (Bell and Bjørkhaug, 2006)

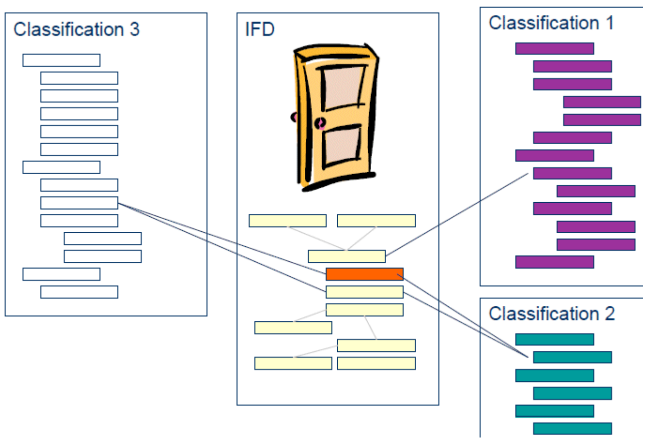

Figure 6. IFD as a mapping mechanism for classification tables. (Bell and Bjørkhaug, 2006).

In order to automatically verify the information in an exchange process we need to detail the information further than the general level of the IFC standard. For example, when the architect supplies information about the type of materials in the beams and columns, she must do so using a plain text string. Even if she spells this correctly, there is no guarantee that the receiving application will understand exactly what this text string means. And what if she uses a different language, dialect or uses the plural form of the word? Ideally the computer should be able to understand even this type of information in the IFC formatted information received. This is typically the scenario addressed in semantic searches on the web. (Bell et.al., 2008, Bell and Bjørkhaug, 2007b)

The IFD standard is based on the ISO 12006-3:2007 "Building construction -- Organization of information about construction works -- Part 3: Framework for object-oriented information" standard, and has many similarities with the EPISTLE4 standard for the Oil and Gas industry (Bell and Bjørkhaug, 2007a).

3.4 *Theories and methods for finite domains*

But is it still possible to avoid the "Knowledge soup"? One approach can be to consider the AEC-industry sources for rules (standards and likewise described codes and regulations) as finite domain of knowledge and professional language (ontology).

The first element is look if there is a language where true and common understanding. Alfred Tarski (1935) concludes in "The concept of Truth in Formalized Languages" that it is a hopeless exercise with regard to natural language, because of its complex and mutable nature. Given the non-universal nature of formal languages (specifically, the usual absence therein of terms belonging to the theory of language), a distinction must be made between the *object language* (the language under study) and the *metalanguage*. The metalanguage contains the names of the expressions of the object language and of the relations between those expressions, and usually the full vocabulary of the object language. But if one look at finite domains, solutions is possible" *The problem of the definition of truth obtains a precise meaning and can be solved in a rigorous way only for those languages whose structure has been exactly specified"*. For other languages — thus, for all natural, "spoken" languages — the meaning of the problem is more or less vague, and its solution can have only an approximate character (Tarski, 1944). If we let this be the "indicator "that a solutions is possible within the AEC domain of knowledge, The chaos of Babel-like communication can be avoided or at least reduced with a common language of rules.

4 METHODS FOR THE AEC INDUSTRY

4.1 *From English to Rulish versions of standards*

Based on theory (Sowa, 2000, 2006, 2007 and Tarski, 1935, 1944) is should be possible to develop a logic system with a finite domain and a structured language. The languages and semantics in standards are written in a defined way, and are suitable for translating into formal notation in a truthful way. The argument for this statement is based on the ISO normative rules for structuring and drafting international standards in table 1.

Table. 1 — Requirement (ISO, 2004)

Verbal form	Equivalent expressions for use in exceptional cases
shall	is to is required to it is required that has to only … is permitted it is necessary
shall not	is not allowed [permitted] [acceptable] [permissible] is required to be not is required that … be not is not to be
Do not use "must" as an alternative for "shall". (This will avoid any confusion between the requirements of a document and external statutory obligations.). Do not use "may not" instead of "shall not" to express a prohibition. To express a direct instruction, for example referring to steps to be taken in a test method, use the imperative mood in English. Example: "Switch on the recorder."	

For *"Recommendation,* (Table H.2) the ISO standards use the verbal form: *Should / should not*. for *"Permission"* (Table H.3) the ISO standards use the verbal form: *May / need not*, and for *"Possibility and capability"* (Table H.4) the ISO standards use the verbal form: *Can / cannot*, all with equivalent expressions for use in exceptional cases similar to Table H.1. (ISO, 2004).

By use of semantic method it should be possible to developed "Rulish" version ready for implementation into software. Laws and regulations also have a similar way of using modal auxiliary verb. This use of normative reference is also implemented in the BIM-manual version 1.1 from Statsbygg, Norwegian Public Construction and Property Management (same as GSA in USA) (Statsbygg, 2009).

4.2 *ISO supported system for domain knowledge*

According to Sowa (2000) should the general principles for constituting an expert system be based on a background knowledge about the world, including ontology, axioms, and defaults. This includes the following topics:

- Ontology: A classification of the types and subtypes of concepts and relations necessary to describe everything in the application domain.
- Definitions: Necessary and sufficient conditions that define new types of concepts and relations in terms of more primitive concepts and relations in terms of more primitive types.
- Constraints: General principles or axioms that must be true of the instances of those concepts.
- Defaults: Information that is expected to be true of the instances of various concept types.
- Behavior: Rules that govern the actions by and upon each type of object and the interactions of collections of objects.

For mapping of specification to logic is Conceptual Schema Modeling Facilities (CSMF) useful. The ISO JTC 1/SC 32 project on Conceptual Schema Modeling Facilities (CSMF) is developing standards for appropriate languages and tools. (Sowa, 2000). Conceptual modeling is central for systems analysis, database modeling, and knowledge engineering, to support the development processes from original codes into computable rules.

4.3 *MOKA and participants*

MOKA - Methodology and software tools Oriented to Knowledge based engineering Applications) is a name given to a methodology as a part of ESPRIT research program. It was assumed to work out the following:
- knowledge representation forms of a product and its designing process as well as the methods of its record,
- computer application for aiding record, representation and managing of the knowledge,
- possibilities of further automatic generation of KBE application code from computer application.

MOKA is used for obtaining knowledge in designing process, for elaborating KBE (Knowledge Based Engineering) and for creation of knowledge base for this system. It uses MOKA methodology and in particular informal model from this methodology and the concept of ICARE forms (Illustration, Constraint, Activity, Rule, Entity). Knowledge referring to the structure of a designed product and its designing process is collected by means of forms.

Ontology's can be created by means of Protégé application and can be exported to many different formats including RDF(S), OWL XML Schema. Protégé is a free open source tool, it is an application which aids creation of knowledge bases, including ontology edition and knowledge acquisition from experts. (MOKA, 2007).

4.4 *Using the IFC constraint model*

The IFC constraint model can be developed such that both simple and complex constraints can be captured. This is done through the provision of a constraint aggregation where the aggregation can be characterized by a logical AND, logical OR or logical NOT operator. The relationship between IDM and IFC is illustrated in table 2.

Table 2 Relationship between IDM and IFC. (Nisbet 2008)

IDM and IFC	
IDM	IFC representation
Process Map	IFC Process Model
Process/Sub Process	IfcRelAggregates/IfcRelNests
Process/Actor	IfcReiAssignsToActor
Process Sequencing	IfcRelSequence
Process Attribute	IfcRelDefinesByProperties/ HasPropertySet
Process Constraint	IfcRelAssociatesConstraint
Constraint / Sub Constraint	IfcConstraintAggregationRelationship
Model View	IFC Constraint Model

The IFC Constraint concept can be integrated into the IDM development method described in ISO/DIS 29481-1 "Building information models -- Information delivery manual -- Part 1: Methodology and format ". Rules that are applied to functional parts and exchange requirements are collected together into rule-sets. Each rule-set is expected to deal with a particular topic. However, a rule-set may contain rules from many origins provided that they are collected together in an organised way. (Wix et. al., 2008)

4.5 *Translation vs transformation*

If it is possible to get at direct match from original languish into "rulish"; the rule definition process can be performed as a translation. However, not all code related text, even standards is suitable for this. The first step is trying to "normalize" the original text into a formal text that can be translated. If this will result in – or can result in – different consequences than the original code, one must be utmost careful. However, automatic rule checking will in many cases be very useful. A way to go around is to explicit define this as a transformed rule, and, a dialogue box give information about this when the rule is activated in the rule checker software.

4.6 *Official computable rule-sets - Rulish version of standards*

As discussed above there can be minor formal or theoretical differences between the standard in native languish and “rulish” without this leads to any general consequences for assessment of the building. An example on this can be the accessibility for a wheelchair through doors. The ISO/DIS 21542 “Building construction - Accessibility and usability of built environment” defines a minimum width of light opening to be 800 mm. For assessment of this one must have the very detailed information about the door-casing – and the opening angel of the door. If the door is only open in 90 the thickness of the door lead will cover some of the opening compared to complete open parallel to the wall is stand in. This demands very much information about the door and it instance (opening). Knowing that doors are industrially produced in 100 mm intervals (Modules = M), using a 9 M door (outer door-casing width 888 mm) will give adequate opening width. Without this transformation, the accessibility would be very hard to check automatically.

By having an official “Rulish standard” version, these transformations could be transparent, and solutions based on consensus can be applied instead of “tricks” from the software developer. In the example above the Rulish standard should have a warning (information) for door with less than 950 mm outer door-casing width, that the defined properties to the door must be manually checked.

5 NORWEGIAN PROJECT (IN PROGRESS)

5.1 *Scope and participants*

The scope of this Norwegian project is to develop methods for translating and / or transforming building related codes in expressed documents as standards, national codes and regulations for use in digital rule (model) checker software. The methods should be software independent. Implementation and testing of IFC model import (IFC 2x3)/ IFC schema (IFC 2x3 / 2x4) limitations is not included.

The participants in this ongoing project, ending December 2009, are; Standards Norway (national standardization organization), Statsbygg (- acts on behalf of the Norwegian government as property manager and advisor in construction and property affairs), BE (National Office of Building Technology and Administration, with professional assistance from the consulting company Catenda, a spin-of from the research organization Sintef-Byggforsk. (Bell, H., Hjelseth, E., Bjørkhaug, L. , 2009). Link to information abut the project is listed in reference.

5.2 *Experiences for this project are so far:*

1) Inconsistency in the standards is identified. This is mostly related to use of different terms on same object / purpose in different places within the standard. An example for this is a standard for measurements of areas (NS 3940), where they use the terms; floor, covering and level, often in variation combinations with space or area. Within this standard, and with an attitude that this must be the same – causes of course no problems. But when using this terms in combinations with other standards one can get into “uncertainty”.

2) A common ontology within the AEC-industry, and a system for mapping of terms and definitions to existing standards and classification tables will clear up misunderstanding and uncertainty. Use of synonyms must be included.

3) Transparency of the rules is important. The major issue is to have full control of which code have been defined for implementation in software, and which code can not be implemented and must be interpreted by skilled humans.

4) Standard is interlinked by use of normative references to other standards. A standard can therefore no be used as the only source.

5) The standards are developed to be used in “known” situations. A description or “complexity” of the model and demand to information in the model is necessary for not “fooling” (intentional or not) the model.

6) Manual code checking is also error prone. Personal interpretations and overlooking errors are “natural”. Compared to use of digital rule checkers on get the same result with the same input each time, making it possible to indentify errors by experience and improve the system over time.

7) “Computerizing” of the codes is a process itself who leads to better understanding. This can supplement the development of new standards. One makes the computable (“computerized”) versions first, and then “re-writes” it for natural language and use by skilled professional persons. In this way one will have all the details in place and have the possibility to “test the consistency“ of the standard.

5.3 *Suggestion to an AEC-based method for development of computable rules*

During this project we have looked at different methods, both theoretical, and some examples from different industries. The suggested procedure is based on KBE approach and traditional standardization work processes. A simplified description of the 6 stages is listed below. Please not that this stages are interactive and loop back is possible and often essential for the result.

Stage 1) Define the scope and sources for the rule set E.g.. NS 3490 Standard for areal and volume calculations with guidelines, defined documents from BE. This will make the foundation of this rule-set to be based on explicit expressions and interpretations. This will be documented.

Stage 2) "Computability assessment" – rearrange the code so that they are as transparent as possible. We believe this is best don by a skilled person from the AEC industry (and not by software experts). This includes use (development) of supporting systems as terminology, ontology, taxonomy and classification with mapping between different tables, connections to other systems, standards and regulations etc.

Stage 3) Committee assessment / approval (this work will consist of own procedures). This is a QA of the prepared work form stage 2. to ensure that the rules are truly computable. Team work (committee) is especially needed for revealing "forgotten questions and limitation of use (complexity).

Stage 4) Logic rule notation. The computable rules will be transferred to a logical notation (format and support tool are not decided yet).

Stage 5) Choose of "rule format" for presentation of computable rules so it can be implemented into software (can be XML-or EXPRESS based).

Stage 6) Implementing (programming) the rules and information text specific in the rule checker software. This includes also the aspects with documentation, changes and testing.

6 DISCUSSIONS

Even though we see increased use of commercial rule checking software – often presented as now can everything be checked automatically – it is hard to find documentation of rules and how they are implemented in relation to the source of code. This appears like a "black-box" process implemented after the "Generate and test" attitude. We have found several interesting research projects with focus on possibilities and limitation regarding IFC implementation. The benefits by using a "framework" and supporting tool is increasing with its complexity. Developing clash detection, there is of course no little for semantic support, due to its topological nature and use of Boolean algebra. With relative few implementations based on single sources of code, one may not have discovered the problems this paper predicts. On the other hand, developing the foundation require expertise, time and money.

The suggestion for further research is to develop rules according to the 3-tier framework. Finding optimal solution and relevant support tools for different types of rules and complexity is important for practical impact of the 3-tier framework in the AEC industry compared to "Generate and test".

7 CONCLUSIONS

The most important is that the results after an automatic rule checking are true. Avoiding the "Knowledge soup" is therefore crucial when developing rules for software implementations. The 3-tier framework indicates how theory and supporting (logic and semantic) tools can support development of applicable rules.

Our suggestion is to have a transparent method for development of computer interpretable codes expressed as rule sets. In the Norwegian project we have included a committee who can be used to "cut through the Gordian knot" and develop rules that will cover the intention with the code, and not only be true to the letters of code. This can increase the number of computable rules.

It is also important to have a clear distinguish between what can be done in the rule checker software, and what must be done by skilled professional must. If not, one must perform a manual control after all, leaving digital code checkers to the less important (easy) cases. On the other hand, rule checkers can be one of the best ways for utilization of BIM in the AEC industry.

PREFERENCES

Andresen, I. 2000. A Multi-Criteria Decision-Making Method for Solar Building Design. Norwegian University of Science and Technology, Faculty of Architecture and Fine Art. ISBN: 82-7984-044-3.

Bakkmoen, K.I., 2009. Classification of building and civil engineering types for the AEC industry Report developed for Standards Norway, C. F. Møller Architects, Norway.

Beetz, J., van Leeuwen. J P, and de Vries, B. 2005. An Ontology Web Language Notation of the Industry Foundation ClassesAbstract. Presented at CIB W78 in 2005 http://itc.scix.net/cgi-bin/works/Show?w78-2005-d1-4-beetz /Accessed: 2009-06-30)

Bell, H. and Bjørkhaug, L. 2006. A buildingSMART Ontology eWork and eBusiness in Architecture, Engineering and Constructin, page 185, ECPPM 2006.

Bell, H. and Bjørkhaug, L. 2007a. IFD and IFC, http://dev.ifd-library.org/index.php/Ifd:Ifd_and_Ifd

Bell, H. and Bjørkhaug, L. 2007b. IFD in a Nutshell, http://dev.ifd-library.org/index.php/Ifd:IFD in a_Nutshell. (Accessed 2009-06-30)

Bell, H. et. al, 2008. Review of the Development and Implementation of IFC Compatible BIM Erabuild Report

Bell, H., Hjelseth, E., Bjørkhaug, L. , 2009. Standardized Computable Rules. Pilot study of methods for development of computable rules. Standards, Norway, BE (National Office of Building Technology and Administration), Statsbygg and Catenda. http://www.standard.no/no/Fagomrader/Bygg-og-anlegg/Digital-byggeprosess/Digitale-regelsjekkere/

Borrmann, A. and Rank, E. 2008. Topological Operators in a 3D Spatial Query Language for Building Information

Models Computation in Engineering, Technische Universitat Munchen, Germany
Borup, L. 2008. Comments to ISO 12006-02 Lise Borup, Presentation 2008-09-18, ISO/TC 59/SC 13 Workshop
Bruijn, Jos de, 2005. Document title: RIF RDF and OWL Compatibility Editor, Free University of Bozen/Bolzano (Accessed 2009-30-06)
Buchanan, B.G., 2008. A Brief History of Artificial Intelligence. http://www.aaai.org/AITopics/pmwiki/pmwiki.php/ AITopics/BriefHistory (Accessed: 2009-06-30).
Delugach, H. 2006. A World of Conceptual Graphs, http://conceptualgraphs.org/body.html (Accessed 2009-06-30).
Ding, L. Drogemuller, R. Jupp, J. Rosenman, M. and Gero, J. 2004. Automated Code Checking, CRC Construction Innovation. In: Clients Driving Innovation Conference. Gold Coast, Queensland, Australia.
Eastman, C. 2009. Survey on Rule Checking Systems (paper in progress). Georgia Technical University, USA.
Gross M. D. 1996. Why can't CAD be more like Lego? CKB, a program for building construction kits. Automation in construction, Vol. 5, No. 285-300.
Han, C.S. Kunz, J. and Kincho, H. 2009. Law. Making Automated Building Code Checking A Reality, Center for Integrated Facility Engineering Stanford University, USA
IA. 1997. The history of Artificial Intelligence http://library.thinkquest.org/2705/history.html (Accessed 2009-06-30)
Ingvaldsen, T. 1994. Byggskadeomfanget i Norge. Utbedrings kostnader i norsk bygge-/ eiendomsbransje- og erfaringer fra andre land, NBI Rapport 163, Oslo
Ingvaldsen, T. 2001. Skader på bygg: Grunnlag for systematisk måling, NBI Rapport 308, Oslo
ISO. 2004. ISO/IEC Directives, Part 2. Rules for the structure and drafting of International Standards, 5th Edition.
Kiviniemi, A. 2005. Requirements Management Interface to Building Product Models, Ph.D. thesis CIFE Technical Report # 161. Center for Integrated Facility Engineering,. Stanford University.
MOKA. 2007. Methodology and software tools Oriented to Knowledge based engineering Applications) http://kpkm2.polsl.pl/PROJEKTY/KBE/moka_ka/moka_ka_en.html (Accessed: 2009-06-30)
Nisbet, N. 2008. IDM Authoring using XML, document version 0.4, dated 2008-11-09, http://www.iai.no/idm/idm_resources/idm_templates/idm XML_authoring.pdf (Accessed: 2009-06-30)
Pepper, S. 2000. The TAO of Topic Maps, Affiliation: Ontopia AS, Oslo, Norway http://www.ontopia.net/topicmaps/materials/tao.html (Accessed 2009-06-30)
Shakeri C., Flores, I., and Houard, A. 2001. Dassault Systems. Research Topics. Knowledge models. Ontologies. Meta-models. http://www.ti.arc.nasa.gov/projects/vib/day2/ppt/Shakeri_day2.ppt (Accessed 2009-06-30)
SmartMarket Report (2009). Building Information Modelling Trends SmartMarket Report. McGraw-Hill Companies, ISBN: 978-1-934926-25-3. www.analyticsstore. construction.com (Accessed 2009-06-30).
Solibri. 2009. Solibri Model Checker www.solibri.com (Accesed 2009-04-20)
Sowa, J. F. 2000. Knowledge representation: Logical, Philosophical and Computational foundations, Thomson Learning. ISBN 0 534-94965-7
Sowa, J. F. 2006. The Challenge of Knowledge Soup, Published at Research Trends in Science, Technology and Mathematics Education, Mumbai,. http://www.jfsowa.com/pubs/challenge.pdf,
Sowa, J.F. 2007. Fads and Fallacies about Logic. IEEE Intelligent Systems, VivoMind Intelligence, Inc.
Statsbygg. 2009. BIM-manual 1.1, http://www.statsbygg.no/FilSystem/files/prosjekter/BIM/SB-BIMmanual1-1mVedl.pdf (Accessed: 2009-06-30)
Syvertsen, .G., 2009. BIM - The Building Information Mess: Past, present and future Department of Structural Engineering, Norwegian University of Science and Technology Paper submitted to IT-Con, http://www.itcon.org/
Tarski, A. 1935. The concept of truth in formalized languages, Logic, Semantics and Metamathematics, Oxford University Press, Oxford (1956), 1935, JH Woodger (trans.); First published as ‘Der Wahrheitsbegriff in Den Formaliserten Sprachen’, Studia Philosophica I.
Tarski, A. 1944 The Semantic Conception of Truth and the Foundations of Semantics, University of California, Berkeley. Published in Philosophy and Phenomenological Research 4 (1944). http://www.jfsowa.com/logic/tarski.htm (Accessed 2009-06-30)
Treldal, N. 2008. Integrated Data and Process Control During BIM Design. Master Thesis. Department of Civil Engineering, Technical University of Denmark
Wikforss, Ö. (2003). Bygnadens informasjonsteknologi: så används och utveklas IT ibyggandet. Svensk byggtjänst.
Wix, J., Nisbet, N. and Liebich, T. 2008. Using Constraints to Validate and Using Constraints to Validate and Check Building Information Models 7th European Conference on Product and Process Modeling (ECPPM) Sophia Antipolis. 10.-12 Sept. 2008.