Semantic Web Approach to Ease Regulation Compliance Checking in Construction Industry

Khalil Riad Bouzidi, Bruno Fies, Catherine Faron Zucker, Alain Zarli, Nhan Le Thanh

Semantic Web Approach to Ease Regulation Compliance Checking in Construction Industry

Khalil Riad Bouzidi [1,2], Bruno Fies [1] , Catherine Faron-Zucker [2], Alain Zarli [1] and Nhan Le Thanh [2]

[1] Centre Scientifique et Technique du Bâtiment, Sophia Antipolis, France, E-Mails: khalil-riad.bouzidi@cstb.fr (K.R.B.); bruno.fies@cstb.fr (B.F), alain.zarli@cstb.fr (A.Z.)

[2] I3S, Université Nice Sophia Antipolis and CNRS, France; E-Mails: faron@polytech.unice.fr (C.F.Z.); nhan.le-thanh@unice.fr (N.L.T.)

Abstract: Regulations in the Building Industry are becoming increasingly complex and involve more than one technical area. They cover products, components and project implementations. They also play an important role to ensure the quality of a building, and to minimize its environmental impact. Control or compliance checking are becoming more complex every day for industrials but also for organizations in charge of assessing the conformity of new products or processes. This paper will detail the approach taken by the CSTB in order to simplify this compliance control task. The approach and the solutions proposed are based on semantic web technologies. For this purpose, we first elaborate a domain-ontology, which defines the main concepts involved and the relationships among them based on OWL. We rely on SBVR and SPARQL to reformulate the regulatory requirements written in natural language in a controlled and formal language. We structure then our control process based on expert practices. Each elementary control step is defined as a SPARQL query and assembled into complex control processes “on demand” according to the component tested and its semantic definition. Finally, we represent in RDF the association between the SBVR rules and SPARQL queries representing the same regulatory constraints.

Keywords: Ontology; Semantic Web; Knowledge Management; Building Industry; e-regulations ; assisted checking; rule based system

---

1. Introduction

Regulations in the Building Industry are becoming increasingly complex and involve more than one technical area. They cover products, components and project implementations. They also play an important role to ensure the quality of a building, and to minimize its environmental impact. As a consequence; checking the conformity of a new product against the existing regulation is becoming more complex every day for industrials. In this general context, the research communities of Knowledge Engineering and Semantic Web have a key role to play to provide models and techniques to simplify access to technical regulatory information, facilitate its appropriation, and support professionals in its implementation.

In this article, we address the different issues linked to the compliancy checking in the specific context of producing Technical Advices (so-called “ATec” in French). An ATec is a document containing technical information on the usability of a product, material, component of construction or even a process, which has an innovative character. However, this new product or new process must comply with existing regulatory documents. We chose the ATec as a study model because CSTB has the mastership and a wide experience in these kinds of technical documents. We were able to lead interviews on the CSTB site of Sophia-Antipolis with experts directly involved in the drafting of these ATec in the field of photovoltaic panels.

2. Selected regulatory documents

2.1. Importance and difficulties linked to regulations

The regulations are written by humans and are read and applied by humans. Consequently, they were sometimes incomplete or contradictory and their structures are often arbitrarily complex. This has at least one direct consequence in the present scope. If we want to automate the verification process of compliancy of a new product, the textual knowledge and the corresponding constraints contained in these documents have to be translated (and transformed) into rules that can be understood by a machine and thus processable.

2.2. The Technical Guides

In this paper, we will illustrate our approach by considering not directly standards or regulatory documents but Technical Guides (TG). These TGs are documents edited by the CSTB which can be considered as explanations of regulatory documents. They do not replace the regulations. They are a complement to the regulatory documents offering to the industrials an easier reading and understanding of the technical rules of construction. They collect detailed executions featuring a wide range of situations. The TGs are also the reference documents that help us to verify the validity of the technical information provided by the manufacturers. They encompass all of the structural and dimensional variables for the validation of a product. Our goal is to formalize the knowledge they contain under the form of constraints database exploitable by knowledge-based systems.

We use the TG "The tile roofs" issued by CSTB as a study model. This guide of 107 pages outlines the different types of tiles and their characteristics. It defines their status of implementation and verification of various criteria such as slope tiles, support or climatology of where they must be installed. Monitoring of these instructions is drastic because the non-compliance with a requirement leads to a negative on the Technical Assessment Technical Paper.

According to the GP "The tile roofs," we were able to identify nine different types of tile, with different intrinsic characteristics. Each tile has a manufacturing area and a shape, an implementation that depends on the slope and support, and raises an attachment.

The regulatory constraints existing in these TGs are used to decide whether the procedures used by manufacturers meet their obligations.

2.3. The Technical Advice (ATec)

As already introduced in the first chapter, an ATec corresponds to an innovative product or process and contains mainly a technical description (made by industrial wanting to promote its new product or process). The structure of an ATec is always the same and it consists of three parts:

- An overview and identification of the assessment, which could be considered as an administrative part;
- The technical assessment itself, formulated by a group of experts from CSTB. Basicaly, only two options are possible in this section. It must be answered clearly if the element or process described is acceptable or not.
- The technical document (TD) of the product or process which must be delivered in the technical assessment. Fulfilling this part is devoted to the industrial who wants to promote an innovative product or process. For several reasons, this is the most difficult part that requires efforts from the industrial and time from CSTBs' experts.

Therefore, we are particularly interested in assisting the creation of the TD.

A ATec is drafted at the request of an industrial. The industrial sends a request to the relevant department within the CSTB and in return, the CSTB sends back to the industrial a template of the DT. It is a Word file containing chapters, text and instructions. This template is supposed to be self sufficient but it appears clearly that the industrials fail to properly fulfill the template in order to come to a good TD. A dialogue between CSTB and the applicant is necessary before reaching an acceptable version of the TD. As a direct consequence, this leads to a long casework, about 6 to 8 months and requires effort from both sides which must to be reduced to a minimum.

3. Modeling Technical Documents

The regulations as well as the TD are written by humans and therefore are in a flat textual format only understandable and thus exploitable by humans. The corner stone of our approach is to develop a common framework allowing structured and semantically rich representations of regulation contents and product or process structures. After reviewing various studies related to technical regulations and after having interviewed experts involved in the

elaboration of ATec, we defined a generic process for the verification of the TD. This process is a formalization of practices followed by CSTBs' experts. In the continuation of the work performed at CSTB [1], about ontology building and conformity checking of construction projects [2], we propose to model the process of elaborating the TD using an ontology-based approach.

We represent the regulatory constraints using also the above mentioned ontology as the source for a controlled vocabulary and we defined a methodology to transform the rules expressed from natural language (as it is the case in current regulation) to semi-formal language (SBVR) and to a formal language (SPARQL). This part will be explained in section 4 of the current article.

3.1. The OntoDT Ontology

In order to disambiguate the semantic attached to the terms, we defined an ontology of the considered domain. We defined an ontology so-called "OntoDT" to represent both the structure of the TD but also the vocabulary used in the considered technical field.

We have studied the TDs issued by the group within CSTB in charge of validating the ATec concerning photovoltaic panels. The OntoDT ontology contains an exhaustive list of terms from the photovoltaic domain merged with the other terms extracted from a thesaurus developed by CSTB for the building industry which is called the REEF [3].

We developed a model for tile based on information collected from a dedicated TG named "The tile roofs." Each tile is represented in concept and integrated it into our ontology OntoDT. The ontology includes all the semantics that reflects the structural and dimensional criteria of a tile, the criteria are represented by properties.

As a result, our ontology has 121 classes and 39 properties formalised in the OWL Lite language. 35% of these classes are created from REEF terms. The remaining 65% are concepts more specific than those of the REEF thesaurus, which contains general concepts of the building industry. In its current state, it lacks specific terms relative to a particular field (Photovoltaic). However, it remains in constant evolution.

3.2. Modelling the technical document

We use our ontology OntoDT to model the semantics conveyed by the DT to a formal interpretable knowledge. We translated the template of the DT as a set of forms interconnected to each other. The product to be described is decomposed into a set of sub elements called "components". These components are identified in the OntoDT and thus linked by semantic relationships to other components. These relationships will guide the concatenation of the corresponding forms. In other words, in an application developed on this basis, the user will be guided in the process according to the information entered in the forms and the way in the forms follow on from each other is determined by the ontology and by data entered by the user.

For example, a PV glass polymer module (a component) is part of a PV panel (the product). This glass polymer module is composed of several elements: polymer film, photovoltaic cell, etc. (elements).

To model this composition, each concept representing a component or an element or a product is defined by an axiom. Axioms are used in the definition of an ontological class, they are of the form A ⊑ B, where A is a primitive concept (Product) and B description composed concept. For instance, (Figure 1) provides the definition of class VerrePolymere in the OWL language: it is a subclass of class ModulePhotovoltaic and of a class defined as the intersection of the classes of the instances having components of class CableElectrique, Cadre, Cellule PhotoVoltaique, etc.


```
<owl:Class rdf:about="#VerrePolymere">
  <rdfs:subClassOf rdf:resource="#ModulePhotoV"/>
  <rdfs:subClassOf>
     <owl:Class>
      <owl:intersectionOf rdf:parseType="Collection">
         <owl:Restriction>
            <owl:onProperty rdf:resource="#hasComponent"/>
            <!-- CableElectrique -->
            <owl:someValuesFrom rdf:resource="&Reef;#01573"/>
         </owl:Restriction>
         <owl:Restriction>
            <owl:onProperty rdf:resource="#hasComponent"/>
            <!-- Cadre -->
            <owl:someValuesFrom rdf:resource="&Reef;#01593"/>
         </owl:Restriction>
               ...
      </owl:intersectionOf>
     </owl:Class>
   </rdfs:subClassOf>
</owl:Class>
```

Figure 1. Example of a defined concept (high level view and corresponding owl code)

We have described the interdependencies between the different concepts modelling the interactions or interdependencies between parts of PV modules. These have a number of intrinsic characteristics (length, weight, manufacturer, etc.) represented in our modelling by properties attached to the corresponding concepts.

Once the industrial fills in the form relative to a concept with all the elements relative to its definition, they are stored in an RDF annotation file. By doing so, we acquire an interoperable representation of a TD, reusable in other systems. The use of the RDF model will allow us later to check the conformity of the TD with the standards of the photovoltaic domain.

3.3. Modelling the process of writing a technical document

The question which now arises is how to model the process itself of filling the TD, in order to produce a dynamic sequence of forms to fill. The dynamicity stands in the fact that the forms to fill in depend on the way previous forms are filled in: we want to adjust to the information provided without requiring the industrial to fill irrelevant parts of the TD.

Our approach is to ask first of high-level information (name the type of product, etc.) then browse the explicit dependency rules in the ontology to seek all information required for a product. Each information provided by the industrial will be confronted to the ontology and will determine the next interaction. The industrial has to choose in a first form a product among all those concerned with the photovoltaic field. From that first choice, based on our ontology, we determine the list of components used in its manufacture and we offer a list of corresponding forms

```
SELECT ?Composant WHERE{
?x rdfs:subClassOf ?y
FILTER( ?x=dt:Component-Name)
?y owl:intersectionOf ?z
?z rdf:rest*/rdf:first ?f
?f owl:onProperty ?p
?f owl:someValuesFrom ?Composant}
```

Figure 2. Extract of the SPARQL query pattern

The dynamicity of the sequence of forms relies upon a SPARQL query pattern presented in (Figure 2) that we instantiate to query the ontology to determine each next form. More precisely, by using this query template, we query a product on its composition by questioning its definition. The concepts involved in its definition are returned by the query and are so much information that the industrial must provide through entry forms generated on the fly. The chaining of forms thus depends on the query results: each form corresponds to one or more elements of the result.

For example, the query below searches the ontology on the definition of the concept "VerrePolymere”.

```
SELECT ?Composant WHERE{
?x rdfs:subClassOf ?y
FILTER( ?x=dt:VerrePolymere)
?y owl:intersectionOf ?z
```

```
?z rdf:rest*/rdf:first ?f
?f owl:onProperty ?p
?f owl:someValuesFrom ?Composant
```

The result is that "VerrePolymere" is a module which has as components a "Cadre", a "CellulePhotoV", etc.

```
Result : [Cadre, CellulePhotoV, FilmPolymere,
VerreInterieur…]
```

At each step of the process of writing a TD, we display to the industrial a form for entering information related to a concept belonging to the result of such a query. If this concept is itself defined, the same query template is instantiated with a new query in order to provide the Industrial with new forms matching with the components involved in the definition of the current concept. The same query pattern is recursively instantiated until reaching terminal concepts, i.e. primitive concepts (with no definition).

As a result, the industrial browses thereby transparently into our ontology to complete all components of its product by filling out the forms provided; only relevant questionnaires are displayed.

3.4. Generation of structured annotations

To validate our approach, we developed a tool to assist the production of TD. This application has been developed in J2EE. It conforms to the three-tier architecture. The first third consists of a web page on the client side (GUI) that supports various forms requiring to be filled in by the industrial. The second third on server side is a servlet containing business code that interacts with the GUI part and the business part. It is connected to the Corese/KGRAM1 semantic engine [4] to query the ontological knowledge that represents of the TD and generates the RDF model of the filled TD. The last third is relative to the data. It includes the ontology of the TD and information of the product stored as RDF annotations.

Our tool provides the industrials with a rich and interactive interface based on a sequence of forms. All along the drafting process, the industrial is guided in his choices by the OntoDT ontology and the system adapts and goes through the ontology to seek to progressively for more specific information. These sequences of forms are orchestrated through SPARQL queries that run on the ontology.

At the end of the sequence of forms, two files are generated: a human readable file containing the information filled in by the industrial, and a semantic description of the document in RDF. This will be exploited to help in writing the technical assessments itself, by automatically checking the conformity of the information in the TD with the regulatory texts of the domain.

4. Modeling Business Rules

[1] http://www-sop.inria.fr/edelweiss/software/corese/

Several studies and international projects are interested into the transformation of regulatory document [5], [6], [7] among which the SBVR standard. SBVR stands for “Semantics of Business Vocabulary and Business Rules”. It is an OMG standard whose ultimate objective is to provide a meta-model that allows establishing data exchange interfaces for tools that create, organize, analyze and use vocabularies and business rules [8], [9], [10]. The SBVR meta-model facilitates the validation, analysis, alignment, and fusion of business rules for different tools of different constructors. The development of an SBVR base is done in two steps: the development of a business vocabulary and the writing of business rules based on the terms and concepts defined in the vocabulary.

SBVR controlled vocabularies consist in hierarchies of concepts specific to some domain, their relationships, definitions and synonyms. SBVR rules are based upon the Predicate Logic: they capture the "what" of business rules, rather than the “how” in other words the semantics of business rules, not the way they must be executed. SBVR is not an executable formalism, it is particularly addressed to business experts. It uses a controlled natural language that all business experts understand. It does not have a specific rule format.

In this paper, our examples of SBVR Rules are given in structured vocabulary, using several font styles:

- Nouns issued from our ontology are underlined in black color
- Verbs are given in orange color.
- Literal values are shown in red color.

4.1. Transforming standards into SBVR rules

There are some approaches to detect whether a sentence may describe a business rule [11]. The authors propose a linguistic analysis to determine whether a sentence expresses a business rule, for instance in the case where a structure such as “if – then” is detected. Ontology has been specifically defined in case some of the concepts that it describes were also detected in the sentence. Accordingly, the sentence is marked as a candidate to contain a business rule and ontology concepts are also associated to the sentence.

However, technical standards can be understood in different ways that is why the manual intervention of a domain expert is essential. We argue that NLP approaches of knowledge extraction from regulatory texts can significantly alleviate the task of domain experts but cannot replace them. In our work, we do not consider linguistic analysis of texts and focus on the representation of expert knowledge. CSTB experts helped us to identify and classify the constraints expressed in the photovoltaic standards and then the rules which represent them. The goal of this categorization is to determine the levels of interoperability of each sentence or paragraph of the standards and classify it.

Also, despite the intervention of a domain expert to identify the meaning of the constraints, some of them remain uninterruptable. These types of constraints contain information too ambiguous and impossible to formalize.

Once these texts are identified, a step of disambiguation is necessary. The transformation of texts into SBVR rules will provide a normative, unambiguous and reusable source.

The extraction of rules from standards or statutory text is a tedious job, it often requires to the structure the information. The descriptions used have been detailed enough to show how the content of standards can be converted into SBVR vocabulary and business rules. However, a clarification of the text was needed before the transformation into SBVR. The steps below are necessary in order to produce an understandable SBVR text:

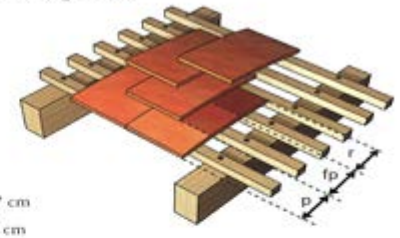
4.1.1. Extraction of conformance rules from table

Let's start with the following TG table:

Pentes des couvertures en tuiles plates de terre cuite

Pentes et recouvrements des tuiles plates

Pose sans écran
DTU 40.23, tableau 1



Emploi admis et recouvrement « r » ≥ 7 cm
Emploi admis et recouvrement « r » ≥ 8 cm
Emploi admis et recouvrement « r » ≥ 9 cm
Emploi non admis

r : recouvrement
fp : faux pureau
p : pureau

Pentes valables pour les rampants de longueur maximale de 8 m (en projection horizontale)

Pente de couverture	Zone I [1]			Zone II [1]			Zone III [1]		
	Situation [1]			Situation [1]			Situation [1]		
	Protégée	Normale	Exposée	Protégée	Normale	Exposée	Protégée	Normale	Exposée
70 %	8 cm			8 cm					
75 %	8 cm			8 cm					
80 %	7 cm	8 cm		7 cm			9 cm		
85 %	7 cm	8 cm		7 cm			9 cm		
90 %	7 cm	7 cm		7 cm	8 cm		8 cm		
95 %	7 cm	7 cm		7 cm	8 cm		8 cm		
100 %	7 cm	7 cm	8 cm	7 cm	7 cm		8 cm	9 cm	
105 %	7 cm	7 cm	8 cm	7 cm	7 cm		8 cm	9 cm	
110 %	7 cm	7 cm	7 cm	7 cm	7 cm	8 cm	8 cm	8 cm	
115 %	7 cm	7 cm	7 cm	7 cm	7 cm	8 cm	8 cm	8 cm	9 cm
120 %	7 cm	7 cm	7 cm	7 cm	7 cm	7 cm	8 cm	8 cm	9 cm
≥ 125 %	7 cm	7 cm	7 cm	7 cm	7 cm	7 cm	8 cm	8 cm	8 cm

1. Les zones et situations de concomitance vent/pluie sont définies dans le chapitre « Climatologie », page 102.

The TG contains tables with different informations relative to the installation of tiles with their dimensional and structural property. In the table above, the first column shows the different applicable slopes. In each row, we can see for a specific slope, its recovering and the area of installation

To transform these constraints into SBVR rules we process on two steps:

First, we transform the row's information into textual constraint. Example, from the first row we can extract the information below:

The applicable tiles slop for a recovering greater than 8 cm, built in zone 1, in protected situation is equal to 70%.

The second step is to rewrite this text into SBVR rules by using the ontology of TDs, the text will be read as follows:

In English:

If the tile have a slop equal to 70% then it is obligatory that the implementation is in Zone 1, in protected situation with a recovering greater than 8 cm.

In French:

Si la pente de couverture est égale à 70 % alors il est obligatoire que la mise en œuvre est en Zone 1 en situation Protégée avec un recouvrement supérieur à 8 cm.

The concepts identified in this fragment are Slop, Zone1 and protected, which belong to the ontology of TD.

4.1.2. Reformulation

Let us consider the following regulatory text:
Les dimensions du châssis principal doivent être :

Largeur intérieur : (847 ± 5) mm
Hauteur intérieur :(1910 ± 5) mm

This standard extract expresses conditions that are difficult to read by non expert readers. It needs a reformulation to be understood:"The maximum width of a main frame must be lower or equal to 853mm and the minimum width higher or equal to 842mm. The maximum height of a main frame must be lower or equal to 1915mm and the minimum height greater than or equal to 1905mm".

If a frame has a minimum width higher or equal to 842mm and a minimum height higher or equal to 1905mm and a maximum width less than or equal to 853mm and a maximum height less than or equal to 1915mm, then it is a main frame

4.2. Transformation SBVR obligation rules in RDF annotation.

Our SBVR rules are Operative Business Rules with "If-then" syntax. This kind of rules can be transformed into systems rules, automated execution of business processes and allow checking how business activities are conducted.

However, the TG's constraints are specific to a relevant situation of a constraint to check. Our goal is to check the compliance of TDs according to constraints expressed in the TG.

So, we built a new RDF annotation (Figure 3) based on abstract language (section 5.1). It allows executing for a specific condition a relevant action using queries representing the antecedent and consequent.

We represented the Antecedent and consequent part with SPARQL “ASK“ queries, this queries model determine if there is a matched triple between the query and the RDF annotation of the TD. “ASK” query returns “TRUE” for positive response and “FALSE” if there is no matching. Using this RDF annotation allowed us to automate a part of the process of compliance check.

```
<rdf:RDF >
                    <Test>
                          <if>
                                <Antecedent>
                          </if>
                          <then>
                                <Consequent>
                          </then>
                    <Test>

</rdf:RDF>
```

Figure 3. The RDF annotations of if-then SBVR rules

4.3. Representation of regulatory constraints in SPARQL

We aim to model the way where experts use CSTB guides and try to automate their know-how. It forces us to follow their interpretation and to establish a formal representation of regulations. SBVR describes the concepts and requirements regardless of their implementations. We chose to represent constraints into “ASK” SPARQL queries. This formalization allows the automation of the compliance of a document against these constraints.

To validate our model, we developed a rule base to verify the compliance of DT with regulatory standards expressed in guides. The availability of verifications information may be addressed as follows:

- Provide explicit information in the DT model. While this is natural for some aspects (material, form) others are derived from more basic information and rely on fallible human judgment that are current causes of error (slope, fastening, fixing). It does not always take into account all the parameters and can cause errors and possible non-compliance thereafter.
- Requests for verification are important they apply on properties that require more analysis like structural, dimensional or climatological constraints.

Figure 4. Transforming SBVR rules into SPARQL queries

To perform a transformation of SBVR rules into SPARQL queries we follow a number of defined steps and use a controlled-vocabulary, in our work we use OntoDT.

As an example the following SBVR rules:

If the tile have a slope equal to 70% then it is obligatory that the implementation is in Zone 1, in protected situation with a recovering greater **than 8 cm**

Step 1: We decompose the SBVR rule into a set of single sentences in order to decompose the different constraints expressed. So we identify the antecedent and the consequent of our "if-then" rules like below:

	English	French
Antecedent	**The tile have a slope**	**La tuile a une pente de couverture**
	The slope is equal to 70%	**la pente de couverture est égale à 70 %**
Consequent	**The tile has an implementation**	**La tuile a une mise en œuvre**
	It is obligatory that implementation is in an area	**il est obligatoire que la mise en œuvre est dans une zone**
	The area types is Zone1	**La zone est de type Zone 1**
	The implementation has a situation	**La mise en œuvre a une situation**
	The situation types is protected	**La situation est de type Protégée**
	The implementation has a recovering greater than 8 cm	**La situation a une recouvrement supérieur à 8 cm**

Step 2: We transform, into triple pattern, each formulation expressed in the antecedent and consequent. We use for this all concepts and properties of OntoDT.

SBVR	SPARQL Triple pattern

The tile have a slope
The slop is equal to 70%
	minus {
	?tile dt:aPourPenteDeCouverture ?slop
	FILTER (xsd:integer(?slope) != 70)
	}

The tile has an implementation	?tile dt:aMiseEnOeuvre ?implemntation
It is obligatory that implementation is in an area	?implementation dt:aPourZone ?area
The area types is Zone1	?area rdf:type dt:Zone1
The implementation has a situation	?implemntation dt:aPourSituation ?situation
The situation types is protected	?situation rdf:type dt:Protégé
The implementation has a recovering greater than 8 cm	?situation1 dt:aPourRecouvrement recovering FILTER (xsd:integer(?recovering)>=8)

Step 3: We build the SPARQL queries:

```
PREFIX
dt:<http://www.semanticweb.org/DossierTechniqueProtegeV.owl#>
ASK
{
?tuile dt:aMiseEnOeuvre ?misenoeuvre
minus {
?misenoeuvre dt:aPourPenteRecouvrement    ?pente
FILTER (xsd:integer(?pente) != 70)
}
}
```

Figure 5. SPARQL query of the antecedent (Example 1)

```
PREFIX
dt:<http://www.semanticweb.org/DossierTechniqueProtegeV.owl#>
ASK
{
?tuile dt:aMiseEnOeuvre ?misenoeuvre

?miseenoeuvre dt:aPourZone  ?zone
?zone rdf:type dt:Zone1

?zone dt:aPourSituation  ?situation
?situation  rdf:type dt:Proteger

?situation dt:aPourRecouvrement ?r
FILTER (xsd:integer(?r)>=8)
}
```

Figure 6. SPARQL query of the consequent (Example)

Step 4: Construction of the RDF annotation of the SBVR rules.

```
<rdf:RDF >
            ...
                  <Test>
                        <if>
                              <Query rdf:about="&OntoDT;#P70"/>
                        </if>
                        <then>
                               <Query
rdf:about="&OntoDT;#Check70"/>
                         </then>
                  </Test>
            ...
</rdf:RDF>
```

Figure 7. RDF annotation of the SBVR rules (Example 1)

5. Modeling of the verification process of regulatory constraints

5.1. Process Model

To capture the know-hows of CSTB experts in charge of providing technical advices on TDs, we developed a process model providing a declarative representation of their process of validating a TD against regulatory constraints. Our model enables to build an RDF description of the sequence of constraints verifications to perform for a given TD.

The RDFS schema of our model comprises four properties: body, if, then and else, and eight classes: `Pipeline`, `Pipe`, `Load`, `Query`, `Rule`, `RuleBase`, `Test` and `And` among which the class `Pipeline` models a process definition and the class Pipe models a call for a process. The execution of queries (`Query`) or rules (`Rule`, `RuleBase`) can be conditional (`Test`) and a process description can recursively call for other processes (`Pipe`), including itself. This recursive feature is used in the modeling of complex processes calling for one or several elementary processes.

The abstract syntax of a process is defined by the following grammar:

```
Pipeline  ::= EXP +
EXP       ::= Load(Name) | Query(Name)
            | Test(Query(Name), Exp, Exp)
            | Rule(Name) | RuleBase(Name)
            | And(Exp +) | Pipe(Name)
```

We have developed a process engine based on the Corese/KGRAM semantic engine [4] which principle is as follows: it analyses a process definition represented in our model and dynamically constructs and executes a sequence of SPARQL queries or rules. It thus enables to supervise, coordinate and sequentially execute a set of queries and rules. The process management relies on a set of predefined SPARQL queries such as the one presented in (Figure 8), dedicated to the management of the body of a process, which enables to list all the components of a process and their types:

```
SELECT * WHERE {
?p rdf:type kg:Pipeline
{
?p kg:body ?q
?q rdf:type ?t
minus {?q rdf:type rdf:List}
}
UNION
{?p kg:body ?a
?a rdf:rest*/rdf:first ?q
?q rdf:type ?t
}
}
```

Figure 8. A SPARQL query template to interpret a process RDF representation

For instance, let us consider the validation process which RDF representation is presented in (Figure 9). Once this RDF data is loaded, our process engine executes the above query template (Figure 8) the resource of class `Pipeline` is identified and the resources denoting sub-processes involved in its definition are listed. Each of these sub-processes is recursively handled in order to identify the operations to be performed. Let us note that we do not use the whole process model since our process representations do not involve rules: in our case, the only basic operations are queries.

```
<rdf:RDF >
      <Pipeline>
            <body rdf:parseType="Collection">
            <Load rdf:about="DocumentTechnique.rdf"/>
                  <Test>
                        <if>
                              <Query rdf:about="&OntoDT;#P60"/>
                        </if>
                        <then>
                              <Query
                        rdf:about="&OntoDT;#Check70"/>
                        </then>
                  </Test>
                  ...
            </body>
      </Pipeline>
</rdf:RDF>
```

Figure 9. Extract of the RDF representation of a verification process

Let us detail the handling of such a RDF description. The process engine interprets a resource of type `Load` (line 4) by loading the RDF description of a TD (located at the `URI` in argument) which validity is to be checked. It interprets a resource of type `Query` (line 7) by loading the SPARQL query located at the URI and executing it on the loaded RDF description of a TD.

A resource of type Test (line 5) calls for the instantiation and execution of the query template presented in (Figure 10) which enables to identify the value of the "`if`, `then`" and "`else`" properties and their types. The process engine then first executes the `ASK` query denoted by the value of the "`if`" property and, depending on a `TRUE` or a `FALSE` answer, it recursively interprets the RDF description of the process denoted by the value of the "`then`" property or by the value of the "`else`" property, if any.

```
SELECT DEBUG * WHERE {
?q kg:if ?qi
OPTIONAL {?q kg:then ?qt . ?qt rdf:type ?tt}
```

```
OPTIONAL {?q kg:else ?qe . ?qe rdf:type ?te}
}
```

Figure 10. A SPARQL query template for identifying conditional sub-processes

5.2. Elementary and complex processes

We distinguish between elementary and complex processes. A process is said to be elementary if it consists in the verification of the attributes of a component described in a TD which is denoted in our ontology by an atomic class. A process is said to be complex if it is associated to a component defined in the ontology as a combination of sub-components. In that case, the process consists in the verification of the attributes of the components and the verification of those of its sub-components.

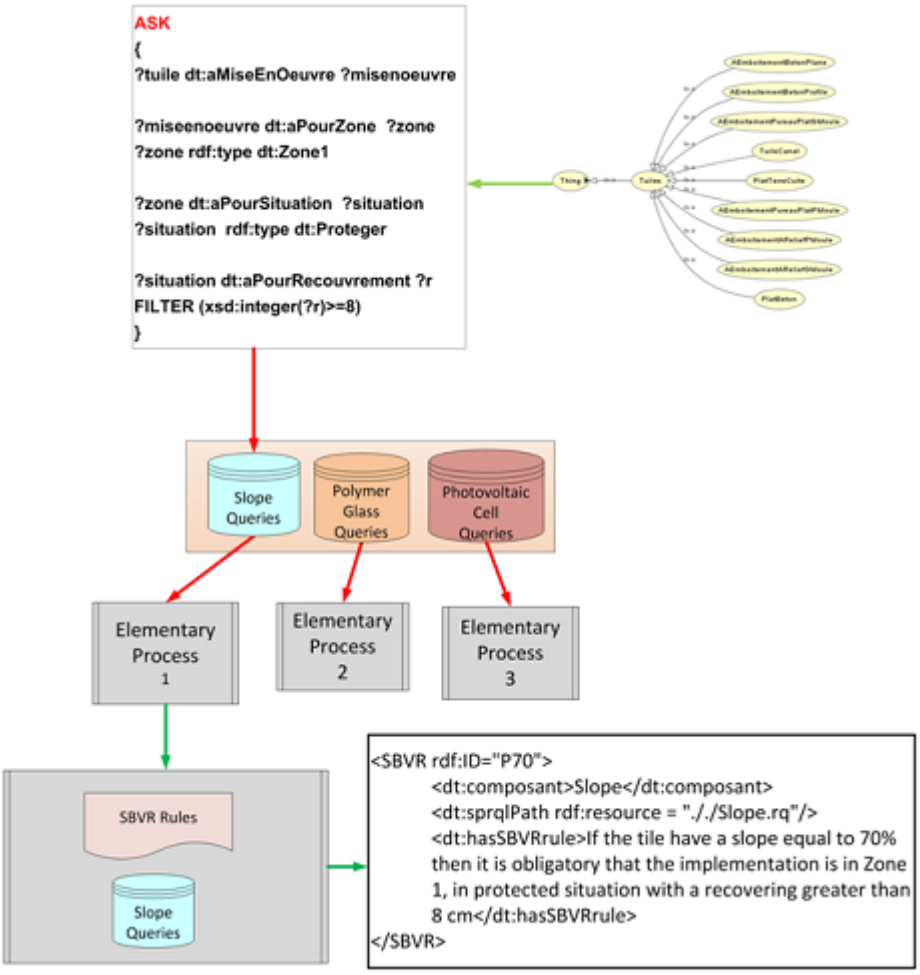


Figure 11. Elementary process of checking constraints

For instance, the validation process associated to a tile (Figure 11) is elementary: its RDF description calls for the execution of SPARQL queries testing its structural and dimensional criteria (the slope, the material, the form, etc.). On the contrary, the validation process associated to PV glass polymer module is complex: its RDF description presented in (Figure 12) calls for the interpretation of the description of its sub-processes relative to the

verification of its sub-components `[Cadre, CellulePhotoV, FilmPolymere, VerreInterieur]` (Section 3.3).

Such an RDF description of a complex process is automatically and dynamically generated, by using a SPARQL query template querying the definition of the component to be validated in the OntoDT ontology.

```
SELECT DISTINCT ?v  where{
?x rdfs:subClassOf ?y
FILTER( ?x=dt:"Component-Name")
?y owl:onProperty ?p
FILTER( ?p=dt:aProcessus)
?y owl:hasValue ?v}
```

Figure 12. A SPARQL query template for generating complex process

The interpretation and execution of such a process consists in the recursive interpretation of the description of its sub-processes. To be precise, in case of a complex process involving sub-processes, the process engine interprets the resource of type Pipe denoting a sub-process by loading its RDF description accessible at the URI and recursively interpreting it.

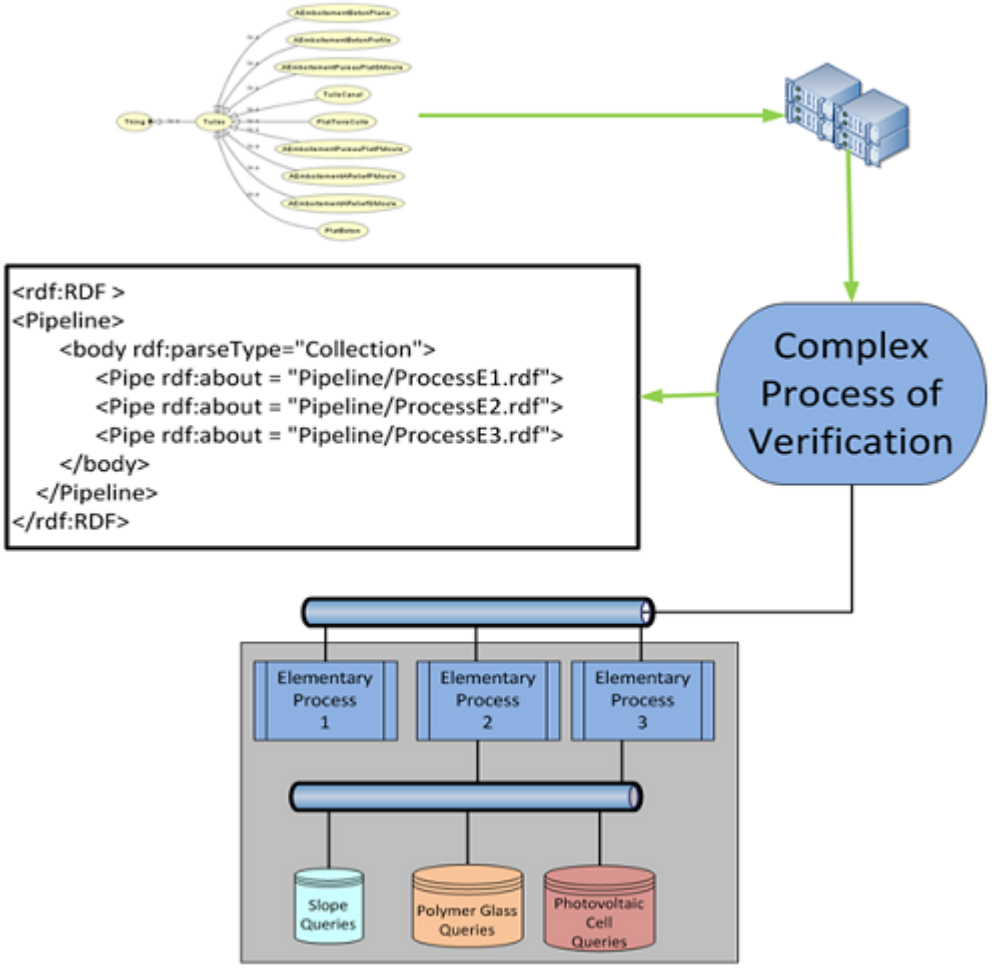


Figure 13. Complex process of checking constraints

5.3. Automating regulation compliance checking

To assist CSTB experts in writing technical advices, we have developed two tools. The first one is dedicated to industrials and helps them in writing TDs. It uses the OntoDT ontology automatically builds and presents to the user the forms to fill up to describe the attributes of all the components of a given product. This tool outputs an RDF description of a product based on the information filled by the users.

This RDF description of the TD of a product comes as an input in a second tool we have developed to help CSTB experts in validating a TD against regulatory constraints. It first automatically builds an RDF representation of the verification process convenient for the product waiting for a technical advice. It uses the OntoDT ontology and the model described in the previous section to recursively build an RDF representation of the process, based on the definition of its components. Then it calls for the process engine developed for our process model and automatically produces to the user a conformity report summarizing which of the queries representing technical constraints succeed when applied to the RDF description of a given TD and which of them do not succeed and therefore reveal a possible non conformity.

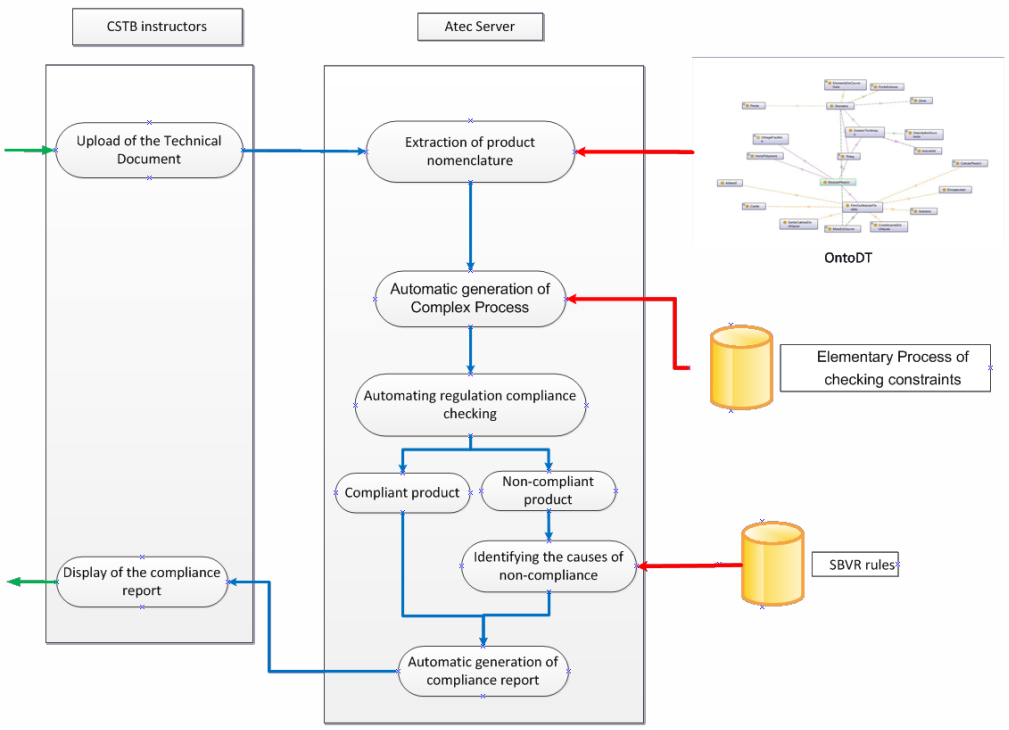


Figure 14. Overview of our assistant tool for writing technical advices

7. State of the art

Improving the logical structure of regulations is a vast research area. [12] have undertaken initial work on the structure of rules in decision tables. Decision trees were later applied in building industry, specifically in the design of steel buildings. [13]. The SASE system. [14] was developed to provide a complete hierarchical structure to classify families of related

regulations or codes. A major study of these early approaches is provided in. [15]. In addition, let us cite [16] who have developed the REGNET application to determine the applicability of building regulations in some given conditions, based on a question and answer interface.

Up to now, regulatory modeling was discussed under two different approaches. The first one aims to automatically analyze the rules and to confront the complexity of natural language [17, 11]. In a second approach, regulatory constraints are directly written according to a normalizing model, with the help of domain experts, which facilitates their translation into formal models [18, 19]. We propose a third approach, which takes into account the regulations written in natural language, offers a tool for writing TDs and automatically analyzes the content of these documents, their conformance to regulation.

On the other hand, various efforts have been made to apply conformance rules to the representations of construction projects, using the structures of drawings specially coded (IFC) or textual descriptions [1, 20, 21]. When compared to these works, the originality of our approach lies in the combination of formal representations and SBVR rules from which they are derived to explain the rules themselves or the decision making. Our approach extends compliance checking with the explanation of the decision making.

8. Results and limitations

We validated our approach on real use case at CSTB. We used TG which summarize the main requirements of 7 DTU currently applied in France. We designed our rules database by extracting regulation constraints and transform them into semi-formal language (SBVR) and formal language (SPARQL) using many steps. These transformations have been handled manually by domain experts. They were able to model 100% of TG constraints into SBVR.

Although the interpretation of experts helps in translating most of the constraints from text to SBVR, some of them remain not transferable into SPARQL. These cases correspond to the “fuzzy” constraints that contain information defined in a qualitative way (eg, "a short distance",), or that contain “common knowledge” [1] (eg, "The recovery of the ridge tiles is in the opposite direction of the winds rain dominant" Technical guides “The tile roofs” page 75).

The current limitation of our technical guides is that they collect detailed execution featuring a wide range of situations. This type of rules called “Implementation rules”, easily representable in SBVR, is the largest part of the rules that cannot be formalised in SPARQL. These rules represent a large proportion of regulatory constraints identified in the TGs (in this case 70%).

Example of implementation rules:

Implementation rules	SBVR formulation
« In the case of Canal tile, it typically runs in a mortar or flashing bardelis embedded and sealed in the wall»	**If tile canal then it is obligatory that it runs in a mortar or flashing**

Technical guides "The tile roofs" page76	**bardelis embedded and sealed in the wall**

As final results, with the help of CSTB experts, we identified about 177 SPARQL queries, 177 elementary processes. This result is about 30% of the SBVR rule modeled.

9. Conclusions

In this paper, we have presented an approach and a tool to represent regulation and technical documents in construction industry and partially automate regulation compliance checking. We propose a domain ontology, OntoDT, representing concepts involved in the description of technical documents and regulations. We combine SBVR and semantic web languages to represent in a controlled vocabulary and formalize regulatory constraints extracted from the Practical Guides edited by CSTB. These two complementary representations both are based on the OntoDT ontology and are associated in RDF annotations. The SBVR-based representation of regulation presented to the users reduces ambiguity and RDF- and SPARQL-based formalizations enable to automate regulation compliance checking. Finally, we propose a process model to organize regulation extracted from Technical Guides. A regulation compliance checking process is associated to each basic component involved in a technical document and a whole checking process is automatically built based on the RDF description of the technical document and the component definitions in the OntoDT ontology.

We have developed a tool for representing technical document and another one for their regulation compliance checking. They have been evaluated by CSTB instructors who have validated both our scenario, the granularity of the information required, the OntoDT ontology. Our regulation compliance checking tool is a component of an assistance tool in writing technical advices we have developed for CSTB instructors that we do not detail in this paper. Our models of regulation, technical documents and conformance checking process have been evaluated through the evaluation of this tool. CSTB instructors have validated the messages output by our tool.

A major perspective of our work is the identification of regulation which cannot be completely formalized and their inclusion through SBVR representations in a semi- automatic conformance checking process.

References and Notes

1. Yurchyshyna, A. Modélisation du contrôle de conformité en construction: une approche ontologique. Thèse de l'université de Nice Sophia Antipolis, France, 2009.
2. Gehre, A.; Katranuschkov, P.; Stankovski, V.; Scherer, R.J. Towards semantic interoperability in virtual organizations. In Proc. of the 22nd Conference on Information Technology in Construction, Dresden, Germany, 2005

3. Bus, N.; Fies, B.; Bourdeau, M.; Charvier, M.; Labedens, R. Reef sémantique, Diffusion et application des textes technico-réglementaires, Accompagnement des pouvoirs publics dans la rédaction des textes officiels. Livrable 4, CSTB, 2009.
4. Corby, O.; Faron-Zucker, C. The KGRAM abstract machine for knowledge graph querying. In Web Intelligence, 2010; pp. 338–341.
5. Lau, G.T.; Law, K.H.; Wiederhold, G. Comparative Analysis of Government Regulations Using Structural and Domain Information. In IEEE Computer, 2005.
6. Bolioli, A.; Dini, L.; Mercatali, P.; Romano, F. For the Automated Mark-Up of Italian Legislative Texts in XML. In Proceedings of Jurix. 15th Annual International Conference on Legal Knowledge and Information Systems, London, UK, 2002.
7. Turk, Z.; Katranuschkov, P.; Scherer, R.J.; Cerovsek, T. The Tools and Services Integration Platform of the ISTforCE Project. In I. Tommelein (Ed.): Concurrent Engineering Conference, Berkeley, 2002.
8. OMG. Business Semantics of Business Rules RFP. br/2003-06-03. http://www.omg.org/cgi-bin/doc?br/03-06-03, 2003.
9. The Object Management Group OMG. Semantics of Business Vocabulary and BusinessRules (SBVR). OMG Speciation, 2006.
10. Chapin, D.; Baisley, D. E.; Hall, H. Semantics of Business Vocabulary & Business Rules (SBVR). In Rule Languages for Interoperability. W3C Workshop on Rule Languages for Interoperability,Washington, DC, USA, 2005.
11. Martínez-fernández, J.L.; González J.C. A Preliminary Approach to the Automatic Extraction of Business Rules from Unrestricted Text in the Banking Industry. In Natural Language and Information Systems, 13th International Conference on Applications of Natural Language to Information Systems, NLDB, London, UK; 2008.
12. Fenves, S.J. Tabular decision logic for structural design. Journal of the Structural Division, 1966; Vol. 92, No. 6. pp. 473–490.
13. Nyman, D.J.; Fenves, S.J.; Wright, R.N. Restructuring study of the aisc specification. Civil engineering standards, srs 393, Urbana-Champaign, IL, USA, 1973; pp. 473–490.
14. Fenves, S.J.; Wright, R.N.; Stahl, F.I.; Reed, K.A. Introduction to sase: Standards analysis, synthesis, and expression. National Technical Information Service, 1987; pp. 473–490.
15. Fenves, S.J.; Garrett, J.H.; Reed K.A. Computer representations of design standards and building codes: U.s. perspective. University of Salford, U.K, 1995; pp. 473–490.
16. Kerrigan, S.; Law, K.H. Logic-based regulation compliance-assistance. In ICAIL, 2003; pp. 126–135
17. Dinesh, N.; Joshi, A.; Lee, I.; Sokolsky, O. Reasoning about conditions and exceptions to laws in regulatory conformance checking. In DEON, Berlin, 2008; pp. 110–124.
18. Reeder, R. W.; Karat, C. M.; Karat, J.; Brodie, C. Usability challenges in security and privacy policy-authoring interfaces. In INTERACT 07, 2007; volume 4663, LNCS, pp. 141–155.
19. Nazarenko, A.; Guisse, A.; Levy, F.; Omrane, N.; Szulman, S. Integrating Written Policies in Business Rule Management Systems. RuleML Europe, 2011; pp. 99-113

20. Pauwels, P.; Van Deursen, D.; Verstraeten, R.; De Roo, J.; De Meyer, R.; Van de Walle, R.; Van Campenhout, J. A. Semantic rule checking environment for building performance checking. Automation in Construction, 2011; pp 506-518.
21. Eastman, C.; Lee, J.M.; Jeong, Y.S.; Lee J.K. Automatic rule-based checking of building designs. Automation in Construction, 2009; pp 1011-1033.