# Formalization of Natural Language Regulations through SBVR Structured English[⋆] (Tutorial)

François Lévy and Adeline Nazarenko

Université Paris 13, Sorbonne Paris Cité, LIPN, CNRS, (UMR 7030)
F-93430, Villetaneuse, France

**Abstract.** This paper presents an original use of SBVR to help building a set of business rules out of regulatory documents. The formalization is analyzed as a three-step process, in which SBVR-SE stands in an intermediate position between the Natural Language on the one hand and the formal language on the other hand. The rules are extracted, clarified and simplified at the general regulatory level (expert task) before being refined according to the business application (engineer task). A methodology for these first two steps is described, with different operations composing each step. It is illustrated with examples from the literature and from the ONTORULE use cases.

## 1 Introduction

Formalizing natural language (NL) regulations is becoming an important challenge for rule systems.

Much more data are available in an electronic form that, let us say, ten years ago. Most governments have set up electronic legal repositories to make their national laws and regulations available (*e.g.* THOMAS in the USA[1], Legifrance in France[2], the Bundestag site in Germany[3]). This is also the case of international institutions (see the European Community[4] or the UNO[5] websites). Smaller organizations such as regions, cities, Länder also produce and publish their own regulations. On the private organizations side, most companies have both public regulations for their customers, and internal ones for their employees.

These rules are involved in applications that mainly concern the opening of rights and the conformance of processes. It is crucial that the formal rules

⋆ We thank to our partners in the ONTORULE project for the fruitful discussions, especially John Hall (Model Systems) for introducing us to the SBVR world, Christian de Sainte Marie for Decision Logic, and Audi for the collaboration on their use case. We are also grateful to American Airline who is the owner of one of our working corpora.

[1] `http://thomas.loc.gov/home/abt_thom.html`
[2] `http://www.legifrance.gouv.fr/`
[3] `http://www.bundestag.de/`
[4] `http://eur-lex.europa.eu`
[5] `http://www.un.org/`

embedded in those applications be consistent with the published regulations. Hence the need for extracting and building the formal rules from the NL ones.

This paper analyses this formalization process. It shows that it can be decomposed into three main steps, in which SBVR Structured English (SBVR-SE) plays an intermediate role between the natural and formal languages. Section 2 presents this challenging process that has been only partially addressed in the state of the art and Section 3 shows that it can be decomposed into three steps. Sections 4 and 5 describe the main operations that are involved in the first two of these formalization steps.

## 2 Formalizing NL Regulations: A Challenging Task

### 2.1 The Translation Task

Rule acquisition is a critical bottleneck for the development of business rules management systems (BRMS) as for most knowledge based systems. The problem is three fold for knowledge engineers. They have to identify all the constraints and rule information that are relevant for the domain of the rule application to develop. They have to specify how these constraints and rule must be handled: some are directly implemented in the decision system; some belong to general policy that apply to the system users but that are not formally expressed in the rule base; some are plain recommendations whereas other are strict constraints; they may concern the structure of the organization to model as well as its procedures and the behavior of actors. Once the relevant information is identified and its enforcement value defined, the knowledge engineers still have to encode this information in a way that is machine-understandable.

This acquisition process is a complex task to handle. Fortunately, in many cases – especially in conformance applications –, the relevant rule information is already encoded in policy documents (*e.g.* contracts, legal regulations, user guides). In those cases, the elicitation work can be based on existing and validated sources instead of experts' introspection and interviews.

However, extracting and translating NL regulations into formal rules remains an open issue [4,13,9]. The translation of text fragments written in NL into formal rules is difficult to automate, due to the reduced expressivity of formal languages and to the complexity of NL discourse, which is redundant and verbose, often elliptic and implicit, frequently ambiguous. Even the translation into SPARQL of LN queries, which are much simpler than texts, is acknowledged as a complex problem [24]. [8] considers a direct translation of legal texts based on a parsing step, but it actually relies on a manual translation of abstract syntax trees in a specific logical language.

### 2.2 The Problem of Uncommunicability

NL and formal rule languages stand on the opposite extremities of the formalization continuum, which raises a problem of uncommunicability between the

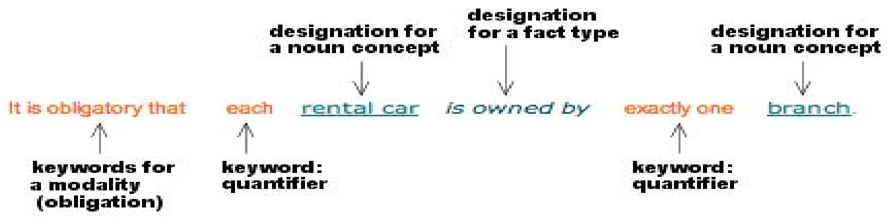
designation for a noun concept
designation for a fact type
designation for a noun concept
It is obligatory that
each
rental car
is owned by
exactly one
branch.
keywords for a modality (obligation)
keyword: quantifier
keyword: quantifier

**Fig. 1.** Example of an SBVR-SE business rule (www.brcommunity.com)

actors. The lack of domain knowledge and the complexity of NL explain the difficulty of writing the formal rules from scratch for the *engineer*. Working with texts requires a certain familiarity with the domain and a thorough understanding of source regulations. The *expert* describes the rules according to his/her knowledge of the organization for which the rule system is being developed. The *engineer* has a different point of view, more focused on the system and its operation as such. The *expert* finds it difficult to read the rule base when it has been written in formal language. This raises many problems, such as regarding the authoring of the rules in conformance to the source documentation and their updating.

Issues related to these deadlocks have been raised [4,13,9] and proposals have been made to improve the actors collaboration. One of them consists in exploiting a new language that acts as an intermediate between NL and formal languages. Controlled languages (CL) have been proposed to play this role. They are more precise, unambiguous, easier to understand – but also, less expressive – than natural language.

## 2.3 The Complexity of Natural Language

Controlled languages are considered as an interesting substitute for NL, assuming that they are readable for users and closer to formal languages than NL. In the Business Rules domain, CLs are used in Oracle Policy Modeling Suite[6], in IBM SPARCLE policy workbench [5]. RuleSpeak [20] is a dedicated in use controlled language, while Atempto Controlled English[7] is more generalist. SBVR (Semantics of Business Vocabulary and Rules) can be seen as a synthesis of several efforts and as a standard independent of any specific natural language. It has been accepted by the OMG (Object Management Group)[8]. We refer to the English version of SBVR CL, namely SBVR Structured English (SBVR-SE). SBVR relies on formulas (Figure 1) combining linguistic basic templates with logical, modal or quantification operators.

[6] www.oracle.com/technology/products/applications/policy-automation

[7] http://attempto.ifi.uzh.ch/site/docs/

[8] http://www.omg.org

However, the NL to CL translation is itself a complex task. Recently, [2] has proposed NL2SBVR[9], a tool to automatically translate NL into SBVR-SE, but the reported experiments show that the complexity of the translation depends on the complexity of the source sentences. Simple rule statements composed of at most two clauses are translated with a 80% success rate but the translation fails for complex ones.

Another way to handle the complexity of NL texts therefore consists in simplifying them. Text simplification has long been studied. It is usually considered to ease translation [22], to help human understanding (esp. in case of understanding disorders [15]), to prepare text summarization [10,7], and in the context of foreign language learning (see [6] for a general presentation). The notion of simplicity actually depends on the target application. For instance, if a succession of small sentences if preferable for foreign language learners, automatic translation can process long sentences as soon as their internal structure is canonical.

Taking advantage of these works on controlled language and text simplification, we propose to decompose the rule formalization into different steps and to rely on SBVR-SE as an intermediate language that ease the transition from natural to formal languages.

### 2.4 Organizing the Formalization Continuum

Several authors have considered degrees in the formalization process. [3] argues that there is a "continuum" between knowledge expressed in NL and formal one. Considering controlled languages as intermediary stages in this continuum, [23] points to two different views of controlled languages constituting steps in the formalization: the naturalist one insists on simplicity, the formalist one on eliminating ambiguity.

More focussed on the semantics of business rules, *Decision Logic* or *Decision Modeling* emphasizes the distinction between rules as specification of what should be, and the operational view where rules state what to do[10]. Operational rules take or help to take decisions, i.e. make rational choices between several outcomes [21] expressed in the language by some decision words (*e.g.* "Estimate", "Determine", "Assess", "Calculate", "Accept" [12]) and which follow the specifications. Operational rules are related to SBVR through the vocabulary, and to a Process Model, since branching points in this model need a decision [14].

Decision Logic has to see with a precise inventory of all the conditions which can influence a decision, and with how a decision is reflected in the system. At the moment, there is no clear and agreed formal view of decision in the BR domain, but OMG is currently elaborating a Decision Modeling Notation (DMN) which is a first step in that direction.

[9] `http://www.cs.bham.ac.uk/ isb855/nl2ocl/projects.html`

[10] Both terms have close but different meanings out of the business rules domain. "Decision Modeling" is used in decision theory for a.k.o. simulation, where the goal is to predict (long term) effects of a given decision. "Decision Logic" also deals how decisions are made by individuals or groups in unforeseen conditions (*e.g.* in emergency cases [16])

## 3 Divide for Conquer

### 3.1 A Three-Step Process

**The Division of Labor.** We focus here on the modelization and formalization of business rules. Some previous works [1] have shown that this tasks involves two typical actors. Both acts as knowledge engineers but one is a domain expert (the *expert*) and the other one is more system oriented (the *engineer*):

– The expert knows the domain of the target application, the organization in which it is expected to take place and the available sources of information. He/she has to identify what is the relevant information to take into account, in the form of a draft rule base.
– The engineer focuses on the rule system, that may not model the whole organization but only a part of it. He/she knows how the system works and he/she can implement rules in an operational language.

**Formalization Steps.** Based on this analysis, we propose to decompose the formalization of rules into three separate steps.

1. The first step produces CL out of NL. The transformation is made by an expert who knows and understands the source texts. The task consists in writing a new CL text that selects the rule information of the source text that is relevant for the organization to model. The resulting text can be considered as the specification of the rule base content.
2. The second step improves the specification text in CL. The transformation is made by the engineer in interaction with the expert. The engineer interprets the rules. He asks the expert to check and complete his/her understanding of the rule base. They may further filter out the text produced in the first step if it contains some rules that are not directly applicable by the rule system.
3. The last step aims at translating the specification text into formal language. It requires a good knowledge of the target application and system.

### 3.2 SBVR, between Natural and Formal Languages

SBVR [18] is an OMG standard. It is not a language *per se* but a metamodel designed for describing the business knowledge of complex organizations such as enterprises, in a formal and detailed way. It enables to

– assist experts in the specification and definition of the semantic model (semantic vocabulary and rules) of the domain;
– describe a business model in a precise, clear and unambiguous way;
– specify various linguistic structures that cover a larger number of languages, even if it is mainly used for English;
– structure[11] an organization business knowledge and business vocabularies;
– make the business rules accessible both to the experts and engineers, thus improving their inter-communication.

[11] `http://www.omg.org/news/meetings/tc/mn/special-events/br/`

The definition of business vocabulary is not dealt with here, our single focus is on using SBVR as an intermediary language. Business rules are written with the help of a controlled language, SBVR-SE, which makes use of a typed business vocabulary (Fig. 1): business terms or concept names, named entities, phrases which express fact types and facts linking concepts[12], as well as a set of (English) grammatical words (*e.g.* '**the**', '**that**', '**another**', '**a given**') used as operators. SBVR offers several types of operators. The most often used are:

– Logical operators, such as negation (it is **not** the case that $p$), conjunction ($p$ **and** $q$), disjunction ($p$ **or** $q$), implication(**if** $p$ **then** $q$, $q$ **if** $p$), equivalence ($p$ **if and only if** $q$).
– Modal operators, that modify the modal value of a fact from [contingently] true to obligatory, possible or necessary. These operators are often inserted in front of rules: "**It is obligatory that**" and "**It is necessary that**", resp. for operative and structural rules.
– Quantifiers: **a, an, each** (universal quantification), **at least one** (existential quantification), **at most one**, **exactly one**.

In the formalization process, our expectation with respect to SBVR is twofold. In the first step, it is used to faithfully translate source texts while reducing the complexity of natural language. The output is a list of autonomous rule (see Fig. 2) that can be further formalized independently of each other. In a second step, it is used to reformulate the business rules so as to take the application into account and to ease their formalization in the last step.

For instance, the textual rule 1 of the EU-Rent example can be first normalized (Rule 2[13]) and then turned into a decision rule (Rule 3). The last step would translate this differently according to the rule language: production rules and logic programming, for instance, have different technicalities to destroy and create objects.

1. *A rental has exactly one renter.*
2. *It is prohibited that the renter of a rental is changed.*
3. *If a renter asks to change the rental to a different renter, cancel the rental and create a new one.*

The first use of SBVR-SE concerns the expert task (from NL to CL) whereas the second one consists in a CL to CL transformation, which refines the description according to the end application. This last transformation usually requires a negotiation between the expert and the engineer. We will see in Section 5 that it mainly consists in eliminating the modal operators, as suggested in:

> The implementation impact of the alethic necessity tag is that any attempted change that would cause the model of the business domain

[12] We do not take for granted, as SBVR-SE does, that facts and fact types are specifically referred to by verb phrases.

[13] Of course, other readings can be proposed but the normalization process involves some interpretation choices.

> to violate the constraint must be dealt with in a way that ensures the constraint is still satisfied (*e.g.*, reject the change, or take some compensatory action). [18, p102]

rental *requests* car model
rental *specifies* car group
car group *includes* car model

**It is necessary that the car model *requested by* a rental *is included in* the car group *specified by* the rental**

**Fig. 2.** Example of an autonomous SBVR rule

## 4 From Natural Language to SBVR Rules

The acquisition methodology relies on the progressive transformation of the source text into a set of self-sufficient rules written in SBVR-SE controlled language. This process relies on four main operations, which are often interlinked: the lexical normalization of the source text, the extraction of the relevant text fragments, the syntactic normalization of these fragments, some semantic transformation for restoring contextual and implicit information. The result of that process is a set of rules written in valid SBVR-SE.

### 4.1 Lexical Normalization and Annotation of the Source Text

The lexical normalization is often performed on the whole source text. It is a critical step for the whole transformation process. An annotation is a meta-data or label attached to a word, phrase, sentence or section of the source document. A normalized vocabulary entry is associated to the annotated segment. In our framework, first annotated segments are elements of the conceptual vocabulary or keywords.

The *conceptual vocabulary* contains all the terms that have a specific meaning in the domain of the source regulation. For the acquisition of rules, we suppose that a domain ontology already exists, that fixes the conceptual elements to use in the rules [17]. The conceptual vocabulary is composed of the set of terms referring to the ontological concepts, individuals and roles. It can be encoded as a SKOS thesaurus referring to a an OWL ontology [19] or as SKOS annotations in the OWL ontology. The semantic annotation is the process that takes a text and a lexicalized ontology as input and outputs the source text enriched with annotations. It consists in localizing in the text all the mentions of the ontological elements, whatever the form that these mentions may take, and annotating them with a reference to the ontological element they refer to. Lexical normalization consists in normalizing the lexicon of the source text since the various variants of a term are all expected to be annotated with the same canonical form.

The *keyword list* contains all the "linguistic symbols [or grammatical words] used to construct statements – the words that can be combined with other designations [from the conceptual vocabulary] to form statements and definitions" [11, p.238]. Keywords are easier to recognize in the documents since they belong to closed class of words and their form is generally more stable.

The initial annotation process therefore produces a text in which the mentions of the ontological elements and the SBVR-SE keywords are annotated. They are colored according to SBVR rules: orange for keywords, green, blue and red respectively for the words and phrases that refer to concepts, roles and individuals. The underlying analogy is that concepts corresponds to "noun concepts", roles to "verb concepts" and individuals to "individual concepts" in the SBVR terminology, although we do not assume that there us a strict parallelism between the part-of-speech categories (nouns *vs.* verbs) and the conceptual ones (concepts *vs.* roles) as SBVR-SE does.

Since the annotation process does not cover the whole text, the resulting annotated text usually mixes black segments and some colored words and phrases. It looks like an SBVR "informal representation", since "not every word is annotated ('tagged') in accordance with a notation that can be mapped to SBVR" [11, p.152]. It is nevertheless a precious input for the extraction, normalization and transformation process that outputs a rule base specification.

### 4.2 Extraction of Rule Fragments from the Source Text

Once the text has been automatically annotated, the knowledge engineer has to identify in the source document the fragments (sentences or sequence of sentences) that convey rule information and to mark them as candidate rules that probably need to be reformulated but are nevertheless relevant for the target rule application.

Identifying these rule fragments in the source text is a complex task and the initial annotation eases this work. The knowledge engineer can focus on the passages that are most marked with annotation and extraction patterns can be designed on the basis of remarkable configurations of keywords.

The selection of a fragment consists in the creation of a candidate rule, which is identical at the beginning to the source fragment but which will be further transformed.

### 4.3 Lexical and Syntactic Normalization of the Rule Fragments

The variability and polysemy of natural language makes it necessary to give a canonical form to the initial statements. This normalization process concerns both the lexicon and the syntax.

One part of the lexical normalization is carried out through the semantic annotation process, since different words and phrases are annotated in the same way if they are alternate labels of the same ontological entity. For instance, in some contexts *member*, *members* and *participants* can be considered as synonyms and may all refer to the same concept.

However, it often happens that some relevant mentions are missed by the semantic annotator: the knowledge engineer has to identify new occurrences of relevant ontological entities. Another problem comes from the polysemic words which may not be properly handled by an automatic semantic annotator. In some context, *city hall* does not refer to a building but to an organization or even to the people involved in that organization. Advanced annotators which take the context into account can take care of these disambiguation cases but they are less performant than others.

Syntactic constructs also have to be simplified and normalized. A lot of syntactic constructs are not supported by a controlled language like SBVR-SE because they are ambiguous or difficult to understand. Some clauses must be reordered within the sentences, enumerations must be split, complex sentences must be simplified by erasing the irrelevant clauses, coordinations have to be decomposed. A single candidate rule may give way to two or more separate and simpler candidate rules. Some rules must also be transformed to stick to the canonical syntactic structure of the target language that is supposed to be unambiguous and easy to understand.

For instance, the following sentences can be decomposed into several ones, which makes the initial statement easier to understand and to exploit.

1. ***Neither*** *accrued mileage,* ***nor*** *award tickets,* ***nor*** *upgrades are transferable by the member upon death*[14]*.*
   *Accrued mileage is not transferable by the member upon death. Award tickets are not transferable by the member upon death. Upgrades are not transferable by the member upon death.*
2. *The membership year,* ***which*** *is the period in* ***which*** *your elite benefits are available, runs from March 1 through the last day of February of the following year.*
   *The* ***membership year*** *is a period.* ***Member's*** *elite benefits are available in the* ***membership year****. The membership year runs from March 1 through the last day of February of the following year.*

## 4.4 Semantic Transformation

Transformations at the semantic level are also often required to restore some elements of context or implicit piece of information that condition the interpretation of the rule. The semantic transformation does not preserve the apparent meaning of the source fragment but aims at decontextualize it, providing context independent formulations, fixing some possible ambiguities, deleting irrelevant stylistic fragments.

For instance, in the UNO regulation n°16 dealing with car manufacturers quality tests, it often happens that a generic term like *test* is used in a sentence where it is clear from the context that it has a specific meaning and actually stands for *e.g. micro-slip test*, which refers to a specific type of tests. When the

[14] This example has been extracted form the American Airlines terms and conditions.

rule statement is isolated from its context, the generic term must be replaced by the more specific one so that the initial meaning is preserved.

More complex transformations involve the whole sentence. In the following case, the knowledge engineer separated the extracted fragment into two different SBVR rule statements and then made explicit the modality "hidden" in the use of a future tense.

1. *No mileage credit will be awarded for canceled flights* ***or*** *if you are accommodated on another airline.*
2. *If a member is accommodated on another airline, then* ***it is obligatory that*** *no mileage credit is attributed. If a flight is cancelled, then* ***It is obligatory that*** *no mileage credit is attributed.*

## 5 From Normalized Rules to Decision Rules

At the end of the normalization and simplification process, the output is SBVR-conformant text. It is a set of business rule statements which are admittedly formal in the sense of the SBVR standard:

> Business rule expressions are classified as formal only if they are expressed purely in terms of fact types in the pre-declared schema for the business domain, as well as certain logical/mathematical operators, quantifiers, etc. Formal statements of rules may be transformed into logical formulations. [18, p.85]

However, the initial goal – having rule directly translatable into rules of an automated decision system – is still not reached. The problem comes from the difference between the regulatory level ontology and the application specific ontology. A semantic gap lies between rules infering deontic modalities and rules infering concrete actions. We consider two points. First, new entities are needed, dedicated to represent the conditions of the decision. Second, a rule modal statement has to be translated into decision terms.

Three examples from different domains are used to illustrate these points.

### 5.1 Introducing New Specialized Entities

Very often, deciding if the conditions are fulfilled needs to introduce new specialized entities. This is best illustrated through examples.

**Example 1.** The first example, bellow, is related to the UNO regulation n°16. Concerning the breaking load test after cold-conditioning (bl-cc test for short), the regulation states the fragment 1. At the normalization level, it appears that 'When' has a temporal value rather than a conditional one. It indicates a step of the process (actually, the fourth step – previous ones being omitted for the sake of brievity). 'And' has also a temporal value. It introduces a next step. Naming the steps is a facility for decomposing the rules (see version 2, duration considerations are provisionally left aside). Taking into account the delays also needs to give a full status of entities. This leads to add two notions, *load-duration* and *removal-measure-delay* (see version 3). More hidden is the fact that the greater the load duration, the more severe is the obligation, so the delay is a *minimum.*

1. *When the strap has been kept under load for 30 minutes in the same low-temperature chamber, the mass shall be removed and the breaking load shall be measured within 5 minutes after removal of the strap from the low-temperature chamber.*
2. *Step 4 of bl-cc test is: the strap is kept under load in the low-temperature chamber.*
   *Step 5 of bl-cc test is: the mass is removed.*
   *Step 6 of bl-cc test is: the breaking load is measured.*
3. *The load-duration is the time between the start of step 4 and of step 5.*
   *It is obligatory that the load duration be greater than 30mn.*
   *The removal-measure-delay is the time between the removal of the strap from the cold-chamber after step 5 and the end of step 6.*
   *It is obligatory that the removal-measure-delay be less than 5 mn.*

**Example 2.** The second example comes from the EU-Rent case of [18]. EU-Rent is a car renting company. The extracted fragment (version 1) is related to branches but nothing is specified for the pick-up and return of cars from and to branches. The knowledge engineer has introduced a specific rule, dealing with the effective and specified drop-off locations (version 2). The obligation is discussed in the following section but we can see here how the ontology is refined to allow writing this version. The fact type "a car is returned to a branch" is broken down into several fact types with the help of added domain vocabulary. New vocabulary (in bold face) and fact types are gathered in fragment 3.

1. *A Local area contains a number of Branches for Rental Car pick-up and return. A rental booking specifies [...] the EU-Rent branch from which the rental is to start. Optionally, the reservation may specify a one-way rental (in which the car is returned to a branch different from the pick-up branch)*
2. *It is obligatory that the rental incurs a location penalty charge if the drop-off location of a rental is not the EU-Rent site that is base for the return branch of the rental.*
3. *A return branch has a* ***base****. The* ***base*** *of the return branch is an* ***EU-Rent site****. A rental has a* ***drop-off location****. The* ***drop-off location*** *is the* ***base*** *of the return branch.*

## 5.2 Exhibiting Decision Variables

When transforming normalized rules into decision rules, the main point consists in getting rid of the modal operators, while preserving as most as possible the meaning of the source fragment in its context. A second operation therefore consists in making explicit some variables related to the decision to take, which often remains implicit.

**Example 1.** The first example is from Haley's blog[15], which describes the formalization of the rules to qualify for the earned income credit (EIC). The source text is a guide provided by the administration to the applicant. The first extracted rule is the version 1 of the following example (slightly simplified for the sake of brievity). The normalization (version 2) requires to restore a premise from the context (the person is applying for EIC), clarify cardinalities (if you have two children, you do not have one) and normalize the obligation. Haley points that "must" is misleading here since the proposition under its scope is not an obligation for the applicant, rather a condition of success. His re-statement of the rule is merged in version 3. "Being qualified for the EIC" is a new concept. It does not describe the specific data related to the applicant, as do the income and the number of children. It states that the data are not conformant wrt. the intended model. We call these variables *decision variables.*

1. *If you have one qualifying child, your Adjusted Gross Income (AGI) must be less than \$29,666.*
2. *If a person applies for the EIC and this person has exactly one qualifying child, then it is obligatory that this person's AGI is less than \$29,666.*
3. *If you apply for the EIC and you have exactly one qualifying child and your AGI is more than \$29,666, then you do not qualify for EIC.*

**Example 2.** Let us return to the breaking load test after cold-conditioning introduced in section 5.1. If the obligations stated in 1 are not fulfilled, no valid conclusion can be drawn from the test. This is different from a failure, which is the case when a valid breaking load is obtained with a value is under a given threshold. It is of course not a success either. A decision variable (validity of the bl-cc test) is introduced to account for that in version 2.

1. *It is obligatory that the load duration be greater than 30mn.*
   *It is obligatory that the removal-measure-delay be less than 5 mn.*
2. *If the load duration is less than 30mn, the bl-cc test is invalid.*
   *If the removal-measure-delay is more than 5 mn, the bl-cc test is invalid.*

**Example 3.** The EU-Rent case has a main obligation specified by the rental booking (statement 1). It is again accounted with the help of a decision variable (version 2).

1. *It is obligatory that the pick-up location be the start branch of the rental.*
   *It is obligatory that the drop-off location be the return branch of the rental.*
2. *If the drop-off location of the rental is not the base of the return branch of this rental, then this rental is non-conformant-for-return.*

The technique proposed here clearly separates two questions. The decision-variables are used to reflect in the model that an obligation has not been fulfilled.

[15] `http://haleyai.com/wordpress/2008/03/28/harvesting-business-rules-from-the-irs/`

In this case, they are introduced in the conclusion of rules, which conditions are obtained from the first SBVR modeling step, from specialized modeling entities and the (negated) first order content of the obligation. This raises a correlated question regarding what to do when the obligation is actually violated. Section 5.3 addresses this point.

## 5.3 Specifying Actions

Deciding what to do when a decision point has been reached involves more application specific knowledge than the previous ones, because regulations remain relatively application independent and do not examine things beyond the obligation. Possible answers can be divided into three groups :

- Abandon, stop the process, do nothing;
- Retry the same process, after modifying one of the conditions;
- Use a remedial subprocess as a continuation after the decision point. The remedial process is often not mentioned by the regulation, and the application specialist generally has a major role in its description.

The first group is illustrated by the EIC case, at least when the application is to help the user to fill a statement of income and a EIC form. The resulting rule could be 1. Of course, other actions are possible. Formally, the EIC case could also yield to 2 and 3.

1. *If you do not qualify for the EIC, then don't fill the EIC form.*
2. *If you do not qualify for the EIC, then reduce your AGI [to $29,666].*
3. *If you do not qualify for the EIC, then apply for one more child.*

The second group is illustrated by the UNO Regulation n°16 case. The resulting rule can be

> *If the bl-cc test is invalid, then the test must be started afresh with a new strap.*

The last group is illustrated by the EU-Rent case. The following rule is suggested by the knowledge engineer. The process is neither abandoned, nor redone. The drop-off is accepted as is, but the charge is increased.

> *If a rental is non-conformant-for-return, this rental incurs a location penalty charge.*

The above examples show that what remains to be done, once the rules are well-formed SBVR-statement, is of a different nature from the normalization steps. This is the reason why we argue that the knowledge engineer who normalizes the source text and formalizes the resulting candidate rule should be different actors. The first step is driven by the source text and a general idea of the organization to model. The second one directly depends on how the candidate rules must be operationalized, and which part of this operational semantics is relevant for the rule system – for deciding, checking or warning. It makes sense that these interpretations be made before implementation and remain accessible to business people through SBVR.

## 6 Conclusion

In order to formalize NL regulations into production rules, we have argued for a three steps process. We have focused here on the first two, which both rely on a controlled language but reflect different tasks devoted to different actors. The clarification of the regulation is under the responsibility of a general-level expert. Its refinement also involves an engineer who knows how the rule system works. We have argued that SBVR is convenient as a controlled language. It allows expressing the new formulations while remaining understandable by business people who want to remain in charge of the rule authoring. Then we have described a methodology for acquiring rules from regulatory texts and transforming them into production rules, providing different operations for each step. The methodology is illustrated with examples from the literature and from the Ontorule use cases. It shows the importance of SBVR and more generally controlled languages as flexible intermediate languages at the border between natural and formal languages.

## References

1. Bajec, M., Krisper, M.: Issues and challenges in business rule-based information systems development. In: ECIS (2005)
2. Bajwa, I.S., Lee, M.G., Bordbar, B.: Sbvr business rules generation from natural language specification. In: AAAI Spring Symposium 2011 Artificial Intelligence 4 Business Agility, pp. 541–545, AAAI, San Francisco (2011), `www.aaai.org/ocs/index.php/SSS/SSS11/paper/download/2378/2918`
3. Baumeister, J., Reutelshoefer, J., Puppe, F.: Engineering intelligent systems on the knowledge formalization continuum. International Journal of Applied Mathematics and Computer Science (AMCS) 21(1) (2011)
4. BRG: Defining business rules – what are they really? The Business Rules Group : formerly, known as the GUIDE Business Rules Project - Final Report revision 1.3 (July, 2000)
5. Brodie, C., Karat, C.M., Karat, J.: An empirical study of natural language parsing of privacy policy rules using the sparcle policy workbench. In: SOUPS 2006 (2006)
6. Chandrasekar, R., Doran, C., Srinivas, B.: Motivations and methods for text simplification. In: Proceedings of the Sixteenth International Conference on Computational Linguistics (COLING 1996), pp. 1041–1044 (1996)
7. Chandrasekar, R., Srinivas, B.: Automatic induction of rules for text simplification (1997)
8. Dinesh, N., Joshi, A., Lee, I., Sokolsky, O.: Reasoning about conditions and exceptions to laws in regulatory conformance checking. In: van der Meyden, R., van der Torre, L. (eds.) DEON 2008. LNCS (LNAI), vol. 5076, pp. 110–124. Springer, Heidelberg (2008), `http://repository.upenn.edu/cis_papers/371`
9. Dubauskaite, R., Vasilecas, O.: An open issues in business rules based information system development. In: Innovative Infotechnologies for Science, Business and Education, vol. 1 (2009)
10. Gasperin, C., Specia, L., Pereira, T.F., Aluisio, S.M.: Learning when to simplify sentences for natural text simplification. In: ENIA 2009 (VII Encontro Nacional de Inteligência Artificial) (2009)

11. The Object Management Group: Semantics of business vocabulary and business rules. Tech. rep., The Object Management Group (2008), `http://www.omg.com/`
12. von Halle, B., Goldberg, L.: The Decision Model: A Business Logic Framework Linking Business and Technology. Taylor & Francis, LLC, Auerbach (2009)
13. Halle, B., Goldberg, L., Zackman, J.: Business Rule Revolution: Running Business the Right Way. Happy About (2006), `http://books.google.com/books?id=I3mvAAAACAAJ`
14. Linehan, M.H., de Sainte Marie, C.: The relationship of decision model and notation (dmn) to sbvr and bpmn. Business Rules Journal 12(6) (June 2011), `http://www.BRCommunity.com/a2011/b597.html`
15. Max, A.: Simplification interactive pour la production de textes adaptés aux personnes souffrant de troubles de la compréhension. In: Proceedings of TALN, Poster Session (2005), `http://www.limsi.fr/Individu/amax/recherche/articles/Max-TALN05.pdf`
16. Mendonça, D., Wallace, W.A.: Development of a decision logic to support group improvisation: An application to emergency response. In: 35th Hawaii International Conference on System Sciences (2002)
17. Nazarenko, A., Guissé, A., Lévy, F., Omrane, N., Szulman, S.: Integrating Written Policies in Business Rule Management Systems. In: Bassiliades, N., Governatori, G., Paschke, A. (eds.) RuleML 2011 - Europe. LNCS, vol. 6826, pp. 99–113. Springer, Heidelberg (2011)
18. OMG: Sbvr (2008), `http://www.omg.org/spec/SBVR/Current`
19. Omrane, N., Nazarenko, A., Rosina, P., Szulman, S., Westphal, C.: Lexicalized ontology for a business rules management platform: An automotive use case. In: Olken, F., Palmirani, M., Sottara, D. (eds.) RuleML - America 2011. LNCS, vol. 7018, pp. 179–192. Springer, Heidelberg (2011)
20. Ross, R.G.: Principles of the Business Rule Approach, ch. 8-12. Addison-Wesley, Boston (2003)
21. Ross, R.G.: Decision analysis using decision tables and business rules. Tech. rep., Business Rule Solutions (2010)
22. Siddharthan, A., Caius, G.: Syntactic simplification and text cohesion (2003)
23. Spreeuwenberg, S.: Rule authoring is a creative process. Business Rules Journal 10(9) (September 2009), `http://www.BRCommunity.com/a2009/b497.html`
24. Unger, C., Bühmann, L., Lehmann, J., Ngomo, A.C.N., Gerber, D., Cimiano, P.: Sparql template based question answering. In: 21st International World Wide Web Conference, WWW 2012 (April 2012), `http://data.semanticweb.org/conference/www/2012/paper/1290`