Visual SBVR

Prakash Musham prakashm@iitk.ac.in
Sharad Singh sharad.singh@theingen.com
Rashi Bahal rashi.bahal@gmail.com
Prabhakar TV tvp@iitk.ac.in

Abstract

The SBVR (Semantics of Business Vocabulary and Rules) vocabulary and rules can be represented in different ways like Structured English, RuleSpeak Business Rule Notation and ORM Notation etc. These are textual notations and they won't help in understanding the meaning of a business rule at a glance. In this paper we are proposing a visual notation for SBVR which enables easy comprehension of the rules. We also demonstrate its feasibility and utility by building an editor tool.

1. Introduction

SBVR provides the vocabulary to describe the business rules. It can be represented in different ways, some of them are Structured English, RuleSpeak Business Rule Notation and ORM Notation [1], but there is no representation by which we can easily understand the rule at a glance. In this paper, we are proposing a visual syntax for the SBVR. This visual syntax is developed on the Structured English.

This paper is organized as follows. Section II provides introduction to SBVR, Section III explains SBVR's Structured English representation, Section IV contains the proposed visual syntax for SBVR and finally Section V describes the SBVR Visual Editor tool which is used for representing and developing SBVR Visual syntax.

2. SBVR

SBVR has been declared as a standard by OMG (Object Management Group) on December 11, 2007 [2]. It can be used as a meta-model to describe the business vocabulary and business rules [1].

SBVR is intended for business community to describe business rules. For effective development of the business, they need complete analysis of their business. For this purpose business analysts express their ideas in their own ways. In this process of transferring their ideas between different communities may lead to misunderstandings. Such problems can be overcome by using SBVR which provides a generalized vocabulary to describe their business rules.

The basic mantra behind the SBVR is, "Rules are built on facts, and facts build on concepts as expressed by terms" [1] which is described in Figure 1 with example.

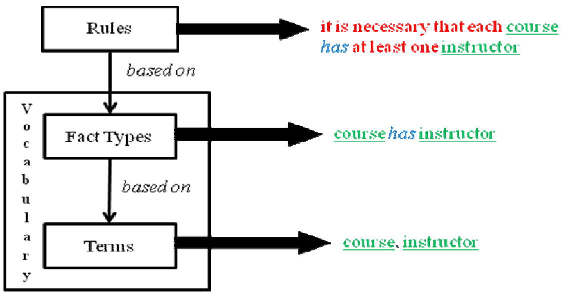


Figure 1: Business Mantra

SBVR has its own set of terminology and keywords to describe the business vocabulary and business rules. The next two subsections give an introduction with its terminology given in SBVR's specification [1].

2.1. SBVR Business Vocabulary

SBVR Business Vocabulary includes terms, names and Facts.

Term is a noun concept and it is represented by a word or a group of words used to represent a business entity.

e.g. 'course' or 'personal computer'

Name is an individual concept and it is represented by a word or a group of words which can be used to represent an instance of a particular term.

e.g. 'Compaq', which is a type of 'laptop'

Fact types are the sentences in SBVR vocabulary which establishes a relationship between the terms/Name. Every fact can be represented in the form of a *term/Name-verb-term/Name* template.

e.g. 'course has instructor'

2.2. SBVR Business Rules

Business Rule is a rule that is under business jurisdiction which portrays the structure and behavior of an organization [1]. The business rules guiding structure of the organization are known as "Structural business rules" and the business rules guiding the behavior of the organization are known as "Operative business rules".

3. SBVR Structured English

SBVR Structured English is a procedure of using English that maps mechanically to SBVR concept It uses a small number of English structures and common words to provide a simple and straightforward mapping [1].

SBVR Structured English provides different fonts to describe the SBVR business rules [1] like.

Term, Name, *verb*

'term', 'Name' and 'verb' are tokens of SBVR vocabulary which are used to create fact of business vocabulary and further used to build the business rules by adding appropriate keywords. These 'keywords' are the tokens of business rules.

3.4. Keyword

Keywords are used for linguistic symbols to construct statements. Depending on the type of keyword, they are categorized as 'quantification keywords', 'logical keywords', 'modal keywords' and 'other keywords' which are listed below.

The letters '*n*' and '*m*' represents use of a literal whole number. The letters '*p*' and '*q*' represents expressions of propositions.

3.4.1. Quantification keywords

Quantification keywords represent the quantity of terms involved in business rules.

Available quantification keywords are given below.

"each", "some", "at least one", "at least *n*", "at most one", "at most *n*", "exactly one", "exactly *n*", "at least *n* at most *m*", "more than one".

e.g. "each course *has* at least one instructor"

3.4.2. Logical keywords

Logical keywords represent the logical operations involved in business rules. Available logical keywords are:

"it is not the case that *p*", "*p* and *q*", "*p* or *q*", "*p* or *q* but not both", "if *p* then *q*", "*q* if *p*", "*p* if and only if *q*", "not both *p* and *q*", "neither *p* nor *q*", "*p* whether or not *q*".

e.g. "each course *has* at least one project or term paper".

3.4.3. Modal keywords

Modal keywords add a modal operation to business rules. It can be of two types one is 'prefix modal keywords' and the second one is 'embedded modal keywords'.

Prefix modal keywords are classified into structural and operative prefix modal keywords.

Structural prefix modal keywords:
"it is necessary that",
"it is possible that", and
"it is impossible that".

Operative prefix modal keywords:
"it is obligatory that",
"it is permitted that" and
"it is prohibited that".

e.g. "it is necessary that each course *has* at least one instructor"

Embedded modal keywords are also classified into structural and operative embedded modal keywords.

Structural embedded modal keywords:
"… always …"
"… can …" and
"… never …"

Operative embedded modal keywords:
"… must …"
"… may …" and
"… must not …"

e.g. "each course always *has* at least one instructor"

3.4.4. Other keywords

These are supporting keywords which help in building business rules.

e.g. “the”, “a”, “an”, “another”, “a given”, etc.

The ‘concept map’ [3] in Figure 2 shows all the available keywords in SBVR Structured English. More information about the SBVR Structured English is available in [1].

4. Visual Syntax for SBVR

SBVR can be represented in different ways, some of them are Structured English, RuleSpeak Business Rule Notation, and ORM Notation [1], but there is no representation by which we can easily understand the rule at a glance. In this paper, we are proposing a visual syntax for SBVR which helps us to understand the meaning of a business rule by looking at it once.

Our visual syntax is developed on top of Structured English which includes a separate graphical symbol for each expression of the Structured English.

4.1. Need of Visual Syntax

“*A picture is worth a thousand words*”

The main aim behind visual syntax is that it is easy to comprehend. Each expression of SBVR Structured English is represented with an interactive structure, which improves user-friendliness. Many of the complex rules can be easily formed by the proposed visual syntax. A long rule can be visualized instantaneously. All the related facts with a particular term/ name can be viewed in one spot. For example, by looking in Figure 3 we can easily understand the relation among the terms. The meaning of each symbol is explained in further sections.

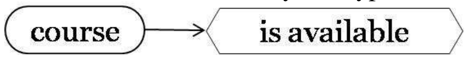
Next sub sections provide SBVR Structured English expressions and corresponding proposed visual syntax for that expression.

4.2. Tokens of SBVR

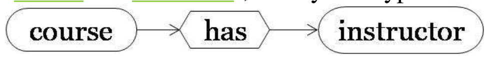
First we will see the tokens of the business vocabulary. Table 1 shows some elementary tokens.

Facts are in the form of *term/name-verb-term/name* template. They are represented in visual syntax as below:

‘course *is available*’, unary fact type



‘course has instructor’, binary fact type



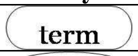
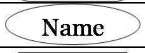
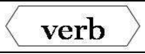
Structured English	Visual syntax
term	
Name	
verb	

Table 1: Tokens of SBVR Business Vocabulary

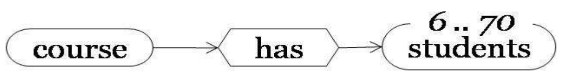
Facts are part of the business vocabulary and business rules are created by adding appropriate keywords to fact.

Tokens of SBVR Business Rules are the ‘keywords’ and their classification is given in section 3.4., next few subsections describes their Structured English notation (SE) and corresponding Visual Syntax (VS) with examples.

4.2.1. Quantification keywords

Quantification keywords (see Table 2) are applied on ‘terms’. They restrict the quantity of terms. In visual syntax, they are represented with the appropriate symbols where ‘*n*’ and ‘*m*’ is the positive integers. The symbols for the quantification keyword is been inspired from the Unified modeling language [4].

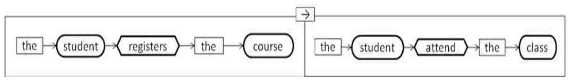
e.g. course has at least 6 at most 70 students



4.2.2. Logical keywords

These Logical keywords (see Table 3) introduce the logical operations. Logical operation can be unary or binary depending on the number of logical operands. In visual syntax the logical keyword is having its symbol representation placed on the top centre of the logical keyword with only exception in ‘it is not the case that’ keyword where the symbol is on the corner of the keyword. The symbols for these keywords are been inspired from mathematical logic.

e.g. if the student *registers* the course then the student *attends* the class



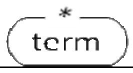
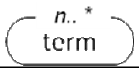
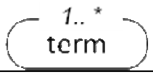
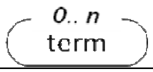
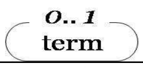
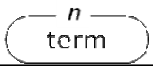
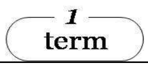
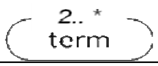
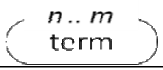
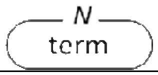
Structured English	Visual syntax
each	
at least *n*	
at least one	
at most *n*	
at most one	
exactly *n*	
exactly one	
more than one	
at least *n* at most *m*	
some	

Table 2: Quantification Keywords

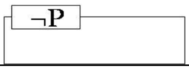
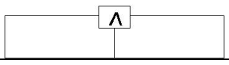
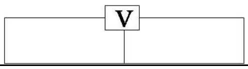
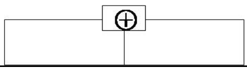
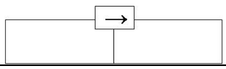
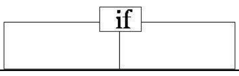
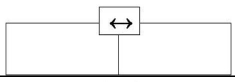
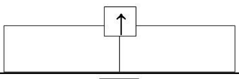
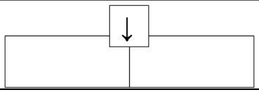
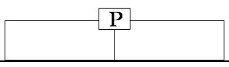
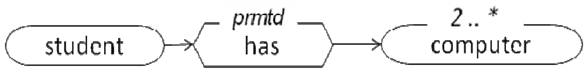
Structured English	Visual syntax
it is not the case that	
p and *q*	
p or *q*	
p or *q* but not both	
if *p* then *q*	
q if *p*	
q if and only if *p*	
not both *p* and *q*	
neither *p* nor *q*	
p whether or not *q*	

Table 3: Logical Keywords

4.2.3. Modal keywords

Modal keywords are applied on 'verbs' which introduces the modal operation in the business rules. In visual syntax they are represented with an appropriate text on top of verb.

e.g. it is permitted that student *has* more than one computer



Structural Prefix Modal Keywords

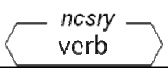
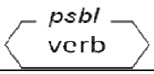
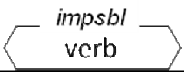
Structured English	Visual syntax
it is necessary that	
it is possible that	
it is impossible that	

Table 4: Structural Prefix Modal Keywords

Operative Prefix Modal Keywords

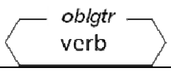
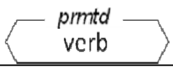
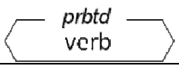
Structured English	Visual syntax
it is obligatory that	
it is permitted that	
it is prohibited that	

Table 5: Operative Prefix Modal Keywords

Structural Embedded Modal Keywords

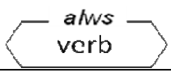
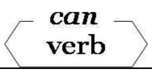
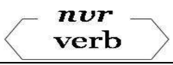
Structured English	Visual syntax
… always …	
… can …	
… never …	

Table 6 Structural Embedded Modal Keywords

Operative Embedded Modal Keywords

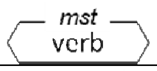
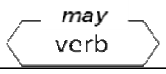
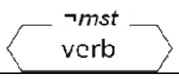
Structured English	Visual syntax
… must …	
… may …	
… must not …	

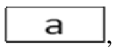
Table 7: Structural Operative Modal Keywords

4.2.4. Other keywords

These are represented with a simple box. Text in the box depends on the keyword.

The role of other keyword is to just provide a structure and a readable form to the rule and doesn't play any important role.

e.g. a, an, the

VS: , ,

5. SBVR Visual Editor

One of the methods for representing SBVR was the SBVR visual syntax. There was a need for a tool to represent and develop SBVR structured English using the visual syntax for which the SBVR Visual Editor was developed.

This java based editor is been made on top of the equinox framework [5] on top of which the eclipse editor [6] is also been made. That's why the editor will provide nice look and feel and a better working environment.

5.1. Need of SBVR Visual Editor

There was a need of a tool to represent and develop the SBVR Visual Syntax. The tool needed to be user friendly and appeals a good look and feel for the user. The flexibility and the robustness of the SBVR Visual Syntax was an important aspect that is to be maintained. The SBVR Visual Editor provides a good streamlined environment for the user to build the SBVR Structured English. Large graphical rules and facts can also be built very easily in this editor.

5.2 Perspectives of SBVR Visual Editor

The SBVR visual editor primarily has two perspectives.

- SBVR Vocabulary editor
- SBVR Rules editor

The vocabulary editor is used to create the SBVR Vocabulary and the rules editor is used to create the SBVR Rules.

There are some views in the editor which assist the user in SBVR Structured English development. The views are given below

- Palette view
- Property view
- Elements view

The palette view contains all the objects that need to be dragged and dropped to the editor. The property view is used to set the property of the elements created on the editor and the elements view is a tree view which contains an entry for each object created on the editor.

5.2.1 SBVR Vocabulary Editor

The SBVR Vocabulary Editor mainly deals with four objects which are

- Term
- Name
- Verb
- Fact

The term, name and verb are been provided in the palette and can be dragged and dropped to the editor (see Figure 4). When we connect a *term/name-verb-term/name* a fact is created and an entry is made in the element view of the fact. So the fact views model contains all the information about the facts created which will be needed while rule creation.

A direct connection between *term/name-term/name* will update some properties of these elements. The properties update will be concept type and general concept.

The term/name elements can be visualized in a tree structure where all the facts will be visible at the same time in which the given term/name have participated (see Figure 5).

The elements view will contain all the elements data which have been created on the editor. These element's entries can be used to again drag the element back to the editor if the element have been removed. The elements view can also be used in searching the elements on the editor. The view will automatically move to the selected object on the editor when the element entry will be selected in the tree view.

5.2.1.1. SBVR Fact

SBVR Facts are created by connecting *term/name-verb-term/name* and so on for creating binary and n-array facts. In a combined editor where all the functionalities are been provided together; the functionalities hampers the flexibility of each other. So the methodology of creation of n-array fact is been implemented to maintain the robustness of the architecture.

Every fact has a main verb placeholder which is like the subject of the fact and all other parts of the fact including other verbs are only for supporting the meaning represented by the fact.

If the user wants to create a fact, he will search for the main verb with which the fact will be created. The editor will search for the whole chain connected with that fact and display it on the tree dialog box. The user will select all the placeholders needed for the fact and will click confirmation to create the unary, binary or n-array fact.

5.2.2. SBVR Rules Editor

The SBVR Rules Editor has three tabbed editors. These three editors create three types of rules and also feed each other for creating more complex rules. In any point of time the rule can be declared as a final rule and so the rule will be added in the final rules list. The rule which was finalized will still be there for further rule creation.

The three types of tabbed rule editor for creating the three types of rules are:

- Embedded rules
- Modal rules
- Logical rules

These rule developing environment are different from each other and gives a more clear presentation and understanding of the rule creation with SBVR Graphical Syntax.

Each tabbed editor is like a different dimension of the graphical view. The user will create rules in these dimensions and so only the permitted rules will be created of a given dimension. Later the user will only merge these different dimensions of rules to create multidimensional rules.

Now the tabbed editor of the SBVR Rules editor will be described

5.2.2.1. Embedded Rules Editor

Embedded Rule editor will only create rules with embedded keywords. The embedded keywords include:

- Quantification keywords
- Other keywords

Quantification keywords are also known as embedded keywords since they are embedded in between the rule and so the dimension is also known as embedded rules editor. These keywords are not applied on any verb object.

Embedded rules editor will only be created by facts. Same fact can be used several times to create different types of embedded rules. Each created embedded rule is entered in the embedded element view so that the rule can be further used to create larger rules.

5.2.2.2. Modal Rules Editor

Modal Rule editor will create Modal Rules which have only modal keywords.

Modal keywords act over the verb placeholder of the fact or any other rule's embedded fact's verb. These rules can be created by:

- Facts
- embedded rules
- Logical rules.

The modal rules are divided in two primary types known as operative keywords for operative rules and structural keywords for structural rules. The editor is been divided in three parts, if the user will be working on structural rules he will see the structural modal keyword parts on the editor and if on operative rules then he will see operative modal keyword parts. There will be no palette present in this editor and the user will just drag drop elements from the fact elements view or embedded elements view or logical element view (see Figure 6)

5.2.2.3. Logical Rules Editor

Logical Rule editor will create logical rules using logical keywords.

Logical keywords mainly bind to parts of a rule. These rules are been made by using:

- Facts
- Embedded rules
- Modal rules
- Logical rules

In this dimension, the logical rule created can be again used to create another logical rule. The logical keyword object is been divided in two parts, like the keyword *p and q* will have the first compartment for p and the second for q. The compartment can hold any of the four types of input mentioned above. The user will drag and drop the logical keyword object from the palette, and then the input object is dropped in the two compartments to create the logical rule.

This logical rule can be again used as inputs to other dimension editors and so the cycle will continue until a desired final rule is not obtained.

6. Conclusion

This paper proposes a visual syntax for SBVR which gives a better knowledge representation of an organization's vocabulary and a visual editor tool which will show its feasibility.

7. Acknowledgements

We wish to thank the European Union's 6th Framework Program of Research for sponsoring this research under OPAALS [7] project.

8. References

[1] Semantics of Business Vocabulary and Business Rules (SBVR), Second SBVR Interim Specification without change bars, dtc/06-08-05 http://www.omg.org/spec/SBVR/1.0/PDF/

[2] http://www.businessrulesgroup.org/sbvr.shtml

[3] CmapTools: A Knowledge Modeling and Sharing Environment, A. J. Cañas, G. Hill, R. Carff, N. Suri, J. Lott, T. Eskridge, G. Gómez, M. Arroyo, R. Carvajal (2004). In: Concept Maps: Theory, Methodology, Technology, Proceedings of the First International Conference on Concept Mapping, A.J. Cañas, J.D. Novak, and F.M. González, Editors , Universidad Pública de Navarra: Pamplona, Spain. p. 125-133.

[4] Unified modeling language specification. Specification, Object Management Group, 2004. URL: http://doc.omg.org/ptc/2004-10-05.

[5] The equinox framework for running OSGi-based system http://www.eclipse.org/equinox/

[6]The Eclipse editor for building IDE's http://www.ibm.com/developerworks/opensource/library/os-eclipse.html

[7] Open philosophies of associative autopoietic digital ecosystems (opaals). Network of Excellence, funded by the European Union's 6th Framework Programme of research. http://www.opaals.org

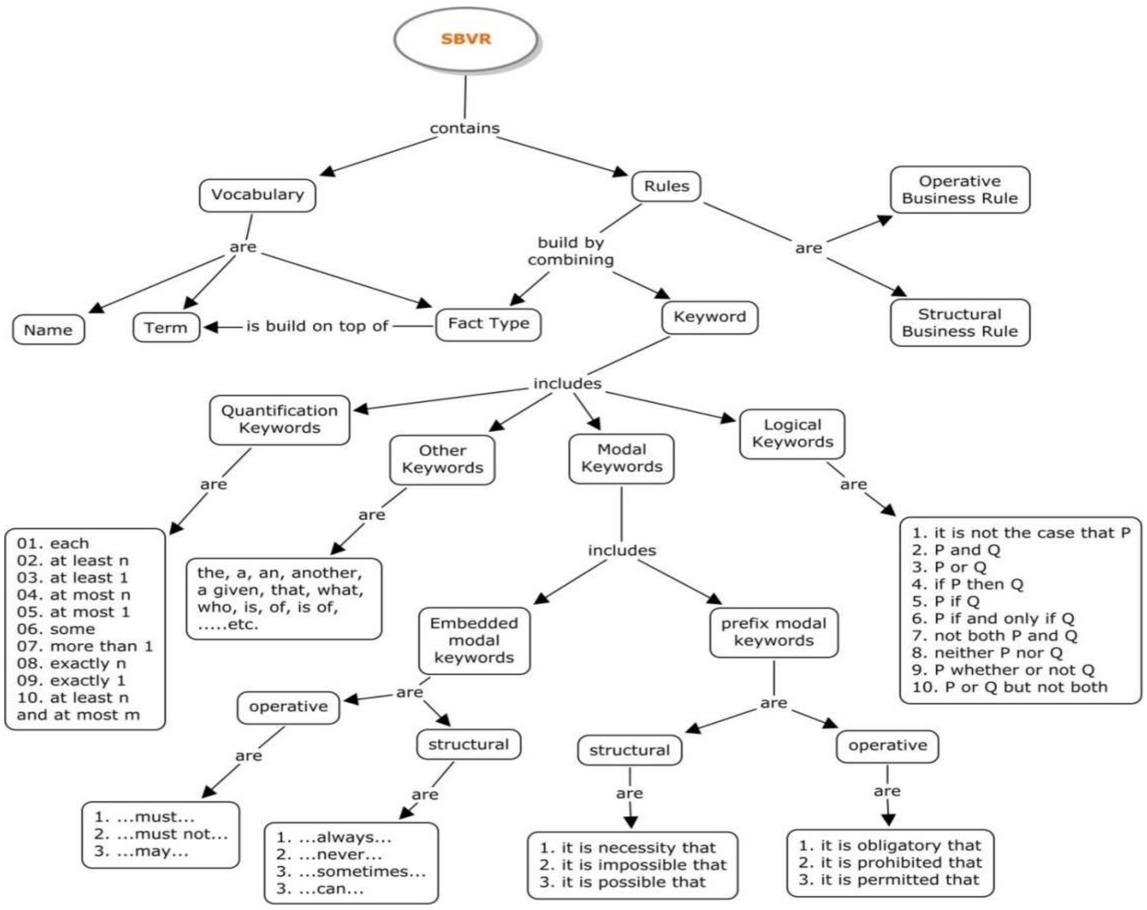


Figure 2: Concept Map for SBVR Structured English

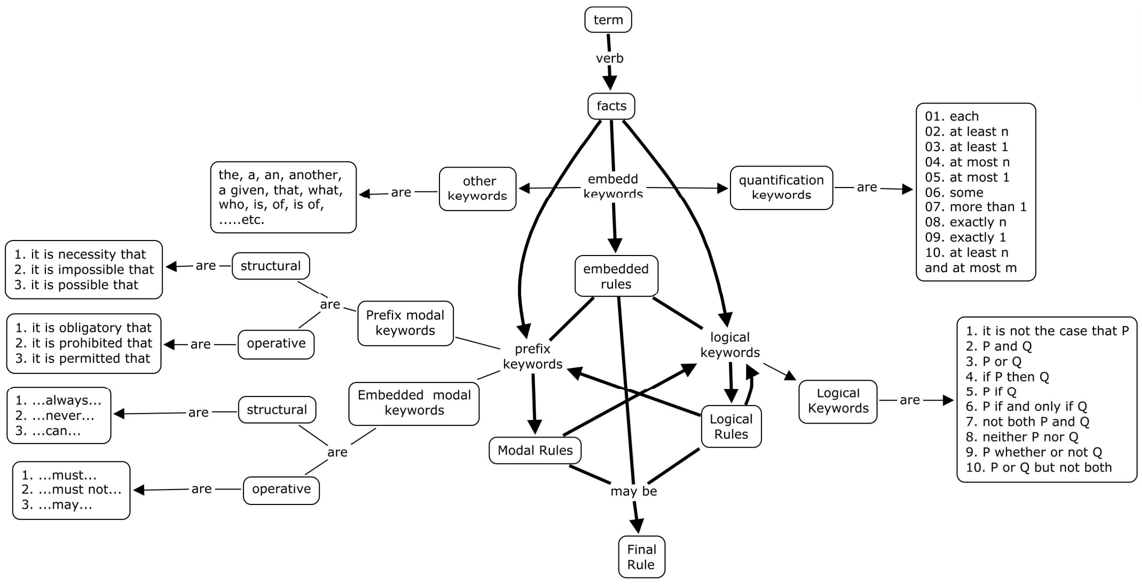


Figure 3: Concept map for the rule creation flow

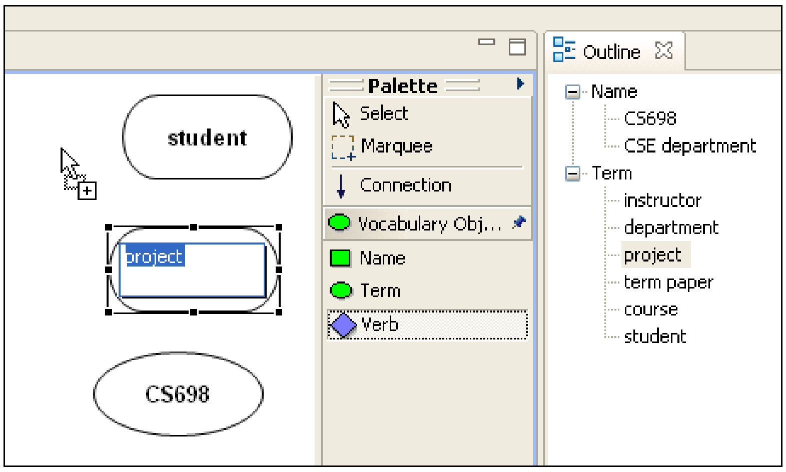
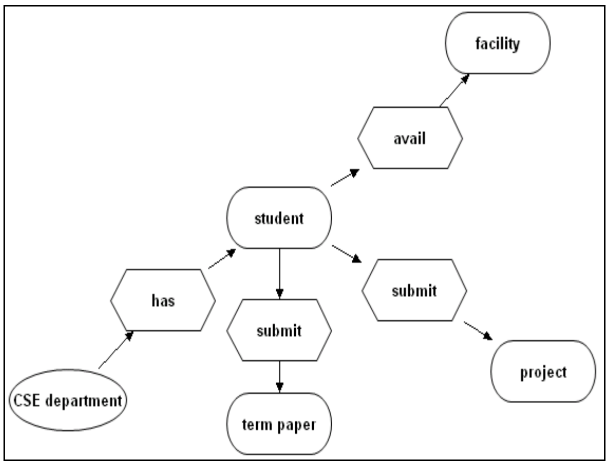


Figure 4: Vocabulary editor functionality



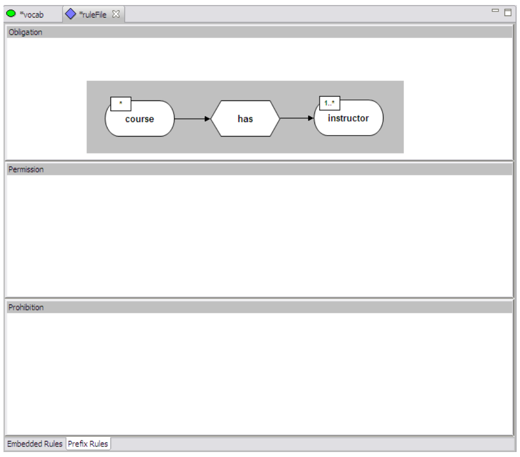


Figure 5: Tree structure of Vocabulary entries

Figure 6: Modal Rules creation