SWRL: RULE ACQUISITION USING ONTOLOGY

SWRL: LIKUMU IEGŪŠANA, IZMANTOJOT ONTOLOĢIJU

D. Plinere, A. Borisov

Keywords: ontology, rule acquisition, Protégé, SWRL rules Tab, Jess

Abstract - Nowadays rule-based systems are very common. The use of ontology-based systems is becoming ever more popular, especially in addition to the rule-based one. The most widely used ontology development platform is Protégé. Protégé provides a knowledge acquisition tool, but still the main issue of the ontology-based rule system is rule acquisition. This paper presents an approach to using SWRL rules Tab, a plug-in to Protégé, for rule acquisition. SWRL rules Tab transforms conjunctive rules to Jess rules in IF…THEN form.

Introduction

Nowadays ontology-based systems are becoming more and more popular. Many researches are done on ontology-based system application in different areas. The main advantage of ontology-based systems is the ability of communication and sharing knowledge between people and between computer systems.

The most widely-used ontology development platform is Protégé. It includes knowledge acquisition technique (Noy et al, 2000). Knowledge acquisition using Protégé has been discussed by different authors (Plinere et al., 2009).

The main issue of the ontology-based system is rule acquisition for problem solving. The research on this issue has produced the following results: rule acquisition using ontology-based system can be achieved in two ways – the first one is rule acquisition as knowledge acquisition using Protégé in three steps (Plinere et al., 2009, Noy et al, 2000), the second one is the use of SWRL rules Tab, a plug-in for Protégé, which transforms conjunctive rules into the Jess rules. It was decided in this research to choose the second way because the transformation mentioned above seemed to look more like rule acquisition.

This research was inspired by the paper by O'Connor et al (O'Connor et al, 2005). The current paper consists of six sections and is organized as follows: Section 2 presents the basics of Semantic Web Rule Language, Section 3 addresses rule acquisition technique. Section 4 discusses rule transformation from conjunctive form into Jess rule in IF…THEN form and Section 5 shows an example of the proposed approach. Finally, Section 6 provides conclusions and directions for future research.

Semantic Web Rule Language

Semantic Web Rule Language (SWRL) was designed to be the rule language of the Semantic Web. SWRL is based on a combination of the OWL DL and OWL Lite sublanguages of the OWL Web Ontology Language the Unary/Binary Datalog sublanguages of the Rule Markup Language. SWRL allows users to write Horn-like rules expressed in terms of OWL concepts to reason about OWL individuals. The rules can be used to infer new knowledge from the existing OWL knowledge bases.

The SWRL Specification does not impose restrictions on how reasoning should be performed with SWRL rules. Thus, investigators are free to use a variety of rule engines to reason with the SWRL rules stored in an OWL knowledge base. They are also free to implement their own editing facilities to create SWRL rules. In this way, SWRL provides a convenient starting point for integrating rule systems to work with the Semantic Web.

According to O'Connor et al (O'Connor et al, 2005), SWRL rules are written as antecedent consequent pairs. In SWRL terminology, the antecedent is referred to as the rule body and the consequent is referred to as the head. The head and body consist of a conjunction of one or more atoms. At present, SWRL does not support more complex logical combinations of atoms.

SWRL rules reason about OWL individuals, primarily in terms of OWL classes and properties.

For example, the most popular one, a SWRL rule expressing that a person with a male sibling has a brother would require capturing the concepts of 'person', 'male', 'sibling' and 'brother' in OWL. Intuitively, the concept of person and male can be captured using an OWL class called *Person* with a subclass *Man*; the sibling and brother relationships can be expressed using OWL properties *hasSibling* and *hasBrother*, which are attached to Person. The rule in SWRL would then be:

```
Person (?x1) ^ hasSibling (?x1,?x2)
^ Man(?x2) → hasBrother(?x1,?x2)
```

Executing this rule would have the effect of setting the *hasBrother* property to *x2* in the individual that satisfies the rule, named *x1*.

SWRL rules can also refer explicitly to OWL individuals. For instance, the following example is a variant of the above rule, inferring that a particular individual Fred has a brother:

```
Person(Fred)^hasSibling(Fred, ?x2)
^ Man(?x2) → hasBrother(Fred,?x2)
```

In this case *Fred* is the name of an OWL individual. SWRL also supports data literals. For example, assuming an individual has a *hasAge* property, it is possible to ask if *Fred* has a 40 year-old brother:

```
Person(Fred)^hasSibling(Fred,?x2)^
Man(?x2) ^ hasAge(?x2,40) →
has40YearOldBrother(Fred,?x2)
```

Rule Acquisition

SWRLTab is a plug-in for Protégé, it transfers SWRL rules using ontology to Jess (Java Expert System Shell) rules. Jess rules are represented in IF… THEN form; they use the knowledge base (ontology) for problem solving. Jess uses an enhanced version of the Rete algorithm to process rules.

SWRL rules are acquired from experts; rule acquisition is in its turn similar to knowledge acquisition: knowledge acquisition includes the elicitation, collection, analysis, modelling and validation of knowledge (Epistemics).

Some of the most important issues in knowledge acquisition are:

1. Most knowledge is in the heads of experts;
2. Experts have vast amounts of knowledge;
3. Experts have a lot of tacit knowledge;
4. They don't know all that they know and use;
5. Tacit knowledge is hard (impossible) to describe;
6. Experts are very busy and valuable people;
7. Each expert doesn't know everything.

In order to acquire knowledge (or rules, in our case) the following should be done: take the experts off the job for a while, allow non-experts to understand the knowledge, focus on the essential knowledge, capture tacit knowledge, allow knowledge to be collated from different experts and allow knowledge to be validated and maintained.

The rules were acquired from experts in the following way: firstly, an initial interview with the experts, which gave these results – knowledge of the domain area and key terminology became more understandable. It was determined what kind of rules and for what purpose they are going to employ, and the resulting protocol was built.

Secondly, the resulting protocol and the initial interview were analyzed. Concepts were created and a set of questions which cover essential issues across the domain was prepared.

The next step was a semi-structured interview with experts using pre-prepared questions. The resulting protocol with provided structure of knowledge and rules of the domain area was built and then analyzed. Knowledge types (concepts, attributes, relationships, rules and values) were defined.

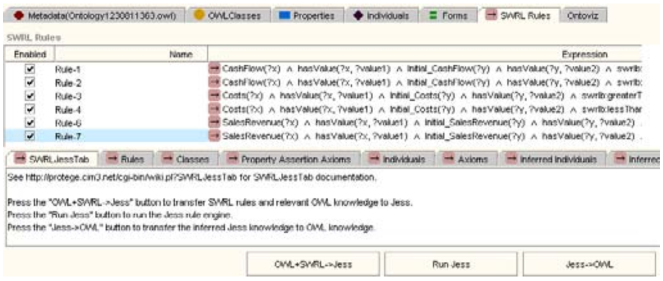
After that, the knowledge achieved was represented graphically and problem solving was presented and the experts were allowed to modify and expand on the knowledge already acquired. Analysis and model building was repeated until the experts and knowledge engineer achieved the goal of this project. And finally, the knowledge and rules acquired were validated with another expert, and no modification was done.

The rules for this research were acquired from experts; these rules are in the form of conjunction and represent all possible relations between classes or between individuals. For example:

```
CashFlow(?x) ^ hasValue (?x,
?value1) ^ Initial_CashFlow(?y) ^
hasValue(?y, ?value2) ^
swrlb:lessThan(?value1,?value2) →
CashFlow_falls(?x, ?y)
```

SWRL (Conjunctive) Rule Transformation to JESS (IF…THEN) Rule

The mechanism of interacting with SWRL in Protégé is through the SWRL rules Tab. This Tab shows all the SWRL rules in a loaded OWL knowledge base in a tabular form (see Fig. 1 and Fig. 2).

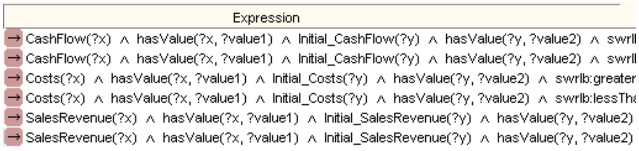


Enabled	Name	Expression
☑	Rule-1	CashFlow(?x) ∧ hasValue(?x, ?value1) ∧ Initial_CashFlow(?y) ∧ hasValue(?y, ?value2) ∧ swrlb:
☑	Rule-2	CashFlow(?x) ∧ hasValue(?x, ?value1) ∧ Initial_CashFlow(?y) ∧ hasValue(?y, ?value2) ∧ swrlb:
☑	Rule-3	Costs(?x) ∧ hasValue(?x, ?value1) ∧ Initial_Costs(?y) ∧ hasValue(?y, ?value2) ∧ swrlb:greaterT
☑	Rule-4	Costs(?x) ∧ hasValue(?x, ?value1) ∧ Initial_Costs(?y) ∧ hasValue(?y, ?value2) ∧ swrlb:lessThar
☑	Rule-6	SalesRevenue(?x) ∧ hasValue(?x, ?value1) ∧ Initial_SalesRevenue(?y) ∧ hasValue(?y, ?value2) .
☑	Rule-7	SalesRevenue(?x) ∧ hasValue(?x, ?value1) ∧ Initial_SalesRevenue(?y) ∧ hasValue(?y, ?value2) .

See http://protege.cim3.net/cgi-bin/wiki.pl?SWRLJessTab for SWRLJessTab documentation.

Press the "OWL+SWRL->Jess" button to transfer SWRL rules and relevant OWL knowledge to Jess.
Press the "Run Jess" button to run the Jess rule engine.
Press the "Jess->OWL" button to transfer the inferred Jess knowledge to OWL knowledge.

Fig. 1. Protégé SWRL rules Tab



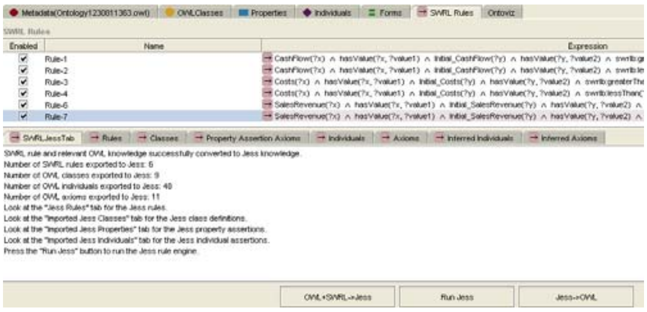
Expression
- CashFlow(?x) ∧ hasValue(?x, ?value1) ∧ Initial_CashFlow(?y) ∧ hasValue(?y, ?value2) ∧ swrll
- CashFlow(?x) ∧ hasValue(?x, ?value1) ∧ Initial_CashFlow(?y) ∧ hasValue(?y, ?value2) ∧ swrll
- Costs(?x) ∧ hasValue(?x, ?value1) ∧ Initial_Costs(?y) ∧ hasValue(?y, ?value2) ∧ swrlb:greater
- Costs(?x) ∧ hasValue(?x, ?value1) ∧ Initial_Costs(?y) ∧ hasValue(?y, ?value2) ∧ swrlb:lessTha
- SalesRevenue(?x) ∧ hasValue(?x, ?value1) ∧ Initial_SalesRevenue(?y) ∧ hasValue(?y, ?value2)
- SalesRevenue(?x) ∧ hasValue(?x, ?value1) ∧ Initial_SalesRevenue(?y) ∧ hasValue(?y, ?value2)

Fig. 2. Protégé SWRL rules Tab (enlarged)

The SWRL Editor allows users to enter rules as text; on the other hand, it also allows users to select OWL entities from a currently loaded knowledge base and to insert them into the rule, for example, it allows selecting OWL classes, properties and individuals. SWRL Editor also includes SWRL built-ins and performs syntactic and semantic checking.

It is possible to transfer SWRL rules and relevant OWL knowledge to Jess by pressing “OWL+SWRL->Jess” button (see Fig. 1, left button below).

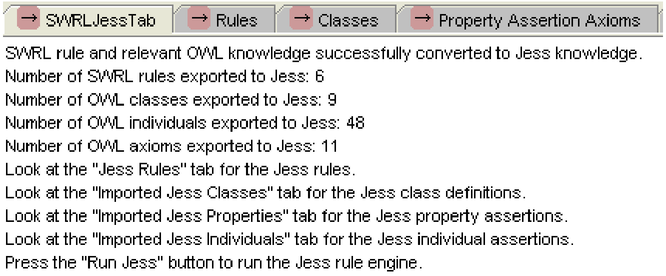
After pressing the “OWL+SWRL->Jess” button, SWRL Editor has the following interface (see Fig. 3 and Fig. 4): SWRL rule and relevant OWL knowledge successfully converted to Jess knowledge. It is possible to see the rules converted to Jess rules, Jess class definitions, Jess property and individual assertions.



Enabled	Name	Expression
☑	Rule-1	CashFlow(?x) ∧ hasValue(?x, ?value1) ∧ Initial_CashFlow(?y) ∧ hasValue(?y, ?value2) ∧ swrlb:g
☑	Rule-2	CashFlow(?x) ∧ hasValue(?x, ?value1) ∧ Initial_CashFlow(?y) ∧ hasValue(?y, ?value2) ∧ swrlb:le
☑	Rule-3	Costs(?x) ∧ hasValue(?x, ?value1) ∧ Initial_Costs(?y) ∧ hasValue(?y, ?value2) ∧ swrlb:greaterTh
☑	Rule-4	Costs(?x) ∧ hasValue(?x, ?value1) ∧ Initial_Costs(?y) ∧ hasValue(?y, ?value2) ∧ swrlb:lessThan(
☑	Rule-6	SalesRevenue(?x) ∧ hasValue(?x, ?value1) ∧ Initial_SalesRevenue(?y) ∧ hasValue(?y, ?value2) ∧
☑	Rule-7	SalesRevenue(?x) ∧ hasValue(?x, ?value1) ∧ Initial_SalesRevenue(?y) ∧ hasValue(?y, ?value2) ∧

```
SWRL rule and relevant OWL knowledge successfully converted to Jess knowledge.
Number of SWRL rules exported to Jess: 6
Number of OWL classes exported to Jess: 9
Number of OWL individuals exported to Jess: 48
Number of OWL axioms exported to Jess: 11
Look at the "Jess Rules" tab for the Jess rules.
Look at the "Imported Jess Classes" tab for the Jess class definitions.
Look at the "Imported Jess Properties" tab for the Jess property assertions.
Look at the "Imported Jess Individuals" tab for the Jess individual assertions.
Press the "Run Jess" button to run the Jess rule engine.
```

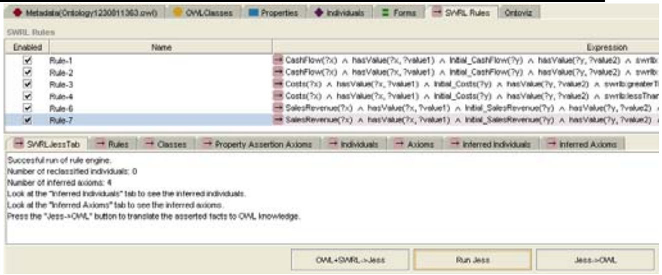
Fig. 3. SWRL rule Tab interface after rule convertation



```
SWRL rule and relevant OWL knowledge successfully converted to Jess knowledge.
Number of SWRL rules exported to Jess: 6
Number of OWL classes exported to Jess: 9
Number of OWL individuals exported to Jess: 48
Number of OWL axioms exported to Jess: 11
Look at the "Jess Rules" tab for the Jess rules.
Look at the "Imported Jess Classes" tab for the Jess class definitions.
Look at the "Imported Jess Properties" tab for the Jess property assertions.
Look at the "Imported Jess Individuals" tab for the Jess individual assertions.
Press the "Run Jess" button to run the Jess rule engine.
```

Fig. 4. SWRL rule Tab interface after rule convertation (enlarged)

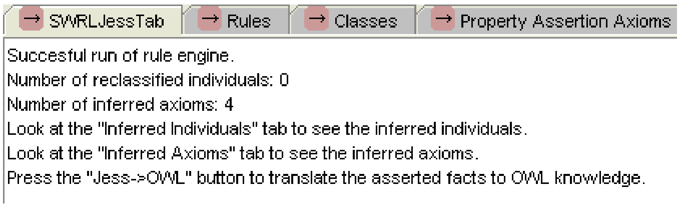
SWRL rule Tab offers to press the “Run Jess” button in order to run the Jess rule engine. The result of pressing this button is shown in Fig. 5 and Fig. 6.



Enabled	Name	Expression
☑	Rule-1	CashFlow(?x) ∧ hasValue(?x, ?value1) ∧ Initial_CashFlow(?y) ∧ hasValue(?y, ?value2) ∧ swrlb:
☑	Rule-2	CashFlow(?x) ∧ hasValue(?x, ?value1) ∧ Initial_CashFlow(?y) ∧ hasValue(?y, ?value2) ∧ swrlb:
☑	Rule-3	Costs(?x) ∧ hasValue(?x, ?value1) ∧ Initial_Costs(?y) ∧ hasValue(?y, ?value2) ∧ swrlb:greaterTl
☑	Rule-4	Costs(?x) ∧ hasValue(?x, ?value1) ∧ Initial_Costs(?y) ∧ hasValue(?y, ?value2) ∧ swrlb:lessThar
☑	Rule-6	SalesRevenue(?x) ∧ hasValue(?x, ?value1) ∧ Initial_SalesRevenue(?y) ∧ hasValue(?y, ?value2)
☑	Rule-7	SalesRevenue(?x) ∧ hasValue(?x, ?value1) ∧ Initial_SalesRevenue(?y) ∧ hasValue(?y, ?value2)

```
Succesful run of rule engine.
Number of reclassified individuals: 0
Number of inferred axioms: 4
Look at the "Inferred Individuals" tab to see the inferred individuals.
Look at the "Inferred Axioms" tab to see the inferred axioms.
Press the "Jess->OWL" button to translate the asserted facts to OWL knowledge.
```

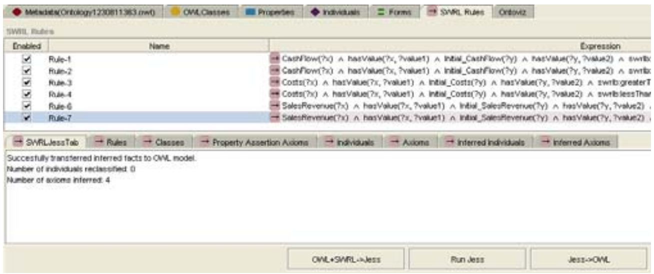
Fig. 5. SWRL rule Tab interface after successful run of Jess rule engine



```
Succesful run of rule engine.
Number of reclassified individuals: 0
Number of inferred axioms: 4
Look at the "Inferred Individuals" tab to see the inferred individuals.
Look at the "Inferred Axioms" tab to see the inferred axioms.
Press the "Jess->OWL" button to translate the asserted facts to OWL knowledge.
```

Fig. 6. SWRL rule Tab interface after successful run of Jess rule engine (enlarged)

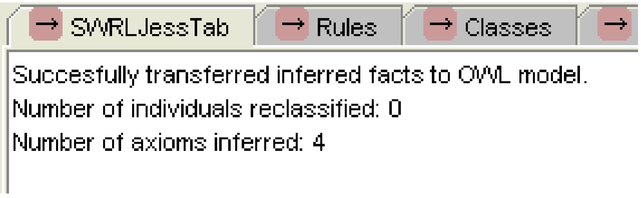
Interaction between the SWRL Editor and the Jess rule engine is user-driven. The user controls when OWL knowledge and SWRL rules are transferred to Jess, when inference is performed using that knowledge and rules, and when the resulting Jess facts are transferred back to Protégé-OWL as OWL knowledge, by pressing “Jess-> OWL” button (see Fig. 7 and Fig. 8).



Enabled	Name	Expression
☑	Rule-1	CashFlow(?x) ∧ hasValue(?x, ?value1) ∧ Initial_CashFlow(?y) ∧ hasValue(?y, ?value2) ∧ swrlb:
☑	Rule-2	CashFlow(?x) ∧ hasValue(?x, ?value1) ∧ Initial_CashFlow(?y) ∧ hasValue(?y, ?value2) ∧ swrlb:
☑	Rule-3	Costs(?x) ∧ hasValue(?x, ?value1) ∧ Initial_Costs(?y) ∧ hasValue(?y, ?value2) ∧ swrlb:greaterT
☑	Rule-4	Costs(?x) ∧ hasValue(?x, ?value1) ∧ Initial_Costs(?y) ∧ hasValue(?y, ?value2) ∧ swrlb:lessThar
☑	Rule-6	SalesRevenue(?x) ∧ hasValue(?x, ?value1) ∧ Initial_SalesRevenue(?y) ∧ hasValue(?y, ?value2)
☑	Rule-7	SalesRevenue(?x) ∧ hasValue(?x, ?value1) ∧ Initial_SalesRevenue(?y) ∧ hasValue(?y, ?value2)

```
Succesfully transferred inferred facts to OWL model.
Number of individuals reclassified: 0
Number of axioms inferred: 4
```

Fig. 7. SWRL rule Tab interface after successful transferring of the new knowledge to Protégé OWL



```
Succesfully transferred inferred facts to OWL model.
Number of individuals reclassified: 0
Number of axioms inferred: 4
```

Fig. 8. SWRL rule Tab interface after successful transferring of the new knowledge to Protégé OWL (enlarged)

O’Connor et al. (O’Connor et al, 2005) explain how it works as follows: the Jess system consists of a rule base, a fact base and an execution engine. The execution engine matches facts in the fact base with rules in the rule base. These rules can assert new facts and put them in the fact base or execute Java functions.

SWRL rules speak about OWL individuals, primarily in terms of OWL classes and properties. When a SWRL rule is fired, it can create new classifications for existing individuals. For example, if a rule consequent asserts that an individual is to be classified as a member of a particular class, that individual must be made a member of that class within OWL when the rule fires. Similarly, if a SWRL rule asserts that two individuals are related via a particular property, then that property must be associated with each individual that satisfies the rule.

According to O’Connor et al., four main tasks must be performed to allow Jess to interoperate with the SWRL Editor: (1) represent relevant knowledge about OWL individuals as Jess facts; (2) represent SWRL rules as Jess rules; (3) perform inference using those rules and reflect the results of that inference in an OWL knowledge base; and (4) control that interaction from a graphical interface.

Example of the Proposed Approach

Domain Area

The product life cycle concept suggests that a product passes through four stages of evolution: introduction, growth, maturity and decline (see Fig. 9). As a product evolves and passes through these four stages, profit is affected and different strategies (Rudenko et al., 2008, Marketing teacher) have to be employed to ensure that the product is a success within its market.

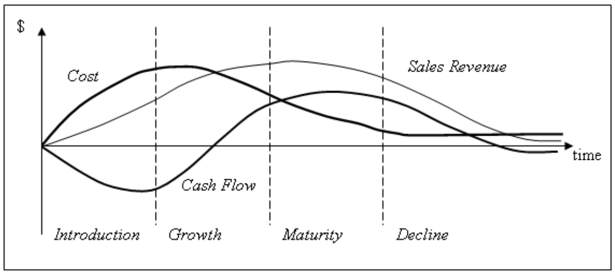


Fig. 9. Product life cycle

According to (Marketing teacher, Online postgraduate courses), in reality very few products follow such a prescriptive cycle. The length of each stage varies enormously. The decisions of marketers can change the stage, for example from maturity to decline, by price-cutting. It is not easy to tell which stage the product is in. Companies often try to use extension strategies in order to try to delay the decline stage of the product life cycle. The maturity stage is a good stage for the company in terms of generating cash. The costs of developing the product and establishing it in the market are paid and then it tends to be at a profitable stage. The longer the company can extend this stage, the better it will be for them (Marketing teacher, Online postgraduate courses).

Marketers and managers cannot define which of the stages the product is in at the moment without some additional technique; all they have is statistics of the previous time periods. For example, values and appropriate time periods of the following curves are available for marketers and managers – cash flow, sales revenue, and development and manufacture costs.

The main question for marketers and managers is – what is the stage the product is in at the moment?

Problem Solving Using SWRL Rules and Ontology

According to O’Connor et al., relevant knowledge about OWL individuals must be represented as Jess knowledge. The two primary properties that must be represented are the following: 1) the classes to which the individual belongs and 2) the properties the individual possesses.

The Jess template facility provides a mechanism for representing an OWL class hierarchy. A Jess template hierarchy can be used to model an OWL class hierarchy using a Jess slot to hold the name of the individual belonging to the hierarchy. Thus, for example, a user must define a Jess template to represent the OWL:Thing class:

```
  (deftemplate OWL:Thing (slot name))
```

A hierarchy representing a class SalesRevenue that subclasses a direct subclass of OWL:Thing called Curves could then be represented as follows in Jess:

```
  (deftemplate     Curves     extends
OWL:Thing)
  (deftemplate  SalesRevenue  extends
Curves)
```

Using this template definition, an OWL individual can be asserted as a member of the class SalesRevenue:

```
  (assert     (SalesRevenue     (name
SalesRevenue_1)))
```

The representation of SWRL rules in Jess using these facts is relatively straightforward. For example, take the following SWRL rule:

```
  SalesRevenue  (?x)  ^  hasValue  (?x,
?value1) ^ Initial_SalesRevenue (?y)
^    hasValue    (?y,    ?value2)    ^
```

```
swrlb:greaterThan  (?value1,  ?value2)
→  SRevenue_grows (?x, ?y)
```

This rule can be represented in Jess — using the representation of individuals outlined above — as follows:

```
   (defrule  aRule  (SalesRevenue  (name
?x)) (Initial_SalesRevenue (name ?y))
(hasValue  ?x  ?value1)  (hasValue  ?y
?value2) (test (> ?value1 ?value2))
   => (assert (SRevenue_grows ?x ?y))
```

Starting this research, a simple example with input data shown in Table 1 was used for the proposed approach. During the research different initial data were used in order to check the correctness of the rules work.

CashFlow_0, Costs_0 and SalesRevenue_0 in the ontology became Initial_CashFlow_0, Initial_Costs_0 and Initial_SalesRevenue_0 because of their initial values 0.

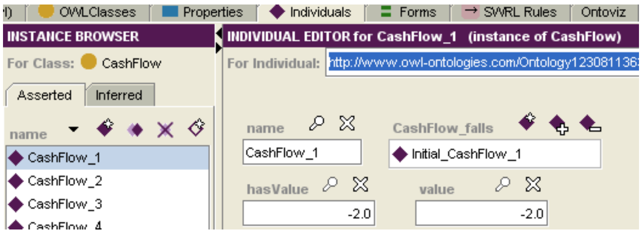
SWRL rules represent all possible relationships between ontology parts, but in case of using SWRL rules Tab in Protégé, it can be defined as rule acquisition using ontology in IF…THEN form and because only those rules that are necessary for problem solving, are fired.

Table 1

Initial data in the research

CashFlow_0	0
CashFlow_1	-2
CashFlow_2	-5
CashFlow_3	-2
CashFlow_4	0
CashFlow_5	3
Costs_0	0
Costs_1	4
Costs_2	8
Costs_3	10
Costs_4	7
Costs_5	5
SalesRevenue_0	0
SalesRevenue_1	1
SalesRevenue_2	2
SalesRevenue_3	3
SalesRevenue_4	7
SalesRevenue_5	10

Once the relevant OWL concepts and SWRL rules have been represented in Jess, the Jess execution engine can perform inference. As the rules fire, new Jess facts are inserted into the fact base. Those facts are then used in further inference. When the inference process is completed, those facts can then be transformed into OWL knowledge (see Fig. 10); this process is the inverse of the mapping mechanism (O’Connor et al., 2004).



For Class: CashFlow

For Individual: http://www.owl-ontologies.com/Ontology123081136

name: CashFlow_1

CashFlow_falls: Initial_CashFlow_1

hasValue: -2.0

value: -2.0

name
CashFlow_1
CashFlow_2
CashFlow_3
CashFlow_4

Fig. 10. Protégé interface after successful transformation of the new knowledge into Protégé-OWL

After the steps mentioned above are executed, new OWL knowledge is visible in Protégé graphical interface (see Fig. 10).

Relations between classes or individuals can be visualized graphically using OntovizTab, another one plug-in for Protégé.

Conclusions

Ontology-based systems are very popular in different research areas and applications. Rule acquisition in ontology-based systems is the main issue. The use of SWRL rule Tab was considered to be the most appropriate rule acquisition technique. SWRL rules are in the form of conjunctive.

SWRL rules represent all possible relations between curves and/or between points on the curve in this research. SWRL rules use ontology; therefore ontology of the domain area was developed. SWRL Editor allows transformation of OWL knowledge and SWRL rules into Jess knowledge represented in Jess facts and rules. When the inference is performed, resulting new facts are transferred back to Protégé-OWL as OWL knowledge.

Future Research

This approach has shown good results in rules acquisition and their further use in problem solving. Protégé and its plug-ins are considered to be future research directions in different domain areas.

References

1. Noy, N. F., Fergerson, R. W., Musen, M.A., 2000. The knowledge model of Protégé-2000: Combining interoperability and flexibility, *Proceedings of the 12th International Conference on Knowledge Engineering and Knowledge Management (EKAW'2000)*, pp 69-82.
2. O'Connor, M., Knublauch, H., Tu, S., Grosof, B., Dean, M., Grosso, W., Musen, M., 2005. Supporting Rule System Interoperability on the Semantic Web with SWRL. *The Semantic Web – ISWC 2005* (2005), pp. 974-986.
3. Plinere, D., Borisov, A., 2009. Ontology-based knowledge acquisition system for product life cycle task. *Proceedings of the 15th International Conference on Soft Computing (Mendel 2009),* pp. 292-297, Brno, Czech Republic
4. Rudenko, D., Borisov, A., 2008. Blackboard architecture for product life cycle stage definition. *Proceedings of the 14th International Conference on Soft Computing (Mendel 2008),* pp.252-257, Brno, Czech Republic.
5. Marketing teacher, Est 2000, http://www.marketingteacher.com/Lessons/lesson_plc.htm, Last visit day May 2009.
6. Online postgraduate courses for the electronic industry, Design and Manufacture of Electronic Systems, Engineering Design, http://www.ami.ac.uk/courses/ami4900_ed/u01/unit_1_sec_2/index.asp, Last visit day May 2009.
7. Epistemics, knowledge is our business, 1993, http://www.epistemics.co.uk/Notes/63-0-0.htm, Last visit day May 2009.

Darya Plinere was born in Riga, Latvia. She received her bachelor's degree in Computer Science from Riga Technical University in 2004 and earned her master's degree in Computer Science from Riga Technical University in 2006. Currently Darya Plinere is PhD student at Riga Technical University, Faculty of Computer Science and Information Technology and a researcher at the Department of Modelling and Simulation of Riga Technical University. Her research interests include agent systems, ontologies, product life cycle, supply chain management and rule-based systems.

Arkady Borisov is Professor of Computer Science in the Faculty of Computer Science and Information Technology at Riga Technical University. He holds a Doctor of Technical Sciences degree in Control in Technical Systems and the Dr.habil.sci.comp. degree. His research interests include fuzzy sets, fuzzy logic and computational intelligence. He has 205 publications in the area.

Darja Plinere, Arkādijs Borisovs. SWRL: likumu iegūšana, izmantojot ontoloģiju

Mūsdienās uz likumiem balstītas sistēmas tiek izmantotas ļoti bieži. Uz ontoloģijām balstīto sistēmu izmantošana kļūst arvien populārāka, it īpaši izmantojot to kopā ar uz likumiem balstītām sistēmām. Visplašāk izmantotā ontoloģiju attīstības platforma ir ontoloģiju redaktors Protégé. Protégé nodrošina zināšanu ieguves līdzekli, taču joprojām galvenais jautājums uz ontoloģiju balstīto likumu sistēmu jomā ir likumu ieguve. Šis raksts piedāvā pieeju izmantot SWRL rules Tab, kas ir plug-in priekš Protégé un nodrošina likumu iegūšanu. SWRL rules Tab transformē konjunktīvus likumus Jess likumos IF ... THEN formā.

Дарья Плинер, Аркадий Борисов. SWRL: извлечение правил с помощью онтологии

В настоящее время очень часто используются системы, основанные на правилах. Использование систем, основанных на онтологиях, становится все более популярным, особенно вместе с системой, основанной на правилах. Наиболее широко используемая платформа развития онтологий – редактор онтологий Protégé. Protégé предлагает инструмент приобретения знаний, но все-таки основной задачей системы, основанной на онтологии, является приобретение правил. Эта статья предлагает следующий подход - использовать SWRL rules Tab, плагин к Protégé, для приобретения правил. SWRL rules Tab преобразовывает конъюнктивные правила в Jess правила в форме IF…THEN.