SWRL Rule Editor
A Web Application as Rich as Desktop Business Rule Editors

João Paulo Orlando[1], Adriano Rívolli[1], Saeed Hassanpour[2], Martin J. O'Connor[2], Amar Das[2] and Dilvan A. Moreira[1]

[1]Dept. of Computer Science, ICMC - Universidade de São Paulo, Campus de São Carlos, Caixa Postal 668, 13560-970, São Carlos, SP, Brazil

[2]Stanford Center for Biomedical Informatics Research, Stanford, CA, U.S.A.

Keywords: SWRL Tool, SWRL API, Rule Tool, Rule SWRL, Rule Visualization, Rule Composition, Semantic Web, SWRL.

Abstract: The Semantic Web Rule Language (SWRL) allows the combination of rules and ontology terms, defined using the Web Ontology Language (OWL), to increase the expressiveness of both. However, as rule sets grow, they become difficult to understand and error prone, especially when used and maintained by more than one person. If SWRL is to become a true web standard, it has to be able to handle big rule sets. To find answers to this problem, we first surveyed business rule systems and found the key features and interfaces they used and then, based on our finds, we proposed techniques and tools that use new visual representations to edit rules in a web application. They allow error detection, rule similarity analysis, rule clustering visualization and atom reuse between rules. These tools are implemented in the SWRL Editor, an open source plug-in for Web-Protégé (a web-based ontology editor) that leverages Web-Protégé's collaborative tools to allow groups of users to not only view and edit rules but also comment and discuss about them. We evaluated our solution comparing it to the only two SWRL editor implementations openly available and showed that it implements more of the key features present in traditional rule systems.

1 INTRODUCTION

The Semantic Web has the goal of making the content available on the Web not only understood by people but also by machines (Berners-Lee, Hendler, and Lassila, 2011). To this end, it uses ontologies for knowledge representation and for the generation of semantically annotated data.

The Web Ontology Language (OWL) is the W3C standard language for Semantic Web ontologies (McGuinness and van Harmelen, 2004). It is a powerful language for building ontologies to specify high-level descriptions of web content. These ontologies are built using class hierarchies representing domain concepts and their related properties. OWL also provides a powerful set of axioms to define precisely how to interpret concepts in an ontology and infer information from these concepts.

However, the expressiveness of OWL is not always sufficient for modeling all kinds of problems. To overcome this deficiency, the SWRL (Semantic Web Rule Language) was created. It extends the set of OWL axioms to include Horn-like rules. It thus enables Horn-like rules to be combined with an OWL knowledge base to create assertions in the form of rules, using concepts defined in OWL to represent, organize and share specific domain knowledge by means of conditional statements (World Wide Web Consortium [W3C], 2004).

As the complexity of systems modeled using SWRL grows, developers face difficulties in managing the resulting rule set properly. A system with a large rule number becomes difficult to understand and error prone, especially when used and maintained by more than one person (Hassanpour, O'Connor, and Das, 2009). But SWRL lacks a rule-editing tool that is little more than a simple text editor and that incorporates the best techniques in rule creation/management tools to support users in creating and managing large rule sets. That is the problem we wanted to alleviate.

Research in rule management and development happens in several areas, such as databases, business systems, etc. So, to begin with, we conducted an

Paulo Orlando J., Rívolli A., Hassanpour S., J. O'Connor M., Das A. and A. Moreira D..
SWRL Rule Editor - A Web Application as Rich as Desktop Business Rule Editors.
DOI: 10.5220/0003999402580263
In *Proceedings of the 14th International Conference on Enterprise Information Systems* (ICEIS-2012), pages 258-263
ISBN: 978-989-8565-11-2

extensive survey on this kind of business rule systems (which are generally rule-based systems used by large corporations) and found the key features and interfaces found on them. After that, we proposed new features and interfaces to recreate/ adapt the ones we have found, in the business sytems, for use in a web application environment tailored for SWRL (all the tools studied were for desktop use). Finally, we implemented an editor, the SWRL Editor, to use these proposed features and interfaces to offer the SWRL community a rule editor with the same capability as the best editors available to other communities (mainly the business rule community).

In this paper, we present the SWRL Editor. It includes new or adapted visual representations and techniques to enhance the viewing and composition of rules. It aims to help Semantic Web developers in the process of creating, viewing and managing SWRL rules. It allows:

- Knowledge acquisition with less inconsistencies, ambiguities and duplicated rules;
- A better viewing experience for rules and rule sets to facilitate the understanding, knowledge and manipulation of them;

2 SWRL

SWRL is an expressive rule language that combines Horn clauses with concepts defined in OWL and can be used to increase the capacity of inference about individuals in a knowledge base in OWL (W3C, 2004). SWRL rules are composed by two parts: the antecedent (body) and consequent (head). Each rule is an implication between the body and head that can be understood as if the body conditions are true then make the conditions of the head also true. Both parts consist of a conjunction of zero or more atoms, not allowing disjunctions or denial.

Atoms, on the one hand, are formed by a predicate and one or more arguments (the number and type of arguments are determined by the atom type). Although the W3C specification defines seven atom types, in this paper, we adopt the convention used in (Hassanpour et al., 2009) that considers *SameAs* and *DifferentFrom* as having the same type because they are syntactically identical, even though they have different semantics (what leaves us with six types).

The six types of atoms are: the Class, Object property and Data valued property types correspond to elements (concepts) defined in the ontology. Data range type corresponds to the data types used in the ontology. The type Same/Different is used to explain the equality or difference between two individuals, as OWL needs to work with the open-world paradigm. Finally, Built-ins are types that encapsulate utility functions and can even be extended by users. Although SWRL rules can be represented in more than one format, the human-readable format is adopted in this paper. The arrow (→) is used to separate body and head, the caret (^) represents the conjunction of atoms and the question mark (?) distinguishes variable names from individuals. Using this syntax, a rule that states that the brother of the father of an individual is his uncle can be written as follows:

```
hasParent(?x,?y) ^ hasBrother(?y,?z) →
        hasUncle(?x,?z)
```

3 RELATED WORK

Before developing the SWRL Editor, we conducted an extensive research on tools to edit rules in general (not only SWRL rules), as well as general-purpose tools for SWRL rules (a total of 16 tools were reviewed). We were interested in their key features and interfaces adopted (Rivolli, Orlando and Moreira, 2011). In this section, two of these tools, the SWRL Tab and Axiomé, are briefly presented. They are directly related to our SWRL Editor and were the only two openly available tools we found to edit SWRL. We also discuss some of the key features found in tools for creating business rule systems (which are rule-based systems used by large corporations).

The SWRL Tab (SWRLTab, 2012) is a plug-in for the Protégé ontology editor and its main purpose is to allow editing and management of SWRL rules. The rules are presented in tabular format, so they can be edited directly where they are displayed or in an editor. The tool performs the identification of syntax errors, using the SWRL specification (W3C, 2004), and validates the correct use of ontology terms. In addition, a set of Application Programming Interfaces (APIs) is available, allowing, for example, rule inference and syntax validation. On the other hand, the tool does not offer any facilities to organize or group the rules.

Axiomé (Hassanpour et al., 2009) is also a plug-in for Protégé and uses the APIs provided by the SWRL Tab tool. From the user perspective, Axiomé offers more tools to organize and view rules and also allows rule edition using templates that are generated based on the structure of the rules. The main technical tools offered are: Automatic rule grouping; Visualization of dependences between rules; Rule Paraphrase (a dynamically generated textual explanation of a rule); and Templates for editing and acquiring rules.

There are a number of techniques and tools related to the creation of business rules that may be reused for the improvement or creation of new techniques related to SWRL rules (Rivolli et al., 2011). Although tools related to the creation of business rules are still evolving (Zacharias, 2008), their current development stage is far ahead compared to SWRL tools.

The analysis of these tools shows different forms of user interaction with rules, for example, decision trees and tables, templates for editing rules, natural language interfaces, automatic term suggestion, error correction, visualization diagrams and integrated editors that use grouping techniques.

4 SWRL EDITOR

4.1 Infrastructure and Architecture

The SWRL Editor was developed as a plug-in for Web-Protégé, the web version of the ontology editor Protégé (Tudorache, Vendetti and Noy, 2008a). Protégé is a free, extensible, open source, ontology editor developed by the Stanford Center for Biomedical Informatics Research (Stanford University). Web-Protégé uses the Google Web Toolkit (GWT) to provide a desktop-like user experience on the web, actually similar to the desktop version of Protégé. The architecture, based on plug-ins, is one of the main Protégé/Web-Protégé features. The extension mechanism allows new features to be added gradually, what made the Protégé/Web-Protégé a flexible but yet solid platform for developing semantic technologies (Tudorache, Noy, and Musen, 2008b). Web-Protégé plugins make it possible to add new tabs containing one or more portlets, reusable graphic components.

As a Web-Protégé plugin, the SWRL Editor uses the Java language and the GWT development tool. It consists of a tab and a portlet that interact, over the network, with the Protégé API allowing the manipulation of SWRL rules in a friendlier way using a higher abstraction level of representation than the SWRL language.

Unlike the SWRL Tab and Axiomé tools, which are very coupled to the Protégé tool, the SWRL Editor was developed applying appropriate design patterns (such as MVP) to make it reusable and decoupled from the Web-Protégé platform.

At the Web-Protégé side, the server allows access to its services through the Ontology API, to read/write ontologies, and the Collaboration API, to support collaboration services, such as ontology term annotation and change control (Tudorache et al., 2008a). SWRL Editor users can apply these collaboration services to annotate components used in their rules. The CHAO (Changes and Annotations Ontology) (Noy, Chugh, Liu, and Musen, 2006) has classes to define different annotation types, such as comments, or changes, such as class created (Tudorache et al., 2008a).

In the Web-Protégé client side, there are the user GUI, a ontology model and a RPC module (to communicate with the server). The ontology model has a partial representation of the server ontology that is updated as needed by the client.

The SWRL Editor plug-in has its own rule representation model, generated by the SWRL Manager from the SWRL rules it gets using the Ontology API. It also has the SWRL Collaboration module to manage the versioning of rule changes and the Plug-in Manager to load code (in jar files) with different algorithms for rule Grouping and Decision Tree view generation. The client side has a MVP (Model-view-presenter) architecture that is similar to the Protégé client.

4.2 Visualization

The solutions created to enhance visualization and rule understanding consist of visual representations for rules (Figure 1, 2 and 3), organization techniques and resources to present rule sets. Lets use a rule set form the Ontology for Breast Cancer Grading (Bulzan, 2010) to show our tool features. This rule set is composed of 12 rules, which are used to assign a note to a tumor from the three criteria from NGS (Next-Generation Sequencing). We show the same rule using three different visual representations:

- The SWRL highlight visual representation (Figure 1) displays the rule atoms and adds color to the arguments according to their type;

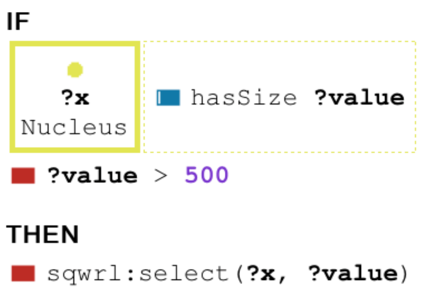
```
Nucleus(?x) ^
hasSize(?x, ?value) ^
swrlb:greaterThan(?value, 500)

->

sqwrl:select(?x, ?value)
```

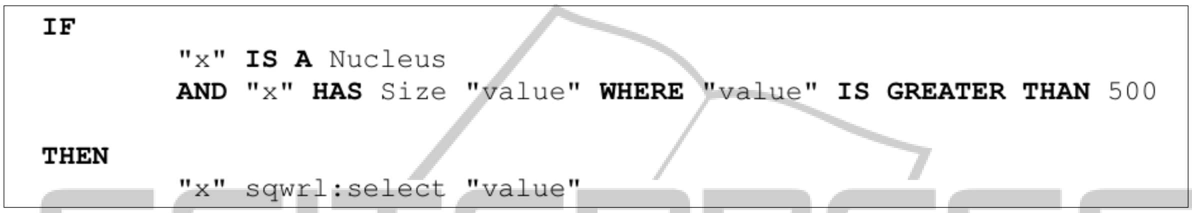
Figure 1: Visual representations for rules - Highlight.

- The hierarchical view (Figure 2) abstracts rule syntax and presents rules in a hierarchical tree related to their meaning and type of atoms;



IF
?x Nucleus
hasSize ?value
?value > 500
THEN
sqwrl:select(?x, ?value)

Figure 2: Visual representations for SWRL rules - Hierarchical View.



```
IF
          "x" IS A Nucleus
          AND "x" HAS Size "value" WHERE "value" IS GREATER THAN 500

THEN
          "x" sqwrl:select "value"
```

Figure 3: Visual representations for SWRL rules - View in Paraphrase.

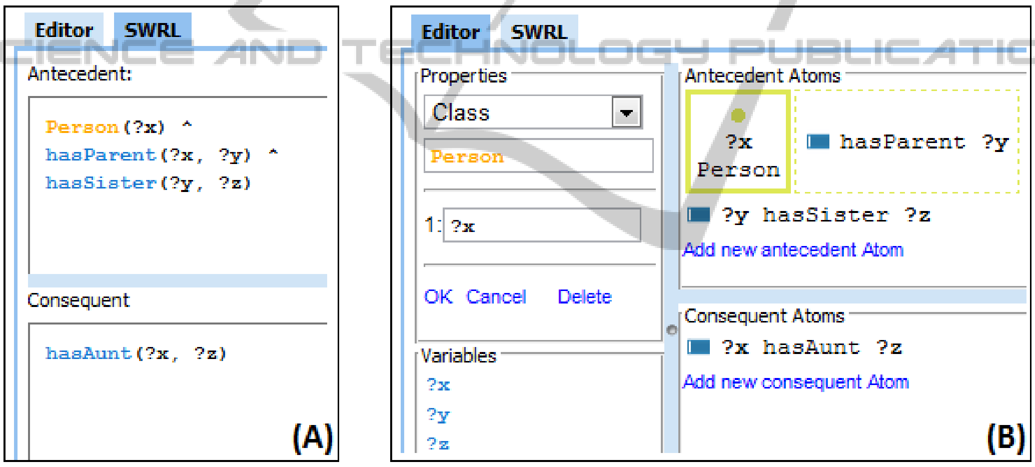


Figure 4: Family Ontology (Peace, 2008) – (A) SWRL Editor with textual Highlight; (B) Hierarchical SWRL Editor.

- Paraphrase visualization (Figure 3) shows a paraphrase of the rule axioms using a technique developed for Axiomé (Hassanpour et al., 2009). We added bold characters to mark connection terms;

All three representations make the rule presentation clearer and enrich its meaning with colors and symbols.

These visualization techniques allow the user to better manage and understand the rule sets.

In addition, the tool has the following features:

- Automatic procedures for extracting information and generating views for each visual representation, without user intervention;
- The capability of listing rules in any of the visual representations: there is a tab for each representation and users can switch them at will;

These features add functionality for the use of rule developers, allowing them to easily locate and understand specific rules.

4.3 Rule Creation and Edition

The SWRL Editor has two rule edition modes to assist users:

- Editor with textual Highlight (Figure 4 (A));
- SWRL hierarchical editor (Figure 4 (B));

The textual editor with highlight is similar to the standard type of editor normally used in tools, such as Eclipse or Netbeans, to edit C or Java programs. Its color scheme is related to the syntactic elements and can be changed by users (antecedents and consequents are also separated).

The SWRL hierarchical graphic editor, shown in figure 4 (B), has boxes that change according to the selected atoms and icons to separate different atom types. It has separate boxes for consequents and antecedents. The editor also suggests terms from the ontologies being used, based on atom types and position.

Users can freely switch between the two editors, what can be a very good tool to teach rule structure to novices. In addition, both editors offer the following features:

- Detection of lexical and syntactic errors;
- Identification of similar rules;
- Hierarchical navigation into otology classes, properties and bultins using a tree (on the editor left side), with the possibility of their inclusion in a new rule.

5 TOOLS COMPARISON

In this section, we present a comparison between the features of the SWRL Tab, Axiomé and our tool, SWRL Editor, as the other tools in the review do not work with SWRL. The Axiomé and SWRL Tab tools have resources coupled to the Protégé 3 API.

They are also not available for the Web-Protégé version (only the desktop version) and, for that reason, cannot benefit of some of the collaboration tools available for the web. The SWRL Editor manipulates rules using an internal logical representation that can be easily mapped to other APIs, such as the OWL-API (and, in the future, even to SPARQL Rules).

To make this comparison, we used the main features that we found in more traditional rule based desktop tools (developed mainly for business users) (Rivolli et al., 2011) and (Zacharias, 2008). Table 1 shows the comparison between the three tools:

We can see that the SWRL Editor has an overwhelming advantage over the other tools as it implements most of the traditional rule based tools features.

Compared to the SWRL Editor, the other two tools either lack most of its capabilities or have a more restricted implementation. The SWRL Editor is an evolution of them. We determined, after input from some users, that a more sophisticated usability test, comparing the three tools, would not be very informative due to the lack of features to compare.

We tested the tool using three ontologies (with SWRL rule sets). When using the Autism ontology, we were able to easily find problems, like repeated rules, that had escaped their developers using the other two tools.

Table 1: Comparing the features of the tools.

Feature	SWRL Editor	SWRL Tab	Axiomé
Platform	Web	Desktop	Desktop
Decision tables	✗	✗	✗
Decision trees	✓	✗	✗
Graphic diagram	✗	✗	✓
Text editor	✓	✓	✓
Text editor with Highlight	✓	✗	✗
Rule templates	✓	✗	✓
Natural language	Partially	✗	Partially
Visual representation	✓	✗	✓
Grouping	✓	✗	✓
Term suggestions	✓	✓	✗
Error Detection	Better Coverage	✓	✓
Filter	✓	✓	✓
Customization Tool	✓	✗	✗
Add new algorithms	✓	✗	✗

It is also important to mention the fact that the SWRL Editor is available on the Web and integrates its various resources (and enhancements) with Web-Protégé collaboration features into a single tool. Users can annotate components used in their rules, adding comments, proposals etc, and discuss online (in real time) about them using the *Notes and Discussion* Tab. As big rule sets are likely to be developed by groups of distributed users, that resulting single tool can more effectively support the work of these groups. That is one of SWRL Editor important contributions as none of the tools in the literature (including more general tools for rule editing) support that.

6 FINAL CONSIDERATIONS

Rule sets in SWRL have the fundamental role of being the inference engine to create new knowledge inside the Semantic Web. But SWRL lacked a rule-editing tool that was more than a simple text editor and incorporated the best techniques for rule creation/management, available in tools for other rule domains.

After we conducted an extensive survey of business rule systems and compiled their most important features and interfaces, we designed the SWRL Editor tool to use, what we considered, the key features to offer a SWRL rule editor with the same capabilities as the best editors available to other communities.

But we went a step further and implemented this tool as a Web-Protégé plugin (using GWT/AJAX technology) to create a desktop like tool running on the web, so we could add to it: zero footprint installation, automatic updates, easy cooperation among user groups, operating system independency, and other well known advantages of web applications.

The SWRL Editor uses three approaches for visualizing rules:

1. The SWRL Text Highlight.
2. The Hierarchical View.
3. Paraphrase Visualization.

For rule composition we created two editing tools:

1. Text Editor with Syntax Highlight
2. Hierarchical SWRL Editor

Both editors have error detection, warnings, search (for similar rules), self-completion and hierarchical navigation of terms in the ontology.

Our main contribution is to show, using the SWRL Editor, that it is possible to make a tool for SWRL rule edition that have the equivalent key features and interfaces of more traditional rule based desktop tools (developed mainly for business users) as a "desktop like" web application. That would not be possible a year ago; it is only possible now due to new internet technologies, such as HTML 5. Such a tool is far more advanced than the current state-of-the-art for SWRL edition.

We also show that it is possible to add collaboration to this tool by leveraging the fact that it runs distributed over the web, it keeps information about rule versioning and uses Web-Protégé tools for cooperation. No other tools described in the literature have such feature but that is important, as big rule sets are likely to be developed by groups of distributed users.

As future work, we intend to develop tools to target redundancy errors, highlight relevant rules (given some criteria) and ways to establish connections among rules (so if you modify one the others in the set are highlighted).

ACKNOWLEDGEMENTS

This work has been a collaborative project between the Intermídia Lab – ICMC – University of São Paulo and the Stanford Center for Biomedical Informatics Research (BMIR) – Stanford University. This project has been funded by the National Institutes of Health, under grant R01MH87756. It has also been funded by grants from the National Council for Scientific and Technological Development - CNPq, a Brazilian research agency.

REFERENCES

Berners-Lee, T., Hendler, J., & Lassila, O. (2001). The Semantic Web. *Scientific American*, 34-43. Retrieved from http://www.scientificamerican.com/article.cfm?id=the-semantic-web

Bulzan, A. (2010). *Breast Cancer Grading Ontology*. Retrieved from http://bioportal.bioontology.org/ontologies/1304

Hassanpour, S., O'Connor, M. J., & Das, A. K. (2009). Exploration of SWRL Rule Bases through Visualization, Paraphrasing, and Categorization of Rules. *International Symposium on Rules*, 246–261. doi: 10.1007/978-3-642-04985-9_23.

McGuinness, D. L., & van Harmelen, F. (2004). OWL Web Ontology Language Overview. W3C Recommendation 10 February 2004. Retrieved from http://www.w3.org/TR/2004/REC-owl-features-20040210

Noy, N. F., Chugh, A., Liu, W., & Musen. M. A. (2006). A framework for ontology evolution in collaborative environments. *ISWC*. 544-558, Athens, Georgia. doi: 10.1007/11926078_39

Peace, J. (2008). *Family Health History Ontology*. Retrieved from http://bioportal.bioontology.org/ontologies/1126

Rivolli, A., Orlando, J. P., & Moreira, D. A. (2011). An Analysis of Rules-Based Systems to Improve SWRL Tools. *International Conference on Enterprise Information Systems*, 191-194 Beijing, China.

SWRLTab. (2012). *SWRL Tab*. Retrieved March 21, 2012, from http://protege.cim3.net/cgi-bin/wiki.pl?SWRLTab#nidC5G

Tudorache, T., Noy, N. F., & Musen. M. (2008b). Supporting collaborative ontology development in protégé. *International Semantic Web Conference*, 17-32, Karlsruhe, Germany, 17-32. doi: 10.1007/978-3-540-88564-1_2

Tudorache, T., Vendetti, J., & Noy, N. F. (2008a). Web-Protégé: A Lightweight OWL Ontology Editor for the Web. *Workshop on OWL: Experiences and Directions, collocated with the ISWC*, Karlsruhe, Germany. doi: 10.1.1.142.8568

World Wide Web Consortium. (2004). *SWRL: A Semantic Web Rule Language Combining OWL and RuleML*. W3C Recommendation 21 May 2004. Retrieved from http://www.w3.org/Submission/SWRL

Zacharias, V. (2008). Development and verification of rule based systems – a survey of developers. *International Symposium Rule Representation, Interchange and Reasoning on the Web*, 6-16, doi: 10.1007/978-3-540-88808-6_4