

# Querying the Semantic Web with SWRL

Martin O'Connor, Samson Tu, Csongor Nyulas, Amar Das, Mark Musen

Stanford Medical Informatics, Stanford University School of Medicine,  
Stanford, CA 94305 [martin.oconnor@stanford.edu](mailto:martin.oconnor@stanford.edu)

**Abstract.** The SWRLTab is a development environment for working with SWRL rules in Protégé-OWL. It supports the editing and execution of SWRL rules. It also provides mechanisms to allow interoperability with a variety of rule engines and the incorporation of user-defined libraries of methods that can be used in rules. Several built-in libraries are provided, include collections of mathematical, string, and temporal operators, in addition to operators that can be used to effectively turn SWRL into a query language. This language provides a simple but powerful means of extracting information from OWL ontologies. Used in association with a relational data importation tool that we have developed called DataMaster, this query language can be also used to express knowledge-level queries on data imported from relational databases.

## 1 Introduction

The Semantic Web project is a shared research plan that aims to provide explicit semantic meaning to data and knowledge on the World Wide Web [1]. Semantic Web applications aim to be able to integrate data and knowledge automatically through the use of standardized languages that describes the content of Web-accessible resources. OWL was developed as a formal language for constructing ontologies that provide high-level descriptions of these web resources [2]. Recent work has concentrated on adding rules to OWL to provide an additional layer of expressivity. The Semantic Web Rule Language (SWRL; [3]) is one of the results of these activities. SWRL allows users to write Horn-like rules that can be expressed in terms of OWL concepts and that can reason about OWL individuals.

We have been developing open-source tools to work with SWRL. One of the primary results is the SWRLTab [4, 9], an extension to the Protégé-OWL ontology development toolkit [5]. Recently we have been concentrating on developing mechanisms to allow SWRL to be used as an OWL query language. We are using this language to query both standard OWL ontologies and ontologies that are populated from relational databases. We have developed a tool called DataMaster [6] to perform this importation. We believe these tools can be used to tackle some of the challenging data integration issues that are faced when developing Semantic Web based software.

## **2 SWRLTab**

The SWRLTab [4] provides a set of APIs that support the building of tools that work with SWRL rules. It has several software components, including (1) an editor that supports interactive creating, editing, reading, and writing of SWRL rules; (2) a rule engine bridge that provides the infrastructure necessary to interoperate with third-party rule engines and reasoners; (3) a built-in bridge that provides a mechanism for defining Java implementations of SWRL built-ins; and (4) a set of built-in libraries containing mathematical, string, temporal, and ABox and TBox operators.

### **2.1 SWRL Editor**

The Protégé-OWL SWRL Editor is an extension to Protégé-OWL that permits interactive editing of SWRL rules. Users can create, edit, and read/write SWRL rules. With the exception of arbitrary OWL expressions, this editor supports the full set of language features outlined in the SWRL Submission. It is tightly integrated with Protégé-OWL and is primarily accessible through a tab within it. When editing rules, users can directly refer to OWL classes, properties, and individuals within an OWL ontology. They also have direct access to a full set of built-ins described in the SWRL built-in specification and to all of the XML Schema data types.

### **2.2 Rule Engine Bridge**

The SWRL Rule Engine Bridge is a subcomponent of the SWRLTab that provides a bridge between an OWL model with SWRL rules and a third party rule engine or reasoner. Its goal is to provide the infrastructure necessary to incorporate rule engines and reasoners into Protégé-OWL to execute SWRL rules.

The bridge provides mechanisms to (1) import SWRL rules and OWL classes, individuals, properties and descriptions from an OWL ontology; (2) write that knowledge to a rule engine or reasoner; (3) allow the rule engine to perform inference and to assert its new knowledge back to the bridge; and (4) insert that asserted knowledge into an OWL ontology. The bridge also provides mechanisms to add graphical user interfaces to the SWRLTab to allow interaction between a particular rule engine implementation and users. A bridge to the Jess rule engine [7] is provided together with a user interface component called the SWRLJessTab.

### **2.3 Built-in Bridge**

SWRL provides a very powerful extension mechanism that allows the use of user-defined methods in rules [8]. These methods are called built-ins and are predicates that accept one or more arguments. Built-ins are analogous to functions in production rule systems. A number of core built-ins are defined in the SWRL specification. This core set includes basic mathematical operators and built-ins for string and date manipulations. SWRL users can also define their own built-in libraries. Example libraries could include built-ins for currency conversion, or for statistical, temporal or spatial operations. Again, once implemented, these user-defined built-ins can be used directly in SWRL rules.

We have developed an extension to the SWRLTab called the SWRL Built-in Bridge. This extension provides support for defining built-in implementations written in Java and dynamically loading them. Users wishing to provide implementations for a library of built-in methods can define a Java class that contains definitions for all the built-ins in their library. The bridge has a dynamic loading mechanism to import these built-in definitions and provides an invocation mechanism to execute these loaded definitions from rule engines. A mechanism to marshal and unmarshal arguments to and from built-in is also provided. If additional rule engines are integrated into the SWRLTab they can use these existing built-in libraries without modifying them.

## 2.4 Built-In Libraries

Using the built-in bridge we have implemented a set of libraries for common methods required by rules [10]. These include implementations for the core SWRL built-ins defined by the SWRL submission, a temporal library that can be used to reason with temporal information in SWRL rules, and libraries with ontology TBox and ABox operators.

## 3 Extending SWRL to Support OWL Queries

We have also developed a built-in library that allows SWRL rules to be used to query OWL ontologies [11]. The library contains SQL-influenced built-ins that can be used in a rule to construct retrieval specifications. For example, the following rule, written with these built-ins, retrieves all persons in an ontology whose age is less than 5, together with their ages:

```
Person(?p) ^ hasAge(?p,?a) ^ swrlb:lessThan(?a,5) • query:select(?p,?a)
```

This query will return pairs of persons and ages. The following query lists all persons together with their ICD9 codes:

```
Person(?p) ^ hasICD9(?p, ?icd) • query:select(?p, ?icd)
```

This query will return pairs of persons and their ICD9 codes. Assuming a person can have more than one ICD9 code, multiple pairs would be displayed for each person—one pair for each code.

The query library also provides basic counting, aggregation, ordering, and duplicate elimination operators.

Query built-ins can be used with other built-in libraries provided by the SWRLTab. For example, the TBox built-in library can be used in SWRL queries to extract the structure of an OWL ontology; the temporal built-in library can be used to express complex temporal conditions in queries. The ability to use built-ins freely in a query provides a means of continuously expanding the power of the query language.

### 3.1 SWRLQueryTab

The SWRLQueryTab provides a convenient way to visualize the results of these SWRL queries. It has a control sub-tab that can be used to control the execution of SWRL rules containing query built-ins. A query can be selected from the rule table in the Protege-OWL SWRL Editor and executed to display results of the query. Users can navigate to that sub-tab to review the results displayed in tabular form.

### 3.2 SWRLQueryAPI

The SWRLQueryAPI provides a JDBC-like Java interface to retrieve the result of SWRL queries. Results for a particular query can be retrieved from a SWRL rule engine bridge. Rows in a result can be iterated through and the contents of each column in a row retrieved using accessor methods.

For example, if we wish to process the results of the earlier query (which we will name “Query-1”) that extracts all persons under the age of 5 from our ontology, we can write:

```
Result result = bridge.getQueryResult("Query-1");

while (result.hasNext()) {
    System.out.println("Person: " + result.getDatatypeValue("?p"));
    System.out.println("Age: " + result.getDatatypeValue("?age"));
    result.next();
} // while
```

## 3 DataMaster

Importing data from relational databases into ontologies is frequently required, particularly when an ontology is used to semantically describe the data used by a software application. Another growing category of applications requires database-ontology integration and/or interoperation, where a mapping between the database schema structure and ontology concepts is the main focus. In the latter cases the import of the data residing in relational databases may not be necessary or desired.

To meet these requirements, we have developed DataMaster [6], a Protégé-OWL plug-in that allows the user to import relational database structure or content into an OWL ontology. DataMaster can be used with any relational database with JDBC/ODBC drivers. A user interface is provided that supports user-driven configuration of the importation process. A user can use this interface to connect to a database and select the portions of the database that they wish to import. If the user decides on a schema-only import the database schema is read and represented in OWL using the Relational-OWL [12] ontology. If a content import is requested, the user can select the a number of mapping options, such as, for example, how table and column names are mapped, and how relational column types are mapped to XSD Schema types.

## 4 Conclusion

We are using these tools to help meet the data specification requirements of several ontology-driven biomedical applications. SWRL is used to help unify the domain-level specification of system data with the run-time operational data needs of system components. In conjunction with OWL, SWRL provides both a formal domain-level description of data in the biomedical systems that we are developing and a run-time query mechanism to execute domain-level queries on data in these systems. We have also implemented a dynamic OWL-to-relational mapping mechanism that allows data to be selectively retrieved from live operational database in response to SWRL queries [13]. These extensions will be released in an upcoming Protégé-OWL release. The ability to meet the high data throughput demands of these applications is crucial to enable this technology to meet the scalability requirements of the Semantic Web.

**Acknowledgements.** This work was supported in part by the Immune Tolerance Network, which is funded by the National Institutes of Health under Grant NO1-AI-15416, and also by the Centers for Disease Control and Prevention under grant number SPO-34603. We thank Valerie Natale for her editorial comments.

## References

1. Berners-Lee, T. The Semantic Web, *Scientific American*, May 2001.
2. OWL Overview: <http://www.w3.org/TR/owl-features/>
3. SWRL Submission: <http://www.w3.org/Submission/SWRL/>
4. SWRLTab: <http://protege.cim3.net/cgi-bin/wiki.pl?SWRLTab>
5. Knublauch, H., Ferguson, R.W., Noy, N.F. and Musen, M.A. The Protégé OWL Plugin: An Open Development Environment for Semantic Web applications. Third International Semantic Web Conference, Hiroshima, Japan, 2004.
6. DataMaster: <http://protegewiki.stanford.edu/index.php/DataMaster>
7. Jess: <http://herzberg.ca.sandia.gov/jess/>
8. SWRL Built-in Specification: <http://www.daml.org/rules/proposal/builtins.html>
9. O'Connor M.J., Knublauch, H., Tu, S.W., Grossof, B., Dean, M., Grosso, W.E., Musen, M.A. Supporting Rule System Interoperability on the Semantic Web with SWRL. Fourth International Semantic Web Conference, Galway, Ireland, 2005.
10. SWRL library: <http://protege.cim3.net/cgi-bin/wiki.pl?SWRLTabBuiltInLibraries>
11. SWRL queries: <http://protege.cim3.net/cgi-bin/wiki.pl?SWRLQueryBuiltIns>
12. C. P. de Laborda, S. Conrad. RelationalOWL - A Data and Schema Representation Format Based on OWL. *Conceptual Modelling*, 43:89-96, 2005.
13. O'Connor MJ, Shankar RD, Tu SW, Nyulas C, Parrish DB, Musen, MA, Das AK. Using Semantic Web Technologies for Knowledge-Driven Querying of Biomedical Data. 11th Conference on Artificial Intelligence in Medicine (AIME 07), Amsterdam, Netherlands, 2007.