The SWRLAPI: A Development Environment for Working with SWRL Rules

Martin O’Connor, Csongor Nyulas, Ravi Shankar, Amar Das, Mark Musen

Stanford Center for Biomedical Informatics Research
Stanford, CA 94305 martin.oconnor@stanford.edu

SWRL is an expressive OWL-based rule language that can be used to increase the amount of knowledge encoded in OWL ontologies. While semantically a SWRL rule can be considered as an additional type of OWL axiom, the authoring and management of SWRL rule bases requires specialized tools that are not typically present in standard OWL development environments. In this paper, we describe such a tool—called the SWRLAPI—that provides a rich development environment for working with SWRL rules. The SWRLAPI is built on the widely-used Protégé-OWL ontology toolkit. It provides both a set of highly interactive user interfaces for working with rules and a set of low-level APIs for embedding rules in knowledge-driven applications. This SWRLAPI has been used to develop a number of technologies to support data integration on the Semantic Web, including a SWRL-based query language called SQWRL and a set of tools that support interoperation between OWL and a variety of information formats.

1 Introduction

We have developed a number of open-source tools to work with SWRL. One of the primary results is the SWRLAPI [1], an extension to the Protégé-OWL ontology development toolkit. The SWRLAPI provides both an authoring environment for developing rules and a set of application programmer interfaces that support the building of rule-driven applications. It has several core software components, including an editor, a rule engine bridge that supports interoperation with a variety of rule engines and OWL reasoners, a mechanism for defining Java implementations of user-defined functions that can be used in rules, and an extensive set of libraries that can be used to dramatically extend the expressive power of SWRL rules.

This core functionality has been used to build two complementary technologies: (1) a SWRL-based query language called SQWRL that can be used to query OWL ontologies; and (2) a set of data integration tools that support interoperation between OWL, SWRL and other information formats, including XML, relational databases, and a variety of others. We believe these two technologies can be used to tackle some of the challenging data integration issues that are faced when developing Semantic Web based software.

2 The SWRLAPI: Core Features

The SWRLAPI has several core 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 OWL reasoners; (3) a bridge that provides a mechanism for defining libraries of built-ins; and (4) an extensive set of built-in libraries.

(1) 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. The editor supports the full set of language features outlined in the SWRL Submission.

(2) 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 OWL reasoner [2]. Its goal is to provide the infrastructure necessary to incorporate rule engines and reasoners into Protégé-OWL to execute SWRL rules. Interoperation with the Jess rule engine and the Pellet reasoner is currently provided. Support for the Jena and Algernon rule engines are being developed.

(3) One of SWRL's most powerful features is its ability to support *built-ins*, which are user-defined predicates that can be used in rules. A number of core built-ins are defined in the SWRL Submission but users may also define their own libraries. The SWRLAPI's **Built-in Bridge** provides support for defining these libraries. 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 built-ins in the library at run-time. If additional rule engines are integrated into the SWRLAPI they can use these existing built-in libraries without modification.

(4) The SWRLAPI provides set of **Built-In Libraries** [3]. These libraries include implementations for the core SWRL built-ins defined in the SWRL Submission, a temporal library that can be used to reason with temporal information in SWRL rules, a mathematical library, an XML processing library, and libraries with ontology TBox and ABox operators. Libraries for dealing with spreadsheet documents, WSDL, SOAP, and RSS feeds are under development.

3 The SWRLAPI: Querying with SQWRL

SWRL is a rule language, not a query language. However, many ontology-based applications require the ability to extract information from ontologies in addition to reasoning with the information in those ontologies. To support this knowledge extraction, we have developed a query language called SQWRL (Semantic Query-

Enhanced Web Rule Language; [4]) that extends SWRL to support querying of OWL ontologies.

SQWRL is implemented as a built-in library using the standard SWRL built-in mechanism. The SQWRL built-in library contains SQL-influenced built-ins that can be used in a rule to construct retrieval specifications for information stored in an OWL ontology. For example, the following SQWRL query retrieves all persons in an ontology whose age is less than 9, together with their ages:

```
Person(?p) ^ hasAge(?p,?a) ^ swrlb:lessThan(?a,9) -> sqwrl:select(?p,?a)
```

Aggregation, ordering, and counting are also supported. For example, a query to return the average age of persons with a known age in an ontology can be written:

```
Person(?p) ^ hasAge(?p, ?age) -> sqwrl:avg(?age)
```

Like SWRL, SQWRL supports the use of OWL class descriptions. For example, to retrieve all classes in an OWL ontology that are associated with an `hasChild` property with a cardinality greater than or equal to one, we can write:

```
(hasChild >= 1)(?i) -> sqwrl:select(?i)
```

SQWRL has access to all available SWRLAPI built-in libraries. The ability to freely use built-ins in a query provides a means of continuously expanding the power of the query language. Crucially, it provides an interoperation bridge to deal with other knowledge formats. SQWRL itself is formally based on OWL-DL and SQWRL queries can be expressed only in terms of core OWL concepts. However, the built-ins provides the ability to refer to non OWL DL concepts in queries. A TBox built-in library, for example, allows direct querying of OWL classes and properties, something that would not be possible in OWL DL. Similarly, an ABox built-in library can support the direct querying of ABox entities. For example, to list all data valued properties in an ontology and the number of their direct subproperties, we can write:

```
tbox:isDatatypeProperty(?p) ^ tbox:isDirectSubPropertyOf(?sp,?p)
  -> sqwrl:select(?p) ^ sqwrl:count(?sp)
```

The SWRLAPI provides a graphical interface to execute SQWRL queries. A JDBC-like Java interface is also provided to execute SQWRL queries in Java applications.

4 The SWRLAPI: Data Integration

Data integration is central challenge of the Semantic Web. Semantic Web applications must typically deal with a variety of information formats. While there are many dimensions to the data integration challenge, two functionalities are crucial: (1) the ability to *query* diverse data formats; and (2) the ability to *map* between different formats. The types of mapping required range from low level structural or syntactic mappings to transformations that require extensive domain knowledge.

We have developed tools to support a variety of mappings to support data integration with SWRL. These tools are: (1) XMLMapper, a library that supports the mapping of information described in XML documents to and from OWL [5]; (2) DataMaster [6], and Dynamic DataMaster [7], tools that supports mapping of relational data to OWL. While we presently only support relational and XML data formats, SWRL's built-in mechanism provided the extension point for dealing with additional formats.

5 Discussion

One of SWRL’s most useful features is its ability to incorporate user-defined built-in libraries. This extension mechanism provides a very powerful means of expanding SWRL’s expressiveness and of increasing the types of information that can be reasoned with using rules. In particular, this mechanism can be used to tackle the issue of data integration, which is one of the central challenges of the Semantic Web. The ability to meet this challenge requires the development of a variety of mapping technologies to allow interoperation between the various formats that will be encountered when developing Semantic Web applications. The tools outlined in this paper provide a set of basic building blocks that can be used to construct these mapping technologies.

References

[1] SWRLAPI. Retrieved on August 8th, 2008 from http://protege.cim3.net/cgi-bin/wiki.pl?SWRLAPI

[2] O'Connor MJ, Knublauch H, Tu SW, Grosof B, Dean M, Grosso WE, and Musen MA (2005). Supporting rule system interoperability on the Semantic Web with SWRL. Fourth International Semantic Web Conference.

[3] SWRLTab Built-In Libraries. Retrieved on August 8th, 2008 from http://protege.cim3.net/cgi-bin/wiki.pl?SWRLTabBuiltInLibraries

[4] SQWRL (2008). SQWRL. Retrieved on August 8th, 2008 from http://protege.cim3.net/cgi-bin/wiki.pl?SQWRL

[5] XMLMapper (2008). SWRLTab XMLMapper Tool. Retrieved on August 8th, 2008 from http://protege.cim3.net/cgi-bin/wiki.pl?SWRLTabXMLBuiltIns

[6] Datamaster (2008). DataMaster. Retrieved August 8th, 2008 from http://protegewiki.stanford.edu/index.php/DataMaster

[7] O'Connor MJ, Shankar RD, Tu SW, Nyulas C, Parrish DB, Musen, MA, Das AK (2007). Using Semantic Web Technologies for Knowledge-Driven Querying of Biomedical Data. 11th Conference on Artificial Intelligence in Medicine.