A Tool for MDD of Rule-based Web Applications based on OWL and SWRL

Joaquín Cañadas[1], José Palma[2] and Samuel Túnez[1]

[1] Dept. of Languages and Computation. University of Almeria. Spain
jjcanada@ual.es, stunez@ual.es
[2] Dept. of Information and Communications Engineering. University of Murcia. Spain
jtpalma@um.es

Abstract. TOOL PRESENTATION*: Rule languages and inference engines incorporate reasoning capabilities to Web information systems. This demonstration paper presents a tool for the development of rule-based applications for the Web based on OWL and SWRL ontologies. The tool applies a model-driven approach to an ontology representing a domain conceptualization and inference model of the problem domain. It automatically generates a rich, functional Web architecture based on the Model-View-Control architectural pattern and the JavaServer Faces technology, embedding a Jess rule engine for reasoning and deriving new information.

Key words: Model Driven Development, OWL, SWRL, Rule-based systems for the Web

1 Introduction

Rule languages and inference engines provide Web information systems with reasoning capabilities. Rule-based applications integrate rule engines to deal with rules and execute inference methods for firing appropriate rules in order to deduce new information and achieve results. Rules combined with ontologies enable the declarative representation of the expert knowledge and business logic in an application domain.

Web Ontology Language (OWL) [1] and Semantic Web Rule Language (SWRL) [2], which play a major role in the Semantic Web [3], are also growing in importance in software development [4]. Ontologies can describe the relevant concepts and data structures of an application domain and rule-based languages can be used to formalize the business logic, increasing the amount of knowledge that can be represented in ontologies.

Ontologies are created using authoring tools like Protégé. Recently, a tool to bridge the gap between OWL ontologies and Model Driven Engineering has

* This work was partially financed by the Spanish MEC through projects TIN2009-14372-C03-01 and PET2007-0033, and by the Andalusian Regional Government project P06-TIC-02411

been presented, the TwoUse Toolkit [3] [5]. It is a free, open source tool bridging the gap between Semantic Web and Model Driven Software Development. It has been developed using Eclipse Modeling Project[4] and implements current OMG and W3C standards for developing ontology-based software models and model-based OWL ontologies. Among its functionality, TwoUse provides a graphical editor for specifying OWL and SWRL models based on the Ontology Definition Metamodel (ODM) [6], and an textual editor for the OWL functional syntax [7]. Since it is deployed as an Eclipse plugin, other Eclipse-based tools for designing and executing transformations can be straightforwardly applied to ontologies created in TwoUse, enabling us to design an MDD process based on OWL and SWRL models.

This demo presents a tool which provides a model-driven approach to develop rule-based Web applications based on OWL and SWRL models created with TwoUse. The tool applies MDD to produce the implementation of a functional, rich Web architecture which embeds the Jess rule engine for inferencing tasks. The functionality for the generated rule-based Web application is predefined to enable end-users to create, retrieve, update and delete instances (CRUD). In contrast to current tools for automatic generation of CRUD systems that perform those functions on relational databases, our contribution is that this functionality is executed on the rule engine working memory, enabling the execution of a forward-chaining inference mechanism to drive the reasoning process.

This paper is organized as follows: Section 2 introduces the model-driven approach applied in the tool. Next, Section 3 describes the architecture for the rule-based Web application generated. Finally, the main conclusions and future work are summarized.

2 Model-driven process implemented in the tool

The proposed tool uses OWL and SWRL ontologies created with TwoUse as source models for the model-driven approach. We focus on *OWL DL* sublanguage of OWL. Classes, properties, and individuals define the structure of data, whereas rules describe logical dependencies between the elements of the ontology refereed in the rule's antecedent and consequent.

Figure 1 shows a simplified schema of the MDD process implemented in the proposed tool. Two different results are found from the single ontology model. On one hand, a Jess [8] rule base is generated, a text file that contains the rules converted to Jess syntax. On the other hand, a set of JavaBeans and JSP web pages, making up a JavaServer Faces (JSF) [9] architecture that embodies the Jess rule engine into the Web application.

Both MDD processes can be executed separately, enabling the decision logic of rule-based applications to be changed regardless of the ontology structure. When the rule model changes, the new rule base can be regenerated and deployed

[3] http://code.google.com/p/twouse/
[4] http://www.eclipse.org/modeling/

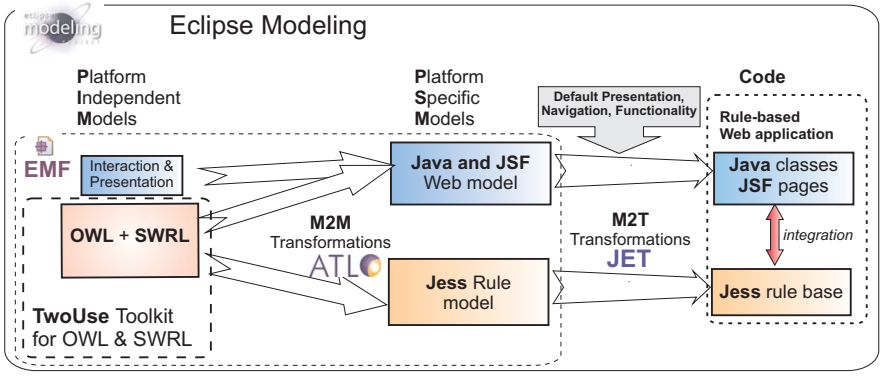

Fig. 1. MDD schema for rule-based Web system generation

into the Web application without affecting the full architecture. This approach makes Web applications easier to maintain and evolve.

The tool was developed using MDD tools provided in Eclipse, and it is supported by the TwoUse toolkit for the creation of ontology and rule models. Metamodels for representing platform-specific models in Jess and the Java/JSF are defined using EMF[5] (Eclipse Modeling Framework).

Model-to-model (M2M) transformations are designed with ATL[6] (Atlas Transformation Language). The first one (bottom flow in Fig. 1) maps an ontology and rule model to a Jess platform-specific model. The second one (top flow in Fig. 1) transforms the ontology model into a Java/JSF Web specific model.

The outputs of both ATL transformations are the respective inputs of two model-to-text (M2T) transformations implemented in JET[7] (Java Emitter Templates). As a result, the code for the rule-based Web application is obtained. On one hand, source files with Jess rules and facts, and on the other hand, the Web application components, the configuration files, the Java classes, and a Jess-Engine Bean which uses the Jess API (Application Programming Interface) to embed the rule engine into the architecture. Moreover, a set of JSP/JSF web pages is generated for the user interface which are based on the RichFaces library [10] framework that adds AJAX capability to JSF applications. Default configuration is injected to provide presentation templates for pages, predefined navigation between them and default functionality for the generated application.

3 Architecture of rule-based Web applications

Figure 2 shows the target architecture for the generated rule-based Web applications.

[5] http://www.eclipse.org/modeling/emf/
[6] http://www.eclipse.org/m2m/atl/
[7] http://www.eclipse.org/modeling/m2t/?project=jet

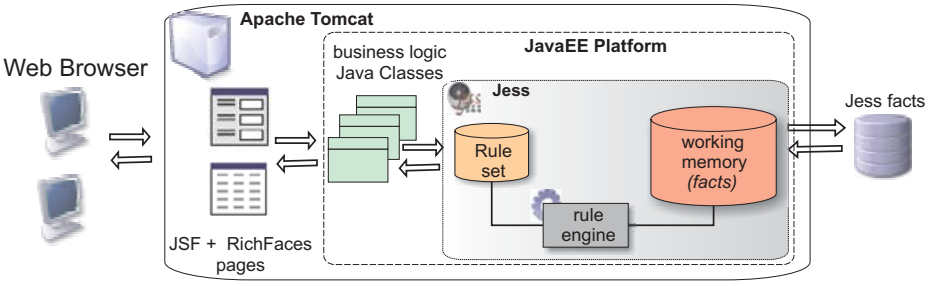

Fig. 2. Architecture of a Web rule-based application

The embedded rule engine manages the Jess rule base and the text file of persistent instances of concepts, called *facts*. Basically, the rule engine consists of three parts: the working memory contains all the information or *facts*, the rule set are all the rules, and the rule engine checks whether the rules match the working memory and executes them then.

The Web application enables the user to perform basic functions for instance management (create, list, update and delete instances). Current tools for automatic generation of that kind of Web applications perform those operations on relational databases. The contribution of our approach is that instance management is executed on the rule engine working memory. The rule engine executes a forward-chaining inference mechanism to drive the reasoning process, firing the rules with conditions evaluated as true, and executing their actions to infer new values or modify existing ones.

The use of both rules and AJAX technology improves instance management since each single value is validated and submitted as it is entered by the user, then the rule engine can fire suitable rules and deduce new information on the fly, driving the instance creation or edition.

4 Conclusion and future work

This tool presentation paper presents a tool that applies MDD to rich Web system development, incorporating a rule engine for deduction and inference. OWL and SWRL formalisms are used as modeling languages by the model-driven approach.

Since expressiveness in rule-based production systems such as Jess is poorer than OWL and SWRL semantics, only a subset of those formalisms is currently supported. Issues related with the combination of rules and ontologies have also an effect on the proposed approach. For example, the semantics of OWL and SWRL adopts an open world assumption, while logic programming languages such as Jess are based on a closed world assumption. As a consequence, only DL-Safe SWRL rules [11] are transformed to Jess. DL-Safe SWRL rules are a restricted subset of SWRL rules that has the desirable property of decidability,

by restricting rules to operate only on known individuals in an OWL ontology. Similarly, Jess rules only operate on facts in the working memory.

Another semantic difference between OWL and classic rule engines such as Jess is related to the fact base and state assertions. While in OWL the ABox containing the assertions about the individuals in the domain can not be modified, on the other hand, in a rule-based systems facts can be asserted and modified during the inference.

Although these semantic differences are present, the proposed tool demonstrates how OWL and SWRL can be used as specification formalisms in rule-based Web applications development. The benefits come from the widely use of these formalisms and the many tools available for editing and reasoning over OWL and SWRL specifications.

The tool is planned to be evaluated in several domains such as pest control in agriculture and medical diagnosis. Future work extends the generated Web application with semantic Web functionalities, such as semantic search based on ontology classes hierarchy as well as instance search.

References

1. W3C OWL Working Group: OWL 2 Web Ontology Language: Document Overview. W3C Recommendation (2009) Available at http://www.w3.org/TR/owl2-overview/.
2. Horrocks, I., Patel-Schneider, P., Boley, H., Tabet, S., Grosof, B., Dean, M.: SWRL: A Semantic Web Rule Language combining OWL and RuleML. W3C Member Submission (2004) Available at http://www.w3.org/Submission/SWRL/.
3. Eiter, T., Ianni, G., Krennwallner, T., Polleres, A.: Rules and ontologies for the semantic web. In Baroglio, C., Bonatti, P.A., Maluszynski, J., Marchiori, M., Polleres, A., Schaffert, S., eds.: Reasoning Web. Volume 5224 of Lecture Notes in Computer Science., Springer (2008) 1–53
4. W3C: A Semantic Web Primer for Object-Oriented Software Developers. W3C Working Draft (2006) Available at http://www.w3.org/TR/sw-oosd-primer/.
5. Parreiras, F.S., Staab, S., Winter, A.: TwoUse: integrating UML models and OWL ontologies. Technical report, Universität Koblenz-Landau (2007)
6. Object Management Group: Ontology Definition Metamodel. Version 1.0. OMG (2009) Available at http://www.omg.org/spec/ODM/1.0/.
7. W3C OWL Working Group: OWL 2 Web Ontology Language: Structural Specification and Functional-Style Syntax. W3C Recommendation (2009) Available at http://www.w3.org/TR/owl-syntax/.
8. Friedman-Hill, E.: Jess in Action: Java Rule-Based Systems. Manning Publications (2003)
9. Geary, D., Horstmann, C.S.: Core JavaServer Faces. 2 edn. Prentice Hall (2007)
10. JBoss: RichFaces (2007) http://www.jboss.org/jbossrichfaces/.
11. Motik, B., Sattler, U., Studer, R.: Query Answering for OWL-DL with rules. Journal of Web Semantics **3**(1) (2005) 41–60