

See discussions, stats, and author profiles for this publication at: <https://www.researchgate.net/publication/220727230>

# RuleModeler: A rich internet application for knowledge modeling

Conference Paper · January 2010

DOI: 10.1145/1839707.1839742 · Source: DBLP

CITATIONS

0

READS

18

4 authors, including:



**Joaquín Cañadas**

Universidad de Almería

36 PUBLICATIONS 106 CITATIONS

[SEE PROFILE](#)



**Francisco Javier Orellana**

Universidad de Almería

19 PUBLICATIONS 102 CITATIONS

[SEE PROFILE](#)



**Samuel Túnez**

Universidad de Almería

49 PUBLICATIONS 124 CITATIONS

[SEE PROFILE](#)

Some of the authors of this publication are also working on these related projects:



Aplicación de Aprendizaje Basado en Proyectos en materias de INGENIERÍA DEL SOFTWARE [View project](#)

# RuleModeler: A Rich Internet Application for Knowledge Modeling

Joaquín Cañadas  
Dept. Computer Science  
University of Almería  
Spain  
jjcanada@ual.es

F. Javier Orellana  
Dept. Computer Science  
University of Almería  
Spain  
fjorella@ual.es

Raquel Salmerón  
Dept. Computer Science  
University of Almería  
Spain  
rsalmeron@ual.es

Samuel Túnez  
Dept. Computer Science  
University of Almería  
Spain  
stunez@ual.es

## ABSTRACT

Traditionally, the use of standard Web-based architecture for knowledge modeling and management suffers from important weaknesses. This work describes the use of current technology of Rich Internet Applications (RIAs) in tools for knowledge modeling and management. As a proof of the concept we have implemented *RuleModeler*, a RIA tool prototype for knowledge modeling where knowledge models composed of ontologies and production rules can be specified in a distributed way. The tool has been used to create and maintain the knowledge model for a pest control decision-support system in agriculture.

## Categories and Subject Descriptors

I.2.4 [Artificial Intelligence]: Knowledge Representation Formalisms and Methods—*Representations (procedural and rule-based)* ; D.2.2 [Software Engineering]: Design Tools and Techniques—*Computer-aided software engineering*

## General Terms

Design

## Keywords

Knowledge Modeling, Supporting tools, RIAs

## 1. INTRODUCTION

Currently the development of knowledge-intensive applications require huge, complex knowledge models which are usually created and maintained by teams of experts and knowledge engineers working in cooperation. In order to involve experts in modeling tasks, supporting tools with two

key features are necessary: (1) distributed, simultaneous multi-user access to the models and (2) an easy to learn, to use, powerful and friendly user interface. Since domain experts are generally unfamiliar with knowledge representation languages and techniques, efficient supporting tools are desirable so they can participate actively in the specification of the knowledge necessary for the project.

Formalisms for knowledge representation and ontology development are widely extended in software development. Rules and ontologies play a key role in the architecture of the Semantic Web, since they are used to provide meaning and reasoning facilities to semantic web applications [4]. The use of standard Web-based server-client architecture for knowledge modeling suffers from being slow, especially for large ontologies and depending on network traffic, and difficult for maintaining consistency while editing [8]. These limitations can be reduced using current approaches for Rich Internet Applications (RIAs) [3]. RIAs provide sophisticated and more interactive user interfaces for web applications, similar to desktop applications, while minimizing network traffic overhead and increasing user usability and efficiency [10].

This work focuses on the use of RIAs for knowledge modeling through an authoring tool prototype supporting knowledge model elicitation, the *RuleModeler* tool, in which ontologies and production rules can be specified. It makes use of RIA technology to minimize client-server data transfers by moving the interaction from the server to the client. Thus, it covers the need for a simple and richly web-based tool supporting stakeholders involved in a project to collaborate in the construction of knowledge models, providing rich functionality and a built-in mechanism for user data validation, reducing input errors.

*RuleModeler* is currently being used in the SAVIA project, a decision-support system for pest control in greenhouse crops that is being developed using rule-based reasoning. This project is being helpful to test *RuleModeler* in a real case with real users, in order to gather the stakeholder's opinions, find bugs and determine future enhancements. It is available online at <http://www.dkse.ual.es/rulemodeler>. The tool was developed for supporting *InSCo* [2], a methodology for software development that intertwines knowledge engineering and software engineering approaches.

The rest of this paper is organized as follows: Section 2

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

I-SEMANTICS 2010, September 1–3, 2010 Graz, Austria  
Copyright 2010 ACM 978-1-4503-0014-8/10/09 ...\$10.00.

describes the prototype tool, its functionality and the technology applied in its development. Section 3 provides a description of the system. Related work is reviewed in Section 4, and finally, main conclusions and future work are summarized.

## 2. THE RULEMODELER TOOL

*RuleModeler*, the main result presented here, is a RIA tool prototype for knowledge modeling that enables authoring and maintenance of knowledge models composed of light-weight ontologies [6] and rules, used in the development of knowledge-based systems. Knowledge models are based on Conceptual Modeling Language (CML) proposed in CommonKADS [9]. Models are organized in modeling projects, which are composed of one or several domain schemas and one or several knowledge bases. Domain schemas allow to model objects and domain relationships:

- Concepts and their attributes, as a representation of objects or actions of the real world. Concepts can be hierarchically classified using inheritance relations. An attribute refers to a value which is atomic and is defined through a value type, which can be a primitive type, a concept or a enumerated type.
- Binary relations, used to define relationships between concepts, specifying a role and a cardinality. Relations can have subtypes as well as attributes.
- An enumerated type consists of a list of constant values established at the time of the declaration of the type.
- Rule types, used to define rule sets containing rules with similar structure, that manages expressions of the concepts, not the concept itself.

These elements defining a domain schema are instantiated in a knowledge base which describes the knowledge of the specific domain, comparable to database content. In a knowledge base the concepts are instantiated specifying a name and their attribute values; relations are instantiated defining the concrete concept instances that are related, and rule instances are defined following the rule pattern described in the rule types. Rules are considered to have most of the importance of a knowledge base because since we start from some requirements, objectives and restrictions, using that rules, we will generate an action plan.

*RuleModeler* enables the creation of knowledge models in different and independent modeling projects. Each project has its own development team, a group of users defined by an administrator in charge of managing the access permissions for each project. Every member of a development team can participate in a project at the same time to specify the knowledge models, interacting with the rich predefined web templates that enable the creation of each element of the model.

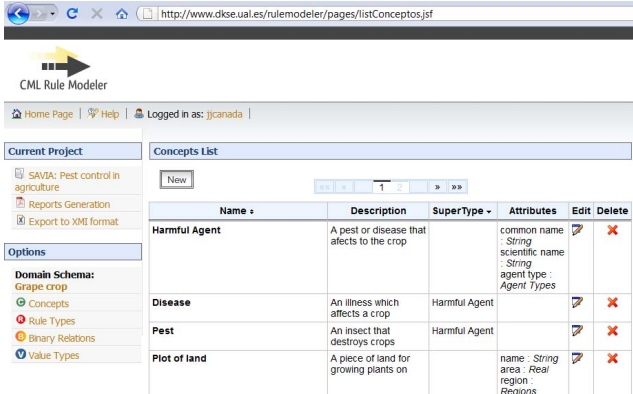
Information stored in a project can be gathered and used to generate project reports in pdf format. Exportation functionality is also available enabling to reuse models in other development tools. Thus, *RuleModeler* knowledge models can be exported using an interchange format which provide interoperability with a tool for web rule-based systems construction [1].

*RuleModeler* was developed with JavaEE. The JavaServer Faces framework (JSF) [5] was used to materialize the Model-View-Controller pattern in the architecture and uses AJAX which adds capabilities to a web application, thereby providing user interface facilities. AJAX technology, provided by JBoss RichFaces framework [7], performs asynchronous calls to the server without refreshing the whole page, improving network bandwidth use and providing greatly improved user experience more reliably and more quickly, since the interface is updated in the client browser, which is one of the main features of RIAs. In this context, *RuleModeler* is intended to be a Web-based tool providing distributed knowledge elicitation by a powerful user interface based on components that previously were only found in desktop applications.

## 3. SCENARIO

This section describes some of the main features of the tool *RuleModeler*, using for this purpose a preexisting real model of agricultural application domain. Screen captures are from SAVIA project, a decision-support system for pest control in crops of the south-east of Spain that is being developed using rule-based reasoning. The goal of the SAVIA system, accessible via Internet and mobile devices, is to help growers and agricultural technicians to operate correctly in their crops with independence of their geographic location.

Figure 1 illustrates a specific section from the SAVIA model where a list of concepts from the domain schema is displayed. As the figure shows, the SAVIA knowledge model consists of concepts such as Harmful Agent, Crop, Grower, Agricultural Technician or Phenological Stage, all of them representing objects from the real world. This view also shows some properties of the concepts: name, description, superType, and attributes, which will be used later for the creation of rules. The concepts and their properties are displayed using a rich component from RichFaces which generates a HTML table, including some functionality to sort the data by any field, and a pagination function that allows to show the results in several pages.



Name	Description	SuperType	Attributes	Edit	Delete
Harmful Agent	A pest or disease that affects to the crop		common name : String scientific name : String agent type : Agent Types		
Disease	An illness which affects a crop	Harmful Agent			
Pest	An insect that destroys crops	Harmful Agent			
Plot of land	A piece of land for growing plants on		name : String area : Real region : Regions		

Figure 1: Snapshot of SAVIA concept list

The instances of elements of the domain knowledge are described in the SAVIA knowledge base. For example, related to the concepts previously explained, Grapes would be an instance of Crop; Grape moth, Thrips and Botrytis would be instances of Harmful Agent; and Dormant bud, Begin flowering and Cap falling would be instances of Phe-

nological Stage.

Rule instances also belong to the knowledge base, and are used to specify logical constraints between concept attributes. An example of the creation of a rule instance is shown in Figure 2. The figure shows a form that allows the users to define a rule instance, indicating some antecedent and consequent expressions. The editor provides some negation, conjunction and disjunction operators which makes possible the creation of complex rules. An example rule has been defined, using some attributes from the Visit concept: "if the date of a Visit is in January, then the phenological stage is Dormant bud".

**Figure 2: Snapshot of SAVIA rule instance definition with RuleModeler**

The rich functionality included in this tool makes all these management tasks easier. Every operation involving the edition of a list, adding a new item or removing an existing one, can be performed without refreshing the whole page. This makes possible to save time in every task maximizing the user satisfaction. This advantage can also be found when forms data are validated, since errors are shown in the current form, or whenever a user needs to filter any data shown in a table.

## 4. RELATED WORK

Numerous tools are currently available for specification of knowledge models based on ontologies and rules. Protégé is the most widely used tool. It is open source and supports several ontology formalisms to construct domain models and knowledge-based applications, using a desktop environment. Focusing on Web-based tools, two representative samples are Web-Protégé<sup>1</sup> and OntoWiki<sup>2</sup>. Web-Protégé is a lightweight ontology editor for the Web that uses Protégé as its backend, supporting a collaborative ontology development process. OntoWiki is one of the recent approaches for using social Social Web and semantic techniques as knowledge acquisition approach. In this context, *RuleModeler* is intended to be a Web-based tool providing a distributed knowledge elicitation environment by a powerful user interface based on rich components, in order to make the interaction with the tool easier.

<sup>1</sup><http://protegewiki.stanford.edu/index.php/WebProtege>

<sup>2</sup><http://ontowiki.net>

## 5. CONCLUSIONS AND FUTURE WORK

This paper introduces *RuleModeler*, a prototype tool for knowledge modeling based on RIA technology to provide distributed access and easy user interaction based on web-form templates with an efficient, intuitive interface. This first version of the tool was tested specifying a pest control knowledge model in an agricultural domain. Its possibilities and deployment in other domains may be evaluated. Moreover, it is intended to be our first attempt at providing basic collaboration between experts and knowledge engineers.

As future work, enriching current templates for specifying more extended properties of ontologies and rules would be desirable, as well as adding the capacity of exporting knowledge models to other representation languages, such as OWL (Web Ontology Language) or SWRL (Semantic Web Rule Language). Finally, we are considering the incorporation of collaborative functionality to promote communication among the users.

## 6. ACKNOWLEDGMENTS

This work was supported by the Spanish Ministry of Education and Science under the project TIN2004-05694 and by the Junta de Andalucía (Andalusian Regional Govt.) project P06-TIC-02411.

## 7. REFERENCES

- [1] J. Cañadas, J. Palma, and S. Túnez. InSCo-Gen: A MDD tool for Web Rule-Based Applications. In *ICWE*, volume 5648 of *Lecture Notes in Computer Science*, pages 523–526. Springer, 2009.
- [2] I. M. del Águila, J. Cañadas, J. Palma, and S. Túnez. Towards a methodology for hybrid systems software development. In *Proceedings of the Int. Conference on Software Engineering and Knowledge Engineering (SEKE)*, pages 188–193, 2006.
- [3] M. Driver, R. Valdes, and G. Phifer. Rich internet applications are the next evolution of the web. Technical report, Gartner Research Note. G, 2005.
- [4] T. Eiter, G. Ianni, T. Krennwallner, and A. Polleres. Rules and ontologies for the semantic web. In *Reasoning Web*, volume 5224 of *Lecture Notes in Computer Science*, pages 1–53. Springer, 2008.
- [5] D. Geary and C. S. Horstmann. *Core JavaServer Faces*. Prentice Hall, 2 edition, May 2007.
- [6] A. Gómez-Pérez, M. Fernández-López, and O. Corcho. *Ontological engineering*. Springer, July 2004.
- [7] JBoss. RichFaces, 2007. <http://www.jboss.org/jbossrichfaces/>.
- [8] A. Kalyanpur, B. Parsia, E. Sirin, B. C. Grau, and J. Hendler. Swoop: A web ontology editing browser. *Web Semantics: Science, Services and Agents on the World Wide Web*, 4(2):144–153, June 2006.
- [9] G. Schreiber, H. Akkermans, A. Anjewierden, R. de Hoog, N. Shadbolt, W. V. de Velde, and B. Wielinga. *Knowledge Engineering and Management: The CommonKADS Methodology*. The MIT Press, 2000.
- [10] J. M. Wright and J. Dietrich. Requirements for rich internet application design methodologies. In *WISE*, volume 5175 of *Lecture Notes in Computer Science*, pages 106–119. Springer, 2008.