K-SITE RULES
Integrating Business Rules in the Mainstream Software Engineering Practice[1]

José L. Martínez-Fernández, José C. González and Pablo Suárez
DAEDALUS – Data, Decisions and Language, S.A., Avda. Albufera 321, Madrid, Spain

Keywords: Business rules, rule engine, rule language, software development lifecycle models.

Abstract: The technology for business rule based systems faces two important challenges: standardization and integration within conventional software development lifecycle models and tools. Despite the standardization effort carried out by international organizations, commercial tools incorporate their own flavours in rule languages, making difficult the migration among tools. On the other hand, although some business rules systems vendors incorporate interfaces to encapsulate decision models as web services, it is still difficult integrating business rules in traditional object-oriented analysis and design methodologies. This is the rationale behind the development of K-Site Rules, a tool that facilitates the cooperation of business people and software designers in business applications.

1 INTRODUCTION

A Business Rule (BR) can be seen as an expression used to make explicit the knowledge about the business present in an organization. According to the Business Rules Group[2] two definitions can be given for the term business rule, depending on the point of view adopted: from the business point of view, where a business rule is "*a directive, intended to influence or guide business behaviour, in support of business policy that has been formulated in response to an opportunity, threat, strength, or weakness*", from the Information Technology (IT) point of view a BR can be defined as "*a statement that defines or constrains an aspect of the business. It is intended to assert business structure, or to control or influence the behaviour of the business*".

A Business Rules Engine (BRE) or Business Rules System (BRS) is a software system in charge of executing IT business rules. These systems usually include tools to support the definition, verification, validation and management of business rules.

In the last few years, the use of business rules systems is being widely adopted by companies to encode business knowledge. But, why? What are the benefits of using business rules? There are three main reasons: First, the knowledge about the business is stored in a somehow centralized, easily accessible storage, and this knowledge can be modified, updated and maintained in a quick and simple way. Second, companies need ways to foster their reaction to new business opportunities, and the ability to adapt their IT systems to take benefit of these opportunities becomes a definitive advantage. Third, from the IT perspective, the use of business rules allows a reduction of the effort needed to adapt software systems to new requirements. The ideal situation is one in which no code modifications are needed to satisfy these new restrictions. Let's try to explain this fact with an example: suppose you are managing a web site selling electronic devices and you want to make a 10% special discount to customers coming for their first time to the web site. You can implement this restriction using an if-then sentence in your code. Now, suppose you want to extend the special discount to every customer for a period of time. Then you should change your code and recompile it to include this new restriction. On the other hand, if you express the restriction as a business rule, you only have to change the business rule definition without changing the code of the application, and with no need to recompile your program.

Nowadays, there are a lot of commercial products implementing business rules engines. Blaze Advisor (Fair Isaac, 2007), ILOG JRules (ILOG

[1] This work has been partially supported by the Spanish Center for Industry Technological Development (CDTI Ministry of Industry, Tourism and Trade), through the project ITECBAN (Architecture for Core Banking Information Systems), INGENIO 2010 Programme.

[2] http://www.businessrulesgroup.org

L. Martínez-Fernández J., C. González J. and Suárez P. (2008).
K-SITE RULES - Integrating Business Rules in the Mainstream Software Engineering Practice.
In *Proceedings of the Tenth International Conference on Enterprise Information Systems - AIDSS*, pages 446-449
DOI: 10.5220/0001687504460449
Copyright © SciTePress

JRules, 2006) and JBoss Rules (Red Hat, 2007) are some of the most important ones. Of course, you have to make a decision about which product you are going to include in your IT architecture. Each product uses its own language to define business rules, so a standardization problem arises. Although there are several efforts to define a standard rules language such as RuleML (Boley, 2005), SWRL (Semantic Web Rule Language) (Horrocks, 2003) or RIF (Rules Interchange Format) (Boley, 2005), these languages are not implemented by commercial products.

On the other hand, if business rules are going to be used to implement software components in companies, there should be a way to include them in the software development lifecycle. For this purpose, tools supporting the harvesting, definition and implementation of business rules should be provided, as well as their utilization in the corresponding stages of the development lifecycle.

This work presents K-Site Rules, an industrial effort to build a tool that covers these two main problems. The first one, by helping in the standardization of business rules representations by providing automatic translations among SWRL and the languages interpreted by commercial products. The second one, providing a simple way to assure configuration management in the business rules development process and development tools, which can be easily integrated within standard software development lifecycle models.

The reminder of this paper is structured as follows: Section 2 describes key issues relating the integration of BRSs in the development cycle. Section 3 describes the architecture of the K-Site Rules solution, focused in assuring the independence of the commercial BRS selected to implement business rules. Section 4 includes some conclusions extracted during the design and development of K-Site Rules.

2 STAGES IN THE BUSINESS RULES DEVELOPMENT PROCESS

There exists a methodology to harvest and develop rules called PROTEUS (Ross, 2006). The development steps defined in K-Site Rules are a simplification of the PROTEUS approach. Before describing the methodology, some definitions must be given according to the point of view considered in K-Site Rules: a decision service is a set of atomic business rules and actions (i.e.: operations included in business objects); an atomic business rules is a rule that can be expressed in one sentence.

Four basic steps are considered:

- *Workflow definition*
  A flow must be defined among sets of atomic rules, making easier to understand the operations performed by the rule and allowing the reuse of already defined atomic rules and rulesets.
- *Atomic rule definition*
  Definitions for atomic rules are supported in different formats. In K-Site Rules an atomic rule can be defined using natural language, decision tables and decision trees. Concepts and facts available to define rules are taken from JAVA object models. Atomic business rules are expressed in the standard SWRL language.
- *Decision service validation*
  Once atomic rules have been defined and included in the decision service workflow and this decision service is complete, validation tasks must be done. The main goal is to test if the functional behaviour of the decision service is correct or not. In order to perform this validation, a target rule engine must be selected and K-Site Rules will automatically translate the decision service to the selected engine, performing predefined validations. Mechanisms to provide data for these validations are, of course, supported by the tool.
- *Decision service publication*
  A version control system is also integrated in K-Site Rules, to manage different versions of business rules and also to publish or deploying rule definitions.

The PROTEUS methodology considers an initial phase to build the business model for the organization. In a first version of K-Site Rules, this business model is inherited from the organization, so valid concepts and facts are those already defined in the target organization and included in some kind of JAVA business objects repository. Nowadays, organizations are taking the first steps to define ontologies to act as some kind of dictionaries to compile business knowledge. The ability to represent this business knowledge using ontologies is also being considered in the development of K-Site Rules. This is the reason to choose SWRL as the language to build standard representations of business rules.

3 A COMPREHENSIVE ARCHITECTURE FOR BR SYSTEMS DEVELOPMENT AND INTEGRATION

Most of Rule-Based Systems offer an architecture similar to the one showed in Figure 1.

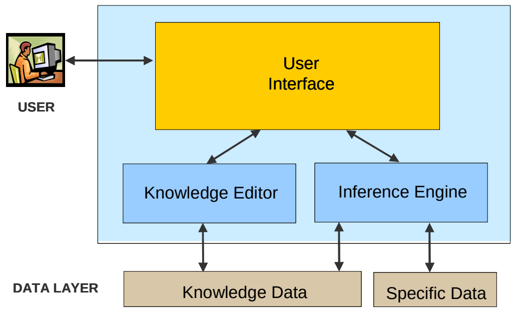

Figure 1: Rule-Based System general architecture.

The user interacts with the Rule-Based system through a User Interface. It allows access to a Knowledge Editor that supports the creation or modification of new rules, possibly using natural language. These rules are stored in a repository which constitutes the Knowledge Data element in Figure 1. Along this process, the Knowledge Editor could make use of previous data stored in this Knowledge Data repository. Rules stored in this repository are typically structured in form of *if-then-else* logical propositions. On the other hand, the Inference Engine or Rules Engine obtains inferences starting from the Knowledge Data and the Specific Data. Obviously, the Inference Engine can use intermediate data generated in successive steps along the inference process.

K-Site Rules implements a layer allowing the integration of Rule-Based Systems, and the architecture of the tool was thought in accordance with the following objectives:

- Make rules development independent from rules engines, allowing the integration with several different engines (view Figure 2).
- Develop and maintain rules in a simple and intuitive way.
- Coordinate all the processes involved in rules definition for making easier the reuse of the rules.
- Control access to rules according to the user profile.

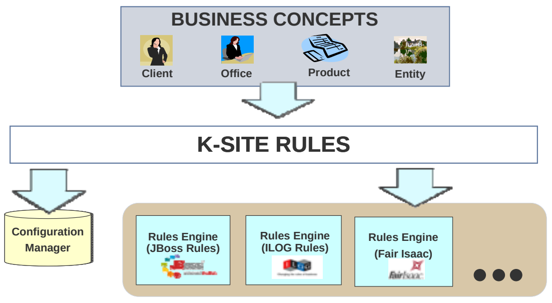
BUSINESS CONCEPTS
Client
Office
Product
Entity
K-SITE RULES
Configuration Manager
Rules Engine (JBoss Rules)
Rules Engine (ILOG Rules)
Rules Engine (Fair Isaac)

Figure 2: A global vision of K-Site Rules Integration.

With the aim of reaching all these objectives, the basic architecture of K-Site Rules is depicted in Figure 3.

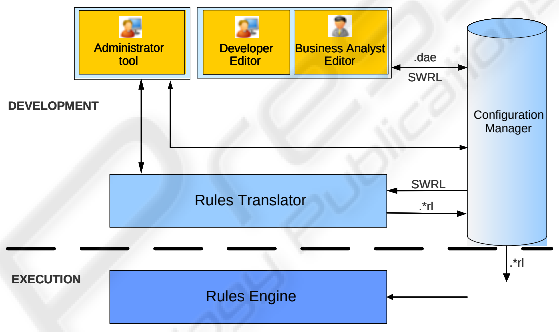

Figure 3: K-Site Rules Architecture.

K-Site Rules offers an editor for each different role. In particular, we include in our architecture a Business Analyst Editor and a Developer Editor. The Developer Editor was developed as a plug-in in an eclipse-based IDE. This plug-in allows the integration of the editor in a modelling tool for the definition of business rules. The Business Analyst Editor was developed as a friendly web tool allowing access to the system through a standard browser. It can also be integrated with the corporate access control system. Both editors allow the user to create atomic *if-then-else* rules in three different ways: through guided natural language, decision tables or decision trees. The editors generate Knowledge Data in an intermediate standard language known as SWRL.

The Rules Translator component provides independence among the rules under development and the Rule Engines used to implement them. It performs the translation of the intermediate SWRL files to the specific Rule Engine files. These languages are those provided by rule engines like ILOG JRules (IRL) or JBoss Rules (DRL). The Rules Translator is also integrated with the Configuration Manager (Figure 3). The methodology defined to develop business rules with K-Site Rules includes a validation stage, where rules functionality must be tested. For this purpose, interaction with Rule Engines must be provided and the standard interface JSR-94 (Toussaint, 2003) is applied.

The Configuration Manager facilitates the reuse of Business Rules, validates their deployment, audits changes in Rules and controls and maintains historic data about Business Rules versions. In addition to these features, the Configuration Manager communicates all involved user roles (business analyst, developer and administrator) in a transparent way, so that the changes carried out by a role will be visible for the other. It controls the access permissions to rules for each user too, and makes it possible the integration with the corporate configuration manager. K-Site Rules provides support for different configuration servers, like Subversion (Collins-Sussman et al., 2007) or Clear Case (IBM, 2007).

Of course, administration tasks have also been considered and a specific tool to manage the tool has been developed.

Finally, the dashed line in Figure 3 represents the separation between the development process and the execution procedure. K-Site Rules only takes part in the development stage and the execution or production stage is out of the scope of the tool.

4 CONCLUSIONS

This paper presents K-Site Rules, a tool and a methodology for business rules integration and deployment. The main features of K-Site Rules are summarized in this couple of definition equations:

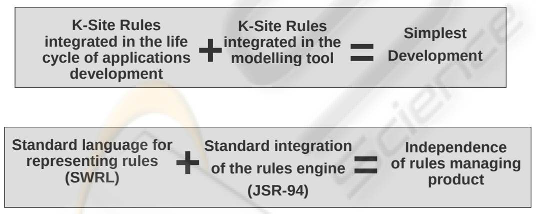
K-Site Rules integrated in the life cycle of applications development
+
K-Site Rules integrated in the modelling tool
=
Simplest Development
Standard language for representing rules (SWRL)
+
Standard integration of the rules engine (JSR-94)
=
Independence of rules managing product

Figure 4: K-Site Rules definition equations.

The main issue addressed by K-Site Rules is the effective cooperation of business people and software engineers in the design of software components, especially in the case of components incorporating declarative business knowledge in the form of business rules. In this sense, the main contribution of this work to the state of the art is the possibility of collaboration between business experts and software developers in a typical Integrated Development Environment (IDE), addressing key issues as configuration management. Another contribution of this work resides in the application of a combination of Semantic Web and Rule-based Knowledge technologies in an industrial environment. K-Site Rules tries to shorten the time gap between the moment when business decisions are taken and the moment when those business decisions get implemented.

K-Site Rules does not incorporate its own inference engine. It relies on commercial or open source software for the debugging and execution of business rules. Instead, it permits the specification of decision processes in the standard SWRL language, translating business rules to the proprietary languages of commercial rule engines.

REFERENCES

Boley, H., Grosof, B., Tabet, S. (2005) RuleML Tutorial, Draft. Retrieved 3/12/2007 from http://www.ruleml.org

Boley, H., Kifer, M., (2007). RIF Core Design. *W3C Working Draft 30 March 2007*, W3C Consortium. Retrieved 3/12/2007 from http://www.w3.org/TR/2007/WD-rif-core-20070330/

Collins-Sussman, B., Fitzpatrick, B.W., Pilato, C.M. (2007) *Version Control with Subversion: For Subversion 1.5*, Retrieved 3/12/2007 from http://svnbook.red-bean.com/nightly/en/

Fair Isaac, (2007) *Blaze Advisor Business Rules Management System: How it works*. Fair Isaac Corporation

Horrocks, P. F. Patel-Schneider, H. Boley, S. Tabet, B. Grosof, M. Dean, (2003) *SWRL: A Semantic Web Rule Language Combining OWL and RuleML, draft 0.5*. Retrieved 3/12/2007 from http://www.daml.org/2003/11/swrl/

IBM, (2007), *IBM Rational ClearCase*, Retrieved 3/12/2007 from ftp://ftp.software.ibm.com/software/rational/web/datasheets/clearcase.pdf

ILOG JRules, (2006). *Business Rule Management (BRM) with ILOG JRules 6. BRMS without compromise*. ILOG Inc.

Red Hat, (2007). JBoss Rules Fact Sheet. Red Hat Inc.

Ross G., Ronald, 2006, *Business Rule Concepts. Getting to the point of knowledge*, Business Rules Solutions LLC.

Toussaint, 2003. *Java Rule Engine API Specification JSR-94*, Draft 1.0, Retrieved 3/12/2007 from http://jcp.org/en/jsr/detail?id=94