2011 UKSim 5th European Symposium on Computer Modeling and Simulation

A Methodology for Graphical Modeling of Business Rules

Daniele Di Bona, Giuseppe Lo Re
Dipartimento di Ingegneria Chimica, Gestionale, Informatica, Meccanica
Università deli Studi di Palermo
Viale delle Scienze, 90128 Palermo (Italy)
E-mail: danieledibona@gmail.com, giuseppe.lore@unipa.it

Giovanni Aiello, Adriano Tamburo, Marco Alessi
Research and Development Laboratory
Engineering Ingegneria Informatica S.p.A.
Viale Regione Siciliana n. 7275, 90146 Palermo (Italy)
E-mail: {giaiello, atamburo, alessi }@eng.it

***Abstract* - This work proposes a novel methodology based on the Business Process Modeling Notation (BPMN) standard capable of graphically modeling business rules. A set of new representation patterns allows business analysts to map processes described through BPMN into conditions and actions of business rules.**
Our approach exploits Domain Specific Language techniques in order to make the methodology independent from the programming language supported by the specific rule engine.
Moreover, this work proposes a web graphical editor, instantiated on a specific sample scenario, where the selected rule engine is Drools, one of the most used open source products. The developed editor allows business analysts to graphically define business rules and to automatically generate executable code compliant with the selected rule engine. The case study and the resulting benchmarking show the effectiveness of the proposed methodology.

Keywords - rule engine; business rule; BPMN; graphical modeling; methodology; web graphical editor; BRMS

I. INTRODUCTION

The adoption of a rule engine for the execution of business rules is a widely considered solution to develop enterprise application software. The definition of business rules is normally performed by the business analyst (e.g. domain expert) by using natural language or graphical notations. However, the formal coding of business rules is generally performed by skilled technicians. Finally, in order to be executed, the business rules have to be translated - according to the particular language processed by the rule engine - into executable code.

In this scenario, in order to improve the development process, it is necessary to propose methodologies and tools specifically designed for business analysts, allowing the rule modeling in a graphical mode and the automatic production of runnable code that can be processed by the rule engine without being translated by an experienced technician.

Literature reports several approaches - based on the adoption of standard notation - capable of modeling Business Rules. In [1] a modeling tool that supports a UML-Based Rule Modeling Language (URML) is presented. URML language is an extension of the UML standard, which also supports rules in UML class diagrams. In [2] authors propose a UML graphical notation for the rules, based on Object Oriented modeling; here the concept of ruling on object diagrams to express constraints and dynamic behavior is introduced.

Both approaches require significant knowledge of UML modeling, which may result difficult for many business analysts.

The Protégé tool [3] includes several functionalities to represent business rules, by using standard ontology modeling notation, such as RDF [4] and OWL [5]. However Protégé does not provide facilities for visual representation of ontologies.

The methodology presented in [6] is based on the XTT approach, which aims at providing an integrated methodology for the design, the implementation, and the analysis of rule-based systems. In [7] authors propose the integration of Drools and XTT, in order to support visual modeling of the rule base. The results of XTT modeling are translated in DRL (Drools Language) files, which can be executed by the Drools engine. However, the weakness of these approaches is that the XTT rule language has not been standardized.

We propose a methodology capable of representing conditions and actions of a business rule by using a graphical notation.

Our methodology is based on the adoption of a standard notation that is well known by the business analysts: the Business Process Model and Notation (BPMN) [8]. BPMN provides a set of graphic elements that allow the business analyst to model a generic business process.

In order to define business rules we propose some patterns of representation to entirely map a rule with a BPMN process. To define the details of each specific condition and/or action contained in the rule we propose to use natural expressions according to DSL (Domain Specific Language) specification language.

For the validation of our methodology we developed a web-based graphical editor integrated with a Business Rule Management System (BRMS): it allows the generic business analyst to define and model the business rules by using the graphical elements of BPMN.

978-0-7695-4619-3/11 $26.00 © 2011 IEEE
DOI 10.1109/EMS.2011.68

In this paper we describe the integration of the graphical editor in Drools Guvnor [9]. This approach will enable the automatic translation of the rules modeled by the editor into Drools rule engine compliant language.

The remainder of the paper is organized as follows: Section II presents the most commonly used rule engines and current tools to graphically model business rules.

The proposed methodology is described in details in Section III. In this section, we also explain why a new methodology, adopting the BPMN standard to design the graphical modeling of rules, is necessary. Thus, we show BPMN representation patterns of the *when* and *then* branches of a generic business rule. Section IV presents a case study which describes the web graphical editor produced on the basis of our methodology, and shows, by example, how to model graphically a business rule. Finally, in Section V we report our conclusions and the future works.

II. Graphical Modeling of Business Rules

A business rule is any piece of knowledge that can be expressed in the following form:

When *'something' is true,* ***Then*** *do 'this'.*

According to the typical pattern of first-order logic, a business rule takes the form of a condition (or set of conditions) followed by an action (or set of actions). This allows describing every business rule by identifying the conditions for the activation of the rule and the actions to be executed after the activation is performed. The former part is known as *when branch* or *left hand side* (LHS), the latter part is called *then branch* or *right hand side* (RHS) of the rule.

A business rules engine is a software system that exploits artificial intelligence techniques, capable of executing business rules with respect to a knowledge base (also known as working memory). This represents the domain of interest of rules, and contains several assumptions over which rules act.

The literature contains various works concerning rule engines, such as Microsoft BizTalk [10], IBM WebSphere JRules [11], Jess [12], Blaze Advisor [13], OpenRules [14] and JBoss Drools [15].

Microsoft BizTalk is a business process management tool that includes a forward chaining inference engine and a business rule editor. Such editor allows writing rules in a guided text mode, in order to produce a decision tree. Nevertheless, the editor is suited to developers and administrators, but it is not friendly for non-technical people.

IBM WebSphere JRules is a family of products for creating and deploying rule-based applications for mainframe and SOA environments. In order to define business rules, JRules includes Rule Studio [16], an Eclipse [17] plug-in, designed to guide the user through the writing and the deployment of rules. Unfortunately, Rule Studio is a text-based environment; therefore software-programming skills are required. Furthermore, JRules has an easy to use Business Rules Management System (BRMS). It allows non-technical users: a) to generate rules starting from Microsoft Word and Microsoft Excel documents and, b) to write rules in a guided textual mode from templates or decision tables or decision trees. Nevertheless, JRules BRMS does not allow graphical editing of conditions and actions of rules.

Jess is a stable well-known rule engine. The rule description language of Jess is similar to the one of Lisp: often confusing and not friendly for non-technical people.

Blaze Advisor is a rule engine and Business Rules Management System that provides many tools for the development of rule based decision support systems. It includes an Eclipse plug-in to write rules in SRL (Structured Rule Language), which is not suitable for business people.

OpenRules is an open source rule engine and a Business Decision Management System (BDMS) that provides many tools for the development of rule based decision support applications. It allows business analysts to import and edit business rules from Microsoft Excel, Microsoft Word, OpenOffice or Google Docs documents. OpenRules includes an Eclipse plug-in designed for technicians to edit rules in Java programming language. It provides non-technical people with a guided text editor that leads to the drafting of rules through decision tables. However, OpenRules does not offer any graphical tools to define conditions and actions of rules.

JBoss Drools is an open source project developed by JBoss and Red Hat, Inc. It includes a business rules engine, and offers several tools for editing, managing and executing business rules. The Drools inference engine fires rules in the form of when/then statements, expressed in DRL, a language which is similar to Java. Drools, like JRules, is the only rule engine that provides a graphical editor known as Drools Flow. It allows graphical modeling of the execution flow among a previously defined set of rules. Nevertheless, Drools Flow, like JRules, does not let the business analyst to graphically model conditions and actions of rules.

In order to define business rules, several graphical notations have been analyzed. The flowcharts can be useful to describe the dynamics of a business rule. However, this notation does not allow the user to express sequences of OR conditions. The UML Activity Diagram, although very intuitive, does not provide the user with suitable graphics elements to define business artifacts and it is not designed to be used by business people.

III. BPMN representation of rules

Our work proposes a methodology enabling to graphically describe conditions and actions of a rule through an intuitive graphical notation.

The first issue we had to face was the choice on the graphical notation to be adopted.

The BPMN is an Object Management Group (OMG) standard known both by the business analysts, who create the initial drafts of the processes, and by the technicians, who develop the needed technology artifacts to perform those processes.

For this reason, we propose BPMN as the graphical notation to be used in the description of the business rules.

The definition of a methodology to map the graphical elements of BPMN with the various artifacts of a business rule was necessary in order to translate a graphically modeled rule into executable code, compliant with a rule engine. Furthermore, we propose a representation pattern showing how a generic user could define a rule by using our methodology.

According to the BPMN standard, complex processes can be modeled through Business Process Diagrams (BPD). A BPD is composed by a set of graphic elements which normally the business analysts are familiar with. In our work, a rule is represented by a simple BPD.

The proposed methodology exploits a subset of BPMN core items. In detail they are: Pool, Lane, Sub-Process, Task, Start Event, End Event, Or Gateway and Sequence Flow.

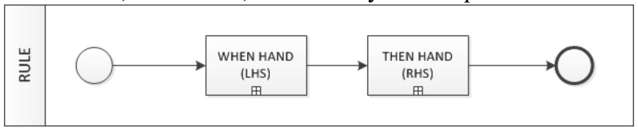


Figure 1. BPMN high-level graphical representation of a business rule.

Figure 1 shows the high-level BPMN representation of a business rule:

1. An entire rule is mapped on a single Lane contained into a Pool, because a Pool represents a single participant and a Lane represents a single activity;
2. The Start Event represents the beginning of the rule evaluation;
3. The first Sub-Process represents the LHS of the rule, so it contains the activation conditions;
4. The second Sub-Process represents the RHS of the rule, that is the actions to be executed when rule fires;
5. The End Event represents the end of operations.

The graphical element chosen to describe both conditions and actions of a rule is the BPMN Task Flow Object.

A. *LHS Representation Pattern*

Figure 2 shows the BPMN representation pattern of the *When* branch.

The LHS of the rule is represented by the *When BPMN Sub-Process.* Thus, it is possible to graphically design the sequence of conditions, which causes the activation of the rule. These conditions are modeled by BPMN Task flow elements inserted after a Start Event marking the beginning of the LHS.

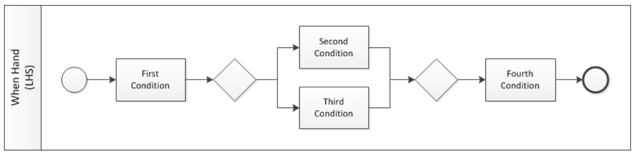


Figure 2. Example of BPMN representation of LHS : "1C and (2C or 3C) and 4C".

The individual conditions can be generally *and/or* related between them. Therefore, we defined AND and OR boolean graphical operators. The logical AND is obtained by connecting two generic Tasks via a Sequence Flow. The logical OR of two or more boolean conditions is done by placing the alternative tasks in parallel and by linking them to the heads of two Or Gateways. They represent the beginning and the end of the OR hand of the boolean conditions.

However, the graphical elements of BPMN do not allow describing the conditions of the rule, so it is necessary to find a method to assign a generic boolean expression for each Task condition. This association is performed by using DSL expressions. Specifically, several DSL statements have to be formalized in order to express a set of boolean conditions in the domain of interest.

The choice to adopt the DSL expressions is motivated by features of the DSL themselves. In fact, they allow mapping the proprietary format expected by the particular rule engine with simple natural language sentences. At compilation time, the DSL statements will be translated by the specific rule engine into runnable code.

B. *RHS Representation Pattern*

The graphical definition of rule actions is similar to the LHS one. Figure 3 shows the BPMN representation pattern of the RHS of the rule.

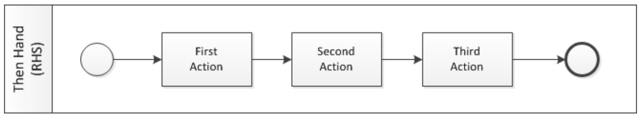


Figure 3. Example of then hand equivalent to the sequence of actions "action1; action2; action3".

As above mentioned, the RHS is represented by the *Then BPMN Sub-Process.* It is possible to graphically design the sequence of actions to be performed when the rule fires. The sequence is composed by the connection of the BPMN Tasks chain to the initial Start Event. Finally, an End Event has to be linked to the last element of the Tasks chain.

As previously stated for LHS, the proposed methodology requires the assignment of an executable action to a Task flow through the adoption of DSL expressions. Thus they map natural language phrases with executable actions. All Flow Objects are interconnected each other with Sequence Flows, which represent the execution order of the actions.

IV. Case Study

In order to verify the effectiveness of our methodology, we have integrated a new graphical editor in Drools Guvnor BRMS. Our work also included the definition of DSL expressions in order to allow non-technical people to associate them with BPMN elements, by using an extension of the guided text editor provided by Guvnor.

The case study describes the graphical modeling of a business rule designed for the ICT platform developed in the eXtended Net.lab research project [18]; this project focused on the development of a platform supporting the digital business ecosystems composed by small-medium enterprises.

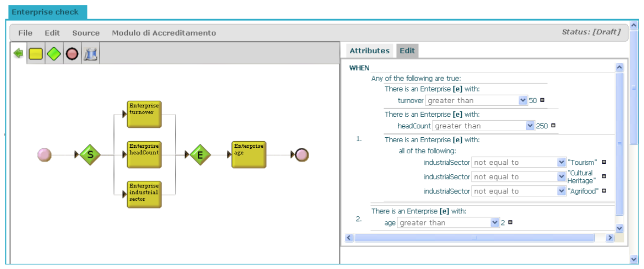
In this work we propose the BPMN design of a business rule used in the accreditation service provided by the ICT platform:

Business rule: *"**if** there is an enterprise whose turnover is more than 50 k€ or an enterprise whose head count is more than 250 or an enterprise whose industrial sector is different from Tourism and from Cultural Heritage and from Agrifood, and finally the enterprise founded is more than two years old, **then** this enterprise must be eliminated and a relative report must be produced".*

Our graphical editor shows the whole business rule as the BPD (Figure 1). Then, the LHS and the RHS views will be represented respectively through the *When* and *Then Sub-Process*.

The BPMN representation of the "Left Hand Side" of the rule is shown in Figure 4.

As the business rule contains three OR conditions ("*enterprise turnover*", "*enterprise head count*" and "*enterprise industrial sector*"), a BPMN OR Gateway branch is added after the Start Event. After that: a) three BPMN Tasks are included into the OR Gateways branch; b) the BPMN Task representing the AND condition ("*enterprise age*") is connected to the OR Gateways branch; c) finally the BPMN End Event is linked to the last BPMN Task.

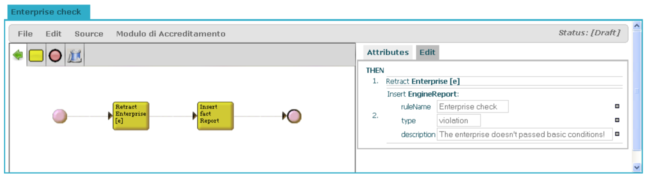


```
WHEN
    Any of the following are true:
        There is an Enterprise [e] with:
            turnover greater than 50
        There is an Enterprise [e] with:
            headCount greater than 250
1.      There is an Enterprise [e] with:
            all of the following:
                industrialSector not equal to "Tourism"
                industrialSector not equal to "Cultural Heritage"
                industrialSector not equal to "Agrifood"
    There is an Enterprise [e] with:
2.      age greater than 2
```

Figure 4. BPMN representation of the Left Hand Side of the rule.

The right side of Figure 4 shows the customized view of the Guvnor guided text editor. The left side of Figure 4 shows the BPMN representation of LHS of the rule, according to our methodology.

When a new condition is added to the graph through a BPMN Task, a new line appears in the guided text editor, that corresponds to the DSL expression "There is *something*". Here, "*something*" always refers to a fact included in the knowledge base of the rule engine. The details of conditions are fully specified by using other DSL statements. When the OR gateways branch is added to the graph, the DSL expression *"Any of the following are true"* automatically appears on the right side of the editor. All conditions within the OR Gateways branch will be indented below the above mentioned DSL expression.



```
THEN
1.  Retract Enterprise [e]
    Insert EngineReport:
        ruleName     Enterprise check
2.      type         violation
        description  The enterprise doesn't passed basic conditions!
```

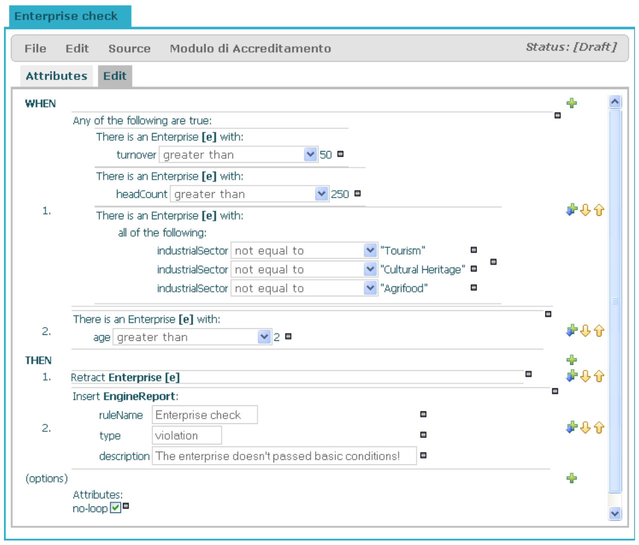
Figure 5. BPMN representation of the Right Hand Side of the rule.

The BPMN representation of the "Right Hand Side" of the rule is shown in Figure 5.

To represent the RHS of the rule, two BPMN Tasks are inserted into the BPD. When the business analyst adds a new Task in the BPD to express an action, she/he can choose among several DSL expressions already included in Guvnor guided text editor. Thus, the "*retract Enterprise*" task and the "*Insert fact Report*" Task can be associated with related DSL expressions, respectively, to remove the Enterprise fact from the knowledge base, and insert the new *EngineReport* fact into the working memory of the rule engine.

The following text is the DRL code generated by the editor:

```
rule "Enterprise check"
no-loop true
dialect "mvel"
when
            ( e : Enterprise( turnover > 50 ) or
              e : Enterprise( headCount > 250 ) or
              e : Enterprise( industrialSector != "Tourism" &&
                    industrialSector != "Cultural Heritage" &&
                    industrialSector != "Agrifood" )
            )
            e : Enterprise( age > 2 )
then
            retract( e );
            EngineReport fact0 = new EngineReport();
            fact0.setRuleName( " Enterprise check " );
            fact0.setType( "violation" );
            fact0.setDescription( "The enterprise doesn't
                      passed basic conditions!" );
            insert( fact0 );
end
```



```
WHEN
    Any of the following are true:
        There is an Enterprise [e] with:
            turnover greater than 50
        There is an Enterprise [e] with:
            headCount greater than 250
1.      There is an Enterprise [e] with:
            all of the following:
                industrialSector not equal to "Tourism"
                industrialSector not equal to "Cultural Heritage"
                industrialSector not equal to "Agrifood"
    There is an Enterprise [e] with:
2.      age greater than 2
THEN
1.  Retract Enterprise [e]
    Insert EngineReport:
        ruleName     Enterprise check
2.      type         violation
        description  The enterprise doesn't passed basic conditions!
(options)
    Attributes:
    no-loop ✓
```

Figure 6. Design of the rule with the guided text editor of Drools Guvnor.

Figure 6 shows the entire formalization of the same rule through the guided text editor of Guvnor. A simple rule, consisting of a few conditions and actions, can be confusing if defined through the guided text editor only. In our approach, the *When* and *Then* branches are separated and it is possible to keep a graphical overview of the rule.

Table 1 reports a comparison between our web graphical editor and other BRMS in terms of covered features. The comparison suggests Jess as the rule engine with less features. IBM ILog JRules and JBoss Drools offer a graphical tool to design the execution flow of the rules but they not allow graphical modeling of conditions and actions of rules. Our web graphical editor, integrated with Drools Guvnor is the only able to allow the business analyst to define conditions and actions of rules with a graphical approach.

Rule Engine	***BRMS***	***Guided Text Editor***	***Graphical Modeling of Rules Flow***	***Graphical Modeling of Conditions and Actions***
BizTalk	no	yes	no	no
JRules	yes	yes	yes	no
Jess	no	no	no	no
Blaze Advisor	yes	yes	no	no
OpenRules	yes	yes	no	no
Drools	yes (Guvnor)	yes	yes	no
	yes (Guvnor + our graphical editor)	yes	yes	yes

Table 1. Benchmark between our approach and other BRMS.

V. Conclusions

This paper describes a methodology to graphically model business rules by using the BPMN standard notation., The mapping of BPMN elements with left side of DSL statements allowed us to automatically generate code runnable on a rule engine. Such methodology is independent by the specific rule engine thanks to the DSL features. Namely, in order to produce rules compliant with a vendor specific rule engine it is only necessary to redefine the right side of the DSL expressions.

The effectiveness of our methodology has been tested on a case study. The proposed web graphical editor, integrated in Drools Guvnor BRMS, allowed us to define a correspondence between the BPMN elements and the DSL expressions used in the guided text editor. Through this user interface, the business analyst has a clear graphical overview of the designing branch and the logical interconnections among conditions and actions.

As future developments, we are improving our methodology in order to provide business analysts with enhanced modeling features. In particular, we plan: a) to support more BPMN items (e.g. Sub-Process into LHS and RHS views); b) to dynamically associate DSL expressions to graphical elements and export them.

References

[1] Lukichev, S., Wagner, G. (2006). UML-Based Rule Modeling with Fujaba. Proceedings of the 4th International Fujaba Days 2006, University of Bayreuth, Germany. Pages: 31 – 35.

[2] Gabriele Taentzer, Adding Visual Rules to Object-Oriented Modeling Techniques, 1999.

[3] Stanford University, Protégé v.4.1 opensource ontology editor, http://protege.stanford.edu/, July 2011.

[4] Klyne G., Caroll J.J. (Eds.), Resource Description Framework (RDF): Concepts and Abstract Syntax, W3C, 2004.

[5] Patel-Schneider, Peter F., Horroks I., OWL Web Ontology Language Semantic and Abstract Syntax, http://www.w3.org/2004/OWL, 2004.

[6] Grzegorz J. Nalepa, Maria A. Mach, Business Rules Design Method for Process Management. Proceedings of the International Multiconference on Computer Science and Information Technology pp. 165–170.

[7] Kluza, K., Nalepa, G.J., Łysik, Ł.: Visual inference specification methods for modularized rulebases. Overview and integration proposal. In Nalepa, G.J., Baumeister, J., eds.: Proceedings of 6th Workshop on Knowledge Engineering and Software Engineering (KESE2009) at the 32nd German Conference on Artificial Intelligence: September 21, 2010, Karlsruhe, Germany, pp. 6–17, 2010.

[8] Object Management Group, Documents Associated with Business Process Model and Notation (BPMN) Version 2.0,http://www.omg.org/spec/BPMN/2.0/, January 2011.

[9] Dennis Byron, JBoss Community, "Understanding the benefits of business rules management software in an open source ecosystem" http://www.jboss.com/pdf/brms/OpenSourceBRMSWhitepaper.pdf, 2010.

[10] Microsoft, "BizTalk Server Overview", http://www.microsoft.com/biztalk/en/us/overview.aspx, 2011.

[11] IBM, "WebSphere ILOG JRules", http://www-01.ibm.com/software/integration/business-rule-management/jrules/, 2011.

[12] Sandia National Laboratories, "Jess, the Rule Engine for the Java Platform", http://www.jessrules.com/, November 2008.

[13] Eager A., "FICO Blaze Advisor 6.7", http://www.fico.com/en/FIResourcesLibrary/ButlerGroupTechnologyAudit.pdf, September 2009.

[14] OpenRules, "Open Rules", http://openrules.com/, 2011.

[15] JBoss Community, "Drools – The Business Logic integration Platform", http://www.jboss.org/drools/, 2011.

[16] ILOG Inc., "Business Rule Management (BRM) with ILOG JRules 6 – Business Rule Management System without compromise", http://logic.stanford.edu/POEM/externalpapers/iRules/JRules_6_BRMS_Without_Compromise_2006032.pdf, March 2006.

[17] Eclipse Foundation, "Eclipse development environment", http://www.eclipse.org/, 2011

[18] Engineering Ingegneria Informatica, University of Lecce, "eXtended Net.Lab research project". Italian Research Innovation Ministry, ref.n. 602/Ric., March 2005.