JOINING BUSINESS RULES AND BUSINESS PROCESSES

Dirk Habich[1], Sebastian Richly[2], Birgit Demuth[2], Franziska Gietl[3], Joachim Spilke[3], Wolfgang Lehner[1], Uwe Assmann[2]

[1]*Dresden University of Technology, Database Technology Group, Dresden, Germany*
dirk.habich@tu-dresden.de, wolfgang.lehner@tu-dresden.de

[2]*Dresden University of Technology, Software Engineering Group, Dresden, Germany*
sebastian.richly@tu-dresden.de, birgit.demuth@tu-dresden.de, uwe.assmann@tu-dresden.de

[3]*University of Halle, Biometrics and Informatics in Agricultural Group, Halle, Germany*
franziska.gietl@landw.uni-halle.de, joachim.spilke@landw.uni-halle.de

Abstract. In today's world, the spheres of business rules and business processes are too loosely coupled. From our point of view, an appropriate joining of both spheres would prove beneficial for various application domains. Therefore, we propose an integrated approach to join rules and processes in a business-aware manner from a modeling and execution perspective. Our developed solution consists of four steps: (1) enhancement of classical business processes with SBVR annotations, (2) automatic integration of SBVR vocabulary into the business process world using domain-specific language transformations, (3) transformation of business rules to OCL constraints, and (4) manual mapping of domain-specific concepts to Web-service data structures. Based on our developed approach, several advantages arise, e.g., the independent modification of business rules without touching the business process structure or the runtime adaptation of rules. Our proposed solution is completely realized within the Open Service Process Platform.

Keywords: Business Rules, Business Processes, BPMN, RuleXpress, OCL.

1 Introduction

Business people describe the structures and processes of their reality in order to support continuous analyses and managerial decisions. Such modeling of the structures and processes results in a model describing a part of the real business world. In this case, the resulting model is an external and explicit representation of a certain part of reality, as seen by the business people who wish to use the model to understand, change, manage, and control this very part of the real business world. Normally, the structures and business processes include many rules, and these business rules are implicitly defined in the business model. Nowadays, two fundamental approaches exist to model and automate various aspects of the business behavior. The first approach focuses on flows of activities that generate some value for the considered part of reality. This approach, known as Business Process Management (BPM), describes "what" should be done [9]. Furthermore, BPM visualizes the "what"-aspect with diagrams and offers technologies to execute business processes. The second approach uses business rules to define desired business behavior. In this case, the rules describe "why" something has to happen [9].

On the one side, the sphere of business processes is well investigated, and the provided tools support the whole BPM life-cycle from the design, to the execution, to the monitoring. Business process modeling enables business people to define the desired look of the flow of activities. Prominent business process languages are the Business Process Execution Language (BPEL) [11] and the Business Process Modeling Notation (BPMN) [12], with BPEL being more technical and BPMN being more abstract. In terms of the MDA stack (Model-Driven Architecture) for business models, BPMN belongs to the platform-independent level (PIM), while BPEL belongs to the platform-specific level (PSM). To execute BPMN processes, a transformation into BPEL has to be performed. With the upcoming BPMN 2.0 specification [12], such a transformation becomes redundant when BPMN models can be executed directly. In general, BPEL as well as BPMN offers a standardized way to describe the functional composition of activities to create comprehensive process definitions.

On the other side, the sphere of business rules is well investigated, too. According to the Business Rules Group [13], a business rule is a statement that defines or constraints some aspect of the business. In this case, business rules are intended to assert business structures or to control the behavior of a business [10]. As several papers state, the business rules usually reside within the computationally independent model [4] to create the business design. The OMG's SBVR (Semantics of Business Vocabulary and Business Rules) approach [8, 16] is a business-aware method that allows a business analyst to create the business design in terms of business vocabulary and rules in some natural language format. However, the propagation of the business semantics from the business analyst to the IT people is not straightforward due to several issues.

Now, while concepts and approaches for both spheres are well known and the provided tools are heavily used for business model creation, a major drawback is the isolated treatment of both spheres. A more appropriate coupling – or as we call it, the joining of business rules and business processes – is especially useful in domains with rule-intensive business processes. In such scenarios, a lot of business constraints that have to be enforced throughout the process must be expressed in terms of activities, and they have to be explicitly integrated in the process definition. Hence, a rule-intensive business process definition contains plenty of nested-conditional activities (e.g., gateway chains in BPMN) that model decision-making points in the process [10]. Such process definitions are complex, non-modular, hard to maintain, and limited in their flexibility. To overcome these shortcomings, we have developed an integrated approach to join rules and processes in a business-aware manner from the modeling and execution perspective. Our solution, as illustrated in Figure 1, consists of four steps: (1) enhancement of BPMN 2.0 business process definitions with SBVR annotations, (2) automatic integration of SBVR vocabulary into the business process world with domain-specific language transformation, (3) transformation of business rules to OCL constraints, and (4) manual mapping of domain-specific concepts to Web-service data structures. Based on this joining concept, several advantages arise, e.g., the ability to modify business rules independently without touching the business process structure or to adapt rules at runtime. To demonstrate our solution, the whole concept has been realized within the Open Service Process Platform [7].

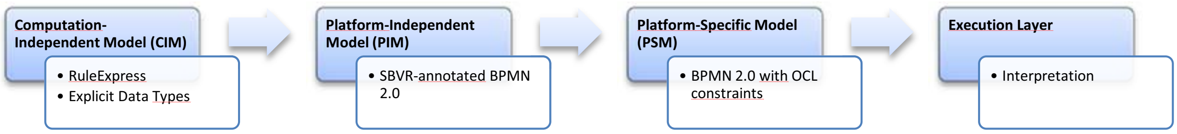

Figure 1. MDA-Stack of Our Integrated-Join Approch

The remainder of our paper is organized as follows. We start with a description of our background use-case scenario, which then drives us to our integrated-join solution in Section 2. In that section, we highlight the challenges and advantages of joining the spheres of business processes and business rules. Based on this background section, we introduce the modeling concept of our developed solution in Section 3. Section 4 focuses on execution concepts of our integrated-join approach. A revised use-case description in Section 5 serves as our evaluation section. We conclude this paper with related work in Section 6 and a summary in Section 7.

2 Use Case Scenario – Precision Dairy Farming

The reconciliation (and combination) of the economic and the ecological aspect of food production poses a major challenge of today's agricultural world. Consumer protection and transparency of production are also gaining in importance. These general conditions imply demands that also have to be met in dairy production. Therefore, big structural changes have taken place in the agricultural sector over the last years. In the field of dairy production, e.g., automated milking systems have been developed and herd sizes have grown. All such structural changes have led to much less contact between the farmer and individual animals.

Hence, new methods to compensate for this loss of contact and associated information are required. Also, a shift away from the whole group or herd towards the individual cow serving as the recipient of measures has taken place. Thus, the monitoring of individual animals has become very important. With the development of new technological equipment (e.g., sensors), the collection of data for each individual animal has been simplified. Derived from the above mentioned monitoring issue, the concept of *Precision Dairy Farming* [14] has been developed. Its implementation aims at the realization of (1) in-house goals and (2) greater goals. Examples of in-house goals include the optimization of the milk production branch, controlled feeding, or the early detection of diseases. Quality assurance, sustainability, animal welfare, and consumer protection as well as transparency of production are important issues for the greater goals.

Moreover, economic constraints are an eminent aspect within the domain of dairy farming. The value of an individual cow during its short useful period in life leads to increased pressure on good governance. For example, in this context and from the perspective of animal welfare, applied medication should be kept at a minimum. The early detection of diseases supports this goal and is based on the analysis of specific and well-defined data in order to derive an early-warning system to estimate and signal potential diseases. Another example is heat detection, where pedometers monitor the activity of the individual animal. This data is then analyzed with statistical approaches, with the rate of heat detection being at 70%. For all these monitoring and sensor-based data, a coordinated data management system is essential to cover all relevant aspects.

However, not all analysis and prediction algorithms can be run in the agricultural enterprise itself. One reason is the lack of analytical domain knowledge in small enterprises. For example, the calculation of the necessary mixed linear models requires specific mathematical knowledge, which is usually only present at a specific third-party service provider. A second reason is the missing underlying data basis or infrastructure. Some domain-specific processes (e.g. inter-company comparisons) require data from different companies in order to compute a comparison report. Such inter-company comparisons represent a trend within precision dairy farming to compare different agricultural enterprises and their strategies to find the best fitting procedure.

In general, precision dairy farming is characterized by a large amount of data caused by monitoring each individual cow. Every agricultural enterprise is located within a big network of external partners, which interact and exchange data amongst each other. External partners are, e.g., creameries, breed and control associations, and public authorities. An important aspect is the derivation of business-relevant information from the monitoring data. This derivation requires knowledge from different domains and involves complex processes. Another challenge lies in the heterogeneity of the partners and their IT systems. In order to highlight the necessity of our developed solution, we introduce a concrete business process of our use-case scenario in detail, with the help of BPMN, in Section 2.1. Furthermore, we summarize the drawbacks of the current realization concept.

2.1 Example Business Process

Figure 2 illustrates a typical example process – a so-called inter-company comparison – of our use-case scenario. As already mentioned above, this process compares different agricultural enterprises based on their animal health data. The control flow of this example is straightforward at first sight, since it is a simplified version of the process with only one enterprise. The simplification is to highlight the interesting features for this paper. In general, each participating agricultural enterprise sends health data about their animals to a third-party service provider. This service provider conducts the comparison based on the received health data and sends a comparison report back to the enterprises. As shown in Figure 2, the service provider executes two successive activities, *Validation* and *Comparative Study*, where the *Validation* is responsible to check the received data according to some specific constraints. Example constraints may be the following: (1) each cow must be assigned to exactly one herd, or (2) the health data must be from the previous month. If all constraints are satisfied, the ultimate evaluation can be executed. At this point, we can state that the *Validation* is a prerequisite for activation of the core business activity *Comparative Study*. However, this special property is not visible due to equal treatment.

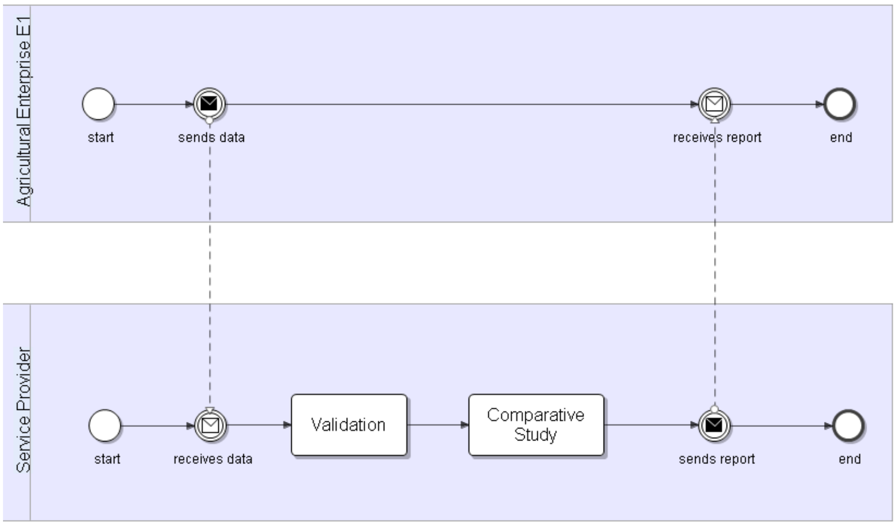

Figure 2. Inter-Company Comparison Process (BPMN Notation).

2.2 Constraints Consideration

As mentioned above, the pure control flow of the example process is straightforward. However, one question that has not been adequately addressed yet is the specification of the *Validation* activity. The activity *Comparative Study* normally includes an invocation of a specific service executing the desired business logic. The *Validation* can also be done by a specialized service. In this case, the constraints are not included in the process definition, and this violates the single-source principle (one source for all information). Changing the constraints involves external partners and may lead to different process documentations. To overcome this drawback, another option is to directly integrate constraints in the control flow using activities, e.g., the RULE activity of BPMN 2.0 [12]. This procedure is suitable for scenarios where few constraints have to be checked in the process definition. In case of many constraints, the resulting process will contain plenty of nested-conditional activities (gateway chains in BPMN), as depicted in Figure 3. In this case, shortcomings of such process definitions are that they are usually too complex for business people, not modular, hard to maintain, and restricted in their flexibility.

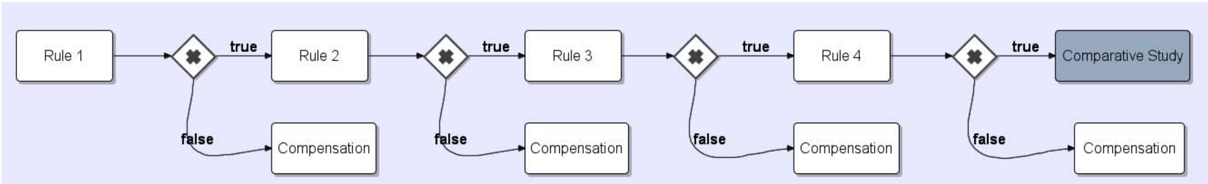

Figure 3. Nested-Conditional Activity Chain – Validation Realization.

To summarize, both available methods for integrating business rules in business processes are not efficiently applicable for rule-intensive business processes where a lot of rules have to be checked throughout the process. In particular, one shortcoming is the modification of business rules. The changing of rules results either in a change of external partners or a modification of the process definition. This shortcoming becomes severe when business rules change more frequently than the core business control flow does. In our use-case scenario evaluation, we found that this aspect holds for the considered precision dairy farming domain, and a more appropriate solution would be beneficial.

3 Modeling Perspective of the Integrated-Join Approach

As shown in Figure 1, we want to present an integrated-join approach that covers all aspects from the modeling to the execution. From the modeling perspective, we have to consider two parts: (1) how to specify rules, and (2) how to integrate them in business processes. The first part is realized using OMG's SBVR approach [8] as already proposed in several related papers [9, 10]. Fundamentally, the basic concepts of SBVR are noun concepts, roles, and fact types (all of them describing vocabularies) as well as rules (in the sense of integrity rules), and all the information is given by facts instantiated on the basis of fact types. Fact types contain one, two or more roles. For instance, the binary fact type "concept has designation" of the SBVR specification itself contains the two roles "concept" and "designation". Rules are special facts that still have another relation to fact types. They are propositions using fact types on the same modeling level to express constraints for instances of these fact types. There are two kinds of rule modalities: the alethic ones, e.g., a necessity, and the deontic ones, e.g., an obligation. Figure 4 illustrates an overview of the various language constructs of SBVR.

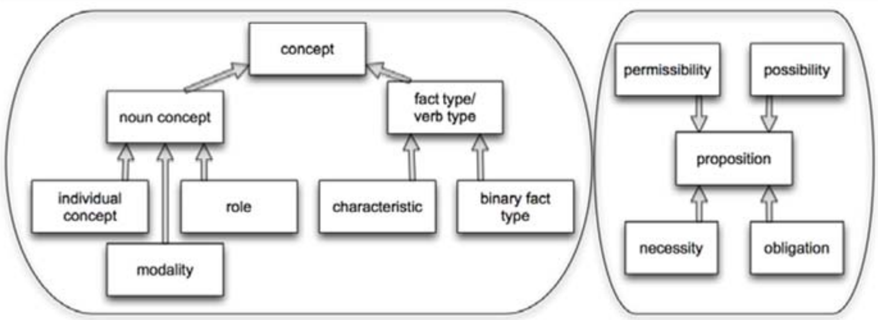

Figure 4. SBVR Main Elements.

For the second part of the modeling perspective, we propose a novel join approach, and this part focuses on how to integrate business rules in business processes. Fundamentally, we have to integrate alethic as well as deontic modalities. Therefore, we inherit all structural components as data definitions in a first step. In a second step, we require an opportunity to embed all behavioral rules. However, instead of directly integrating these rules using activities – resulting for many rules in a long nested-conditional activity chain – we use an annotation concept for joining rules and processes. If we take a closer look at the nested-conditional chain approach, we can observe that the end of a chain marks an activity that has to be executed when all rules are satisfied (see Figure 3). The end chain activity is usually a core business activity like the *Comparative Study* of our running example process. Therefore, the chain and, particularly, the included rules are tightly coupled with a core business activity, and the rules are prerequisites for the activation of this activity. To highlight this prerequisite in a more appropriate way, we annotate the corresponding rules directly at the core business activity, as depicted in Figure 5. The contents of these annotations are SBVR rule definitions (using the RuleXpress[1] notation).

Fundamentally, this annotation-oriented join approach offers several advantages. One such advantage is that the business rules and business processes are tightly coupled without complicating the process structure. Instead of integrating the business rules in the control flow, which often results in complex process definitions, we prefer an annotation of the corresponding business activities. A second major advantage is that the business rules can be changed without modifying the process definition. From our point of view, and using both presented advantages, our annotation-oriented modeling concept is perfectly suitable for business people. On the one hand,

[1] http://www.rulearts.com/RuleXpress

business people can express their business processes. On the other hand, they are able to enhance the modeled business process definition with SBVR rules at the corresponding business activities without modifying the control flow.

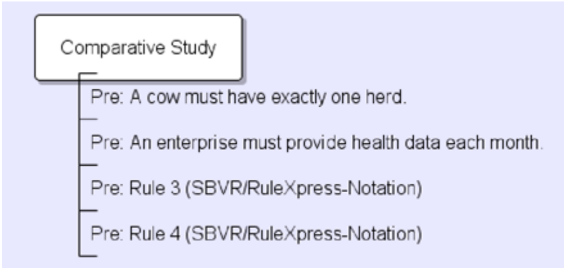

Figure 5. SBVR-annotated Business Activity.

4 Execution Perspective of the Integrated-Join Approach

While the previous section focused on the platform-independent model (PIM) according to the MDA stack by presenting an SBVR-annotated BPMN approach, this section now describes all essential steps to execute such modeled processes. In general, for this conveyance, we have to consider the level of platform-specific models (PSM) as well as the code or execution level. Important to note is that in the context of business processes, an interpreter approach is state-of-the-art for the execution. Therefore, we want to derive executable process definitions that are seamlessly interpretable at the code level. In this case, the most challenging issue is to define a way to transform SBVR rules into an interpretable constraint language. In our work, we decided to use the Object Constraint Language (OCL) [17] as target language for rule execution. The availability of an OCL interpreter offered by the Dresden OCL toolkit [17] is definitely a major reason. Another reason for this procedure is the corporate handling approach for business processes as well as business rules.

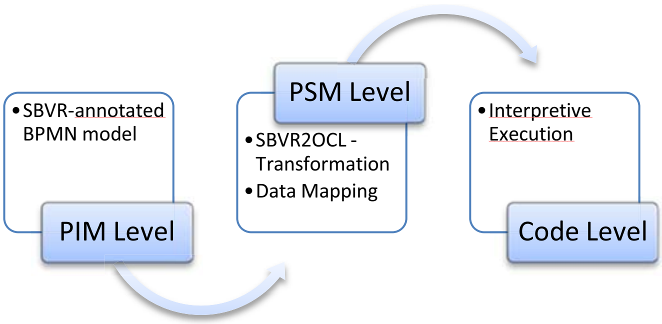

Figure 6. Transformation Details.

Our detailed transformation approach is depicted in Figure 6. We start with an SBVR-annotated BPMN process model as described in the previous section. The imposed annotations represent a shortcoming from the execution perspective. To overcome this issue, we transform BPMN processes with SBVR annotations into BPMN processes with nested-conditional rule chains during the transition from the PIM level to the PSM level. In this transformation procedure, we create for each SBVR annotation a RULE activity in a first step, where the RULE activity then contains one SBVR rule. In a second step, single RULE activities for a business activity are concatenated with gateways, and the resulting chain is integrated in the control flow before the corresponding business activity takes place. In this way, a regular BPMN process with nested-conditional rule chains is the main concept for the PSM level. A further transformation, e.g., into BPEL is not necessary due to the included execution property of BPMN 2.0 processes.

Fundamentally, the platform-specific model has to contain all essential platform details that are necessary for execution. Therefore, the transformation of annotations into gateway chains is only the first step. In a second step, OCL constraints have to be derived from the SBVR rules. The third and last step consists of the manual mapping of domain-specific concepts to Web-service data structures, where Web services are the core execution method for business activities. These two steps are described in detail in the following subsections.

4.1 From SBVR to a Domain-Specific Language

SBVR allows four types of expressions, as shown in the right part of Figure 4, to create the business vocabulary and rules. To parse the business model given in SBVR, we have to identify the four types of expressions, i.e., term, name, verbs, and keywords, as explained in [8, 16]. However, such parsing should be as easy as possible. If we look at the current market for SBVR tools, we can observe that some tools offer interchange formats or direct access to their repositories. Using these possibilities, the structured read-out can be simplified. For example, RuleXpress is such a tool. Designed by RuleArts, it enables business analysts to capture their vocabulary or terms and source business rules. In this case, we find a rule- and vocabulary-centric approach offering easy analyses. In contrast to classical SBVR management tools, it is possible to express the fact model as a UML class diagram with instance associations. In general, RuleXpress is a repository-based tool that can be used offline or in a multi-user environment. Therefore, models are stored in a central repository enabling the check-out to a local copy and subsequent merging-back.

As demonstrated in [4], the transformation from SBVR structural facts is possible. In our case, we transform the SBVR/RuleXpress terms and facts into an Eclipse Modeling Framework (EMF) model[2]. EMF is based on ECore, an MOF-like meta-model. The source of the rules is the RuleXpress repository, which is a relational database. It stores all terms, facts as well as the associations of the fact model. The advantage of the database approach is that we do not have to parse a text file; we can use typed database entries. As a first step, a straightforward transformation tool produces an EMF-based meta-model with all terms and the associations or categorization (structural rules) from the RuleXpress database. This meta-model can be seen as a domain-specific language for the use case of the SBVR model.

Due to the circumstance that we want to transform SBVR rules into OCL constraints in business processes, we have to handle all essential data types. SBVR itself does not distinguish between ordinary data types and classes. It only contains some predefined elementary concepts---integer (positive integer) and text. If we look at business processes that mostly use Web services, we have to support XSD types (string, decimal, integer, float, Boolean, date, time), which are comparable to the UML data types. Therefore, we translate all these types into SBVR object types. These types have to be added to the original SBVR model.

4.2 Transforming Behavioral Rules into OCL

Using the previous step, we are able to convert the text-based vocabulary and the structural rules into an EMF model. This model now allows us to integrate the vocabulary and the structural rules into an EMF-based BPMN model. But the behavioral rules are still missing, so we are unable to express constraints for the business process. To tackle this issue, we need to dive deeper into the context of SVBR fact types and their transformation into OCL.

Fact	Mapping
necessity claim	- -necessity context type inv <F>
obligation claim	- -obligation context type inv <F>
possibility claim	not possible
permissibility claim	not possible

Figure 7. Facts and Their OCL Pendants.

Fundamentally, four types of facts can be expressed in SBVR. As shown in Figure 7, the fact types *Necessitation* and O*bligation* can be mapped to OCL invariants, while the fact types *Possibility* and *Permission* are not derivable to invariants. An OCL invariant is a constraint that should always be true during the whole life-time of an object. Therefore, necessitation and obligation can only be represented as OCL constraints. Possibilities and permission define restrictions making a positive statement on a situation that may occur in the business world. Such a transformation into OCL invariants is not the focus of this paper, since the purpose of invariants is normally different.

To map *Necessitation* and O*bligation* fact types onto our EMF-based representation, one possibility is to develop a straightforward parser. The shortcomings of this procedure are that the parser has to be adapted every time an SVBR rule changes and that it cannot be efficiently reused in other projects, not even if they are very close to the original one. A more generic solution is EMFText [15], which is a tool originally designed to define text syntax for languages described by an EMF meta-model and to describe domain-specific languages. In our scenario, we use EMFText to parse behavioral rules to our EMF model.

The whole EMFText workflow for our transformation is illustrated in Figure 7. As input for EMFText, an EMF meta-model and a conceptual schema (CS) are needed. The conceptual schema includes the syntax for model elements being defined using concepts from the Extended Backus Naur Form (EBNF). As basis for CS, we use standard rule structures and a simplified vocabulary containing only the base set of constraint definitions. The EMF meta-model corresponds to the model that we also assume for SBVR-annotated BPMN processes. Then, EMF produces a complete grammar and parser. We now use the produced grammar to parse the behavioral rules to an EMF model. This includes the vocabulary---which can be easily parsed---and associations to the logical operators. The logical operators like, e.g., *exactly one*, are also represented as EMF classes (e.g., *exactly*) with parameterized attributes (e.g., *one*). The determined model now contains all information from the

[2] http://www.eclipse.org/modeling/emf/

SBVR repository. As a last step of the behavioral rule parsing, a straightforward generator is employed to produce OCL from the EMF model by translating the logical associations into OCL constraints. For example, the SBVR/RuleXpress rule "*A cow must have exactly one herd,*" is transformed into the following OCL invariant constraint:

"*Context Cow inv: self.herd->size() = 1*".

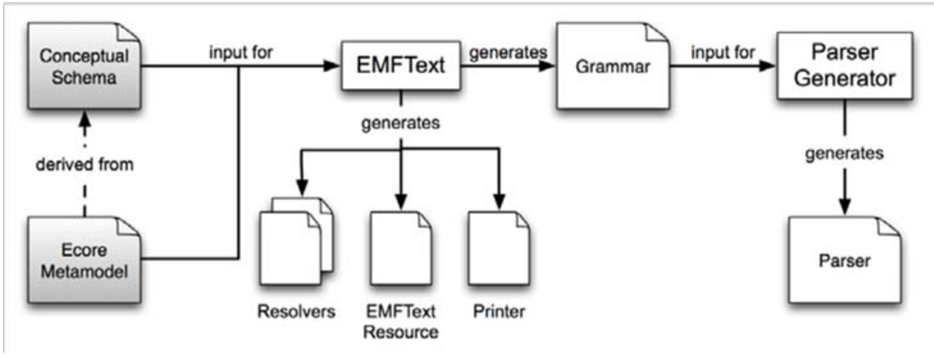

Figure 8. Rule Transformation using EMFText Workflow.

The last challenging issue in this area is that an OCL invariant does not contain any information on what should be done if the constraint is not satisfied. That means appropriate consequential activities are not expressible in OCL, whereas SBVR allows this specification by using enforcement levels. An enforcement level defines an implication activity for a necessitation, usually in an informal way. To consider this issue, we map SBVR enforcement levels to BPMN compensation activities. In this case, we transform SBVR rules into a combination of OCL invariants and BPMN compensation activities.

5 Implementation

Our complete proposed concept has been implemented to realize business processes of our background use case scenario "precision dairy farming". On the one hand, the *Eclipse BPMN modeling tool*[3] has been adjusted to enable our presented joined model concept, whereas we utilize RuleXpress as a business rule tool and repository. Therefore, rule annotations have to be conducted in RuleXpress notation. The described transformation from PIM to PSM is implemented as a separate JAVA program. Then, the resulting BPMN process definition includes nested-conditional OCL rule chains.

As process execution engine for our prototype, we used the *Open Service Process Platform* (short OSPP) [7]. This engine has been developed at TU Dresden as an OSGi[4] application and has two major advantages. OSPP is designed for extensibility and to support different workflow languages. Functionality and business logic can be extended and changed by registering new plug-ins to the OSPP server. The OSGi framework is responsible for the runtime and lifecycle of each plug-in. One central concept is the extension points. These points are slots in a bundle where other bundles can hook up to extend the original implementation. We used this concept to integrate the Dresden OCL toolkit interpreter into the OSPP server and also some extensions to the execution logic to continuously validate the OCL constraints. Additionally, OSPP is based on workflow nets and provides different adapters for various languages. Currently, BPEL, jPDL and BPMN are supported.

6 Related Work

The integration of SBVR rules into an MDA process has been discussed in several papers. In [1, 2], Linehan shows a complete integration of SBVR Structured English into the Model-Driven Architecture stack. The SBVR rules are integrated into the PIM and subsequently refined to the PSM. The code conversion is limited to a well defined subset of SBVR and supports transformations to OCL or Java. In these papers, Linehan does not explain how the OCL generation is realized. Another approach to integrate SBVR in MDA is the business rule templates approach (BRT) [3]. In this specially designed language, developers can define business rules templates that can be used in generative approaches like MDA. Developers using BR templates have to define domain-specific templates for their business rules. Afterwards, these templates can be transformed/refined to the platform-specific model. The concepts of BRT have been implemented in the BRidge IT project.

Another approach [4] focuses on the transformation of SBVR specifications to different kinds of UML diagrams. This includes, for example, UML class diagrams and state chart diagrams. Raj et al. [4] show the straightforward transformation rules and thus also present SBVR-to-model transformations, like we do in our

[3] http://www.eclipse.org/bpmn/
[4] http://www.osgi.org

approach from SBVR to EMF. The transformation or integration of OCL constraints into the UML models have not been considered. Also, the transformation from Structured English to OCL has been investigated, like in [5]. It allows parsing a CS specification into a UML class diagram. For this transformation, only the controlled English is necessary, in contrast to the structured repository of RuleXpress used in our approach. A completely different approach is proposed in [6]. They present a procedure for the reverse transformation from a UML/OCL description to SBVR and Structured English.

The integration of business rules in business process has been investigated, e.g., in [9, 10]. Charfi et al. [10] presented a hybrid Web-service composition approach, where their integration is based on Aspects and AO4BPEL. In contrast to our work, they focused more on the technical level with BPEL and proprietary rule definitions. Graml et al. [10] proposed a pragmatic approach for the integration, but they focused only on the modeling perspective and presented several modeling patterns.

7 Summary

In this paper, we presented an integrated-join approach for the modeling and execution of rule-intensive business processes. On the modeling side, we proposed an SBVR-annotated approach for a combined modeling of processes and rules in a business-aware way. On the execution side, we introduced our developed transformation strategy to derive regular BPMN process definitions with OCL constraints out of an SBVR-annotated process. Furthermore, we described our prototypical realization concept as well as our driving background use-case scenario of the whole work. From our point of view, this is the first work that covers all aspects of joining business rules and business processes from the modeling and the execution layer.

References

[1] **Linehan, M. H.**: *Semantics in Model-Driven Business Design*. In: Proceedings of 2nd International Semantic Web Policy Workshop (SWPW'06, Nov. 5-9, 2006, Athens, GA, USA), 2006.

[2] **Linehan, M. H.:** *Ontologies and Rules in Business Models*. In: Proceedings of the 2007 Eleventh International IEEE EDOC Conference Workshop (EDOCW '07, Oct. 15-16), pages 149-156, 2007.

[3] **Sosunovas, S. and Vasilecas, O.:** *Precise notation for business rules templates*. In: Proceedings of the 7th International Baltic Conference on Databases and Information Systems (DB&IS'06, Vlinius, Lettland), pages 55-60, 2006.

[4] **Raj, A.; Prabhakar, T. V. and Hendryx, S.:** *Transformation of SBVR business design to UML models*. In: Proceedings of the 1st conference on India software engineering conference (ISEC'08, Hyderabad, India, Feb 19-22), pages 29-38, 2008.

[5] **Kleiner, M.; Albert, P. and Bézivin, J.**: *Parsing SBVR-based controlled languages*. In: Proceedings of the 12th International Conference on Model Driven Engineering Languages and Systems (Models'O9, Denver, Colorado, USA .Oct. 4-9), 2009.

[6] **Cabot, J.; Pau, R. and Raventos, R.**: *From UML/OCL to SBVR specifications: A challenging transformation*. In: Information Systems, 2009.

[7] **Habich, D.; Richly, S.; Ruempel, A.; Buecke, W. and Preissler, S.**: *Open Service Process Platform 2.0*. In: Proceedings of the 2008 IEEE Congress on Services (SERVICES I 2008, Honolulu, Hawaii, USA, July 6-11), pages 152-159, 2009

[8] **Linehan, M. H.**: *SBVR Use Cases*. In: Proceedings of 2008 International Symposium on Rule Representation, Interchange and Reasoning on the Web (RuleM'08, Orlando, FL, USA, Oct. 30-31), pages 182-196, 2008.

[9] **Graml, T.; Bracht, R. and Spies, M.**:*Patterns of Business Rules to Enable Agile Business Processes*. In: Proceedings of 11th IEEE International Enterprise Distributed Object Computing Conference (EDOC'07, Oct. 15-19, Annapolis, Maryland, USA), pages 365-378, 2007.

[10] **Charfi, A. and Mezini, M.**: *Hybrid web service composition: business processes meet business rules*. In: Proceedings of 2nd International Conference on Service Computing (ICSOC'04, New York, NY, USA, Nov. 15-19), pages 30-38, 2004.

[11] **Specification of BPEL:** *http://www.ibm.com/developerworks/library/specification/ws-bpel/*

[12] **Specification of BPMN 2.0 (Beta 1):** *http://www.omg.org/cgi-bin/doc?dtc/09-08-14*

[13] **The Business Rule Group**: *Defining Business Rules, What are they really?, http://www.businessrulesgroup.org/, 2010*

[14] **Schulze, C.; Spilke, J. and Lehner, W.**: *Data modeling for Precision Dairy Farming within the competitive field of operational and analytical tasks*. In: Computers and Electronics in Agriculture, Volume 59, Issue 1-2 (Nov. 2007), pages 39-55, 2007

[15] **Heidenreich, F.;Johannes, J.; Karol, S.; Seifert, M. and Wende C.**: *Derivation and Refinement of Textual Syntax for Models*. In Proceedings of the 5th European Conference on Model-Driven Architecture Foundations and Applications (ECMDA-FA 2009), pages 114-129, 2009.

[16] **Semantics of business vocabulary and rules specification**: http://www.omg.org/spec/SBVR/1.0/, Object Management Group, January 2008.

[17] **Dresden OCL Toolkit:** http://dresden-ocl.sourceforge.net/