Proceedings of the 2013 Federated Conference on
Computer Science and Information Systems pp. 939–946

Towards Rule-oriented Business Process Model Generation

Krzysztof Kluza and Grzegorz J. Nalepa
AGH University of Science and Technology
al. A. Mickiewicza 30, 30-059 Krakow, Poland
E-mail: {kluza,gjn}@agh.edu.pl

***Abstract*—Attribute-Relationship Diagrams (ARD) aim at capturing relations, especially dependency relation, between the specified attributes. This paper describes work-in-progress research concerning process and rules integration, which takes advantage of the ARD method and allows for generating executable models. The paper examines the possibility of generating the rule-oriented BPMN model and enriching process models with rules from the ARD diagram.**

***Index Terms*—BPMN, Business Processes, Business Rules**

I. Introduction

BUSINESS Process (BP) models constitute a graphical representation of processes in an organization. Such a process is composed of related tasks that produce a specific service or product for a particular customer [1]. When it comes to practical modeling, Business Process Model and Notation (BPMN) [2], [3] constitutes a standard for this purpose.

The current version of the BPMN notation allows for modeling many aspects of business; nonetheless, it is not suitable for modeling some aspects of the enterprise, especially decision rules or constraints [4]. Recently, the Business Rules (BR) approach has been proposed as a new way of capturing the functional requirements and modeling system logic in a designer-friendly fashion. Moreover, the BR approach is a solution originated from Rule-Based Systems, that are a mature and well established technology. As processes and rules are related to each other, BR are often, but not limited to, used for the specification of task logic in process models.

BR can be acquired based on some data using machine learning techniques [5] or generated from natural language specification [6]; however, they often have to be modeled manually based on the knowledge collected from the domain experts, as usually their knowledge is not written down anywhere. Similarly, processes are designed manually as well. However, the simplified process models can be generated using process mining tools [7] or acquired from natural language description using NLP techniques [8]. Other method of acquiring BPMN models is to transform the existing process models in other languages to the BPMN notation. From a researcher's point of view, this can be a challenge in the case that languages are of different paradigm or presents a different aspect of the system, e.g. UML use-case diagrams [9].

The paper is supported by the HiBuProBuRul Project funded from NCN (National Science Centre) resources for science (no. DEC-2011/03/N/ST6/00909).

In this paper, we present work-in-progress research which is a part of our research concerning process and rules integration [10], [11], [12]. We examine the possibility of generation of the rule-oriented BPMN model as well as the possibility of enriching BP processes with rules from the ARD diagram.

As the ARD method allows a domain expert for gradual identification of the properties of a system being designed, we argue that having the system properties identified and described in terms of attributes, it is possible to generate executable BPMN model with the corresponding BR tasks as well as enrich the BP model with such BR tasks. Such an approach would allow for generating business processes and rule schemas for logic task specification at the same time. The generated rule prototypes comply with the XTT rule representation [13], [14], [15], [16] from the Semantic Knowledge Engineering approach [17].

The paper is organized as follows. In Section II we present the motivation for our research. Section III provides a short overview of the related approaches. Section IV presents the details of the ARD method. In Section V, we give an overview of the proposed method for process model generation and enriching the BP model with BR tasks based on a design example. The hybrid execution environment is presented in Section VI. Section VII summarizes the paper.

II. Motivation

The complexity of software has been constantly increasing for the last decades. To deal with this growth, new design methods and advanced modeling solutions are required [18]. For this purpose, modern applications use business processes and business rules as business logic specification [19].

According to the BPMN 2.0 specification [2], the notation is not suitable for modeling such concepts as rules. Therefore, this reveals the challenges in modeling and executing processes with rules. It is so because processes and rules are developed or modeled separately and are not well matched.

Our research aims at developing the integrated method for modeling BP with BR to provide a consistent method for modeling business systems. Such a method will allow for modeling processes with rules in a straightforward way, and then for executing such a developed model.

The main contribution of this paper is a presentation of the possibility of generation of the rule-oriented BPMN model, which can be used for enriching the BP models with BR tasks.

978-1-4673-4471-5/$25.00

III. Related work

Several approaches can be considered as related to the method presented in this paper. As our method proposes automatic generation of a BPMN model, the approach can be compared with such approaches as: process mining [7], generating processes from text in natural language (based on NLP methods) [8], or finally transforming process from other notations to BPMN, especially from the notations that are not process-oriented, e.g. the UML use case diagrams [20].

The process mining methods [7] allow for very flexible process models generation, and in some cases this technique does not require any human activity. However, the result of the method is a very general process that is not suitable for direct execution. In order to be an executable BPMN model, it has to be significantly enhanced and refactored. In the case of our method, it is not as much flexible as process mining technique, but it produces a BPMN model which is executable and provides support for Business Rule tasks.

Generating processes from text description provided in natural language [8] can have practical results and allows for generating a high quality BPMN model. High quality models can also be obtained through translation from other representations, such as the UML use case diagrams [21], [22]. Unfortunately, a method based on the natural language description has to be supported by an advanced NLP system, thus practical applications of this method is very complex. Translation from other representations, in turn, requires process models designed using such representations, which often do not exist. In our approach, a process model is generated based on the carefully prepared ARD diagram. Although this requires the ARD diagram, it is very simple model and in some cases it can be obtained from text description using some mining technique to acquire attributes. This requires additional research yet. However, there has been trials of mining such attributes from text in natural language [23].

There are also other approaches, such as generating business process models from Bill Of Materials (BOM) [24], Product Based Workflow Design (PBWD) [25], [26], based on Product Data Model (PDM), or Decision Dependency Design (D3) [27], [28]. These are more similar to the method proposed in this paper. However, in the case of our approach, apart from the fact that it generates an executable BPMN model, it supports the rule prototypes generation for Business Rule tasks, what makes the BPMN model data consistent with the rule engine requirements for data. Therefore, we claim that our approach is partially rule-based [29].

It is important to mention that the presented approach can be used either for generating a new rule-oriented BPMN model or for enriching the existing process model with rules based on the corresponding ARD diagram.

Although the work presented in this paper is work-in-progress research, the method overview presented in the paper reveals significant differences from the techniques mentioned above, especially in the case of method simplicity and support for rules in process models.

IV. Attribute Relationship Diagrams

Attribute Relationship Diagram (ARD) [30] constitute a method which allows a user (especially a domain expert) for gradual identification of the system properties during design.

The goal of this method is to capture functional dependencies between attributes. The attributes are expressed in terms of Attributive Logic [31], [17], [32] and denote particular system properties identified by the domain expert. The identified dependencies form a directed graph in which properties are represented as nodes and dependencies are represented as transitions. In the following definitions, we present more formal description of ARD.

A typical atomic formula (or fact) takes the following form

$$a(p) = d$$

where a is an attribute, p is a property and d is the current value of a for p. More complex descriptions take usually the form of conjunctions of such atoms and are omnipresent in the AI literature [33].

An **attribute** $a_i \in A$ is a function (or partial function) of the form:

$$a_i \colon P \to \mathbb{D}_i$$

where

- P is a set of property symbols,
- A is a set of attribute names,
- $\mathbb{D}$ is a set of attribute values (the domains).

An example of an attribute can be the `carAge`, which denotes the age of a car, and the attribute value is within the domain $\mathbb{D}_{carAge} = [0, \inf]$.

A **generalized attribute** $a_j \in A$ is a function (or partial function) of the form:

$$a_j \colon P \to 2^{\mathbb{D}_j}$$

where $2^{\mathbb{D}_j}$ is the family of all the subsets of $\mathbb{D}_{\mathbb{j}}$.

An example of a generalized attribute can be the `ownedInsurances`, which is a set of the customer insurances, and the attribute value is a subset of the domain $\mathbb{D}_{ownedInsurances}$, which consists of the possible insurances that a particular customer can posses.

In the case of abstraction level, the ARD attributes and generalized attributes can be described either as conceptual or physical ones.

A **conceptual attribute** $c \in C$ is an attribute describing some general, abstract aspect of the system.
Conceptual attribute names are capitalized, e.g.: `BaseRate`. During the design process, conceptual attributes are being *finalized* into, possibly multiple, physical attributes.

A **physical attribute** $a \in A$ is an attribute describing a specific well-defined, atomic aspect of the system.
Names of physical attributes are not capitalized, e.g. `payment`. A physical attribute origins from one or more (indirectly) conceptual attributes and can not be further *finalized*.

A **simple property** $p_s \in P$ is a property described by a single attribute.

A **complex property** $p_c \in P$ is a property described by multiple attributes.

A **dependency** $d \in D$ is an ordered pair of properties (f, t), where $f \in P$ is the **independent property** and $t \in P$ is the **dependent property** that depends on f. For simplicity $d = (f, t) \in D$ will be presented as: $dep(f, t)$.

An **ARD diagram** R is a pair (P, D), where P is a set of properties, and D is a set of dependencies, and between two properties only a single dependency is allowed.

To illustrate the ARD concepts, an exemplary ARD diagram with properties and the dependency between them is presented in Figure 1. The diagram should be interpreted in the following way: `payment` depends somehow on `carCapacity` and `baseCharge`.

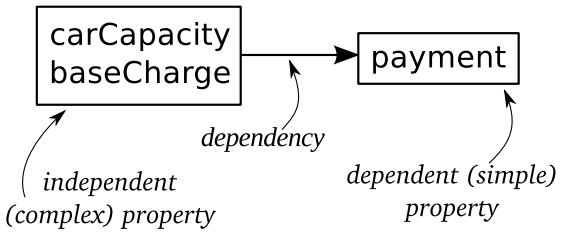

Figure 1. An example of the ARD diagram

The core aspects of the ARD method are diagram transformations, which regard properties and serve as a tool for diagram specification and development. Transformations are required to specify additional dependencies or introduce new attributes for the system. For the transformation of the diagram R_1 into the diagram R_2, the R_2 is more specific than the R_1.

Finalization *final* is a function of the form:

$$final : p_1 \rightarrow p_2$$

that transforms a simple property $p_1 \in P$ described by a conceptual attribute into a property $p_2 \in P$, where the attribute describing p_1 is substituted by one or more conceptual or physical attributes describing p_2, which are more detailed than the attribute describing a property p_1.

In Figure 2, an exemplary finalization transformation is presented. It shows that the simple property `BaseRate` (described by a single conceptual attribute) is finalized into a new complex property described by two physical attributes `carCapacity` and `baseCharge`.

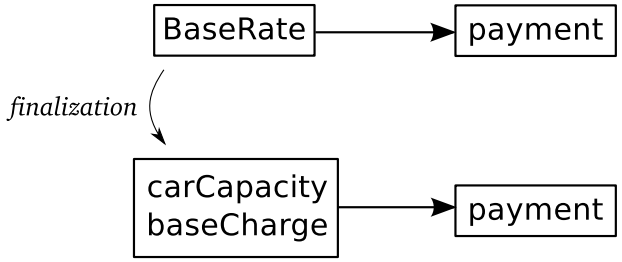

Figure 2. An example of the ARD finalization transformation

Split *split* is a function of the form:

$$split : p_c \rightarrow \{p^1, p^2, \ldots, p^n\}$$

where a complex property p_c is replaced by n properties, each of them described by one or more attributes originally describing p_c. Since p_c may depend on some other properties $p_o^1 \ldots p_o^n$, dependencies between these properties and $p^1 \ldots p^n$ have to be stated.

To illustrate this transformation, Figure 3 shows the complex property described by two physical attributes (`carCapacity` and `baseCharge`), which is split into two simple properties described by these attributes.

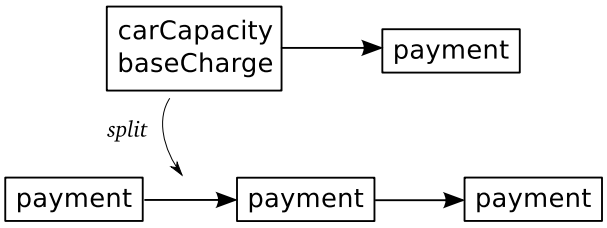

Figure 3. An example of the ARD split transformation

Upon splitting and finalization, the ARD model is more and more specific (see Figure 4). The consecutive levels of ARD forms a hierarchy of progressively detailed diagrams, which constitutes Transformation Process History (TPH) [34]. The implementation of this hierarchical model is provided through storing the lowest available, most detailed diagram level at any time, and additional information needed to recreate all of the higher levels. Such model captures information about changes made to properties at consecutive diagram levels.

A. Polish Liability Insurance Case Study

Let us now present an illustrative example of the Polish Liability Insurance (PLLI) case study. The example was developed as a benchmark case for the Semantic Knowledge Engineering (SKE) approach for rule-based systems [17]. Based on this simple case study, we will then present how ARD can be used for BPMN model generation.

In the PLLI case study, the price for the liability insurance for protecting against third party claims is to be calculated.

The price is calculated based on various reasons, which can be obtained from the domain expert. The main factors in calculating the liability insurance premium are data about the vehicle, such as the car engine capacity, the car age, etc. Additionally, the impact on the insurance price have such data as the driver's age, the period of holding the license, the number of accidents in the last year, etc. Moreover, in the calculation, the insurance premium can be increased or decreased because of number of payment installments, other insurances, continuity of insurance or the number of cars insured. All these pieces of data can be specified using the ARD method and presented using the ARD diagram (see Figure 5). As specification of ARD is an iterative process, the corresponding TPH diagram, presenting split and finalization transformations, can be easily depicted, as shown in Figure 6.

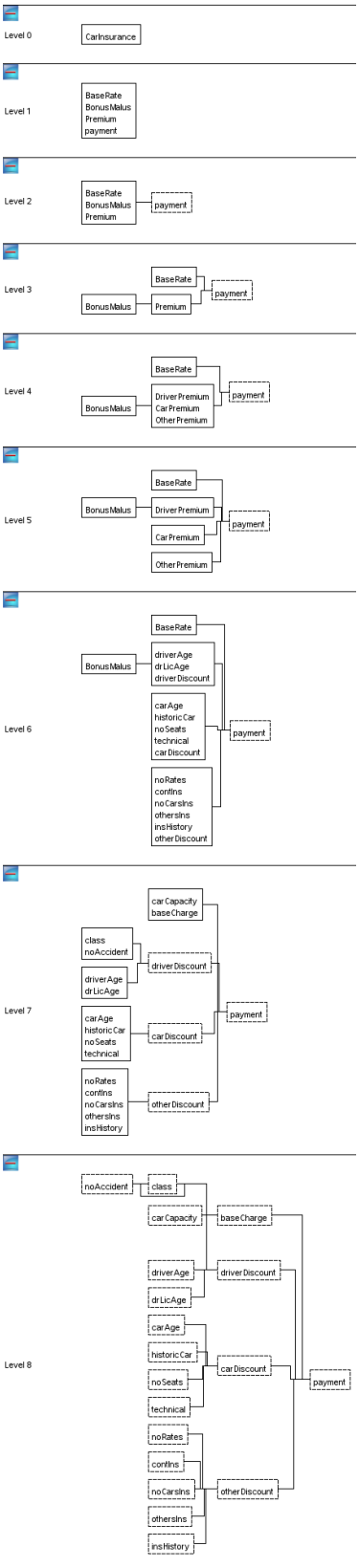

Figure 4. The ARD levels for the PLLI case study

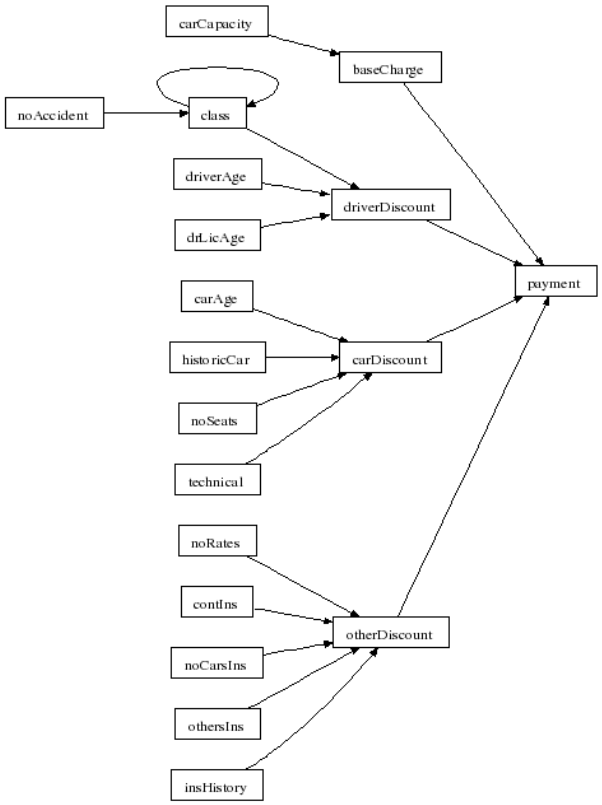

Figure 5. A complete ARD model for the PPLI example

B. Advantages of ARD

There are several advantages of using the ARD method to specify the system. Firstly, this method describes the system in an attribute-oriented way, and thus it is easy to comprehend and generates a simple model.

ARD can be used even if there are not many pieces of information available. It is so because this method does not require anything apart from the specification of dependencies between attributes. It is important to mention that we do not specify the detail semantics of the dependency relationship; thus it is only claimed that one property depends on other property. Although this limitation of ARD can be seen as a drawback, the main focus of this method is on simplicity.

The ARD method can also be extended, e.g. there can be used some mining technique to acquire attributes and dependencies among them. However, this requires additional research tasks yet. There has been trials of mining such attributes from text in natural language [23].

Applying ARD as a design process allows a domain expert to identify attributes of the modeled system and refine them gradually, as well as generates rule prototypes based on the identified attributes. Thanks to storing the history of transformations, it is possible to refactor such a system [35].

In the following section, we give a short overview of the proposed method of process model generation and enriching the BP model with BR tasks based on a design example.

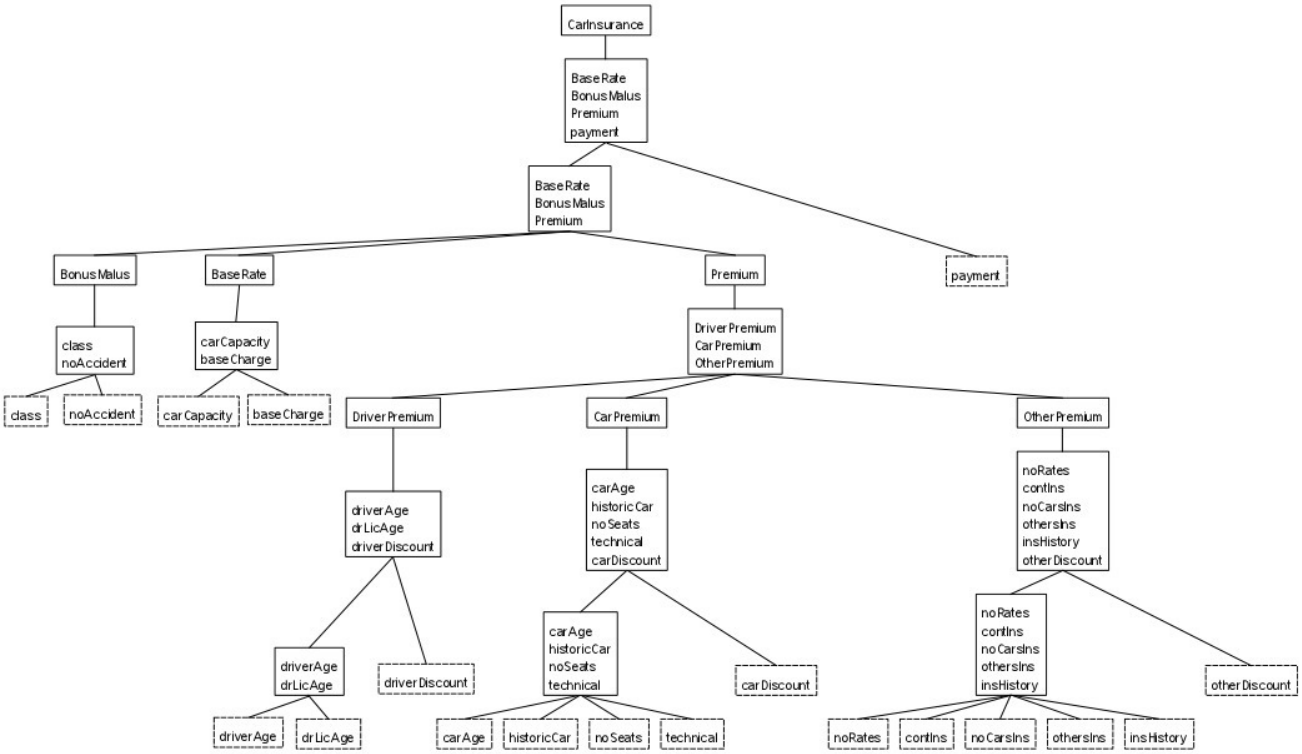

Figure 6. An example of the TPH diagram, corresponding to the ARD diagram presented in Figure 5

V. RULE-ORIENTED BPMN MODEL GENERATION

In the proposed rule-oriented approach for BPMN model generation, we consider:

1) **Generating the whole BPMN model based on ARD**, presented in the previous section. Such an approach requires specific input and generates particular output:

   Input: Attribute-Relationship Diagram (ARD), and additionally Transformation Process History (TPH).
   The input is the most detailed ARD+ diagram, that has all of the physical attributes identified (in fact, this can also be applied to higher level diagrams, generating rules for some parts of the system being designed).

   Output: A rule-oriented BPMN process model.
   The output is a process model in the BPMN notation with User tasks, BR tasks and additional elements of the control flow objects. BR tasks contain rule prototypes in a very general format:

   ```
   rule:
   condition attributes | decision attributes
   ```

   Goal: The goal of this approach is to automatically build a rule-oriented BPMN process model on the basis of the ARD diagram (optionally supported by the TPH diagram). The algorithm will generate both User Tasks with form attributes for entering particular pieces of information and Business Rule Tasks with prototypes of decision tables.

   Sketch of the algorithm:

   1. Generate BR tasks from ARD based on the modified version of the algorithm for generating the XTT2 representation from ARD (detailed description of this part is presented below).
   2. Generate proper User tasks which acquire necessary information from the user.
   3. Generate proper User/Mail tasks to communicate process results to the user.
   4. Complete the diagram using control flow with additional flow objects, such as start and end events, and gateways.

2) **Enriching the existing BPMN model with BR tasks based on ARD** (either developed parallely to BP model or generated based on the process description).

   Input: BPMN process model, Attribute-Relationship Diagram (ARD) corresponding to the BPMN model, and additionally Transformation Process History (TPH).

   Output: A rule-oriented BPMN process model.

   Goal: The goal of this approach is to automatically enrich a BPMN process model with rule tasks on the basis of the ARD diagram (optionally supported by the TPH diagram). The algorithm will support refactoring of the process model to rule-oriented way by proposing new BR tasks for the process model.

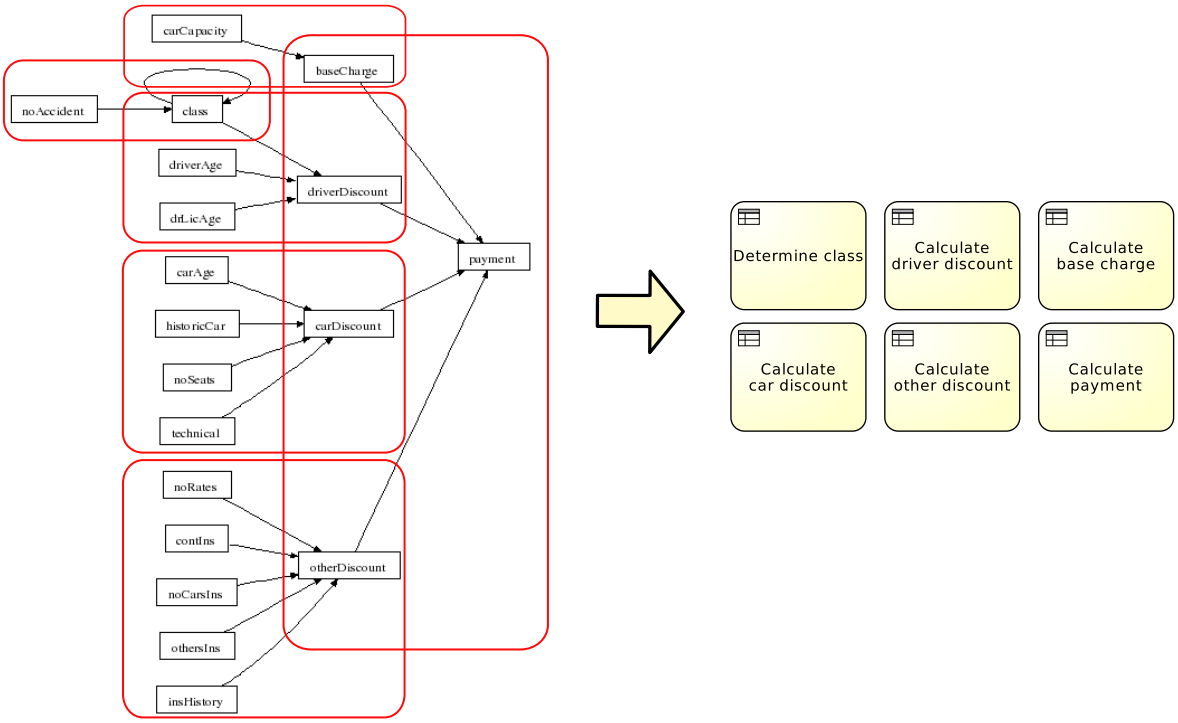

Figure 7. BR tasks generated from the ARD diagram

In the aforementioned cases, the most important aspect is to generate Business Rule tasks with rule prototypes for a process model. This can be done using the modified version of the algorithm for generating the XTT2 representation from ARD (described in [36], [34]). The result of application of this algorithm is presented in Figure 7. A draft of the algorithm for generating Business Rule tasks with rule prototypes for a process model is as follows:

1) Prepare data:
   a) Choose a dependency $d \in D : dep(f,t), f \neq t$, where D is a set of dependencies in the ARD diagram.
   b) Select all independent properties (other than f) that t depends on. Let $F_t = \{f_t^i : dep(f_t^i, t), f_t^i \neq f\}$. Remove the considered dependencies from the set: $D := D \setminus \{d_{f_t^i,t}\}$.
   c) Select all dependent properties (other than t) that depend only on f. Let $T_f = \{t_f^i : dep(f, t_f^i), t_f^i \neq t, \nexists f_x : (dep(f_x, t_f^i), f_x \neq f)\}$. Remove the considered dependencies from the set: $D := D \setminus \{d_{f,t_f^i}\}$.
2) Create BR tasks based on F_t and T_f:
   a) if $F_t = \emptyset, T_f = \emptyset$, create a BR task determining the value of the t attribute and associate the task with the following decision table schema: $f \mid t$.
   b) if $F_t \neq \emptyset, T_f = \emptyset$, create a BR task determining the value of the t attribute and associate it with the following decision table schema: f, f_t^1, f_t^2, ... $\mid t$.
   c) if $F_t = \emptyset, T_f \neq \emptyset$, create a BR task determining the value of the $T_f \cup \{t\}$) attributes and associate it with the decision table schema: $f \mid t$, t_f^1, t_f^2, ...
   d) if $F_t \neq \emptyset, T_f \neq \emptyset$, create two BR tasks determining the value of the T_f and t attributes and associate them with the following decision table schemas respectively: f, f_t^1, f_t^2, ... $\mid t$ and $f \mid t_f^1$, t_f^2, ...
3) Go to step 1 if there are any dependencies left ($D \neq \emptyset$).

The result of application of the BR task generation for the Polish Liability Insurance case is presented in Figure 7. Next, User tasks which acquire necessary information from the user and User/Mail tasks to communicate process results to the user have to be generated (see Figure 8 and 9).

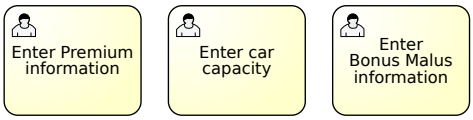

Figure 8. User tasks generated from the ARD diagram

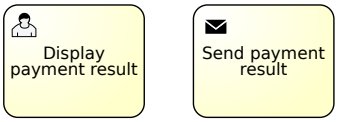

Figure 9. User/Mail tasks generated from the ARD diagram

Finally, the model have to be completed using control flow with additional flow objects, such as start and end events, and gateways. The resulting diagram can be observed in the Activiti-based environment presented in Figure 10.

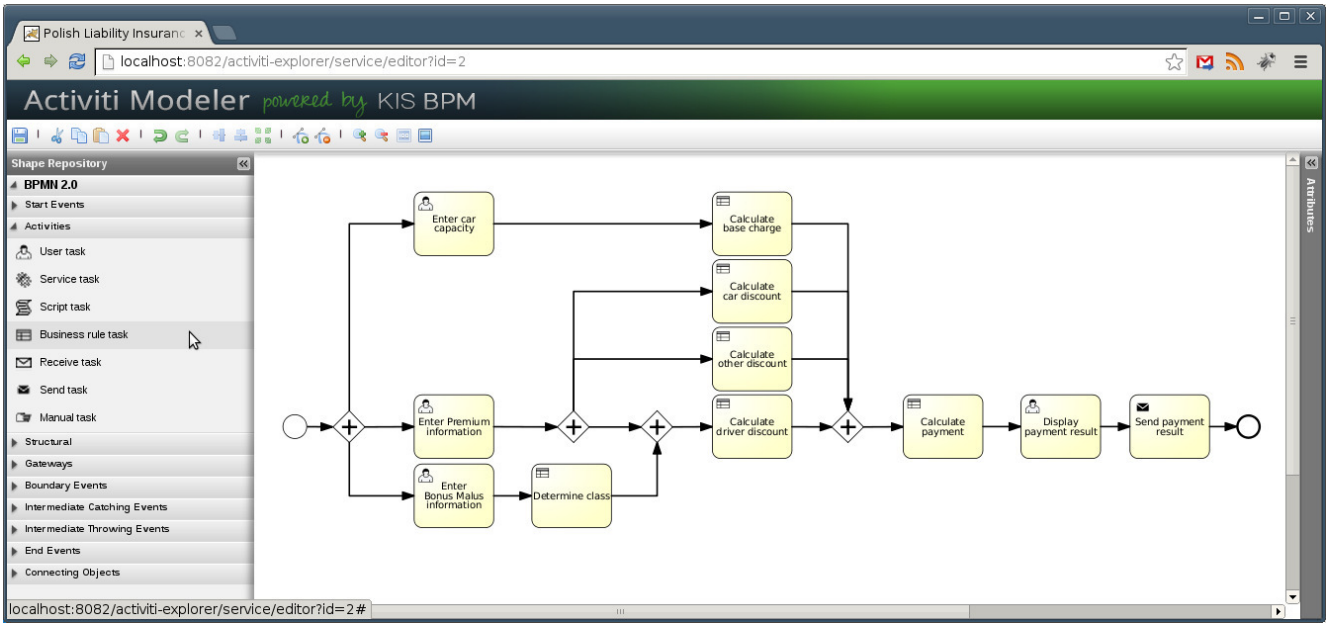

Figure 10. A prototype Activiti-based environment for modeling and executing processes with rules

VI. Hybrid Execution Environment

As a BPMN model generated from ARD constitutes an executable specification of a process, it can be executed in the process runtime environment. However, for complete execution of the model, i.e. execution of the Business Rule task logic, a process engine, such as jBPM [37] or Activiti [38], has to delegate rule execution to the business rule engine. As decision table schemas are generated automatically, the created decision tables have to be complemented with rules. Decision table can be filled in with rules using a dedicated editor [13] or a dedicated plugin for the process modeler [11]. Then, our prototype hybrid execution environment [12], [39], can serve as a specific execution example for this approach.

VII. Concluding Remarks

The aim of this paper is to examine the possibility of generating the rule-oriented BPMN model and enriching process models with rules based on the ARD diagram. We give an overview of the method for process model generation and present a first draft of the algorithm for automatic generation of rule-oriented BPMN process models from Attribute Relationship Diagram. In the algorithm, BR tasks with corresponding decision table schemas are generated and the resulting model can be executed in the hybrid execution environment.

The presented approach can be used either to generate the whole BPMN model based on the existing ARD diagram or to enrich the existing BPMN model with BR tasks based on ARD developed parallely to BP model or generated based on the process description. As the generated rule schemas are complementary to the process model, the solution addresses the two mentioned challenges: separation between processes and rules in the modeling phase and the problem of the execution of such separated data, which usually requires some additional integration or configuration in the execution environment.

As this paper presents a work-in-progress research, our future work will consist in refining and formalizing the presented approach. Then, we plan to extend the approach with new patterns and some optimization elements. We consider also enriching the ARD diagram with selected relations from the similar methods [24], [25], [27], [28], [26] and integrate the method with automatic verification features [40], [41].

References

[1] A. Lindsay, D. Dawns, and K. Lunn, "Business processes – attempts to find a definition," *Information and Software Technology*, vol. 45, no. 15, pp. 1015–1019, December 2003, elsevier.

[2] OMG, "Business Process Model and Notation (BPMN): Version 2.0 specification," Object Management Group, Tech. Rep. formal/2011-01-03, January 2011.

[3] T. Allweyer, *BPMN 2.0. Introduction to the Standard for Business Process Modeling*. Norderstedt: BoD, 2010.

[4] B. Silver, *BPMN Method and Style*. Cody-Cassidy Press, 2009.

[5] T. M. Mitchell, *Machine Learning*. MIT Press and The McGraw-Hill companies, Inc., 1997.

[6] I. S. Bajwa, M. G. Lee, and B. Bordbar, "SBVR Business Rules Generation from Natural Language Specification," in *AAAI Spring Symposium: AI for Business Agility*. AAAI, 2011. [Online]. Available: http://www.aaai.org/Library/Symposia/Spring/ss11-03.php

[7] W. M. P. van der Aalst, *Process Mining: Discovery, Conformance and Enhancement of Business Processes*, 1st ed. Springer Publishing Company, Incorporated, 2011.

[8] F. Friedrich, J. Mendling, and F. Puhlmann, "Process model generation from natural language text," in *Advanced Information Systems Engineering*, ser. Lecture Notes in Computer Science, H. Mouratidis and C. Rolland, Eds. Springer Berlin Heidelberg, 2011, vol. 6741, pp. 482–496.

[9] A. Sinha and A. Paradkar, "Use cases to process specifications in business process modeling notation," in *Web Services (ICWS), 2010 IEEE International Conference on*, 2010, pp. 473–480.

[10] G. J. Nalepa, K. Kluza, and S. Ernst, "Modeling and analysis of business processes with business rules," in *Business Process Modeling: Software Engineering, Analysis and Applications*, ser. Business Issues, Competition and Entrepreneurship, J. Beckmann, Ed. Nova Science Publishers, 2011, pp. 135–156.

[11] K. Kluza, K. Kaczor, and G. J. Nalepa, "Enriching business processes with rules using the Oryx BPMN editor," in *Artificial Intelligence and Soft Computing: 11th International Conference, ICAISC 2012: Zakopane, Poland, April 29–May 3, 2012*, ser. Lecture Notes in Artificial Intelligence, L. Rutkowski and [et al.], Eds., vol. 7268. Springer, 2012, pp. 573–581. [Online]. Available: http://www.springerlink.com/content/u654r0m56882np77/

[12] G. J. Nalepa, K. Kluza, and K. Kaczor, "Proposal of an inference engine architecture for business rules and processes," in *Artificial Intelligence and Soft Computing: 12th International Conference, ICAISC 2013: Zakopane, Poland, June 9–13, 2013*, ser. Lecture Notes in Artificial Intelligence, L. Rutkowski and [et al.], Eds., vol. 7895. Springer, 2013, pp. 453–464. [Online]. Available: http://www.springer.com/computer/ai/book/978-3-642-38609-1

[13] G. J. Nalepa, A. Ligęza, and K. Kaczor, "Formalization and modeling of rules using the XTT2 method," *International Journal on Artificial Intelligence Tools*, vol. 20, no. 6, pp. 1107–1125, 2011.

[14] A. Ligęza and G. J. Nalepa, "A study of methodological issues in design and development of rule-based systems: proposal of a new approach," *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, vol. 1, no. 2, pp. 117–137, 2011.

[15] M. Szpyrka, "Exclusion rule-based systems – case study," in *International Multiconference on Computer Science and Information Technology*, vol. 3, Wisła, Poland, October 20-22 2008, pp. 237–242.

[16] A. Ligęza and M. Szpyrka, "Reduction of tabular systems," in *Artificial Intelligence and Soft Computing - ICAISC 2004*, ser. Lecture Notes in Computer Science, L. Rutkowski, J. Siekmann, R. Tadeusiewicz, and L. Zadeh, Eds. Springer Berlin / Heidelberg, 2004, vol. 3070, pp. 903–908.

[17] G. J. Nalepa, *Semantic Knowledge Engineering. A Rule-Based Approach*. Kraków: Wydawnictwa AGH, 2011.

[18] G. J. Nalepa and K. Kluza, "UML representation for rule-based application models with XTT2-based business rules," *International Journal of Software Engineering and Knowledge Engineering (IJSEKE)*, vol. 22, no. 4, pp. 485–524, 2012. [Online]. Available: http://www.worldscientific.com/doi/abs/10.1142/S021819401250012X

[19] G. J. Nalepa, "Proposal of business process and rules modeling with the XTT method," in *Symbolic and numeric algorithms for scientific computing, 2007. SYNASC Ninth international symposium. September 26–29*, V. Negru and et al., Eds., IEEE Computer Society. Los Alamitos, California ; Washington ; Tokyo: IEEE, CPS Conference Publishing Service, september 2007, pp. 500–506.

[20] OMG, "Unified Modeling Language (OMG UML) version 2.2. superstructure," Object Management Group, Tech. Rep. formal/2009-02-02, February 2009.

[21] J. R. Nawrocki, T. Nedza, M. Ochodek, and L. Olek, "Describing business processes with use cases," in *BIS*, 2006, pp. 13–27.

[22] D. Lubke, K. Schneider, and M. Weidlich, "Visualizing use case sets as bpmn processes," in *Requirements Engineering Visualization, 2008. REV '08.*, 2008, pp. 21–25.

[23] M. Atzmueller and G. J. Nalepa, "A textual subgroup mining approach for rapid ARD+ model capture," in *FLAIRS-22: Proceedings of the twenty-second international Florida Artificial Intelligence Research Society conference: 19–21 May 2009, Sanibel Island, Florida, USA*, H. C. Lane and H. W. Guesgen, Eds., FLAIRS. Menlo Park, California: AAAI Press, 2009, pp. 414–415, to be published.

[24] W. van der Aalst, "On the automatic generation of workflow processes based on product structures," *Computers in Industry*, vol. 39, no. 2, pp. 97–111, 1999.

[25] I. Vanderfeesten, H. Reijers, and W. Aalst, "Case handling systems as product based workflow design support," in *Enterprise Information Systems*, ser. Lecture Notes in Business Information Processing, J. Filipe, J. Cordeiro, and J. Cardoso, Eds. Springer Berlin Heidelberg, 2009, vol. 12, pp. 187–198.

[26] I. Vanderfeesten, H. Reijers, W. Aalst, and J. Vogelaar, "Automatic support for product based workflow design: Generation of process models from a product data model," in *On the Move to Meaningful Internet Systems: OTM 2010 Workshops*, ser. Lecture Notes in Computer Science, R. Meersman, T. Dillon, and P. Herrero, Eds. Springer Berlin Heidelberg, 2010, vol. 6428, pp. 665–674.

[27] F. Wu, L. Priscilla, M. Gao, F. Caron, W. Roover, and J. Vanthienen, "Modeling decision structures and dependencies," in *On the Move to Meaningful Internet Systems: OTM 2012 Workshops*, ser. Lecture Notes in Computer Science, P. Herrero, H. Panetto, R. Meersman, and T. Dillon, Eds. Springer Berlin Heidelberg, 2012, vol. 7567, pp. 525–533.

[28] W. Roover and J. Vanthienen, "On the relation between decision structures, tables and processes," in *On the Move to Meaningful Internet Systems: OTM 2011 Workshops*, ser. Lecture Notes in Computer Science, R. Meersman, T. Dillon, and P. Herrero, Eds. Springer Berlin Heidelberg, 2011, vol. 7046, pp. 591–598.

[29] S. Goedertier and J. Vanthienen, "Rule-based business process modeling and execution," in *In: Proceedings of the IEEE EDOC Workshop on Vocabularies Ontologies and Rules for The Enterprise (VORTE 2005). CTIT Workshop Proceeding Series (ISSN 0929-0672*, 2005, pp. 67–74.

[30] G. J. Nalepa and I. Wojnicki, "Towards formalization of ARD+ conceptual design and refinement method," in *FLAIRS-21: Proceedings of the twenty-first international Florida Artificial Intelligence Research Society conference: 15–17 May 2008, Coconut Grove, Florida, USA*, D. C. Wilson and H. C. Lane, Eds. Menlo Park, California: AAAI Press, 2008, pp. 353–358, accepted.

[31] A. Ligęza, *Logical Foundations for Rule-Based Systems*. Berlin, Heidelberg: Springer-Verlag, 2006.

[32] A. Ligęza and G. J. Nalepa, "Knowledge representation with granular attributive logic for XTT-based expert systems," in *FLAIRS-20: Proceedings of the 20th International Florida Artificial Intelligence Research Society Conference: Key West, Florida, May 7-9, 2007*, D. C. Wilson, G. C. J. Sutcliffe, and FLAIRS, Eds., Florida Artificial Intelligence Research Society. Menlo Park, California: AAAI Press, may 2007, pp. 530–535.

[33] A. A. Hopgood, *Intelligent Systems for Engineers and Scientists*, 2nd ed. Boca Raton London New York Washington, D.C.: CRC Press, 2001.

[34] G. J. Nalepa and I. Wojnicki, "ARD+ a prototyping method for decision rules. method overview, tools, and the thermostat case study," AGH University of Science and Technology, Tech. Rep. CSLTR 01/2009, June 2009.

[35] G. J. Nalepa and I. Wojnicki, "VARDA rule design and visualization tool-chain," in *KI 2008: Advances in Artificial Intelligence: 31st Annual German Conference on AI, KI 2008: Kaiserslautern, Germany, September 23–26, 2008*, ser. Lecture Notes in Artificial Intelligence, A. R. Dengel and et al., Eds., vol. 5243. Berlin; Heidelberg: Springer Verlag, 2008, pp. 395–396, to be published.

[36] G. J. Nalepa and A. Ligęza, *Software engineering: evolution and emerging technologies*, ser. Frontiers in Artificial Intelligence and Applications. Amsterdam: IOS Press, 2005, vol. 130, ch. Conceptual modelling and automated implementation of rule-based systems, pp. 330–340.

[37] *jBPM User Guide*, 5th ed., The jBPM team of JBoss Community, Dec 2011, online: http://docs.jboss.org/jbpm/v5.2/userguide/.

[38] T. Rademakers, T. Baeyens, and J. Barrez, *Activiti in Action: Executable Business Processes in BPMN 2.0*, ser. Manning Pubs Co Series. Manning Publications Company, 2012.

[39] K. Kaczor, K. Kluza, and G. J. Nalepa, "Towards rule interoperability: Design of Drools rule bases using the XTT2 method," *Transactions on Computational Collective Intelligence XI*, vol. 8065, pp. 155–175, 2013.

[40] K. Kluza, T. Maślanka, G. J. Nalepa, and A. Ligęza, "Proposal of representing BPMN diagrams with XTT2-based business rules," in *Intelligent Distributed Computing V. Proceedings of the 5th International Symposium on Intelligent Distributed Computing – IDC 2011, Delft, the Netherlands – October 2011*, ser. Studies in Computational Intelligence, F. M. Brazier, K. Nieuwenhuis, G. Pavlin, M. Warnier, and C. Badica, Eds. Springer-Verlag, 2011, vol. 382, pp. 243–248. [Online]. Available: http://www.springerlink.com/content/d44n334p05772263/

[41] M. Szpyrka, G. J. Nalepa, A. Ligęza, and K. Kluza, "Proposal of formal verification of selected BPMN models with Alvis modeling language," in *Intelligent Distributed Computing V. Proceedings of the 5th International Symposium on Intelligent Distributed Computing – IDC 2011, Delft, the Netherlands – October 2011*, ser. Studies in Computational Intelligence, F. M. Brazier, K. Nieuwenhuis, G. Pavlin, M. Warnier, and C. Badica, Eds. Springer-Verlag, 2011, vol. 382, pp. 249–255. [Online]. Available: http://www.springerlink.com/content/m181144037q67271/