Modeling and Enacting Enterprise Decisions

Laurent Janssens[1,2(✉)], Johannes De Smedt[1], and Jan Vanthienen[1]

[1] KU Leuven, Leuven Institute for Research on Information Systems, Leuven, Belgium
{laurent.janssens,johannes.desmedt,jan.vanthienen}@kuleuven.be
[2] KU Leuven, Declarative Languages and Systems, Leuven, Belgium

Abstract. For years, the capturing of business decisions in enterprise models has not been treated as a separate concern. Rather, decisions were included in business process models or in knowledge models and ontologies. This leaves the overall view of a decision and its interplay with other decision and data requirements dispersed and hard to maintain.

The recently introduced OMG Decision Model and Notation (DMN) standard deals with decisions as a separate concern and presents decision modeling as a sovereign part of enterprise modeling. Decisions are modeled at the logical level, and the model is executable.

This work links the logical decision model to various execution strategies and processes by formalizing decisions within a business context, and by examining execution mechanisms for different availability of input data. These strategies can determine the preferred business process that handles the decision or allow to execute a decision model even when not all inputs are available up front.

Keywords: Decision modeling · Process modeling · Process enactment · Decision execution

1 Introduction and Related Work

Capturing the needs for making consistent, correct, and maintainable decisions is hard. Currently, many works focus on modeling decisions in processes (or as process steps). Most processes and business process models incorporate decisions of some kind, but they are often hidden in process flows, process activities or manual activities. It is not considered good practice to model the detailed decision paths in the business process model. Separating rules and decisions from the process simplifies the process model (separation of concerns).

Decisions are typically based upon a number of business decision rules that describe the premises and possible outcomes of a specific situation. Each decision may depend on a number of input data and on the outcome of one or more other decisions. Typical decisions are: creditworthiness of the customer in a financial process, claim acceptance in an insurance process, eligibility decisions in social security, etc. Since these decisions guide the activities and workflows of all process

J. Krogstie et al. (Eds.): CAiSE 2016, LNBIP 249, pp. 169–180, 2016.
DOI: 10.1007/978-3-319-39564-7_17

stakeholders (participants, owners), they should be regarded as first-class citizens in enterprise modeling.

Sometimes decisions can be included as an activity in a business process. The process then handles a number of steps, shows the appropriate decision points and represents the path to follow for each of the alternatives. In a large number of cases, however, a particular business process does not just contain decisions, but the entire process might be about a decision. The major purpose of a loan process e.g., or an insurance claim process, is to prepare and make a final decision. The process executes the decision by showing different steps, modeling the communication between parties, preparing required input data, recording the decision and returning the result. The purpose of this paper is to examine how such a model of a decision in the new OMG DMN standard can be brought to execution in a fixed process or in a flexible execution setting.

An alternative method, namely Product Based Workflow Design (PBWD) is presented in [1–3]. Similarities can be found between PBWD and case handling workflow management systems [4,5], as they focus on the data elements rather than on the control flow of the process. Our approach is related, but focuses on the decisions rather than on the data. The paper is organized as follows. First, related work including the DMN standard is discussed, as well as a formal definition of a DMN model. Next, the role of a decision model next to a process model is elaborated in Sect. 3 and illustrated in Sect. 4, supported by 3 new ways of executing a DMN model. Finally, a research agenda is formulated and a conclusion proposed.

2 Background

The term "decision modeling" refers to various ways to represent decision rules, constraints and conditional statements that describe the premises and outcomes of a specific situation and govern the actions that take place in applications and systems. Numerous decision models have been proposed to this end [6]. These are also used in many domains, e.g. business processes, credit risk [7,8], and medical diagnosis [9].

2.1 Decision Model and Notation (DMN)

Driven by an expanding need for decision support and automation, the OMG standards management group has developed a new standard: DMN, the Decision Model and Notation [10,11]. A first version of the standard has been made available in September 2015 and version 1.1 is awaiting publication. Many vendors (Oracle, IBM, Signavio, Decision Management Solutions, FICO, BlueRiq, OpenRules, among others) already offer tooling and industry users are starting to use the standard. An overview of some DMN concepts is provided in Fig. 1 where a decision model (right) is linked to a decision activity in a process model (left).

DMN consists of two levels: the decision requirements level (DRD, decision requirements diagram), indicating the requirements of a decision, and the decision logic level. The decision logic level provides a language for specifying decision logic (FEEL, the Friendly Enough Expression Language), and a corresponding notation (boxed expressions and decision tables) which allows such expressions to be linked to elements in the requirements level. Earlier research about decision modeling and management [12–14], decision tables and structures [15–17] is now becoming very important in the context of business processes and DMN.

DMN provides a common decision model notation that is readily understandable by users/modelers in all the development phases: modeling, execution, management and monitoring of those decisions. In DMN, decisions are based on criteria, conclude one or more results, require one or more subdecisions, and refer to a decision logic which is represented using a simple (e.g., decision tables) or a complex technique (e.g. analytics).

In Fig. 1 a simple car rental example is shown as executed next to a Business Process Model and Notation (BPMN) diagram [18]. DMN uses decision blocks (rectangles), business knowledge (cut-corner rectangles), and decision inputs (ovals). In this case, a decision table is used to capture the eligibility rules for assigning a customer a car, based on his/her employment status, country, and age. This eligibility is a decision that can be used as an input for another one, in this case the routing of the customer through the process. For this decision, the application risk is also evaluated and makes up the final decision for offering a product in the workflow.

2.2 Formalization

In DMN, decisions and their inputs are structured using decision requirement diagrams (DRD). These diagrams denote the information requirements of each decision, by connecting them with their subdecisions and inputs. This is represented by a directed acyclic graph. The DMN specication allows a DRD to be an incomplete or partial representation of the decision requirements in a decision model. Thus, the complete set of requirements must be derived from the set of all DRDs in the decision model.

Definition 1. *The decision requirements level R_{DM} of a decision model DM is a single decision requirement graph depicted as a set of decisions requirement diagrams.*

We use R_{DM} to denote this set of DRDs. The DMN standard describes the requirement level of a decision model to consist of a decision requirement graph (DRG). This DRG is represented by a DRD which is self-contained, i.e. for every decision in the diagram all its requirements are also represented in the diagram.

Definition 2. *A decision requirement diagram $DRD \in R_{DM}$ is a decision requirement graph DRG if and only if for every decision in the diagram all its modeled requirements, present in at least one diagram in R_{DM}, are also represented in the diagram.*

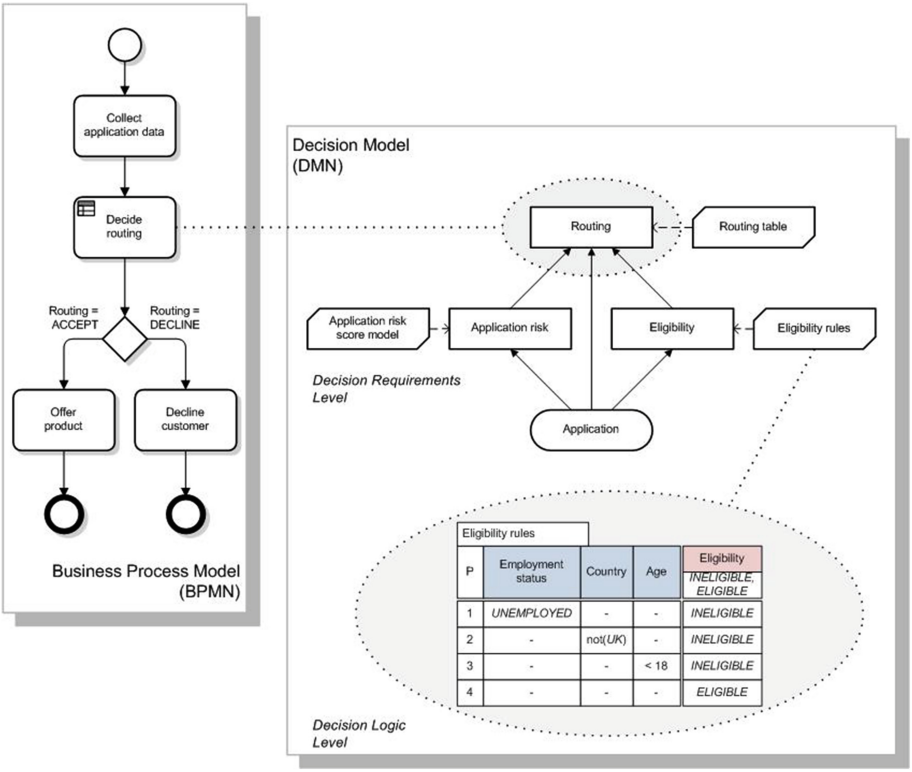

Fig. 1. Car rental decision model example [10].

According to the DMN standard [10] a decision can be dened as in Definition 3. In processes a decision is usually an activity, i.e. the act of using the decision logic. Another common meaning is that a decision is the actual result, which we call the output of a decision.

Definition 3. *A decision is the description of the decision logic used to determine an output from a number of inputs.*

In DRDs these decisions D are represented by the decision nodes $D \in R_{DM}$. We will use D to refer to both the decision and its representing node in a DRD. From the definition of DRGs we can derive an important property of decisions in the decision model.

Property 1. Given a decision model every decision D in that model has a unique decision requirement graph DRG_D with D as its single top-level decision.

From Definition 2 we know a DRG contains exactly all information requirements of its decisions. Thus there can only exist one DRG with D as its single top-level decision. We use DRG_D to denote this DRG. Decisions are often structured to use the results of other intermediate decisions, called subdecisions.

Definition 4. *A decision Ds is a subdecision of decision D if and only if it is part of* DRG_D.

An order can be defined on decisions in a decision model by using the property that DRDs are directed acyclic graphs. From this property we know each DRD has a topological order. The concept of topological orders is closely related to partial orders bringing us to Property 2.

Property 2. The topological order of a DRD induces a partial order $\leq$ on the decisions contained in the DRD.

For two decisions D_1 and D_2 we say $D_2 \leq D_1$ if and only if there is a directed path from D_2 to D_1, i.e. D_2 is a subdecision of D_1. Since decisions are declarative, this partial order does not dictate an execution order, but rather a requirement order.

3 The Role of Decision Models

The role of decision models must be framed within the context of business processes as well. Currently, many approaches have hidden the way in which the decision or chain of decisions contribute to the process to achieve the end result in a holistic way. This section relates the different ways in which a decision (model) can contribute to the overall achievement of integration of the dynamic execution of activities and the requirements that are imposed by a decision model and its inputs as described in [19].

3.1 A Decision Model Corresponding to a Single Decision Activity in a Process Model

Within a business process context, activities dominate the way in which the assignment of resources, data, etc. is done. Mainly due to this atomic division of the workflow, it is often assumed to have data only connected to one activity or one splitting node connected to many activities to avoid any faulty interactions over the global process model. In many cases in which there is no interleaving of data throughout the model, decisions can be used in this form of single branches that use the output of an activity that has implemented and evaluated the criteria for making the decision.

3.2 A Decision Model Spanning over Multiple Decision Activities in an Existing Process Model

Multiple activities in a process model may refer to different decisions that are all part of the same decision model. An important remark is whether the data needed as input for a decision is available before the decision is invoked. Activities producing the input required by decisions should occur before invoking those

decisions. Otherwise, situations appear in which decisions cannot be made consistently due to incomplete or incorrect input. Next to the process, the decision model has as big an impact on how the system evaluates and uses data inputs as the process model. When decisions are dependent of intermediary results that are spawned earlier in the process, the decision model also puts constraints on the sequence ordering of the workflow, although they might not be defined explicitly in the process model itself. These issues have been addressed in [20].

3.3 A Decision Model that can Be Translated to a Straightforward Process for Execution

Sometimes the business process is really about a big decision. Some approaches were proposed [21] to model this in business process models, hence forgoing the purpose of decision models that were designed specifically for this task. Other approaches [22] rather seek to find the balance between data-driven models and business processes, however, the process part still remains a subordinate to the decision model. In this paper we assume that when the process is really about a big decision, the process model can be considered as the chosen execution flow to make the decision.

3.4 Executing a Decision Model Beyond One Fixed Decision: Flexibility

Once a decision model is built, it could be used for multiple purposes, not just the obvious decision that is present in the context of a current business question. The decision model could be designed for the current process, but also for other or future processes.

4 Three Scenarios for the Execution of Decision Models

Three types of scenarios for the execution of decision models can be identified. In this section we describe each of these scenarios in detail, and illustrate them using the example in Figs. 1 and 2. This table is an example instantiation of the Routing decision shown in Fig. 1.

Routing				
U	Eligibility	Application Risk	Age	Routing
1	*INELIGIBLE*	-	-	*DECLINE*
2	*ELIGIBLE*	*HIGH*	-	*DECLINE*
3		*MODERATE*	*< 25*	*DECLINE*
4			*>= 25*	*ACCEPT*
5		*LOW*	-	*ACCEPT*

Fig. 2. Routing decision table.

The identified scenarios can be seen as executing decisions in cases where all input is available, where enough input is available, and where only some input is available. Subsection 4.1 deals with the case of complete input, discussing this situation, and introducing additional formalisms to discuss these execution scenarios. In Subsection 4.2 scenarios where, although incomplete, enough input is available to make a decision. In these scenarios it becomes possible to optimize the decision process, either statically or dynamically. Subsection 4.3 ultimately details scenarios where insufficient input is available to make a decision, but it shows that even in these cases it is often possible to still execute a decision, e.g. to exclude some possible outcomes.

4.1 Standard Forward Decision Execution

Often the decision's input is available up front. The decision logic is captured in a straightforward decision model and the decision is to find the correct outcome for a specific set of input values. These are typical current DMN applications, e.g., determine the discount for specific clients, based on discount policy, client data, history, etc. or determine eligibility for insurance given the company policy. The reasoning mechanism does not need high flexibility, but it is important that company rules and policies can easily be adapted and brought to implementation.

To illustrate this and the following scenarios we extend our earlier formalization, to further describe a decision's inputs and outputs, and define what it means to execute a decision.

Definition 5. *The decision vocabulary Σ_D of a decision D is the set of the union of the decision input symbols Σ_I and the decision output symbols Σ_O.*

In other words, the decision vocabulary contains a symbol for each input and each output attribute. We will identify an attribute σ by its associated vocabulary symbol and say $\sigma \in \Sigma$. With each such symbol we associate an attribute domain.

Definition 6. *Each attribute $\sigma \in \Sigma_D$ has an associated domain D_σ, the set of all its possible values.*

Definition 7. *An attribute structure S of a vocabulary Σ contains for each attribute symbol $\sigma \in \Sigma$ a value $\sigma_S \in D_\sigma$.*

The set of symbols interpreted by S is denoted Σ_S.

Definition 8. *An attribute structure S' of a vocabulary Σ' is called a partial attribute structure of Σ iff $\Sigma' \subseteq \Sigma$. S is called an input structure or output structure of Σ_D if $\Sigma_S = \Sigma_I$ or $\Sigma_S = \Sigma_O$, respectively.*

Definition 9. *A total structure $S = S_I \cup S_O$, where S_I and S_O are input, respectively, output structures of D is an instance of D iff D maps input S_I to output S_O.*

In the case of a decision represented by a single-hit decision table a structure is an instance if a row maps its input to its output.

Definition 10. *A decision D is completely invokable for a partial attribute structure S if S is an input structure for* Σ_D.

The routing decision with the associated decision table shown in Fig. 2 is completely invokable when all its input is available, i.e. when the application is filled in completely, and the Eligibility and Application risk decisions have been made. Clearly this is a very strong restriction which is not always needed, this restriction is relaxed in the next subsection.

4.2 Optimized Forward Decision Execution

Often the decisions inputs are not available up front, but can be obtained at a certain cost (database lookup, user question). The decision logic is captured in a straightforward decision model and the decision is still to **find the correct outcome for a specific set of input values**. But the order of looking up input data and answers to user questions can have cost implications. If a decision can be made with only a partial set of inputs, it is more cost effective. The decision model is the same, but the execution of the specific decision might be optimized. So the question here is: what is the optimal process of executing the specific decision, given the cost of obtaining data and the frequency of cases? Since the optimal process of a decisions execution can be case dependent it becomes necessary to know when a decision can safely be made, i.e. when the outcome of a decision made with only partial input will not change if more data becomes available, this is formalized in the following definitions.

Definition 11. *Given a partial input structure S of vocabulary* Σ *for a decision D an input extension S' of S is a total input structure of* Σ *where* $\sigma'_S = \sigma_S$ *if* $\sigma \in \Sigma_S$ *and* $\sigma'_S = v$ *for some* $v \in D_\sigma$ *otherwise, for all* $\sigma \in \Sigma_I$.

Definition 12. *A decision D is safely invokable for a partial input structure S if there exists exactly one output structure SO such that* $S' \cup S_O$ *is an instance of D, for every input extension S' of S.*

This definition only applies when the definition is represented as a single hit table, and thus highlights an important benefit of using single hit tables, they allow safe invocation when sufficient input is available. This is not the case for multi-hit tables, as all applicable and non-applicable rules must be identified. As various process models can result from a decision model, the choice between different process models becomes an important issue. There is a need for criteria to rate the process models such that models can be compared to each other and the process model that best matches the business strategy can be chosen. Possible process modeling criteria are indicated in [23,24]. Applying these criteria shows some important strategies:

- Customer perspective: minimal points of contact.
- Business process behavioral perspective: starting with a labor-intensive activity is not optimal since the decision could be easily taken otherwise.
- Organizational perspective: the number of handovers can be minimized.
- Informational perspective: all necessary information could be easily acquired at one point in time.
- External environmental perspective: assessing external information should be limited.

With various decision process models to choose from, the decision process model that fits best with the business requirements can be chosen.

These criteria allow for two types of optimization, one static, the other dynamic. The approaches depend on the assumption made about input availability, offering optimization capabilities for both situations where all input is available as well as if only some input is available.

Static Optimization of Forward Decision Execution. In the static approach the assumption is that all input is available, the decisions execution can then be optimized by ordering the different subdecisions using the before mentioned criteria, or the cost, or frequency of these decisions. If we assume that for the example in Figs. 1 and 2 the Application Risk decision is very labor-intensive, then we can optimize the execution process by always invoking the Eligibility decision first, and having the Routing decision result in DECLINE, if the customer is ineligible. This constitutes an optimal process for the execution of the Routing decision when all input is available.

This is similar to the conversion of decision tables into program code, meaning that an efficient execution tree has to be generated for all the combinations of the condition values. Different paths in the execution tree can test the conditions in a different order, therefore the number of possible trees is enormous (and grows fast with a larger number of conditions). Additional information used in the conversion algorithm can be: the test time for each condition (if not available, all test times are considered equal) and case frequencies. In that way, the average execution time for the decision can be minimized. In the era between 1965 and 1980, much effort was devoted to research on this conversion of decision tables into efficient test sequences, leading to a growing list of conversion algorithms. A discussion of the evolution and results of the most important algorithms can be found in [25].

Dynamic Optimization of Forward Decision Execution. Apart from the previous static approach a dynamic approach can be taken if the assumption that not all input is available up front is not valid. Using the above criteria and the requirement order of the decision model, as described in Property 2, an optimal process for the execution of the decision can be generated dynamically on a case-by-case basis. By taking into account available input, Definition 12 can be used to guide the optimization, by invoking a high priority, or low cost decision as

soon as it is safe to do so. In doing so, multiple unnecessary subdecisions may be excluded from the process, increasing its efficiency.

4.3 Flexible Decision Execution Scenarios

The specific **decision to be answered is not always fixed**. When dealing with a credit application, for example, sometimes the decision is not: Does this specific customer (with all input data available) get a loan?, but: What can we already derive from the available data? Another question could be: Which changes should be recommended to the customer to obtain a higher loan? DMN decisions do not specify whether the required input data is needed up front to be able to determine the outcome and assume a given decision path. This both has its advantages and disadvantages. A disadvantage is that an enactment of an ill-constructed DMN model might not be able to derive the results needed, due to missing input. If the user had not intended this, he might be unaware until runtime. On the other hand, it gives DMN a greater flexibility. Furthermore, it might not always be an impediment as there is a way to circumvent partial input. Given enough information it is often possible to exclude certain outcomes, as defined in Definition 13.

Definition 13. *A decision D is partially invokable for a partial input structure S_I if their exists an output structure S_O and there exists no input extension S'_I of S_I such that $S'_I \cup S_O$ is an instance of D.*

When resuming the example of the last subsection, partial invocation offers additional possibilities. Assume a customer who is eligible and of age 25 submits an application. If the outcome High can be excluded by partial invocation of the Application Risk decision, then the Routing decision can be safely invoked to result in the Accept outcome, since only the Moderate and Low outcomes remain for application risk.

5 Conclusions and Future Work

Business decisions are important, but are often not made explicit, hidden in processes or in the manual activities. In fast changing environments, decision models and better decision management will allow to create maintainable and flexible decision execution. This paper has shown how decision models differ, resemble, and complement business process models in enterprise modeling.

For future work, a plethora of opportunities exist in defining and optimizing the execution of decision models. First of all, many inference techniques can provide the enhancement of decision inputs, both in terms of completing partial input, as extending decision outcomes with examples, and so on. Secondly, it is the task of decision models to allow for the interplay of these different mechanisms that are each better tailored towards solving and extending certain decision domains as used in the models. Finally, it is yet to be proven how well decision models can truly relieve business process models in terms of flexibility, resulting in comprehensible though expressive declarative specifications.

References

1. Reijers, H.A., Limam, S., Van Der Aalst, W.M.: Product-based workflow design. J. Manage. Inf. Syst. **20**(1), 229–262 (2003)
2. Vanderfeesten, I., Reijers, H.A., van der Aalst, W.M.: Case handling systems as product based workflow design support. In: Filipe, J., Cordeiro, J., Cardoso, J. (eds.) Enterprise Information Systems. LNBIP, vol. 12, pp. 187–198. Springer, Heidelberg (2007)
3. Vanderfeesten, I.T., Reijers, H.A., van der Aalst, W.M.: An evaluation of case handling systems for product based workflow design. In: ICEIS, vol. 3, pp. 39–46 (2007)
4. Van der Aalst, W.: On the automatic generation of workflow processes based on product structures. Comput. Ind. **39**(2), 97–111 (1999)
5. Traganos, K., Grefen, P.: Hybrid service compositions: when BPM meets dynamic case management. In: Dustdar, S., Leymann, F., Villari, M. (eds.) Service Oriented and Cloud Computing. Lecture Notes in Computer Science, vol. 9306, pp. 226–239. Springer, Heidelberg (2015)
6. Koutsoukis, N.S., Mitra, G.: Decision Modelling and Information Systems: The Information Value Chain, vol. 26. Springer, Heidelberg (2003)
7. Kim, D.J., Ferrin, D.L., Rao, H.R.: A trust-based consumer decision-making model in electronic commerce: the role of trust, perceived risk, and their antecedents. Decis. Support Syst. **44**(2), 544–564 (2008)
8. Li, H., Zhou, X.: Risk decision making based on decision-theoretic rough set: a three-way view decision model. Int. J. Comput. Intell. Syst. **4**(1), 1–11 (2011)
9. Sun, X., Faunce, T.: Decision-analytical modelling in health-care economic evaluations. Eur. J. Health Econ. **9**(4), 313–323 (2008)
10. OMG: Decision Model and Notation (2015)
11. Taylor, J., Fish, A., Vanthienen, J., Vincent, P.: Emerging standards in decision modeling-an introduction to decision model & notation. In: Fischer, L. (ed.) iBPMS (2013)
12. Taylor, J., Raden, N.: Smart Enough Systems: How to Deliver Competitive Advantage by Automating Hidden Decisions. Pearson Education, Upper Saddle River (2007)
13. Taylor, J.: Decision Management Systems: A Practical Guide to Using Business Rules and Predictive Analytics. Pearson Education, Boston (2011)
14. Fish, A.N.: Knowledge Automation: How to Implement Decision Management in Business Processes, vol. 595. Wiley, Hoboken (2012)
15. Vanthienen, J.: What business rules and tables can do for regulations. Bus. Rules J. **8**(7), 2 p. (2007)
16. Ligeza, A., Nalepa, G.J.: A study of methodological issues in design and development of rule-based systems: proposal of a new approach. Wiley Interdisc. Rev. Data Min. Knowl. Discov. **1**(2), 117–137 (2011)
17. Nalepa, G.J., Ligeza, A., Kaczor, K.: Formalization and modeling of rules using the XTT2 method. Int. J. Artif. Intell. Tools **20**(06), 1107–1125 (2011)
18. White, S.A.: BPMN Modeling and Reference Guide: Understanding and Using BPMN. Future Strategies Inc., Lighthouse Point (2008)
19. Vanthienen, J., Caron, F., De Smedt, J.: Business rules, decisions and processes: five reflections upon living apart together. In: Proceedings SIGBPS Workshop on Business Processes and Services (BPS 2013), pp. 76–81 (2013)

20. Janssens, L., Bazhenova, E., De Smedt, J., Vanthienen, J., Denecker, M.: Consistent integration of decision (DMN) and process (BPMN) models. In: CaiSE Forum (Springer Accepted)
21. Kluza, K., Kaczor, K., Nalepa, G.J.: Enriching business processes with rules using the Oryx BPMN editor. In: Rutkowski, L., Korytkowski, M., Scherer, R., Tadeusiewicz, R., Zadeh, L.A., Zurada, J.M. (eds.) ICAISC 2012, Part II. LNCS, vol. 7268, pp. 573–581. Springer, Heidelberg (2012)
22. van der Aa, H., Reijers, H.A., Vanderfeesten, I.: Composing workflow activities on the basis of data-flow structures. In: Daniel, F., Wang, J., Weber, B. (eds.) BPM 2013. LNCS, vol. 8094, pp. 275–282. Springer, Heidelberg (2013)
23. Reijers, H.A., Mansar, S.L.: Best practices in business process redesign: an overview and qualitative evaluation of successful redesign heuristics. Omega **33**(4), 283–306 (2005)
24. Ferme, V., Ivanchikj, A., Pautasso, C.: A framework for benchmarking BPMN 2.0 workflow management systems. In: Motahari-Nezhad, H.R., Recker, J., Weidlich, M. (eds.) Business Process Management. LNCS, vol. 9253, pp. 251–259. Springer, Heidelberg (2015)
25. Henry Beitz, E., et al.: A Modern Appraisal of Decision Tables, a CODASYL Report. ACM, New York (1982)