Bidimensional Process Discovery for Mining BPMN Models

Jochen De Weerdt, Seppe K.L.M. vanden Broucke and Filip Caron

Department of Decision Sciences and Information Management, KU Leuven
Naamsestraat 69, B-3000 Leuven, Belgium
jochen.deweerdt@kuleuven.be

Abstract This paper presents "BPMN Miner", a process discovery technique that uses BPMN as the representational language for the discovery result. The proposed approach is novel in the sense that it is able to represent control-flow with BPMN constructs, but also because it augments the control-flow perspective with an organizational dimension by discovering swimlanes that represent organizational roles in the business process. Additional advantages of the proposed mining approach can be summarized as follows: it provides intuitive and easy-to-use abstraction/specification functionality which makes it applicable to event logs with various complexity levels, it provides instant feedback about the conformance between the input log and the resulting model based on a dedicated fitness metric, it is robust to noise, and it can easily integrate with modeling and other BPM tools with exporting functionality through the XPDL-format. In this way, BPMN Miner will take process mining one step closer to the status of indispensable for business process reengineering as discovered models are immediately available in the preferred language of a majority of practitioners, educators and researchers.

Key words: process mining, process discovery, Business Process Model and Notation, BPMN

1 Introduction

The research field of process mining deals with the extraction of knowledge from event logs—transactional data repositories containing historical information as stored by process aware information systems [1]. Process mining analysis tasks are commonly distributed over three, broad categories: process discovery (derive some sort of descriptive model from a given event log), conformance checking (match the behavior seen in a given event log with that of a process model), and process enhancement (improve and extend an existing model based on additional data). Without doubt, the process mining analysis task of process discovery has received the most attention in the research community. As such, a great deal of algorithms and techniques have been proposed to this end. In many cases, the representational language utilized by techniques to represent discovered models reflect the same semantics as common process modeling standards which are being used by practitioners and researchers. As such, popular output formats

for discovered models include: Petri nets [2], Heuristic nets [3], Causal nets [4] and EPCs [5].

However, one particular process modeling standard which has been somewhat overlooked in the process mining community is the Business Process Model and Notation (BPMN) standard [6]. This is peculiar, as the majority of educators and researchers have adopted BPMN as the language of choice when working with business processes. The reason for this stems mainly from the fact that BPMN has, for a long time, lacked a formal definition of its execution semantics. The initial specifications [7] defined behavioral semantics using the notion of token flow, similar to Petri nets and UML activity diagrams, but solely described the execution semantics in narrative form. Although researchers have defined formalized definitions, ranging from attempts to define a formal semantics for a subset of BPMN [8, 9] to more complete approaches [10, 11, 12], the fact remains that both BPMN's many visual objects and its weak semantic formalization have caused scholars to develop process discovery techniques based on more formalized modeling approaches.

Nevertheless, given BPMN's wide dissemination, we argue that the availability of a native BPMN-based process discovery technique could be of great benefit within process identification, optimization and re-engineering efforts. Therefore, this paper introduces a new process discovery technique—*BPMN Miner*—which represents discovered control-flow aspects using BPMN constructs. We select a subset of constructs, both because discovering some constructs is near-impossible using only historically recorded process execution "traces", as well as because scholars have indicated that only a small subset of BPMN's constructs are used by the majority of practitioners [13]. However, our proposed approach is unique in the sense that it combines the control-flow perspective with an organizational dimension by discovering swim lanes that represent organizational roles in the business process. In addition, our technique provides intuitive and easy-to-use abstraction/specification functionality which makes it applicable to event logs with various complexity levels, provides instant feedback about the conformance between the input log and the resulting model based on a dedicated fitness metric, is robust to noise, and can easily be integrated with modeling and other BPM tools by exporting the discovered model.

The remainder of this paper is structured as follows. Section 2 outlines the rationale behind our proposed approach. Section 3 presents a comparative study regarding currently available process discovery techniques. Section 4 provides a formalization of the developed technique. Section 5 outlines a case study, illustrating the benefits of the BPMN Miner using a concise example. Section 6 concludes the paper.

2 Rationale

In this section, it is argued why automated process discovery with BPMN as the underlying modeling language is of utmost relevance for practitioners, education, and researchers.

2.1 Relevance for Practitioners

BPMN is considered as the de facto standard for process modeling [14] and is widely adopted by both business and IT communities [15]. Practitioners from both communities use the notation standard mainly for documenting, improving, simulating and implementing business processes [16].

While the adoption of BPMN for the purpose of process modeling has been successful, the adoption of process mining as the most valuable tool for business process improvement initiatives is somehow lagging. It is argued that a core factor contributing to this effect consists of a lack of deep technical understanding from typical business practitioners involved in such improvement initiatives with regard to conventional languages used by process mining tools. Even despite the uptake of commercial and highly user-friendly process discovery tools, the mismatch in modeling notation and the subsequent translation effort required to go from the analysis phase to redesign brings about an unnecessary adoption barrier. Therefore, we contend that the mining of BPMN models from execution data will prove highly beneficial for the further adoption of process mining, along the following lines of reasoning:

Automated process identification Practitioners involved in process identification and modeling, can be persuaded into using automated process discovery techniques if such techniques provide effortless integration with popular modeling tools. With the capability to discover BPMN models from event logs, actual time savings can be realized for practitioners who are now typically involved in a two-step process of first interpreting automated discovery results and then making use of the insights gained for designing or adapting process models. Note that the survey in [17] showed that process model editing functionality is amongst the most desired additional features for the ProM-framework. This clearly indicates that (automated) analysis and (re)design are tightly coupled and tools and techniques in both areas should be maximally aligned.

Facilitating the process re-engineering cycle In typical redesign scenarios, people observe the as-is state of a process or set of processes, take certain improvement decisions, analyze the outcomes, and subsequently take additional improvement measures if necessary. Currently available automated process discovery techniques require business (process) analysts to possess additional skills and knowledge about typical output modeling notations such as Petri nets, Heuristic nets, or Fuzzy models. In addition, next to interpretation, practitioners will also be required to compare the results of automated discovery with existing process models. Such a comparison is far from effortless requiring profound technical understanding often unavailable in organizations with lower BPM maturity. With discovered process models in native BPMN format, the mapping of discovered vs. existing models becomes significantly easier.

Improved communication of process mining results Working with a unified process model notation throughout the entire BPM life cycle will enable improved communication between functional units as well as across organizational

hierarchies. Due to the fact that many organizations heavily rely on process modeling for documentation and communication, investments in data collection and data analysis might be perceived more worthwhile because these investments should not be looked at in isolation, but can actually contribute to and improve existing BPM practices.

2.2 Relevance for Education

A second stakeholder group for which BPMN-based process discovery is of value is educators and students. Generally speaking, BPM courses and text books (e.g. [18]) kick off with a thorough discussion on process modeling, with BPMN often receiving a great deal of attention. In later stages or chapters, process mining is brought up as well. However, this often requires educators to introduce new modeling notations, most notably Petri nets, Heuristic nets, and Fuzzy models, given their popularity for process mining. Moreover, the introduction of such new paradigms quickly obfuscates the link with process modeling and execution topics. While several programs can already leverage upon previously acquired knowledge, a majority of students, e.g. in business/management-oriented studies, do not possess such background knowledge. For that reason, a high-quality process discovery tool which presents its results in BPMN is likely to lower the effort required by educators to incorporate process mining in their units. It is pointed out that, from a student perspective, a positive attitude towards the usability and ease of use has been observed with respect to BPMN and its tool support [19].

2.3 Relevance for Research

Key research contributions in the process mining field have traditionally been strongly technical in nature. While valuable application studies have been published as well, there exists a significant opportunity for research about topics such as usefulness, ease of use, user acceptance, etc. of process mining within organisations. A process discovery technique with BPMN as underlying modeling language will lower barriers to conducting such studies, which often involve technically lower skilled individuals. Ultimately, user-centered studies could provide valuable insights into how the full potential of process mining can be realized or in what directions future process mining research should develop.

3 Comparative Study

The quality of discovered process models is inherently determined by the implicit search space implied by the representational bias of a process discovery technique (and thus its associated representation language). In [20], the authors advocate for selecting the "right" representational bias when discovering process models from event logs. They argue that the representational bias should be based on

essential properties of a process model and should not be driven by the desired graphical representation. The process mining manifesto also lists the aspect of representational bias as one of the key challenges in the process mining domain [21].

While we don't contest that the search for an optimal representational bias for process discovery in terms of the implicit search space is of interest, a more pragmatic stance is taken in this paper. This is because, from a knowledge discovery viewpoint [22], patterns discovered from data should adhere to several principles: validity, novelty, usefulness, and understandability. While the use of for instance Causal nets for process discovery is likely to produce rich and high quality results, such an approach suffers from a steep learning curve which often leads to problems of understandability. For this reason, we argue that a more pragmatic, user-centered stance with respect to the suitability of the representational bias is worth pursuing. This pragmatic approach is based upon an assessment of some key characteristics of process modeling notations for the purpose of process discovery, as detailed in Table 1. It is argued that there exist two contrasting effects that make it difficult to agree on one fit-for-all modeling notation for process discovery.

Modeling Notation	**Ease of Interpretation**	**Suitability Rep. Bias Proc. Disc.**	**Popularity (Modeling)**	**Popularity (Mining)**
Petri net	●●○○○	●●○○○	●○○○○	●●●●○
Heuristic net	●●●○○	●●●●○	○○○○○	●●●●●
Fuzzy model	●●●●○	●●●●○	○○○○○	●●●●●
Causal net	●○○○○	●●●●●	○○○○○	●●○○○
EPC	●●●●○	●○○○○	●●●●○	●●○○○
BPMN	●●●●●	●●○○○	●●●●●	○○○○○

Table 1. Key characteristics of process modeling notations for the purpose of process discovery.

Based on a comparative analysis of various modeling notations, it is observed that traditionally popular modeling notations used for process discovery (i.e. Heuristic nets, Fuzzy models, and Causal nets) put a strong emphasis on the suitability of the representational bias. Note that our judgment about the representational bias reflects how well these notations help process discovery techniques at expressing a large number of possible constructs, while at the same time avoiding syntactically incorrect models as much as possible. Therefore, Petri nets, another popular representation choice, are scored lower because it is actually non-trivial to avoid the construction of incorrect Petri nets. On the other hand, representation languages with a less steep learning curve such as EPC and BPMN make it more difficult for process discovery techniques in terms of representational bias because modeling constructs are difficult to map against recorded data and because of the broad set of available constructs in the case of BPMN. A second, even stronger, contrast exists in terms of the level of

popularity for modeling vs. mining. Basically, there exists an important discrepancy in the BPM domain between languages used for modeling and languages used for mining. While some might argue that models can be translated from one language to another (for instance through the use of Petri nets as BPM's Esperanto), this often involves non-trivial procedures. With this paper, we opt to bridge the gap between modeling and mining by making use of BPMN as the representational language.

To conclude, we recognize that BPMN presents several drawbacks as a representational language for process discovery. Most importantly, many of its concepts are difficult or impossible to map with recorded event data. In addition, the broad range of concepts also leads to the absence of a clear and crisp definition of its execution semantics, which is a much desired characteristic for process discovery. However, given the rationale in Section 2 for a native BPMN miner, the next section details how these limitations can be dealt with.

4 Implementation

This section provides a formalization of the developed technique, together with an overview regarding its implementation as a ProM plugin[1].

4.1 Preliminaries

We define the following terms and notations. An *event log* L is defined as a multi set of traces. Each trace $\sigma \in L$ is a finite sequence of events with σ_i the event at position i in trace σ. Let T_L denote a set of activities occurring in the event log L. Let O_L denote a set of originators occurring in the event log L. $a : L \rightarrow T_L$ is the function returning the activity for a given event, $o : L \rightarrow O_L$ is the function returning the originator (i.e. the role, person, group, department...) having executed the event. A trace can simply be denoted in full based on the activities and originators of its event, e.g.: $\sigma = \langle start^{alice}, register^{bob}, \ldots \rangle$.

The *Business Process Model and Notation* (BPMN) is a well-known diagrammatic notation to support the specification of business processes. BPMN consists of a high amount of notational elements, classified into four types: flow objects, connecting objects, artifacts and swim lanes. Artifacts and swim lanes are unrelated to process flows and thus do not come into play for BPMN's token-based execution semantics. Due to space constraints, we do not provide a full description of BPMN, but instead refer the reader to [6].

4.2 Control-Flow Discovery

One of the novel contributions of our process discovery technique is that it directly applies BPMN as the representational language for discovering process

[1] In analogy with the WEKA toolkit for data mining, ProM is an extensible framework that supports a wide variety of process mining techniques in the form of plug-ins. See: `http://www.processmining.org`.

models from event logs, thus not relying on a translation. As stated in the introduction, the discovery of BPMN models has been somewhat overlooked so far, mainly due to fact that BPMN lacks a formal definition of its execution semantics and its many notational constructs pose a challenge for discovery techniques. To overcome this issue, our proposed approach deliberately considers a subset of BPMN's notational constructs in order to perform the control-flow discovery. Other works have illustrated that only a small subset of BPMN is actually being applied in real-life modeling practice [13], those being gateways (XOR and parallel), tasks, sequence flow, start and end event, and swim lanes. All of these constructs are also supported by our approach. In addition, we highlight the fact that discovered BPMN process models by our approach do provide an ideal starting point which can easily be adapted, extended, and modified by modelers and practitioners, as the discovered model lies much closer to the representational language practitioners are already applying, thus preventing conversion steps (with potential loss of accuracy or behavioral representation) or having to learn other modeling notations.

To perform the control-flow discovery, our technique applies an algorithm which is similar to the one applied in Heuristics Miner [3]. The basic idea works as follows. First: dependency information is derived between activities in the event log to construct a so called dependency graph $D = \{(a,b)|a \in T_L \wedge b \in T_L \wedge \exists\sigma \in L : [\exists\sigma_i \in \sigma : [a(\sigma_i) = a \wedge a(\sigma_{i+1}) = b]]\}$. Next, the splits and joins in the dependency graph need to be characterized by semantic information and converted to BPMN constructs. For example, for the split $\{(a,b),(a,c),(a,d)\} \subset D$, we aim to investigate whether b, c, and d all occur in parallel (AND split), independently (XOR split), or a mixture of both. This is done by iterating over the traces in the event log and investigated succession and precedence relations for the given split or join respectively, similar to the approach applied in Heuristics Miner [23]. For example, for the split above, we investigate to see how many times b alone occurred after a, c alone occurred after a, d alone occurred after a, b and c occurred after a, b and d occurred after a, c and d occurred after a, and how many times all three activities occurred after a (always before the next occurrence of a). Based on these frequencies, we derive the semantics of the split or join, and thus define $I : T_L \rightarrow \mathcal{P}(\mathcal{P}(T_L))$ and $O : T_L \rightarrow \mathcal{P}(\mathcal{P}(T_L))$ as the functions returning the input and output patterns for each task. A pattern is defined as a set of sets where each set of activities is interpreted as a disjunction (XOR), and the activities within a set are interpreted as a conjunction (AND).

Based on this information, a BPMN model is constructed as follows. First, start and end events are added. Second, the BPMN graph is constructed. Activities are added for each $a \in T_L$: A_a. Next, a XOR input and output gateway is created for each activity A_a: XOR_a^i and XOR_a^o and connected to the activity with sequence flows (XOR_a^i, A_a) and (A_a, XOR_a^o). For each $a \in T_L$, AND gateways are created for each set of activities in $I(a)$ and $O(a)$: $AND_a^{i,j}$ and $AND_a^{o,k}$, which are connected to the input and output XOR gateways with sequence flows $(AND_a^{i,j}, XOR_a^i)$ for $j = 1, \ldots, |I(a)|$ and $(XOR_a^o, AND_a^{o,k})$ for $k = 1, \ldots, |O(a)|$. Next, a set of XOR connecting gateways

is constructed for each $a_i \in \bigcup(I(a))$ with $a \in T_L | a \neq a_i$: $XOR_d^{a_i,a}$ and for each $a_o \in \bigcup(O(a))$ with $a \in T_L | a \neq a_o$: XOR_d^{a,a_o}. Sequence flows are added for each $(AND_a^{o,k}, XOR_d^{a_i,a_o})$ so that $a_i \in T_L, a_o \in T_L, a \in T_L, k = 1, \ldots |O(a)|$ and with $a_i \in O(a)^k$[2]. Similarly, we add flows for each $(XOR_d^{a_i,a_o}, AND_a^{i,j})$ so that $a_i \in T_L, a_o \in T_L, a \in T_L, j = 1, \ldots |I(a)|$ and with $a_o \in I(a)^j$. Finally, the start and end event are connected with all the activities A_a for which $I(a) = \{\emptyset\}$ and $O(a) = \{\emptyset\}$ respectively. If no such activity can be found, the algorithm determines a single start/end activity based on the frequency of the activity occurring most commonly at the start/end of traces. On the other hand, if multiple start or ending activities can be identified, they are connected through a XOR gateway with the starting and ending event. The third phase of the control-flow discovery algorithm consists of simplification of the BPMN model. This simplification step consists of iterative removal of all gateways which only contain a single incoming and one outgoing sequence flow, merging all AND gateways with the same incoming activities and a single outgoing activity (using a XOR gateway to connect the merged outgoing activities to the AND gateway) and merging all AND gateways with the same outgoing activities and a single incoming activity (using a XOR gateway to connect the merged incoming activities to the AND gateway).

4.3 Filtering and Abstraction

Discovering process models containing a high amount of activities, dependencies and noise leads to spaghetti models with a high amount of variability. In this case, the value of the discovered process models decreases rapidly, as it is no longer possible to derive understandable insights on how the as-is process is behaving.

We have implemented a number of techniques to deal with the aspects of variability and noise. First of all, users have the option to configure a dependency threshold, similar as the dependency thresholds applied in Heuristics Miner, although here, only one dependency threshold is used. The dependency threshold affects which arcs will be taken into the account during the construction of the dependency graph D. Second, we also incorporate a split/join threshold, affecting which patterns will be considered to include in $I(a)$ and $O(a)$ (based on their frequency). Finally, in case event logs contain low-frequent activities, the implemented plugin also offers end users the option to first filter out these low-frequent activities from the log before continuing with the discovery.

4.4 Bidimensional Discovery

The majority of existing process discovery algorithms focus on discovering the control-flow perspective of an event log, meaning that they use the sequence and ordering of activities to derive a process model using a particular representational

[2] We assume here that the input and output sets are ordered. $O(a)^k$ thus returns the jth subset of $O(a)$.

language. Other techniques exist which start from another event log perspective (social network extraction techniques [24], for instance). In BPMN Miner, we apply a bidimensional approach, directly incorporating the social (i.e. originator) perspective in the discovered model, together with control-flow (i.e. the sequence flow between activities).

To model originator information, our technique makes use of the swimlane construct of BPMN, meaning that our technique attempts to create a number of swimlanes containing one or more activities. Each swimlane then represent a "worker pool" (or "role") which is responsible for executing its contained activities. The swimlanes are discovered as follows. Recall $o : L \rightarrow O_L$ as being the function returning the originator (i.e. the role, person, group, department...) having executed an event. Depending on the desired level of granularity, end-users can choose which originator attribute to use to construct the swimlanes. First, for each activity $a \in T_L$, a dedicated swimlane-representing set $S_i = \{a\}$ is constructed, containing this single activity, so that $S = \{S_1, \ldots, S_{|T_L|}\}$. Next, swimlanes S_i and S_j are merged iff $\exists o \in O_L, a_i \in S_i, a_j \in S_j, \sigma_i \in L, \sigma_j \in L : [a(\sigma_i) = a_i \wedge a(\sigma_j) = a_j \wedge o(\sigma_i) = o(\sigma_j) = o]$. Ultimately, this leads to a set of merged swimlanes representing a grouping of activities which are to be contained in their swimlane. The grouping is performed such that each swimlane also represents a "role" (a distinct grouping of originators) responsible for this set of activities.

4.5 Conformance Analysis

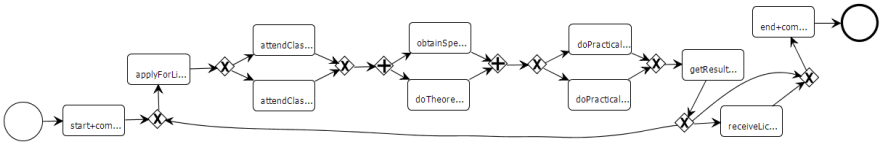
Apart from discovering BPMN models using a bidimensional approach, we also incorporate conformance checking functionality in BPMN Miner. More precisely, we add the ability to replay an event log over a discovered BPMN model (using token-based execution semantics), to determine the fitness level of the BPMN model in respect to the given event log. An overall fitness metric is then defined as the number of events in the event log which could be correctly executed by the BPMN model over the total amount of events. In addition, since this fitness metric is defined in an event-granular manner, we can also, for each activity in the model, indicate its degree of conformance in a visual manner. The following section outlines an example illustrating this functionality.

4.6 Exporting

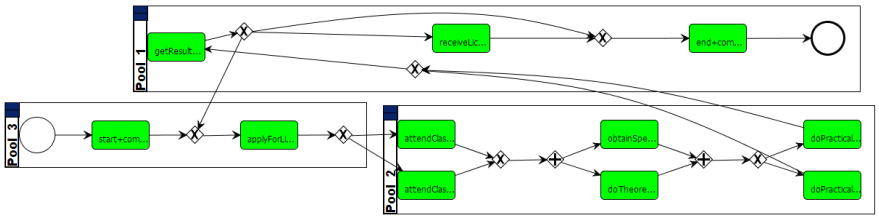
The final functional element of BPMN Miner we wish to emphasize is the ability to export and convert the discovered BPMN models. For exporting, we make use of ProM's built in exporting functionality to enable end-users to save discovered models to XPDL (XML Process Definition Language), which can then be read in by most existing modeling tools (ARIS, Bizagi, Signavio, Activiti, etc.). To accommodate the needs of researchers and scholars, functionality is also provided to convert the discovered BPMN models to Petri nets.

5 Illustrating Example

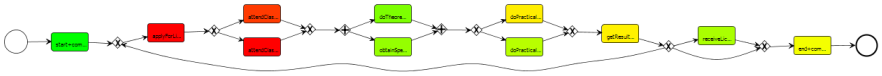
This section illustrates the developed BPMN Miner by means of a concise, fictitious example. To do so, we utilize the *driversLicenseLoop* event log (a commonly used log in benchmarking setups, see [25]), and annotate this event log with originator information. Fig. 1 depicts a number of screen captions illustrating the features of the BPMN Miner plugin. Fig. 1(a) shows the result of mining the BPMN model from the *driversLicenseLoop* event log without creating swimlanes or performing conformance analysis. This model can be exported to XPDL and imported in third-party tools or converted in ProM to a Petri net and used for further analysis. Fig. 1(b) shows the discovered BPMN model with the bidimensional discovery and conformance analysis enabled. As shown, the model has a perfect fitness level (all activities green). Each swimlane represents a pool of originators responsible for the activities contained within the swimlane. Finally, Fig. 1(c) shows the result of a conformance analysis of the mined BPMN diagram (without swimlanes) against the event log in which a high amount of noise was introduced and does thus not fit the discovered model. The coloring of activities (green to red scale) indicate problematic areas with a high amount of deviations.



(a) Mined BPMN model without swimlanes or conformance visualization.



(b) Mined BPMN model with swimlanes and conformance visualization.



(c) Conformance analysis of mined BPMN diagram against noisy event log.

Fig. 1. Screen captions illustrating discovery features of BPMN Miner.

6 Conclusions

This paper describes BPMN Miner, a new process discovery technique that directly applies BPMN as its representational language for discovered models. The technique also provides functionality to discover resource-based swimlanes, thus combining control-flow and resource information in one easily interpretable, bidimensional model. Moreover, BPMN Miner provides support for filtering and abstraction so as to deal with complex event logs presenting a wide variety of behavior. Finally, it will provide the user with immediate feedback regarding the conformance between the input event log and the discovered model through an event-granular fitness metric and deviation visualization.

It is argued that BPMN Miner will lower the adoption barrier of process mining for practitioners, education, and researchers. In future work, we aim at further extending the set of supported BPMN constructs so as to ultimately develop a technique that can produce "rich" discovered process models, for instance with exception handling, sub-processes, data flows (discovery of data artifacts), etc., or the discovery of external process participants based on interactions with the operating environment (this can be represented as collapsed pools with message flows). In addition, the discovery of hierarchical process structures with different invocation methods (subprocesses, call activities etc.) is also put forward as future work.

Acknowledgements

We would like to thank the KU Leuven research council for financial support under grant OT/10/010.

References

1. van der Aalst, W.M.P.: Process Mining - Discovery, Conformance and Enhancement of Business Processes. Springer (2011)
2. Murata, T.: Petri nets: Properties, analysis and applications. Proceedings of the IEEE **77**(4) (1989) 541–580
3. Weijters, A.J.M.M., van der Aalst, W.M.P.: Rediscovering workflow models from event-based data using little thumb. Integrated Computer-Aided Engineering **10**(2) (2003) 151–162
4. van der Aalst, W.M.P., Adriansyah, A., van Dongen, B.F.: Causal nets: A modeling language tailored towards process discovery. In Katoen, J.P., König, B., eds.: CONCUR. Volume 6901 of Lecture Notes in Computer Science., Springer (2011) 28–42
5. Scheer, A.W., Thomas, O., Adam, O.: Process modeling using event-driven process chains. In Dumas, M., van der Aalst, W.M.P., ter Hofstede, A.H.M., eds.: Process-Aware Information Systems: Bridging People and Software Through Process Technology. Wiley (2005)

6. Object Management Group (OMG): Business Process Model and Notation (BPMN) Version 2.0. OMG Document – formal/2011-01-03 (2011)
7. Object Management Group (OMG): Business Process Model and Notation (BPMN) Version 1.2. OMG Document – formal/2009-01-03 (2009)
8. Dijkman, R.M.: Choreography-based design of business collaborations. Technical report, Eindhoven University of Technology (2006)
9. Dijkman, R.M., Dumas, M., Ouyang, C.: Formal semantics and automated analysis of bpmn process models. Technical report, Queensland University of Technology (2007)
10. Wong, P.Y.H., Gibbons, J.: A process semantics for bpmn. In Liu, S., Maibaum, T.S.E., Araki, K., eds.: ICFEM. Volume 5256 of Lecture Notes in Computer Science., Springer (2008) 355–374
11. Lam, V.S.: A precise execution semantics for bpmn. IAENG International Journal of Computer Science **39**(1) (2012)
12. Dijkman, R.M., Gorp, P.V.: Bpmn 2.0 execution semantics formalized as graph rewrite rules. In Mendling, J., Weidlich, M., Weske, M., eds.: BPMN. Volume 67 of Lecture Notes in Business Information Processing., Springer (2010) 16–30
13. zur Muehlen, M., Recker, J.: How much language is enough? theoretical and practical use of the business process modeling notation. In Jr., J.A.B., Krogstie, J., Pastor, O., Pernici, B., Rolland, C., Sølvberg, A., eds.: Seminal Contributions to Information Systems Engineering. Springer (2013) 429–443
14. Recker, J.: Opportunities and constraints: the current struggle with bpmn. Business Proc. Manag. Journal **16**(1) (2010) 181–201
15. Recker, J.C.: BPMN modeling–who, where, how and why. BPTrends **5**(3) (2008) 1–8
16. Chinosi, M., Trombetta, A.: Bpmn: An introduction to the standard. Computer Standards & Interfaces **34**(1) (2012) 124–134
17. Claes, J., Poels, G.: Process mining and the prom framework: An exploratory survey. In Rosa, M.L., Soffer, P., eds.: Business Process Management Workshops. Volume 132 of LNBIP., Springer (2012) 187–198
18. Dumas, M., Rosa, M.L., Mendling, J., Reijers, H.A.: Fundamentals of Business Process Management. Springer (2013)
19. Rozman, T., Horvat, R.V., Rozman, I.: Modeling the standard compliant software processes in the university environment. Business Process Management Journal **14**(1) (2008) 53–64
20. van der Aalst, W.M.P.: On the representational bias in process mining. In Reddy, S., Tata, S., eds.: WETICE, IEEE Computer Society (2011) 2–7
21. van der Aalst et al.: Process mining manifesto. In Daniel, F., Barkaoui, K., Dustdar, S., eds.: Business Process Management Workshops. Volume 99 of Lecture Notes in Business Information Processing. Springer Berlin Heidelberg (2012) 169–194
22. Fayyad, U.M., Piatetsky-Shapiro, G., Smyth, P.: Knowledge discovery and data mining: Towards a unifying framework. In: KDD. (1996) 82–88
23. Weijters, A.J.M.M., van der Aalst, W.M.P., Alves de Medeiros, A.K.: Process mining with the heuristicsminer algorithm. BETA working paper series 166, TU Eindhoven (2006)
24. van der Aalst, W.M.P., Reijers, H.A., Song, M.: Discovering social networks from event logs. Computer Supported Cooperative Work **14**(6) (2005) 549–593
25. Alves de Medeiros, A., Weijters, A., van der Aalst, W.: Genetic process mining: an experimental evaluation. Data Min. Knowl. Discov. **14**(2) (2007) 245–304