Streamlining the education of Declarative Process Management with Process DoCtoR*

Konstantinos Varvoutas[1], Julian Neuberger[1] and Hugo A. López[1]

[1]Technical University of Denmark, Denmark

Abstract

This paper presents Process DoCtoR, an enhanced version of DCR-JS, a browser-based tool for DCR graphs, aimed at supporting novice users in common declarative process management tasks. Declarative models are highly flexible, but their adoption remains limited, as their understandability has been shown to be challenging, especially for novice users. Process DoCtoR enables users to kickstart modeling by extracting initial models from natural-language descriptions or BPMN models, extends the language with multi-perspective constraints over time and data, and improves the performance of discovery and conformance checking by up to 96.9 % in terms of duration and up to 97.3 % in terms of peak memory usage, enabling large real-world logs to be mined directly in the browser.

Keywords

Declarative Processes, DCR Graphs, Multi-Perspective Processes, Conformance Checking, Process Discovery

1. Introduction

For more than twenty years, declarative process modeling notations such as Declare [1] and the Dynamic Condition Response (DCR) graphs [2] have offered an alternative view to the orchestration of highly complex business processes. Unlike imperative process modeling notations, such as BPMN, declarative modeling does not prescribe a typical sequence of tasks, enabling support for complex, highly flexible processes with a high degree of decision-making flexibility. With its strong links to Linear Temporal Logic on Finite traces, Declare has consolidated its position as a research language. DCR graphs are, to the best of our knowledge, the only declarative notation that has been successful in generating uptake in industry, particularly in public sector digitalization [3]. Compared to Declare, DCR graphs' underlying semantics describe regular and ω-regular languages; it has a smaller yet powerful set of constraints (5 vs 18 in Declare), and captures *defeasible constraints* via inclusion and exclusion rules, not directly encodable in Declare. According to Keramidis [3], some perceived advantages of the declarative paradigm include simplification and automation of processes, and easier process agency for domain specialists.

Although declarative process modeling offers significant advantages, its adoption remains limited compared to imperative approaches. This can be attributed, in part, to the understandability of declarative models, which is challenging, particularly for novice users, who often find declarative specifications less intuitive than imperative process descriptions [4]. Second, organizations have made substantial investments in imperative modeling languages such as BPMN, resulting in extensive repositories of process models. Finally, there is a lack of robust, open-source, and easy-to-access tools supporting declarative modeling at different stages of the process lifecycle [5]. Together, these factors create a considerable barrier to the broader adoption of declarative approaches.

DCR-JS [6] was introduced as a lightweight, browser-based environment that aims to lower some of the practical barriers associated with declarative process modeling. In this work, we extend DCR-JS into Process DoCtoR. Apart from the name change, Process DoCtoR adds new features to support

Proceedings of the BPM 2026 Best Dissertation Award, Doctoral Consortium, and Demonstration Resources Forum (BPM 2026), Toronto, Canada, September 27th to October 2nd, 2026.

*Process DoCtoR

konva@dtu.dk (K. Varvoutas); jtone@dtu.dk (J. Neuberger); hulo@dtu.dk (H. A. López)

0009-0005-7272-9014 (K. Varvoutas); 0009-0008-4244-7659 (J. Neuberger); 0000-0001-5162-7936 (H. A. López)

published 2026-09-28

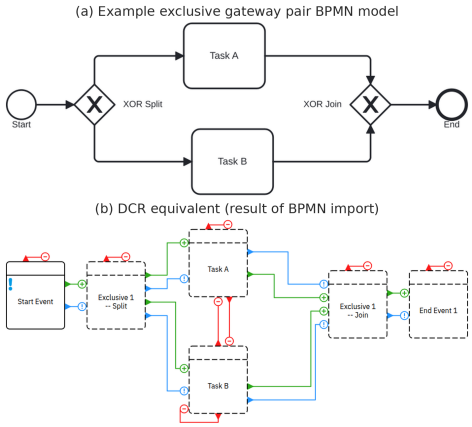

Figure 1: BPMN Import

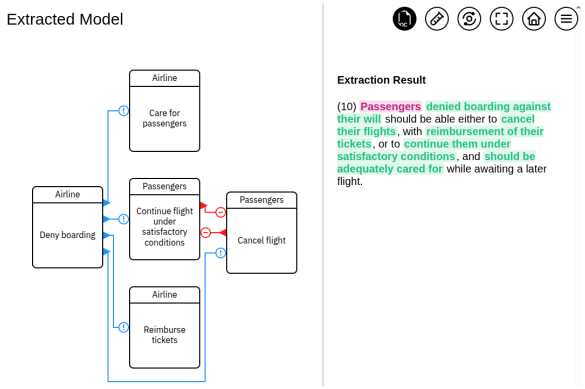

Figure 2: Elicitation

accessibility, expressiveness, and efficiency for declarative modeling and mining. First, it enables novices to start their first models via text-to-process pipelines, or imports from BPMN; second, it increased expressiveness to supporting multi-perspective DCR constraints; finally, the performance of process discovery and conformance checking is significantly enhanced, which is demonstrated through an evaluation on a large collection of publicly available event logs [7].

Structure. Section 2 gives a brief introduction to the syntax and semantics of DCR graphs and illustrates the main innovations of the tool. Section 3 discusses tool maturity and availability. Section 4 concludes.

2. Declarative Process Management in Process DoCtoR

A DCR graph can be seen as a collection of events linked by relations. Each event carries a marking denoting whether it has been executed, is pending, or has been excluded. Relations impose constraints that define the characteristics of accepting traces. A condition relation defines precedence, while a response relation defines an obligatory consequence. An exclusion relation removes an event from the active process, which can later be reversed via an inclusion relation. Process DoCtoR supports MP-DCR graphs, an extension of DCR that includes timing and data constraints; for its formal definition, please refer to [8].

Process DoCtoR extends previous versions of DCR-JS [6], supporting all five existing modules (modeling, simulation, discovery, conformance checking and log generation) and introducing extensions to four of them while preserving the tool's architecture.

Modeling. The modeler was extended to support time and data annotations. Clicking on an event allows the user to associate it with a variable specifying its name and type. Supported variable types are integer, boolean, and string, each of which can be given a default value. Additionally, relations now support both timed and data-aware annotations. Expressions for guards are written in a subset of FEEL expressions[1]. Constraints can be annotated with time to express delays and deadlines. Both can be specified as ISO 8601 duration strings, e.g., P30D for 30 days or PT2H for 2 hours.

Elicitation. Process DoCtoR supports a kickstart in the elicitation of DCR models via LLM-generated models. Our prompting strategy is based on Neuberger et al. [9], adjusted for DCR following Lindner et al. [10]. As empirically evaluated in [10], LLM models should be used as *draft* models, and some aspects (e.g., activity duplicates, relation extractions) should be revised by the user. Compared with [10], we extend elicitation with prompts for extracting data variables, data guards, and deadlines. We expose the definition of process-relevant information to the user, allowing them to adapt it to their specific use cases. During extraction, we discard all information that is not grounded in the source text, and highlight the remaining information, as shown in Fig. 2, which presents an example of an elicited

[1]The *Friendly Enough Expression Language* is part of *Decision Model and Notation*, see https://www.omg.org/spec/DMN, last accessed June 23, 2026.

model after manual revision. Currently, only OpenAI-hosted models are supported, with additional providers planned for future releases.

BPMN Import. Process DoCtoR is a tool supporting the initiation of process modelers in declarative specifications, and many of our intended users have prior knowledge in process modeling standards such as BPMN. To facilitate the transition between paradigms, Process DoCtoR supports importing sound BPMN models and converting them to DCR graphs according to the encoding [11]. A subset of BPMN comprising start and end events, tasks, XOR and AND gateways, and recursion is supported. Each gateway split/join pair must follow the single-entry single-exit (SESE) principle. The intention with the encodings is to show how many of the BPMN constructs result in overly rigid declarative models that can be flexibilized with Process DoCtoR. An imported example model is presented in Fig. 1.

Process Discovery. Process DoCtoR discovers DCR graphs from XES event logs entirely in the browser, offering data privacy at the expense of single-thread and memory limits. The parsing and pre-processing stages around the discovery algorithm were re-engineered, leaving the algorithm itself unchanged: logs are streamed incrementally instead of loaded in whole, and equivalent traces are collapsed [12].

Simulation.

Process DoCtoR supports step-based simulation of DCR graphs. We extended this simulation engine with time and data perspectives of processes. The extended simulation interface is shown in Fig. 3. Clicking on an enabled event to execute it, when a variable is associated with it, a pop-up will prompt the user to input a value. In our running example, this is the case for event *Diagnose Medicine*, and its associated variable *Diagnosis*. Additionally, a control panel at the bottom of the simulator shows the current date and time, and allows forwarding time by a given amount. Events with a deadline or delay have this information shown at the bottom, like event *Buy Medicine* in our example (Fig. 3).

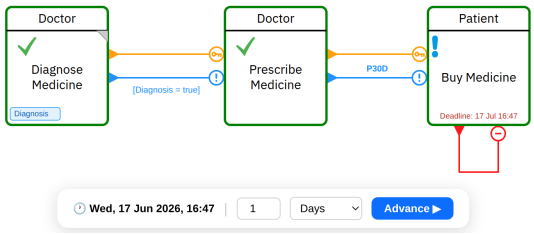

Figure 3: Time- and data-aware simulation of a Medicine Prescription process.

Conformance Checking. Previous conformance checking capabilities were extended with support for data guards and timed constraints, and required novel definitions for partial compliance [8]. Additionally, it was re-engineered following the same trace-variant aggregation principle applied to discovery [12]. This aggregation is applied conditionally: when the model contains no guards, variables, or time constraints, checking runs in control-flow mode, collapsing traces into variants, so that each unique execution pattern is replayed only once. When the model does contain guards, variables, or time constraints, checking instead runs in data-aware mode: traces are still visually grouped by their control-flow signature, but only for display purposes, each trace is replayed individually. Each variant group is displayed with a green check (conforming), red cross (violating) or orange percentage-cross (partially violating). In data-aware mode, expanding a variant group reveals the conformance status of its individual traces. Additionally, for each trace, the number of control-flow and temporal violations is shown, represented by a red broken-chain and a yellow clock, respectively (guarded constraints are covered by control-flow violations as they only affect whether a constraint applies). Selecting a trace shows its events and model constraints, color coded by conformance results.

3. Maturity and Availability

Maturity. Process DoCtoR is an iteration of DCR-JS [6], evolved over the years through re-engineering efforts and software engineering projects at the Technical University of Denmark and the University of Copenhagen. It is currently used in two M.Sc. courses at DTU, with more than 100 students per course using it for modeling and analysis, while some extend it in their M.Sc. theses. Notably, the extensions presented in this work are recent research developments [10, 11, 8], while the performance gains are a result of a performance engineering M.Sc. thesis [12]. Our tests ensure that the behavior corresponds to

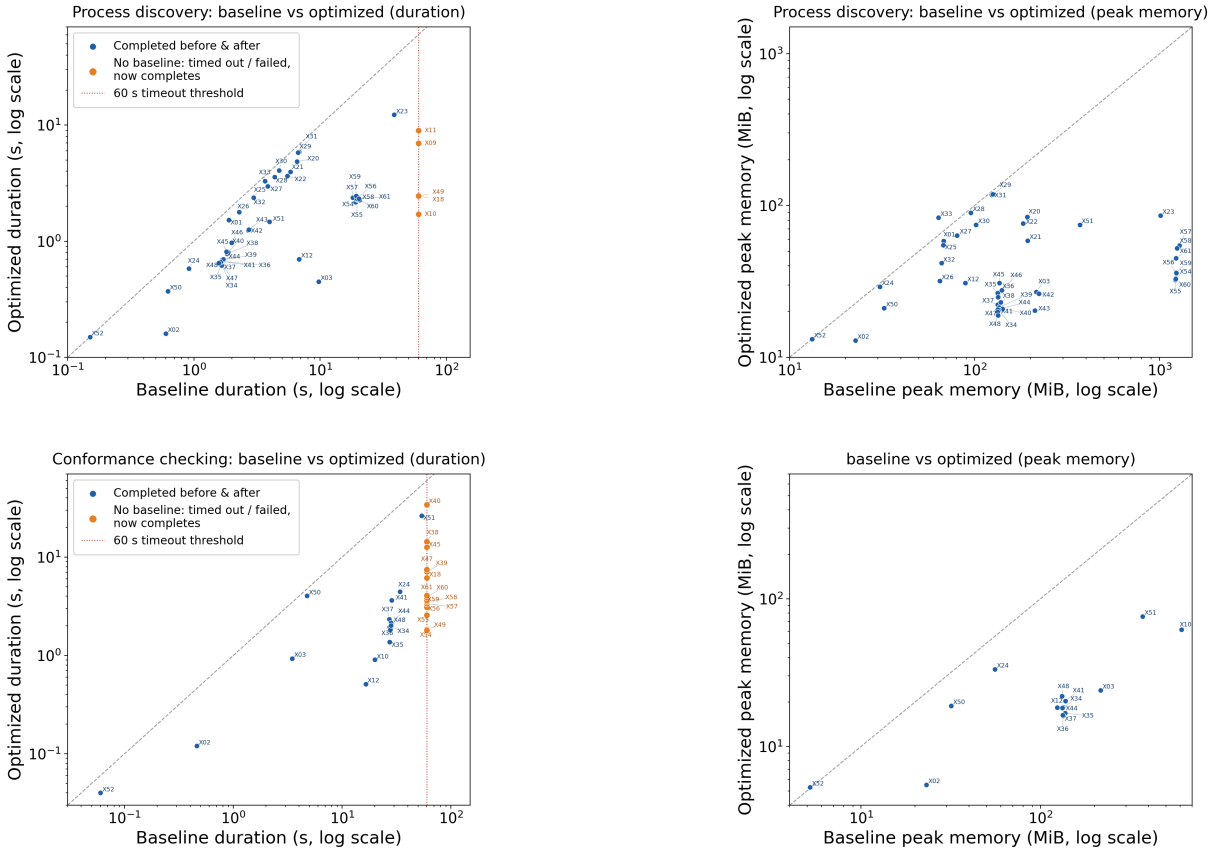

Figure 4: Process discovery and conformance checking across the evaluated event logs, baseline vs. optimized. Points below the diagonal indicate improvement.

the semantics of multi-perspective DCR graphs [8], and is compatible with commercial DCR offerings.

Scalability. Originally, the tool was developed with an academic scope in mind, and usability requirements took precedence over scalability. This iteration required re-engineering the parsing and pre-processing stages around discovery and conformance checking algorithms to improve its performance and support larger logs. Figure 4 reports duration and peak memory before and after these optimizations. Evaluated against a recent collection of 61 publicly available event logs [2] in the process mining community [7], the re-engineering yields geometric-mean reductions, for process discovery, of 62% in duration and 79% in peak memory, and increased the number of logs that complete discovery from 44 to 49. Five large logs that previously exhausted browser memory or timed out, most notably BPIC'19 (over 700 MB), can now be mined entirely in the browser. One limitation is that trace aggregation is only beneficial for logs with repeated behavior, but its overhead is negligible, and logs of near-unique traces still benefit from the re-engineered parsing stage. Significantly, log parsing is no longer the dominant bottleneck. This iteration identifies graph layout as the limiting factor for twelve logs whose resulting models contain large numbers of nodes and edges (approaching 10,000), which, arguably, will be hard to understand by human users due to their size. Conformance checking, evaluated on the logs for which discovery completed, shows comparable gains. It achieves geometric-mean reductions of 87% in duration and 80% in peak memory, doubling the number of completing logs from 15 to 30. The highest individual gains observed were: 95.3% in discovery duration (X03), 97.3% in discovery memory (X54/X55/X60), 96.9% in conformance duration (X12), and 89.9% in conformance memory (X10).

Availability. Process DoCtoR can be used directly from a browser at https:/processdoctor.compute.dtu.dk/. All code is public at https://github.com/hugoalopez-dtu/dcr-js. A short demonstration video can be

[2]The identifiers X01-X61 correspond to https://github.com/hugoalopez-dtu/dcr-js/raw/main/test_logs/event_logs.xlsx

found at https://www.youtube.com/watch?v=5Y-5Y0zOJPg.

4. Conclusion

We present Process DoCtoR as a one-stop tool for the introduction to declarative BPM. With our performance improvements and extended language expressiveness, and elicitation from natural language and BPMN models, we aim at providing an educational tool where novice modelers can quickly start experimenting with the benefits of DBPM in multiple phases of the BPM lifecycle. In future work, we plan to further develop intuitive and novice-friendly ways of interacting with Process DoCtoR, especially conversational interfaces, multi-instance simulation, and visualization of large process models.

Acknowledgments

This work has been supported by the grant "Center for Digital CompliancE (DICE)" (VIL57420) from VILLUM FONDEN, and by the Innovation Foundation project "Explainable Hybrid-AI for Computational Law and Accurate Legal Chatbots" 4355-00018B XHAILe, and Predictive and Prescriptive Process Analytics for Industry 4.0 (P3AI4, grant 4105-00045B)

Declaration on Generative AI

The author(s) have not employed any Generative AI tools.

References

[1] M. Pesic, W. M. Van der Aalst, A declarative approach for flexible business processes management, in: BPM, Springer, 2006, pp. 169–180.
[2] T. Hildebrandt, R. R. Mukkamala, Declarative event-based workflow as distributed dynamic condition response graphs, EPTCS 69 (2011) 59–73.
[3] P. Keramidis, T. T. Hildebrandt, On the business value of declarative business process management, in: HICSS 2026, 2026, pp. 5631–5640.
[4] D. Fahland, D. Lübke, J. Mendling, H. Reijers, B. Weber, M. Weidlich, S. Zugal, Declarative versus imperative process modeling languages: The issue of understandability, in: BPMDS, 2009.
[5] H. A. López, C. di Ciccio, T. Hildebrandt, J. D. Smedt, Teaching declarative BPM: Where, when and to whom it matters, in: Business Process Management: Responsible BPM Forum, Process Technology Forum, Educators Forum, Springer, 2026, p. to appear.
[6] A. K. Christfort, H. A. López, DCR-JS: An Online Environment for Declarative Process Mining, in: BPM, CEUR-WS, 2025, pp. 256–263.
[7] A. Costa, S. Y. Eroglu, K. Andree, L. Pufahl, A collection of publicly available event logs enhanced by metadata, volume 4032, 2025, p. 288 – 295.
[8] K. Varvoutas, J. Neuberger, H. A. López, A neuro-symbolic framework for compliance management with declarative models, in: CAISE workshops, Springer, 2026, pp. 277–290.
[9] J. Neuberger, L. Ackermann, H. van der Aa, S. Jablonski, A universal prompting strategy for extracting process model information from natural language text using large language models, in: ER, 2024.
[10] J. Lindner, H. A. López, Discovering declarative processes in textual descriptions using large language models, in: GenAI4PM, Springer, 2025.
[11] Y. Zhou, H. A. López, A compositional approach to mapping bpmn to dcr for cross-paradigm process modeling, in: FormaliSE 2026, ACM, United States, 2026. doi:10.1145/3793656.3793684.
[12] T. K. Blomsterberg, Efficient client-side processing for declarative process mining: Overcoming computational bottlenecks in browser-based architectures, 2026.