BPMNLearn and Grade: Enabling Automated Grading of BPMN Models

Erik Tolboom[1], Karolin Winter[1,*] and Laura Genga[1]

[1]Eindhoven University of Technology, Groene Loper 3, 5612 AE Eindhoven, The Netherlands

Abstract

Teaching and grading process modeling exercises is a core task in business process management education which, however, comes with several challenges including the identification of both syntactic and semantic errors. Existing modeling tools mostly support syntactic error checking, while semantic error detection remains a manual, time-consuming and error-prone task for teachers. This demo paper addresses this challenge by presenting BPMNLearn&Grade. It enables teachers to model rubric criteria through an interactive interface and automatically check criteria fulfillment of student models. The tool was tested for assignments of courses taught at Eindhoven University of Technology.

Keywords

Process Modeling, BPMN Model Grading, Business Process Management Education

1. Introduction and Problem Statement

In business process management education, teaching process modeling constitutes a core task [1] that can be challenging for both students and teachers [2]. Students need to master the process modeling language with its syntactic rules while at the same time create a good understanding of the semantics of the process that should be modeled. Oftentimes, these processes are described in natural language, which can lead to multiple valid models depending on the interpretation. On the other hand, grading students' models is a time-consuming and error-prone task, mostly performed manually by the teachers. To structure the grading process, ensuring transparency and as much objectivity as possible, teachers usually employ *rubrics* [3]. Those are a coherent set of criteria with performance-level descriptions, e.g., which modeling choices are considered more or less appropriate for a given assignment (and, hence, should receive higher or lower points). When rubrics are employed, the grading process entails manually assessing the process model submitted by the student against the specified criteria.

Using existing tools for such a match is, to the best of our knowledge, not feasible because only syntactic but not semantic checks are supported [4]. The latter would, however, be required to realize automated grading of process modeling exercises.

To address this challenge, this demo paper presents BPMNLearn&Grade, a tool that allows for modeling rubric criteria for Business Process Model and Notation (BPMN) exercises together with possible variants of rubrics, and automatically checking their fulfillment given a student model. The tool was developed and tested in cooperation with lecturers at Eindhoven University of Technology in the context of a Bachelor and a Master course.

2. Innovations and Functionalities of BPMNLearn&Grade

In recent years, teaching process modeling in higher education has attracted increasing research interest. For example, [2] present a teaching module based on, i.a., formative assessment for novice learners of BPMN modeling while [5] presents a chatbot for interactively teaching Declare modeling. Similarly, the Model Judge framework [6] offers interactive modeling from a given text and feedback functionalities.

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

*Corresponding author.

e.s.tolboom@student.tue.nl (E. Tolboom); k.m.winter@tue.nl (K. Winter); l.genga@tue.nl (L. Genga)

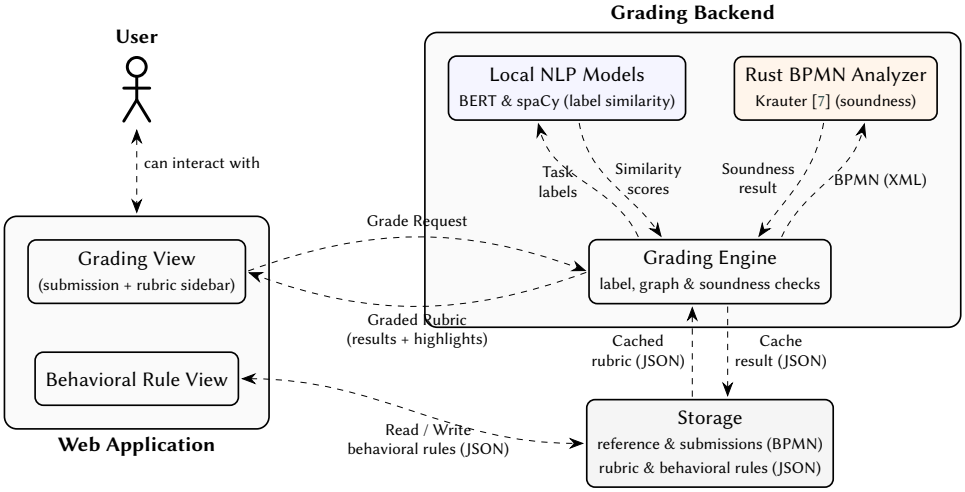

Figure 1: High-Level Overview of the BPMNLearn&Grade Architecture

However, these approaches mainly consider the learning perspective, while BPMNLearn&Grade emphasizes the grading perspective from the teacher's point of view. The tool supports the task of having an assignment that asks students to develop a BPMN model based on a textual description of a process.

In the following we detail the architecture and functionalities of BPMNLearn&Grade. The tool is available at GitHub[1], and a screencast demonstrating the main functionalities is accessible on YouTube[2].

2.1. Architecture

BPMNLearn& Grade is implemented in Python and integrates the Rust BPMN Analyzer taken from [7]. Python was chosen for BPMNLearn&Grade because of its rich ecosystem of libraries and extensibility given its interpreted nature. Figure 1 illustrates the architecture of BPMNLearn&Grade consisting of three components. The *web application* serves as the user interface and consists of the grading and behavioral rule view. The *grading backend* contains the grading engine performing model checks such as task label similarity calculations using local NLP models and soundness checks using [7]. A *storage component* manages data access and retrieval.

2.2. Implemented Functionalities

The tool has two main views: the grading and the behavioral rule view.

Grading View. Once the tool is launched, the teacher sees the Grading View (Figure 2). The view shows the reference or the submitted model on the left, and the list of rubric items to be graded on the right. In addition, teachers can upload a supplement containing for example the assignment description for easy lookup. For each rubric item, it is shown whether it was fulfilled or not, and an overall score reflecting the achieved points is calculated. We also implemented features related to giving feedback, in particular, on each rubric criterion two types of notes can be added. First, for giving students feedback and, second, for giving internal feedback. Internal feedback here is meant to signal to other graders that, for instance, a lower matching threshold was used. This helps ensure inter-rater consistency. As can be seen, the rubric items are divided into simple checks and behavioral checks. Examples of simple checks are those focusing on syntactic control-flow checks such as deadlocks, dead activities, and proper synchronization which are checked using [7]. Various checks related to model quality can be performed as well. For example, checking for label atomicity, exact duplicate tasks, and semantically duplicate tasks. Furthermore, sanity checks are implemented. Those serve to ensure that the student

[1] https://github.com/ETolboom/blg
[2] https://youtu.be/9REmiOhD4uY

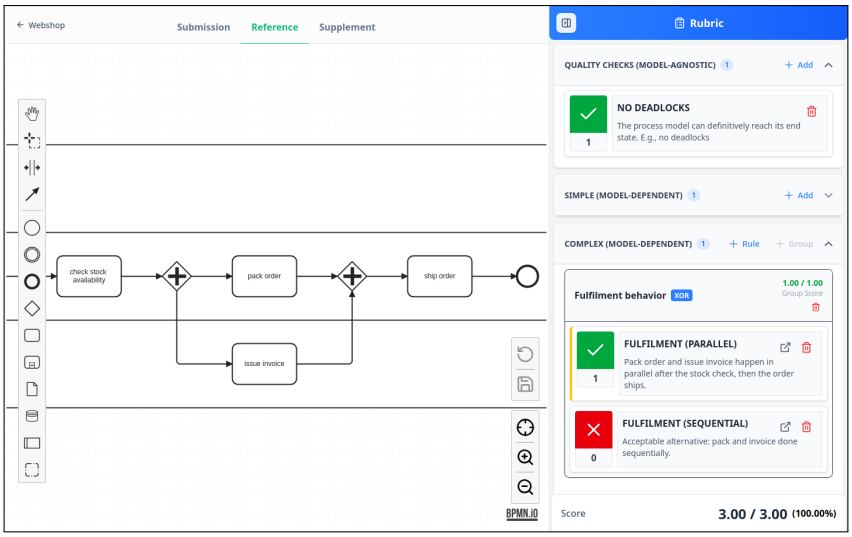

Figure 2: Example of the Grading View

submission at least somehow reflects what should be modeled according to the assignment description. These include, for example, determining correct labels for pools and their lanes as well as ensuring all required tasks are actually modeled.

Behavioral Rule View. Complex items correspond to more complex rules, that are defined in the Behavioral Rule View (Figure 3). Formulating a rule means combining two or more "Element Check" or "Gateway Check" items by means of the provided connectors. An "Element Check" allows to specify the type (task, event) and the label, while a "Gateway Check" asks to specify whether it is a parallel or choice gateway and whether properties of the gateway itself should be part of the items to check for the grading, such as its outcomes. The connectors specify the ordering relation requested among the connected elements, i.e., parallel, concurrent, or sequential. The portion of rule shown in Figure 3 represents a rubric item checking whether there exists a parallel gateway connected to the tasks "issue invoice" and "pack order", and whether these tasks are modeled as concurrent. Although it may seem like a trivial check, it is worth noting that industry-standard tools such as Camunda or Signavio do not support these behavioral checks, which are routine rubric items in BPM courses. The behavioral rules support the teacher in isolating and checking a specific portion of the process model. Evaluation of such behavioral rules at run-time works as follows: First, the first node of the rule is matched with the closest matching element in the BPMN model.

Then, based on the settings in the followed by connector, the next node in the rule is matched. The `followed by` connector in the example specifies the ideal distance and max distance between the current BPMN element and the next. This was done to avoid both matching an arbitrary element as well as backtracking. This procedure repeats until either all nodes in the behavioral rule are exhausted and the end node is reached or if the rule exits early in case it cannot find a matching BPMN element. It is worth noting that multiple behavioral rules can be combined into a single criterion called a behavioral rule group. These make it possible for a teacher to specify multiple alternative solutions possibly worth less points than the original intended solution. As can be seen in Figure 3, the teacher might define a rule to assign only half a point if the relation between "pack order" and "issue invoice" is sequential.

Given that student submissions may differ greatly, various options for leniency are provided. First, the `followed by` connector allows the instructor to determine the allowed distance between elements. This may for example be relevant when extra gateways are inserted to prevent implicit merging. Second, gateways may not always be labeled or have labeled outcomes such as with parallel gateways, as such these are optional toggles. Lastly, activity labels may not be identical due to wording differences or typos. Therefore, labels are matched semantically using a locally run BERT model.[3]

[3]https://huggingface.co/sentence-transformers/all-mpnet-base-v2

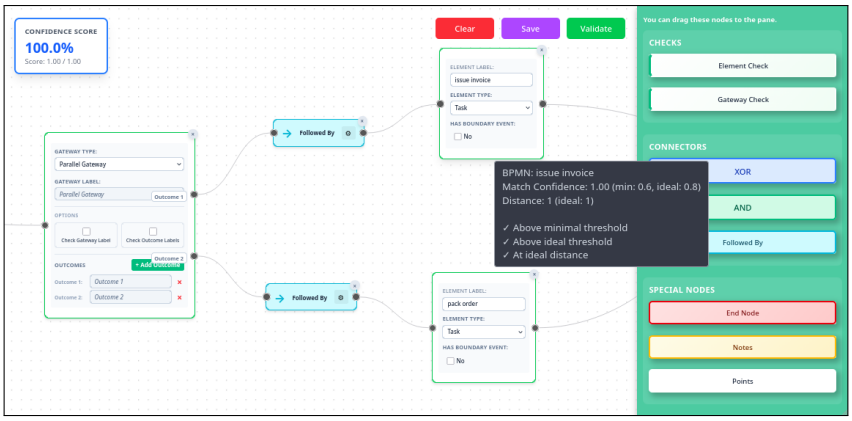

Figure 3: Example of the Behavioral Rule View

3. Maturity

The tool was developed in cooperation with seven teachers at Eindhoven University of Technology. All of them have several years of experience teaching BPMN modeling as well as other modeling languages. We gathered feedback through three different means.

First, during the initial development stage, we conducted interviews with two lecturers in order to collect pain points of lecturers and common student mistakes. This helped us to gather an overview on the functionalities that should be supported. One recurring theme that came up was about the role of the tool: "I would rely upon it like a student assistant. I would still double check the output it gives me. Therefore it's very important that it provides detailed scoring: which rubric criteria are scored, and why, based on specific elements of a student's solution. That traceability, or transparency as you call it, is very important". In its essence, rubrics aim to provide a fair way to grade a submission. Therefore transparency became one of the core design pillars of BPMNLearn&Grade. Similarly, when asked about what was most time consuming it was said that "sometimes the models are just way too far off that it just doesn't make any sense". Further suggesting that "we will need to have some global check first to see: is this even automatically gradable or is this just too far off?". Based on this feedback extra emphasis was placed on implementing quality checks such as task coverage, task type checking, as well as semantic duplicate task checking.

Secondly, we organized a feedback session with other lecturers in which the tool and its functionalities were demonstrated. This helped us to get a broader view and feedback on how we implemented the functionalities as mentioned in the interviews earlier.

Third, after we had a complete version of the tool, we tested it on student submissions from the "Fundamentals of Business Information Systems", and on the reference model of an exam for the "Business Process Management" course. The former is a large Bachelor course where basic concepts of BPMN modeling are taught. The latter is a Master course which focuses on advanced BPMN modeling concepts. This in-depth testing helped us to further improve the tool. For example, it was revealed that initial versions lacked the ability to clearly discern whether pools are missing or just mislabeled. Based on this observation, coverage details were added to make it transparent how the result is obtained. Those are now displayed by, for example, hovering over relevant items.

The current version of BPMNLearn&Grade has some limitations, in particular due to the dependency on the `rust_bpmn_analyzer`. For instance, the library does not allow considering some elements of the BPMN standard, such as message start events, boundary events and timer events. Consequently, the current version of the tool does not support rubric items involving these elements.

4. Conclusion

In this paper, we introduced BPMNLearn&Grade, a tool aimed at supporting the automated checking of rubric criteria employed for courses teaching BPMN modeling. The tool allows teachers to define rubric criteria of different complexity, covering the most commonly used BPM constructs, and supports the definition of alternative criteria, possibly leading to higher or lower points. Interviews and feedback sessions with the involved teachers confirmed the tool's potential in supporting the grading process. We foresee several avenues for future work. For example, extending the supported BPMN modeling elements, e.g., adding timer events with proper semantic parsing as well as result import functionalities for learning management systems such as Moodle, Canvas, and Brightpsace. In addition, exploring further Large Language Model grading capabilities for BPMN exercises as it was done for UML [8], constitutes an interesting future research direction.

Acknowledgments

This research has been funded by the Department of Industrial Engineering and Innovation Sciences through the education innovation fund. We would like to thank all lecturers that contributed to this work through interviews and feedback sessions.

Declaration on Generative AI

During the preparation of this work, the authors used Claude Code to create an initial version of the TikZ code of Fig. 1 and Writefull for grammar and spelling checks. After using these tools, the authors reviewed and edited the content as needed and take full responsibility for the publication's content.

References

[1] M. R. B. Nobre, J. Vilela, Content and skills for teaching BPM in computer science courses: A systematic mapping study, in: Proceedings of the 16th International Conference on Computer Supported Education, CSEDU, SCITEPRESS, 2024, pp. 373–384.

[2] P. Vemuri, S. Poelmans, I. Compagnucci, M. Snoeck, Using formative assessment and feedback to train novice modelers in business process modeling, in: ACM/IEEE International Conference on Model Driven Engineering Languages and Systems, MODELS, IEEE, 2023, pp. 130–137.

[3] Y. M. Reddy, H. Andrade, A review of rubric use in higher education, Assessment & evaluation in higher education 35 (2010) 435–448.

[4] C. Haisjackl, P. Soffer, S. Y. Lim, B. Weber, How do humans inspect BPMN models: an exploratory study, Softw. Syst. Model. 17 (2018) 655–673.

[5] S. Nagel, A. Wolters, D. M. Riehle, P. Delfmann, Educlare - an intelligent tutoring chatbot for teaching declarative process modeling, in: Doctoral Consortium and Demo Track 2024 at the International Conference on Process Mining, CEUR Workshop Proceedings, CEUR-WS.org, 2024.

[6] J. Sànchez-Ferreres, L. Delicado, A. A. Andaloussi, A. Burattin, G. Calderón-Ruiz, B. Weber, J. Carmona, L. Padró, Supporting the process of learning and teaching process models, IEEE Trans. Learn. Technol. 13 (2020) 552–566.

[7] T. Kräuter, P. Stünkel, A. Rutle, Y. Lamo, H. König, BPMN analyzer 2.0: Instantaneous, comprehensible, and fixable control flow analysis for realistic BPMN models, in: Proceedings of the Best Dissertation Award, Doctoral Consortium, and Demonstration & Resources Forum at BPM, CEUR Workshop Proceedings, CEUR-WS.org, 2024, pp. 66–70.

[8] N. Bouali, M. Gerhold, T. U. Rehman, F. Ahmed, Toward automated UML diagram assessment: Comparing llm-generated scores with teaching assistants, in: Proceedings of the 17th International Conference on Computer Supported Education, CSEDU, SCITEPRESS, 2025, pp. 158–169.