Efficient LLM-Based Conversational Process Modeling

Julius Köpke[1] and Aya Safan[1]

University of Klagenfurt, Department of Informatics Systems, Klagenfurt, Austria,
julius.koepke@aau.at, aya.safan@aau.at
WWW home page: https://www.aau.at/en/isys/ics/

Abstract. Since the introduction of OpenAI's ChatGPT, the underlying technology of generative AI and large language models (LLMs) has gained tremendous interest in academia. Researchers began experimenting with LLMs' capabilities in various domains, including business process modeling. While these works indicate a promising potential of LLMs for this task, they do not consider the number of tokens of the prompting strategies and output formats. However, the token amount is such tools' number one cost driver. In addition, an efficient representation of the conversation state has not been addressed so far. This paper addresses these concerns and introduces and evaluates an approach for efficient LLM-based conversational process modeling. We have implemented our approach as a publicly available online tool. In our experiments, we observed average input token reductions of 94% compared to an existing tool while maintaining even better levels of correctness. Furthermore, a user study at a public science fair indicates solid numbers for the tool's usefulness.

Key words: Large Language Models, LLM, Conversational Process Modeling

1 Introduction

Large language models (LLMs) have recently gained tremendous attention from researchers in various fields. Numerous works such as [3, 5, 6, 7] explore the potential of LLMs to transform textual descriptions to business process models. The paper [7] systematically analyzes LLMs for conversational process modeling and evaluates various prompting strategies. These works indicate promising capabilities of LLMs, particularly OpenAI's GPT-4 [11], for this task. With *ProMoAI* [9, 10], there is an online tool for conversational process modeling available. A user enters a textual description of a process; the tool then generates an initial model and presents it graphically to the user, who comments on the model in a feedback loop. While the tool's capabilities seem promising, it comes with significant costs. In initial experiments, with a 3 step process and two iterations of the feedback loop using GPT-4, we paid around 0.8 USD in OpenAI API usage fees.

We argue that such high costs can significantly hinder the broader adoption of such tools. However, existing works only focus on the quality of the generated solutions, ignoring their costs. When using the OpenAI API, the amount of tokens directly determines the monetary costs of a request. While the cost models for self-hosted LLMs may differ, the number of tokens still significantly impacts the computational costs for generating a response [12, 13].

To reduce costs, a context prompt should add minimal overhead to the user input, and the generated output format should be compact. However, the output format can also affect the quality of the generated models [7] and, thus, potentially the number of required context prompt tokens. Furthermore, even if monetary costs are not an issue, conversational modeling tools should be tuned for efficiency, considering scalability and the carbon footprint induced by energy consumption [13].

This paper introduces and evaluates a method for efficient LLM-based conversational process modeling. We present related works in Sect. 2. In Sect. 3, we identify the major cost drivers for conversational process modeling, propose an efficient approach, and introduce an optimized process meta-model for communicating processes with LLMs. Sect. 4 introduces the *BPMN-Chatbot* as an instantiation of our approach. In Sect 5, we evaluate the chatbot against *ProMoAI* [9, 10] and a prompting strategy from [7] regarding model quality and cost-efficiency. We suppose that a cost-efficient LLM-based modeling tool has the potential to make process modeling available to a broader audience, and we, therefore, additionally report on the acceptance of the tool by users of the general public at a public science fair. Finally, Sect. 6 concludes the paper.

2 Related Work

An analysis of potential applications of LLMs in the BPM domain was conducted in [15]. While there are approaches for generating process models from text using various techniques (see [14] for a survey), LLMs can potentially disrupt alternative approaches. Moreover, LLMs can handle scenarios without complete textual descriptions, allowing users to ask for a type of process, and the LLM answers based on its implicit background knowledge. Initial experiments in [3] show quite promising results on the modeling capabilities of ChatGPT (GPT-4) for conceptual modeling and business process modeling. Experiments in [5] use ChatGPT to generate imperative and declarative process models and RPA descriptions from text. In [6], the capabilities of LLMs for extracting relevant tasks from textual descriptions were evaluated. An extension of [6] is provided in [7], where the capabilities for extracting control flow from process descriptions were extensively evaluated. The work does not present a tool but assesses the outputs of various prompting strategies and output formats.

In [4], a framework for transforming textual descriptions into business processes is proposed. It uses Petri Nets as an intermediate format. However, in a second LLM step, the Petri Nets are transformed into JSON-Nets, including the graphical positioning of the nodes.

A publicly available online tool ProMoAI [1] for the LLM-based generation of process models was introduced in [9, 10]. It allows the usage of various OpenAI LLMs. It uses heavy-weight prompts to generate Python code, which, in turn, generates Petri Nets. Users can refine the models in a feedback loop.

3 Efficient LLM-Based Conversational Process Modeling

3.1 Conversational Process Modeling

We now discuss the general phases for conversational process modeling supporting interactions like ProMoAI [10]: The user first provides an input text, and the system generates an initial model. Afterward, the user may repeatedly provide feedback, and the system adopts the generated model accordingly. This leads to two distinct phases:

Generation of an initial model: In this phase, the user enters a textual description d of a process. The tool then generates a prompt $p = (d, i, fi)$, where i is an instruction prompt for model generation and fi is the instruction for the intermediate output format f. The prompt is then sent to the LLM. It generates an answer a_0 in the intermediate format f. The system, in turn, translates a_0 to an output model m_0, which is presented graphically to the user. For this step, the larger p is, the larger the number of context tokens will be. Since we cannot control the size of d, we can only reduce the sizes of i and fi. Consequently, the intermediate format f is highly relevant in two aspects: Instances should be compact to reduce the number of output tokens. At the same time, f must be easily "understandable" by the LLM to achieve high-quality models.

Feedback Loop: In the n-th iteration of the feedback loop, the user comments on the model m_{n-1} with comment c_n. A new prompt $p' = (c_n, i', fi, st)$ is created where st is the conversation state. The prompt p' is sent to the LLM, and a new model a_n in format f is returned. It is then translated to the output model m_n and presented to the user.

Since existing LLM systems are stateless [17], the state of the conversation must be part of the prompt in the feedback loop. A conversation state that includes all prior message exchanges would dramatically increase the context token size. Moreover, since LLMs have a limited context window, the conversation state cannot grow indefinitely.

3.2 Approach

For efficient LLM-based process modeling, we propose the following approach: In the initial step, we instruct the LLM to provide the answer a_0 in an optimized intermediate format. In the feedback loop, we construct the prompt as $p' = (i', f_i, c, a_{n-1})$. Therefore, the state of the conversation is solely represented by the most recent model in the intermediate format.

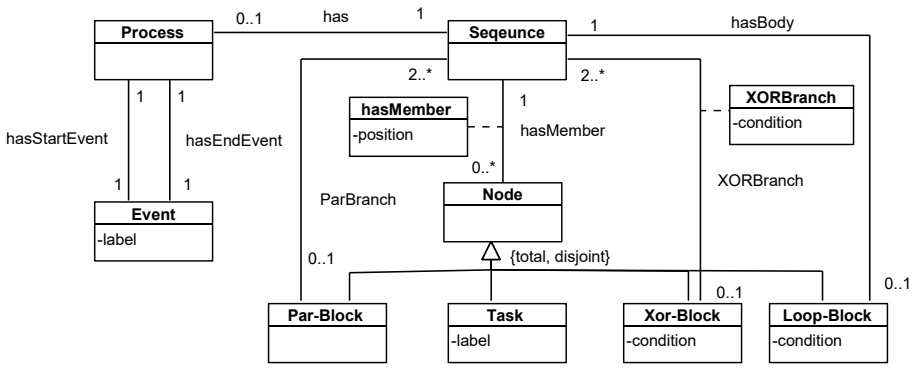

Fig. 1: Process Meta-Model for intermediate models

Process Meta-Model The intermediate format is a key factor in reducing the costs of conversational process modeling tools. It should be compact and "intuitive" for the LLM. Our preliminary experiments with BPMN-XML led to a costly, verbose, low-quality output. To achieve a small number of output tokens (and thus also context tokens for the feedback loop), we base our approach on the process meta-model shown in Fig.1. It is a variant of full-blocked process models [8], where each split gateway has its corresponding join gateway. In addition, each process has exactly one Start and End event and one sequence of nodes. A node can be a task, a Par-Block, a Xor-Block, or a Loop-Block. Xor-Blocks have a condition and are connected to at least two sequences of nodes via Xor-Branches. Each Xor-Branch has its associated condition. Similarly, Par-Blocks are connected to at least two sequences via Par-Branches. Loop-Blocks have while semantics with a condition for repetition and are connected to a sequence via the loop body. This model covers the same BPMN core elements used in conversational process modeling in literature [7, 3, 9]. However, we force the output models to be full-blocked. Such processes have several benefits: They are correct by design by eliminating deadlocks and life locks. In addition, the model is structurally similar to block-structured programming languages. Based on the good programming performance of mainstream LLMs, we suggest that only small amounts of instructions are needed to create high-quality output models following this meta-model.

4 Introducing the BPMN-Chatbot

The *BPMN-Chatbot* is an instantiation of our proposed approach for efficient conversational process modeling. It is freely available online[2]. However, upon the first start, the tool asks for an OpenAI API key.

[1] `https://promoai.streamlit.app/` Last accessed 12.06.2024
[2] `https://isys.uni-klu.ac.at/pubserv/BPMN-Chatbot-Alpha/`

Usage Scenario When a new session is started, the chatbot introduces itself and offers a tutorial on BPMN for novice users. The user can create a new process model by providing a description by typing or voice recording. The system then generates a response displayed as a BPMN diagram. The *BPMN-Chatbot* then asks the user if they would like to extend the model or fix errors. In these cases, the user provides a comment and an updated diagram is generated, and the feedback loop starts again. A screenshot of the tool is shown in Fig. 2. It shows the interface after one iteration of the feedback loop. Moreover, the tool allows navigation between model versions (using the arrow icons) to provide feedback on a particular version.

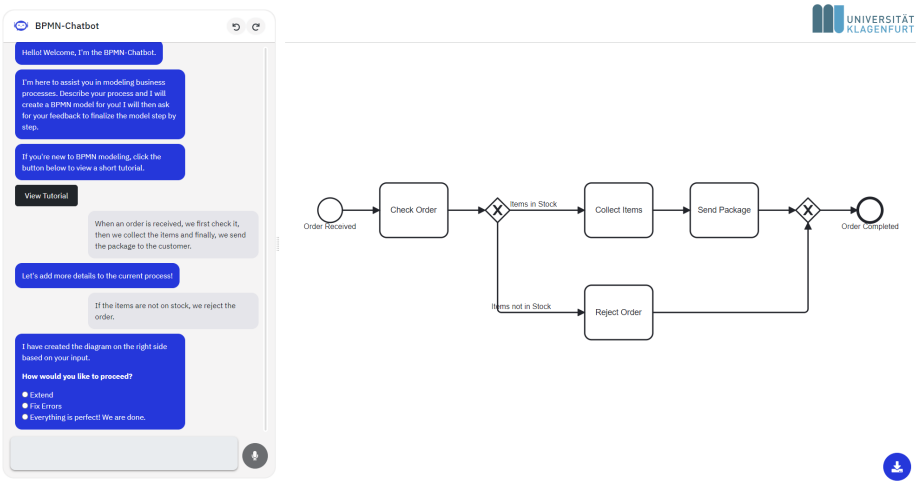

Fig. 2: Screenshot of the tool after one refinement.

Architecture The prototype is implemented as a React single-page web application as shown in Fig. 3. The UI is built using React components, which manage rendering, state updates, and event handling. A Prompt Generator module takes user input and constructs a prompt for the LLM API. Once the prompt is ready, it sends a request to the LLM API, which processes the request and returns a JSON response containing the generated process model. The Model2Model Translator module then converts this JSON response into BPMN XML. Finally, the bpmn-js[3] library is used to render the diagram.

Prompt Generation The implementation uses OpenAI's chat completions API[4], which allows the specification of the response format via JSON Schema. Therefore, we provide a schema derived from our process Meta-Model along

[3] `https://bpmn.io/toolkit/bpmn-js/`
[4] `https://platform.openai.com/docs/api-reference/chat/create`

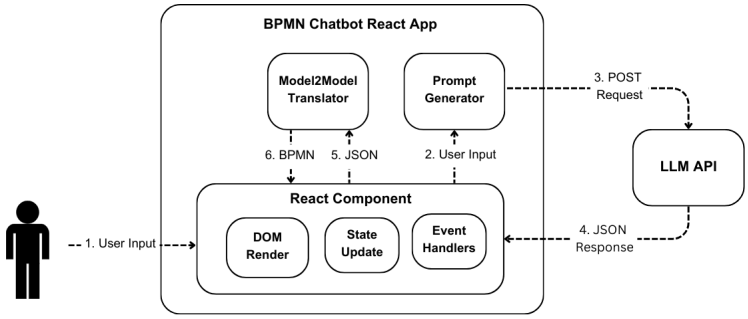

Fig. 3: BPMN Chatbot prototype architecture.

with every request. Furthermore, we include the instruction prompt shown in Listing 1. The instruction prompt follows a combination of the Template, Meta Language Creation, and Persona prompt patterns cataloged in [16]. The prompt establishes the LLM's role as a business process modeling expert and guides it in identifying key elements within the process description. It also emphasizes on nested structures and process elements to guide the LLM in correctly using the intermediate format. The instructions also encourage the identification of parallel tasks for optimization and the use of clear and specific names for labels and conditions. The prompt was iteratively optimized based on preliminary experiments on process descriptions disjoint from the ones in the evaluation.

```
You are a business process modeling expert. I will provide you with a textual description
of a business process. Generate a JSON model for the process.
Analyze and identify key elements:
1. Start and end events.
2. Tasks and their sequence.
3. Gateways (xor or parallel) and an array of "branches" containing tasks. For xor gateways,
there is a condition for the decision point and each branch has a condition label.
4. Loops: involve repeating tasks until a specific condition is met.
Nested structure: The schema uses nested structures within gateways to represent branching
paths.
Order matters: The order of elements in the "process" array defines the execution sequence.
When analyzing the process description, identify opportunities to model tasks as parallel
whenever possible for optimization (if it does not contradict the user intended sequence).
Use clear names for labels and conditions.
Aim for granular detail (e.g., instead of "Task 1: Action 1 and Action 2", use "Task 1:
Action 1" and "Task 2: Action 2").
Sometimes you will be given a previous JSON solution with user instructions to edit.
```

Listing 1: Instructions prompt for Meta-Model-based process modeling.

The OpenAI API accepts prompts in the form of message arrays. This could be used to include all the previous interactions with the chatbot as a state. However, following our approach, we use a much smaller prompt: For the initial prompt, we send an array starting with a system message containing the instructions prompt, followed by a user message containing the user input. In the feedback loop, we include the criticized model in the intermediate format

as an assistant message between the instructions and the user message. In both cases, the definition of the intermediate process format is implicitly included in the prompt.

Model Transformation: Intermediate Model to BPMN This step transforms the intermediate process model into a BPMN-XML-compliant model ready for rendering. By purpose, the input model does not contain any graphical representation information; however, this is included during model transformation. An important usability aspect is that small changes in the input model should also result in small changes in the graphical output model. We achieve this by making use of the block-structured input processes. In the first step, our algorithm deterministically adds the required positioning and size information by traversing the process tree. In particular, the dimensions of the elements are based on their type (event, task, or gateway), and the spacing between them is predefined. The starting coordinates assigned to the start event are initialized to fixed x and y values. The coordinates for each element are then calculated recursively. If an element has branches, the function iterates through them, updating the x and y coordinates accordingly. For each branch, it tracks the maximum height of nested elements. After processing all elements in the branch, the maximum y-coordinate is updated to ensure proper positioning of the following branches. Moreover, intermediate coordinates are calculated for sequence flows, adjusting for vertical and horizontal differences between connected elements.

5 Evaluation

Using our *BPMN-Chatbot* prototype, we assess our method's cost-efficiency by comparing the number of tokens required to generate an initial model with other approaches, as detailed in Sect. 5.1. We also compare the correctness of the generated models. The full system, including the feedback loop, is evaluated in a preliminary user study in Sect. 5.2.

5.1 Efficiency and Correctness - Initial Model Generation

In this experiment, we use the same subset of the PET dataset [1] used in [7]. We have generated outputs for each process description with our *BPMN-Chatbot*, *ProMoAI* [10], and prompt pattern R with Mermaid JS output in [7]. We refer to the latter one as *patternR*. We have chosen *patternR* because this combination showed the best performance for direct model generation from input text without preprocessing in [7]. We have conducted all tests with GPT-4o. Each process model was requested three times by the different systems. We ran the experiments for the *BPMN-Chatbot* on May 15, 2024. For *ProMoAI* [10], we used the OpenAI Dashboard to assess the number of tokens. We, therefore, executed each test case on a different day. These evaluations were conducted

between May 15 and May 23. The experiments with *promptR* were executed on June 7. The dataset and all results are available online[5].

The resulting process models were evaluated anonymously by a commission of two BPMN experts (disjoint from the authors) based on the correctness definition of [7]: *A model is estimated to be correct if all logical aspects of the process description are captured. If some elements are missing, substituted, or new elements are added to the model, and these changes do not violate the information introduced in the process description, the model is considered correct.* Since the colleagues are experienced in assessing student submissions, they additionally graded the models on a 1 (best) to 5 (worst) school grade scale.

Results and Discussion Regarding the number of tokens, our *BPMN-Chatbot* achieved an average reduction of the number of context tokens by 94% and of the output tokens by 79% compared to the baseline tool $ProMoAI$. Regarding *patternR*, we achieved an average reduction of the number of output tokens of 28% while the number of context tokens was 30% larger. Detailed results are shown in Fig. 4.

Our tool provides a substantial advantage over $ProMoAI$. However, the situation is more complex for *patternR*. According to our experiments, *patternR* is beneficial if the produced process models are small. However, in the case of larger models (see Case 1.3 in Fig. 4), the smaller output format compensates for the larger amount of context tokens. We require more context tokens since the schema definition of the output format also accounts for context tokens, while *patternR* makes use of the fact that the Mermaid format is known to GPT-4o. However, with the current pricing politics[6], this does not lead to a monetary advantage of *patternR* as output tokens cost three times more than context tokens. Moreover, in this experiment, we did not consider the costs of the feedback loop, where the output format is highly relevant, as a previous model is sent in each iteration of the loop.

Table 1 shows the correctness percentage and school grades. Overall, we achieved a correctness rate of 95% compared to 86% for $ProMoAI$ and 81% for *patternR*. The high output quality is also reflected in the school grades, with an average grade of 2.09 compared to 2.9 for $ProMoAI$ and 2.5 for *patternR*. Notably, in the school grade, *patternR* performs better than $ProMoAI$. One reason is the inability of $ProMoAI$ to include expressions for conditional gateways and branches.

5.2 Technology Acceptance and Feedback Loop

At the start, the attendees were shown an introductory video, or an interactive introduction was provided. The introduction demonstrated the core BPMN control-flow elements and how modelers currently create process models. Afterward, the attendees designed processes using the tool. We chose an open task

[5] https://github.com/BPMN-Chatbot/bpmn-chatbot-archive

[6] `https://openai.com/api/pricing/` Last accessed 09.06.2024

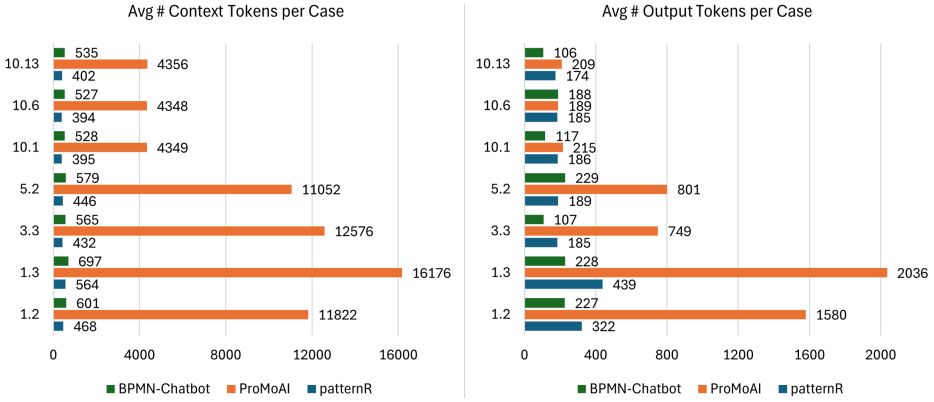

Fig. 4: Comparison of Context Tokens and Output Tokens

Case	**#Words**	**#Tasks**	**BPMN-Chatbot**	**ProMoAI [10]**	**patternR [7]**
10.13	39	3	(100%, 2.0)	(100%, 2.0)	(100%, 1.0)
10.6	30	4	(100%, 2.0)	(100%, 2.0)	(100%, 2.3)
10.1	29	4	(100%, 2.0)	(100%, 2.0)	(100%, 2.0)
5.2	83	7	(67%, 1.0)	(67%, 2.0)	(100%, 2.7)
3.3	71	7	(100%, 2.0)	(67%, 3.7)	(100%, 1.33)
1.3	162	11	(100%, 2.3)	(67%, 4.0)	(0%, 5.0)
1.2	100	10	(100%, 3.3)	(100%, 3.3)	(67%, 3.3)
Average			(95.24%, 2.09)	(86%, 2.9)	(81%, 2.5)

Table 1: (Percentage of correctly generated models, avg school grade (1-5))

to avoid the issue of attendees simply entering the predefined task description. After completing the task, we invited participants to complete a short questionnaire and donate their conversations for research. 40 Participants completed the questionnaire out of the total 76 conversations we collected.

Since the experiment was done at a public science fair, we limited the questions on the demographics to their usage pattern of PC/Mac computers and their knowledge of business processes. 60% of the participants identified themselves as daily PC users, 22.5 % as weekly users, and 17.5% as occasional users. 32.5% saw business processes for the first time in our experiment, 20% had seen business process models before, while 15% had previous knowledge from their work environment. Another 15% had already created business processes, with 12.5% creating them regularly. For 5%, no answer was available.

The questions, average results, and standard deviation (SD) are shown in Table 2. We used a 1 (strongly disagree) to 5 (strongly agree) Likert scale. Our questionnaire covered two aspects: On the one hand, we wanted to know if the prototype provided useful answers in general (Q2) and if, in particular, the answers in the feedback loop were useful (Q3). Question (Q1) asked if the participants completed their task with the tool successfully. On the other hand,

we wanted to assess if the tool has the potential to democratize process modeling. To answer this question, we have opted for a technology acceptance test [2]. Since we surveyed people from the general public, we focused on perceived usefulness, attitude, and intention of use. We used three questions (Q5-Q7) for the perceived usefulness and report on the average values. The attitude and intention of use were assessed by one question (Q4) and (Q8), respectively. Since the original user acceptance questions target professional users, not general attendees, we have rephrased the questions for this scenario.

Category	**Statement**	**Rating**	**SD**
Usefulness of Response	(Q1) I successfully modeled my business process with the BPMN-Chatbot	4.15	0.88
	(Q2) The models created by the BPMN-Chatbot were helpful.	4.18	0.86
	(Q3) The BPMN-Chatbot responded well to my feedback.	4.25	0.8
	Usefulness of Response Average	*4.19*	*0.85*
Attitude	(Q4) It was fun using the BPMN-Chatbot.	4.7	0.50
Perceived Usefulness	(Q5) I think the BPMN-Chatbot makes modeling business processes easier.	4.45	0.67
	(Q6) I think the BPMN-Chatbot increases productivity in modeling business processes.	4.25	0.80
	(Q7) I think the BPMN-Chatbot is useful for modeling business processes.	4.5	0.67
	Perceived Usefulness Average	*4.4*	*0.71*
Intention of Use	(Q8) If I could use the BPMN-Chatbot, I would.	4.28	1.02

Table 2: Questionnaire results of the experiment at the science fair.

Results and Discussion Regarding the usefulness of response questions, we obtained 4.25 for the feedback loop (Q3) and 4.18 (Q2) for the general answers provided by the system. For task completion, a score of 4.15 was reached. Since 27.5% of the participants had modeled business processes before, we also report on this subgroup. Here, the scores were even better, with 4.55 (Q3) for the feedback loop, 4.64 for the answers (Q2), and 4.36 for task completion (Q1). We see this as a strong indication that the tool responds very well to user input in general, and the answers during the feedback loop were also considered very good. This clearly shows that our proposal for efficiently implementing the feedback loop using only a previous process model as the state still leads to highly valued responses.

Regarding the technology acceptance questions, the tool scored overall 4.4 for perceived usefulness, 4.7 for attitude, and 4.28 for the intention of use. In the subgroup of participants with at least some process modeling experience,

the tool reached even better values with 4.7 for perceived usefulness, 4.82 for attitude, and 4.73 for intention of use. For novice users, the tool scored 4.28 for perceived usefulness, 4.63 for attitude, and 4.1 for intention of use. These results indicate substantial acceptance of the tool by participants with at least some process modeling experience. We still see the slightly lower numbers of novice users as a strong indication that LLM-based conversational process modeling has the potential to democratize process modeling.

Threats to validity The setting at the public science fair allowed us to collect feedback from a wide audience. Overall, it was very well received, and the participants enjoyed using it in a recreational setting. However, for a full technology acceptance test, a more defined setting is required, where people spend more time with the tool. Also, the fully open task may have led to higher acceptance scores as participants were not constrained to a fixed modeling goal. Therefore, we see the numbers only as a strong indication of the overall tool's usefulness. Regarding the experiments of the initial models, we have some randomness based on the nature of LLMs and uncontrolled factors. Therefore, we see the results as a strong indication of high output quality.

6 Conclusion and Future Works

LLM-based conversational process modeling systems should use LLMs resources efficiently. We have identified the core cost drivers for LLM-based conversational process modeling and introduced a meta-model for intermediate process models. We have instantiated the approach with our publicly available *BPMN-Chatbot*. The tool was evaluated with two experiments. In the first experiment, we compared the number of tokens and the quality of the solutions for initial models. The results show substantial reductions in the number of tokens (up to 94%) while achieving 95% correct models with an average school grade of 2.09 compared to 86% and 2.5 for the best competitors. An additional user study conducted at a public science fair assessed the quality of the responses in the feedback loop and the overall user acceptance. The results indicate a strong acceptance and satisfying responses in the feedback loop. Future works include an extension of the meta-model with more process elements, a dedicated technology acceptance test with modeling experts, experiments on more input processes with more modeling experts to assess output quality, and a comparison to classical NLP-based approaches. An evaluation with alternative and open Source LLMs is also considered relevant for future work.

References

1. Bellan, P., et. Al.: Process extraction from natural language text: the PET dataset and annotation guidelines. In: Proceedings of NL4AI'2022). CEUR Workshop Proceedings, vol. 3287, pp. 177–191. CEUR-WS.org (2022), https://ceur-ws.org/Vol-3287/paper18.pdf

2. Davis, F., Davis, F.: Perceived usefulness, perceived ease of use, and user acceptance of information technology. MIS Quarterly **13**, 319– (09 1989). https://doi.org/10.2307/249008
3. Fill, H., Fettke, P., Köpke, J.: Conceptual modeling and large language models: Impressions from first experiments with chatgpt. Enterp. Model. Inf. Syst. Archit. Int. J. Concept. Model. **18**, 3 (2023). https://doi.org/10.18417/EMISA.18.3, https://doi.org/10.18417/emisa.18.3
4. Forell, M., Schüler, S.: Modeling meets large language models. In: Modellierung 2024 Satellite Events (2024). https://doi.org/10.18420/modellierung2024-ws-003
5. Grohs, M., Abb, L., Elsayed, N., Rehse, J.R.: Large language models can accomplish business process management tasks. In: De Weerdt, J., Pufahl, L. (eds.) Business Process Management Workshops. pp. 453–465. Springer Nature Switzerland, Cham (2024)
6. Klievtsova, N., Benzin, J.V., Kampik, T., Mangler, J., Rinderle-Ma, S.: Conversational process modelling: state of the art, applications, and implications in practice. In: International Conference on Business Process Management. pp. 319–336. Springer (2023)
7. Klievtsova, N., Benzin, J.V., Kampik, T., Mangler, J., Rinderle-Ma, S.: Conversational process modeling: Can generative ai empower domain experts in creating and redesigning process models? (2024), https://arxiv.org/abs/2304.11065v2
8. Kopp, O., Martin, D., Wutke, D., Leymann, F.: The difference between graph-based and block-structured business process modelling languages. Enterp. Model. Inf. Syst. Archit. Int. J. Concept. Model. **4**(1), 3–13 (2009). https://doi.org/10.18417/EMISA.4.1.1
9. Kourani, H., Berti, A., Schuster, D., van der Aalst, W.M.P.: Process modeling with large language models (2024), https://arxiv.org/pdf/2403.07541
10. Kourani, H., Berti, A., Schuster, D., van der Aalst, W.M.P.: Promoai: Process modeling with generative ai (2024)
11. OpenAI: Gpt-4 technical report (2024), https://arxiv.org/abs/2303.08774
12. Patel, P., Choukse, E., Zhang, C., Shah, A., Goiri, Í., Maleki, S., Bianchini, R.: Splitwise: Efficient generative llm inference using phase splitting. Power **400**(700W), 1–75 (2023)
13. Samsi, S., Zhao, D., McDonald, J., Li, B., Michaleas, A., Jones, M., Bergeron, W., Kepner, J., Tiwari, D., Gadepally, V.: From words to watts: Benchmarking the energy costs of large language model inference (2023), https://arxiv.org/pdf/2310.03003
14. Schüler, S., Alpers, S.: State of the art: Automatic generation of business process models. In: International Conference on Business Process Management. pp. 161–173. Springer (2024)
15. Vidgof, M., Bachhofner, S., Mendling, J.: Large language models for business process management: Opportunities and challenges. In: Di Francescomarino, C., Burattin, A., Janiesch, C., Sadiq, S. (eds.) Business Process Management Forum. pp. 107–123. Springer Nature Switzerland, Cham (2023)
16. White, J., Fu, Q., Hays, S., Sandborn, M., Olea, C., Gilbert, H., Elnashar, A., Spencer-Smith, J., Schmidt, D.C.: A prompt pattern catalog to enhance prompt engineering with chatgpt (2023)
17. Yu, L., Li, J.: Stateful large language model serving with pensieve (2024), https://arxiv.org/pdf/2312.05516