Utilizing Process Models in the Requirements Engineering Process Through Model2Text Transformation

Nataliia Klievtsova
Technical University of Munich
TUM School of Computation, Information and Technology
Garching, Germany
{firstname.lastname}@tum.de ⓘ

Juergen Mangler
Technical University of Munich
TUM School of Computation, Information and Technology
Garching, Germany
{firstname.lastname}@tum.de ⓘ

Timotheus Kampik
SAP Signavio
Berlin, Germany
{firstname.lastname}@sap.com ⓘ

Stefanie Rinderle-Ma
Technical University of Munich
TUM School of Computation, Information and Technology
Garching, Germany
{firstname.lastname}@tum.de ⓘ

Abstract—With the advent of large language models (LLMs), requirements engineers have gained a powerful natural language processing tool to analyze, query, and validate a wide variety of textual artifacts, thus potentially supporting the whole requirements engineering process from requirements elicitation to management. However, the input for the requirements engineering process often encompasses a variety of potential information sources in various formats, especially graphical models such as process models. Hence, this work aims to contribute to the state of the art by assessing the feasibility of utilizing graphical process models and their textual representations in the requirements engineering process. In particular, we focus on the extraction of textual process descriptions from process models as i) input for the requirements engineering process and ii) documentation as the result of process-oriented requirements engineering. To this end, we explore, quantify, and compare traditional deterministic and LLM-based extraction methods where the latter includes GPT3, GPT3.5, GPT4, and LLAMA. The evaluation assesses output quality and information loss based on one data set. The results indicate that LLMs produce human-like process descriptions based on the predefined patterns, but apparently lack true comprehension of the process models.

Index Terms—AI4RE, Process Models, Process Descriptions, Large Language Models

I. INTRODUCTION

The development of an information system (IS) is a complex process which requires an understanding of the application domain by requirements engineers, i.e., of its technical, organizational, and operational specificities [1]. Thus, requirements engineers often deal with a multitude of either already existing artifacts or artifacts that need to be created – ranging from text artifacts, like user stories, use-cases, and scenario descriptions to graphical artifacts like conversation flows, UML diagrams, and graphical business process descriptions [2].

Especially business processes, or models thereof, can be considered crucial artifacts. Given that business process orientation is held as an effective approach to increasing corporate performance [3], IS are often required to support and sustain their business processes transparently [4]. Business processes are a specific type of organizational routine [5] and engage multiple stakeholders [6]. Typically, business process models constitute (graphical) representations of business processes [6] and the increasing scale and complexity of business operations has led to a widespread increase in business process modeling adoption [7].

Thus, the development and implementation of an IS is connected with understanding, analysis and improvement of one or multiple business processes and their dependencies [8].

According to [9] a requirements engineering (RE) process consists of two phases: requirements development (RD) and requirements management (RM). RD typically comprises four phases, which are requirements elicitation, analysis, specification, and validation. RM includes such activities as status tracking, version control, and change control. All of these phases and activities require that all the above-mentioned artifacts are understood, made consistent with each other, and organized in a way that allows for simple management and (re-)validation.

With the advent of natural language processing (NLP), and especially of large language models (LLMs), and their recent application in the form of chatbots such as ChatGPT, requirements engineers have gained a powerful tool to manage, query and transform large requirements bases into different artifacts. While many aspects have been explored, such as conducting requirements elicitation in the form of interviews between domain experts and chatbots, and transforming textual process descriptions into graphical representations [10]–[12] (also referred to as text-to-model or *T2M* transformation), other aspects still require additional research.

In this work we focus on a particular aspect of requirement elicitation and validation, i.e., understanding the content of

business process models, and transforming them into textual descriptions. The motivation for this step is to make the information contained in these graphical process models (artefacts) available to all participants involved in the RE process, enabling them to (1) understand the details of process models even if they are not familiar with process modeling [13], (2) check consistency, and (3) transform the textual information back into up-to-date business process models of possibly higher quality [14]. In order to exploit the capabilities of LLMs, we investigate their feasibility for the generation of process descriptions from process models (referred to as model-to-text or *M2T* transformation), and compare them with traditional (deterministic) M2T methods. The paper addresses the following research questions:

- **RQ1** What are suitable graphical representations, and associated LLM prompts, that lead to successful M2T transformation?
- **RQ2** What are suitable key performance indicators (KPIs) to quantify the quality of the textual descriptions generated through LLMs based on process models, enabling the comparison of the results of LLMs to traditional M2T methods?
- **RQ3** Which level of information loss occurs during M2T transformation, especially when utilizing LLMs?

We tackle **RQ1** – **RQ3** as follows: in Sect. II we discuss existing M2T transformation approaches, i.e., traditional NLP methods and LLM based approaches to introduce the topic and provide the baseline for subsequent comparison. In Sect. III, we explore the scenarios in which M2T transformation can be utilized during the Requirements Engineering (RE) process. In Sect. IV we outline the M2T approach proposed in this work. We discuss the rationale for the selection of different graphical model representations (and their associated file formats), and discuss the LLMs and traditional M2T methods we employed, as well as our experimental setup including KPIs for assessment and comparison. Section IV addresses **RQ1** and **RQ2**. Section V presents the evaluation of the proposed approach for existing LLMs, i.e., GPT3, GPT3.5, GPT4, and LLAMA based on an existing "baseline" data set [15]. We measure the quality of the produced results regarding different LLM prompts, traditional M2T methods ($\mapsto$ **RQ1** and **RQ2**), and information loss ($\mapsto$ **RQ3**), the latter based on similarity metrics and the KPIs proposed in [11]. Finally, section VI concludes and discusses the findings.

II. EXISTING TRANSFORMATION METHODS

Generally, model transformation methods can be classified into three groups, i.e., text-to-model (T2M), model-to-text (M2T), and model-to-model (M2M) [16].

T2M transformation will not be deeply addressed in the context of this paper. Existing methods depend on text pattern search, rule-based approaches, or semantic analysis (e.g., [17]–[19]). Nevertheless, with recent advancements in machine learning (ML) and generative AI, there is a growing interest in applying classical supervised machine learning-based approaches and LLMs [11], [12], [20], [21].

Model transformation is an approach to transform an input model into a target model (M2M) or a target grammar (M2T) [16]. M2M transformation is relatively straightforward, involving the conversion of one structured object into another. It is worth highlighting that M2M transformation is a long-running challenge in business process management, in particular when it comes to the interoperability of BPMN-based process models between tools of different vendors. Here, M2M transformation is continuously improved by software vendors participating in the OMG BPMN Model Interchange Working Group (MWIG)[1].

In contrast, M2T transformations target textual outputs, often possessing arbitrary structures that do not correspond to any specific meta-model. Existing traditional M2T transformation approaches can be categorized into three groups based on their underlying implementation approach: visitor-based, template-based, or hybrid [16].

Visitor-based approaches such as [22] navigate a tree-based internal representation of the input model to generate information based on the visited model elements. The generated text is written after every element is visited. The order of the visited elements is defined by transformation rules. In template-based approaches, the generation of text for input models is defined through templates. Templates include static text and placeholders for data extraction. The transformation is controlled by a meta-program accessing information stored within the target models. Hybrid approaches employ template-based approaches in combination with visitor-based patterns to perform M2T transformation [16], and deliver the best results of all traditional approaches..

As visitor-based approaches mostly integrate a three stage pipeline (content determination, planning, realization) proposed in [23], several challenges occur, mainly regarding text and sentence planning to make the text lively and readable. Template-based methods suffer from low flexibility, often being specific to particular languages and lack generality, which complicates their maintenance [24]. Overall, so far, M2T approaches generate arbitrary, unstructured text, posing difficulties in evaluating proposed approaches [25].

III. M2T TRANSFORMATION IN REQUIREMENTS ENGINEERING PROCESS

RE can be considered a starting point of IS development. In the scope of this work, we focus specifically on the RD phase of the RE process (see Fig. 1). There are many stakeholders involved in the RE process from both customer and developer side, e.g., customers and end users, domain experts, project managers, software developers, software testers, as well as requirements engineers [26].

The level of involvement in a project depends on the role of the particular stakeholder. However, requirements engineers are actively involved throughout the entire project timeline, as they serve as a bridge in communication between customer

---

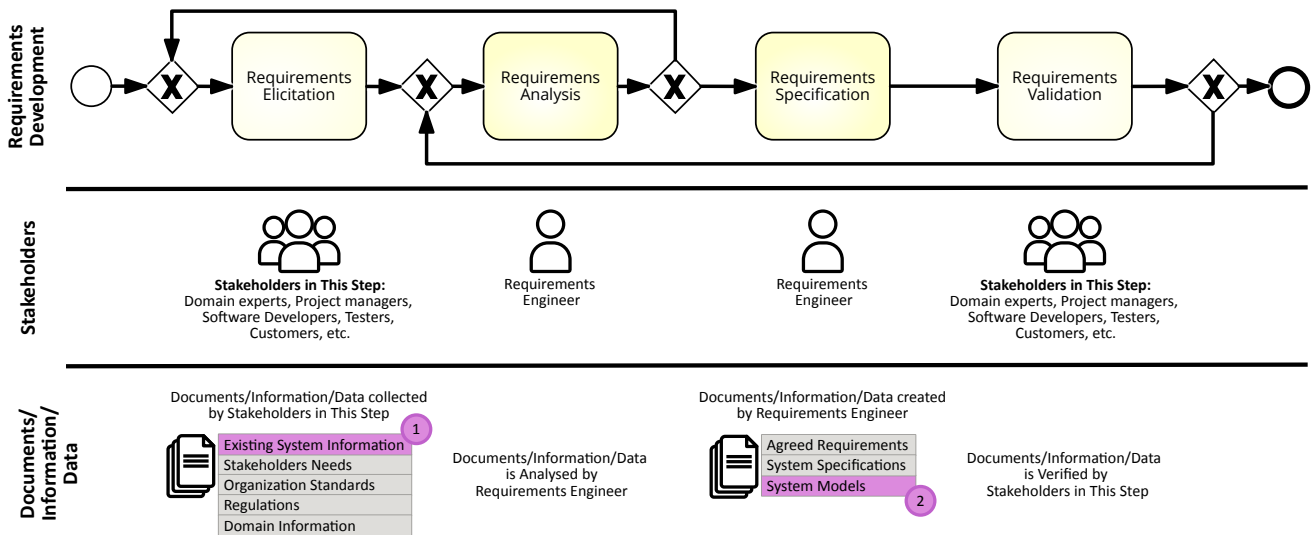[1]https://www.omgwiki.org/bpmn-miwg/doku.php

Fig. 1: Requirements Development with Stakeholders and Documentation Data as a Part of RE Process

and development stakeholders in order to correctly identify and document all requirements [27].

During *requirements elicitation*, all stakeholders who influence the requirements for the IS create various documents containing information about stakeholder needs, organizational standards and regulations, domain information, as well as information about the AS-IS status of the IS (if existing) [9]. These documents serve as a foundation for *requirements analysis*. After the requirements analysis, the requirements engineer can start with the *requirements specification*. The main goal of requirements specification is to extract requirements from the elicited documents, resulting in an appropriate software requirements specification (SRS) [28], i.e., one which declares the crucial characteristics, functions, and capabilities of the IS, along with its constraints. After the creation of the SRS, it is essential to undergo a *requirements validation* process facilitated by stakeholders who rely on this SRS.

The majority of the documents utilized across all four phases of RD are written using natural language due to its versatility, i.e., as it is familiar to all participants, does not require additional skills, and can be used to express any kind of requirements [29]. However, this can lead to imprecise, incomplete, and inconsistent [30] requirements and makes the RE process a more complex and time-consuming task [31].

Moreover, natural language is not always the best representation of organizational routines. Depending on the purpose, the content, and the stakeholders involved in a business process, various representation formats such as process models, spreadsheets, and checklists might be employed in addition to natural language text [2].

Process models can be employed during the software development cycle, particularly in the context of process-, component-, and service-oriented software development. Possible application scenarios might involve the implementation of executable business processes, the generation of artifacts for automated code creation, and the formulation of models

defining the logic of the IS [32]. [33] provides guidance on addressing common challenges during the activities of requirements engineers by exploiting BPMN process models[2].

The complexity associated with creating, integrating, and maintaining multiple representations during the RE process can be challenging. However, selecting only one particular representation can lead to insufficient involvement of stakeholders and requirements engineers in the RE process and may cause an incomplete and inaccurate understanding of organizational routines [34]. According to Cognitive Theory of Multimedia Learning (CTML) [35], describing the concept of learning preference, the simultaneous presentation of multiple representations is recommended.

The representation of information through both natural language and graphical process models can encourage a more inclusive and effective communication process. In the context of this paper, we aim to highlight following two scenarios where M2T transformation (i.e., BPMN model to text) can provide comprehensive support to all participants within the RE process and simplify their routing (see Fig. 1):

① In cases where an existing AS-IS model is present, M2T transformation becomes useful, enabling stakeholders, independent of their modelling skills, to ensure the relevance of the current process model. This capability assists in the avoidance of potential confusion and errors that may arise when a requirements engineer receives an outdated model during requirements analysis.

② Within the analysis phase, a requirements engineer generates a new TO-BE model that reflects the up-to-date state of the system after consideration of all elicited requirements. Here, M2T transformation assists in the creation of process

---
[2]BPMN stands for Business Process Model and Notation and constitutes a standardized notation and de-facto *lingua franca* for business process modeling (bpmn.org). Hence, we will refer to BPMN as process modeling notation in the remainder of the paper.

documentation, which provides benefits for comprehension of the processes and conducting checks for inconsistencies during requirements validation.

How texts generated from process models can be utilzed for RE has been covered in other works such as [24], we instead focus on means to generate such texts.

IV. M2T Transformation with LLM-based and Hybrid Approaches

At first glance, M2T transformation performed by LLMs seems to be pretty straightforward: **(a)** present the desired model to the LLM; **(b)** instruct the LLM to convert this model into the text; **(c)** estimate whether the generated content is consistent with the provided model (i.e., evaluate the quality of M2T transformation). However, several challenges arise during M2T transformation. (a) and (b) refer to the orange tasks and (c) refers to the whole procedure (see Fig. 2).

Graphical Representation. Firstly, to enable the communication between the end user and the LLM, a tokenizer is employed. The tokenizer breaks down input texts into tokens, which can be various unit of text as a word, a subword or a single character [36]. However, a common LLM input/output token window is between $1,000$–$8,000$ tokens. Due to this limitation, it is not always possible to utilize standard XML serialization of BPMN models, especially for the more complex models with multiple elements and attributes. Thus, the transition from the standard XML serialization into a simplified abstract graphical representation of the BPMN model is required.

There are multiple graph description languages available like PlantUML[3], Mermaid.js[4], Graphviz DOT[5] or D2[6] [37] [38]. Graphviz DOT and Mermaid.js have been selected, as both have a simple syntax, support multiple diagram types, and are well-known, as well as well-documented. For example, the following textual process description can be converted from the standard XML serialization of a BPMN model into the selected graphical representation, as shown in Fig. 3.

> **"After task A, either task B or task C are conducted. After the control flow is merged, task D is executed."** (Process Description PD1)

Prompt Engineering. Secondly, the LLM is instructed to perform M2T transformation. The prompt aims to instruct the LLM about the task it has to perform [39]. Several studies highlight the importance of good prompt design and engineering to achieve a reasonable LLM response [40]. Since LLMs are sensible to the prompts and there are always several ways to describe one particular task, we designed two prompts for the M2T transformation. In "Prompt 1" we ask the LLMs to generate a process description for a specific model using simple natural language. "Prompt 2" was modified to prevent the usage of modeling-language specific terms, as we aim for the text to be written naturally for easy comprehension by

[3]https://plantuml.com/
[4]https://mermaid.js.org/
[5]https://graphviz.org/doc/info/lang.html
[6]https://d2lang.com/tour/intro

various RE process stakeholders without modeling knowledge. The prompts are available on GitHub[7].

Prompt 1: Read this `<representation_type>` model: `<model>`. Convert this model to a textual process description using simple natural language. Return only text summary.

Prompt 2: Read this `<representation_type>` model: `<model>`. Convert this model to a textual process description using simple natural language without mentioning types of the model elements (i.e., task, start event, end event, gateway, etc.). Return only text summary.

Quality of M2T Transformation. To assess the quality of LLM-based M2T transformation it is necessary to assess (a) how efficient is the M2T transformation (cf. Sect. V-A); (b) the information loss or excess occurring during continuous transformations (cf. Sect. V-A3); and (c) how well the LLM-based transformation performs in comparison to hybrid approaches (cf. Sect. V)?

The whole evaluation process will be structured as per Fig. 2, the raw evaluation results are available on GitHub[8].

The traditional (hybrid) approach: For comparison, we use a *hybrid approach* of text generation (non-AI). In this approach graphical models are transformed into trees, where each task, or gateway is represented by a node in the tree. The trees are then traversed, and the text/meaning of each node inserted into templates, similar to [22].

In the scope of this work two different templates are used. The first template named "**reduced**" assumes local context, i.e., if the text contains information that task should occur in the second branch of a decision node it is assumed that the most recently decision is meant. If there is nested decision this could potentially lead to confusing texts.

Thus the second template named "**extended**" contains information about an ID for every decision and every parallel node, so that with every information about a task we can refer to where it is inserted. I.e, "The task X occurs in the second branch of decision D1".

As for the approach utilized in [22] no code is available, we implemented a transformation pipeline[9] containing the following types of components:

- *Source Components:* Three different source components (BPMN2, Mermaid, Graphviz), extract data from the respective text formats to a generic node/link data structure.
- *Transform Component:* The node/link representation is converted into a Refined Process Structure Tree

[7]https://github.com/com-pot-93/m2t-trans/tree/main/prompt_engineering, last access: 2024-01-29
[8]https://github.com/com-pot-93/m2t-trans/tree/main/evaluation, last access: 2024-01-29
[9]https://github.com/etm/promptgen which contains the data sets, all the conversion results, and a conversion script; the library https://github.com/etm/cpee-transformation utilized by the conversion script, contains the conversion logic itself; last access: 2023-12-08
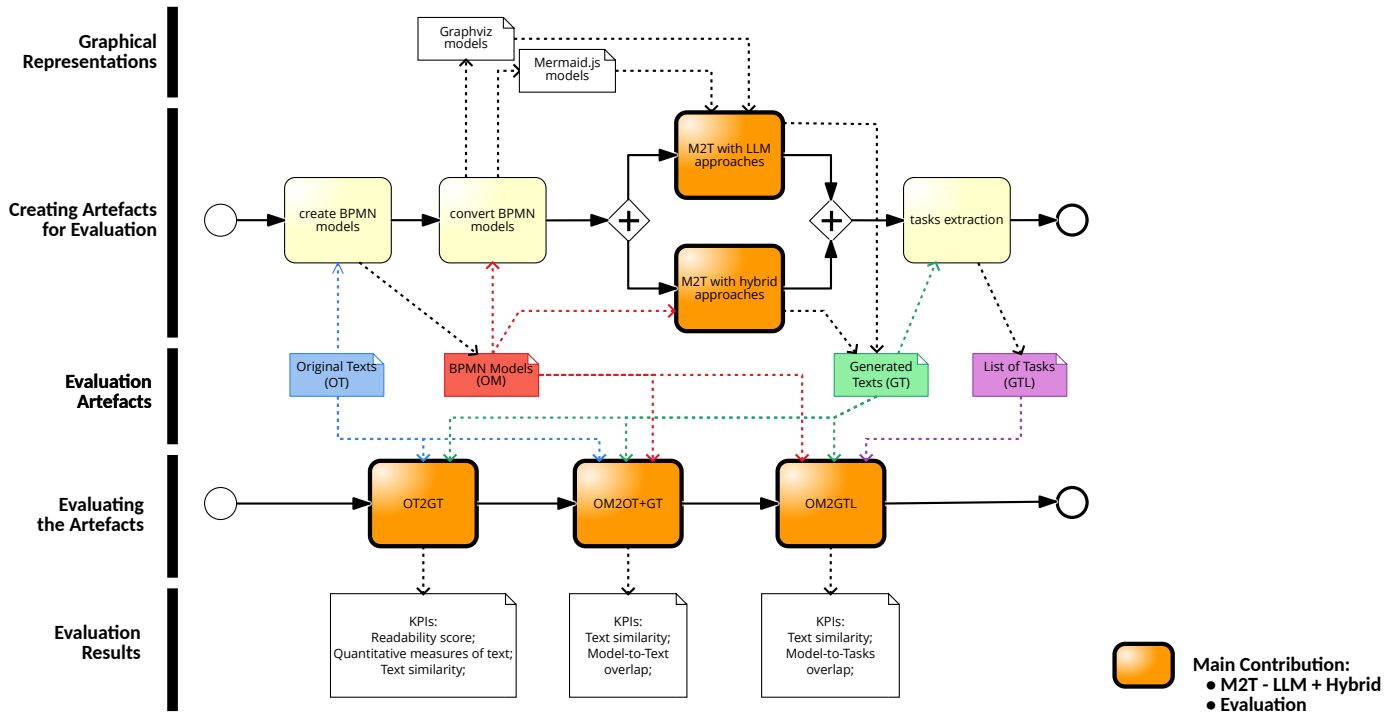
Fig. 2: M2T and Evaluation Process

(RPST) [41], with a focus on fixing simple modeling errors, and producing well-formed BPMN.

- *Target Components:* We use four target components. A CPEE[10] [42] RPST XML representation to be able to visually inspect the RPST, and three different text representations based on slightly different templates and tree traversal algorithms.

Finally, we discuss (a), (b) and (c) introduced at the start of this section, based on the subsequent evaluation steps (also shown in Fig. 2) which yield a number of KPIs:

- **OT2GT**: Compare original to generated texts
  - *readability*: FKGL - Flesch-Kincaid Grade Level and FRE - Flesch Reading Ease;
  - *quantitative measures*: #S - number of sentences, #W - number of words, and #WS- number of words per sentence;
  - *text similarity*: TS-NC - non-contextual text similarity and TS-C - contextual text similarity;
  - *modeling references*: #MR - number of generated texts containing references to the modeling elements;
- **OM2OT+GT**: Compare original models to original and generated texts
  - *model-to-text similarity*: TS-C - contextual text similarity;
  - *model-to-text overlap*: O-OM - percentage of the tasks in the original model, that were not aligned to the sentences in the generated text and O-OT - percentage

of the sentences in the generated text, that were not aligned to the tasks in the original model;
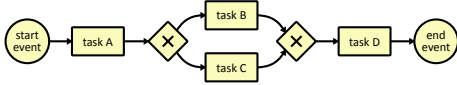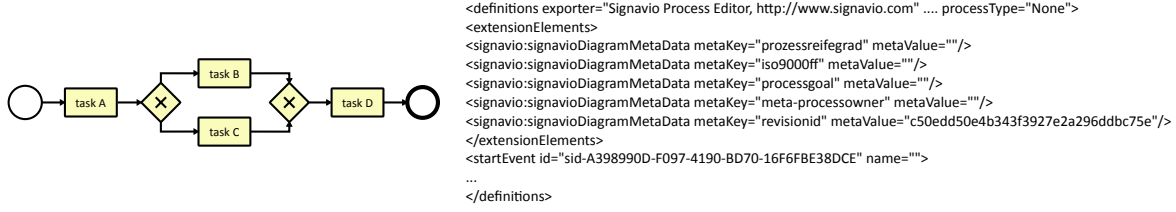
- **OM2GTL**: Compare original models to extracted tasks from generated texts
  - *quantitative measures*: #T - number of tasks in the model and number of extracted tasks;
  - *model-to-tasks similarity*: TS-C - contextual text similarity;
  - *model-to-tasks overlap*: O-OO - percentage of the tasks in the original model, that were not aligned to the tasks from the generated text and O-OG - percentage of the tasks from the generated text, that were not aligned to the tasks in the original model;

For non-contextual similarity, we use algorithms in which only the actual words are considered for the similarity calculation, without taking into account the context in which each word appears. On the other hand, contextual similarity considers the surrounding context in which different words appear. To measure the non-contextual similarity (TS-NC) we apply TD-IDF vectorizer and for the contextual one (TS-C) - BERT base model as a word embedding model. To obtain both similarities we calculate a cosine similarity metric [43].

To measure the model-to-text overlap we compute the TS-C between each task in the model and each sentence in the text. Then we sum up all tasks and all sentences that reach the predefined similarity threshold, i.e., are matched to each other (overlap). After that we count how many task in the model (O-OM) and sentences in the text (O-OT) are misaligned. The calculation of Model-to-tasks overlap follows the same principle.
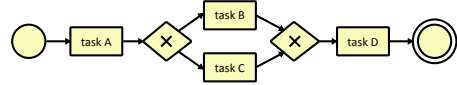
[10]Cloud Process Engine: https://cpee.org/

```
<definitions exporter="Signavio Process Editor, http://www.signavio.com" .... processType="None">
<extensionElements>
<signavio:signavioDiagramMetaData metaKey="prozessreifegrad" metaValue=""/>
<signavio:signavioDiagramMetaData metaKey="iso9000ff" metaValue=""/>
<signavio:signavioDiagramMetaData metaKey="processgoal" metaValue=""/>
<signavio:signavioDiagramMetaData metaKey="meta-processowner" metaValue=""/>
<signavio:signavioDiagramMetaData metaKey="revisionid" metaValue="c50edd50e4b343f3927e2a296ddbc75e"/>
</extensionElements>
<startEvent id="sid-A398990D-F097-4190-BD70-16F6FBE38DCE" name="">
...
</definitions>
```

flowchart LR
```
0:startevent:((startevent))-->1:task:(task A)
1:task:-->2:exclusivegateway:{x}
2:exclusivegateway:--> 3:task:(task B)
2:exclusivegateway:--> 4:task:(task C)
3:task:-->5:exclusivegateway:{x}
4:task:-->5:exclusivegateway:
5:exclusivegateway: --> 6:task:(task D)
6:task:-->7:endevent:((endevent))
```

(ii) Mermaid.js serialization

digraph G {
```
rankdir = LR;
node[style=filled,fillcolor=lemonchiffon];
"start_1"[shape=circle label=""]; "end_1"[shape=doublecircle label=""];
"seg_1"[shape=diamond label="X"]; "meg_1"[shape=diamond label="X"];
"task A"[shape=rectangle]; "task B"[shape=rectangle];
"task C"[shape=rectangle]; "task D"[shape=rectangle];
"start_1"-> "task A"; "task A"-> "seg_1"; "seg_1"->"task B"; "seg_1"->"task C";
"task B"->"meg_1"; "task C"->"meg_1"; "meg_1"->"task D"; "task D"-> "end_1";
}
```

(iii) DOT Graphviz serialization

Fig. 3: Selected Graphical Representations for PD1

V. EVALUATION

For the evaluation we utilize the PET dataset[11]. PET is an annotated dataset for process extraction from natural language text tasks (i.e., process descriptions). It comprises 45 textual process descriptions from different domains alongside human-annotated process elements. Only 7 examples out of the 45 from the PET dataset also utilized by [44] are used. These examples represent subgroups of diverse size and complexity in a sample, allowing us to draw more precise conclusions by representing the entire population and minimizing the possibility of bias. Such a diversity is also significant because it directly influences the quality of the generated response by the LLMs [45].

All tables present average values derived for the 7 selected examples. Considering the average across the entire sample aims to capture the overall tendency observed within it, even though these values may not fully capture the nuances present in individual documents.

Selected examples in the sample are between 30 and 162 words, and on average 79 words, in the selected process descriptions. Based on these process descriptions and annotations provided in the PET data set, BPMN models are created. The BPMN models are then converted into simplified graphical representations, utilizing Mermaid.js (MER) and Graphviz DOT (GV) as the underlying serialization languages.

The created process models consist of two basic categories of BPMN elements: (a) flow objects, including start and end events, tasks, exclusive and parallel gateways, and (b) connecting objects, represented by sequence flows. Each model contains between 3 and 11 tasks, and at least 2 events (one start event and one end event). In addition, 5 out of 7 process models include exclusive gateways, while 1 process model contains parallel gateways as well (all models are available at github[12]).

Each of the created models is utilized to generate its corresponding process description. Subsequently, for the final evaluation the list of tasks is extracted from each generated process description.

We refer to: i) the 7 selected process descriptions from the PET dataset as the **original texts (OT)**; ii) the process models that are created based on these process descriptions and their annotations as the **original models (OM)**; iii) the texts generated using LLMs or traditional approaches based on the OM as the **generated texts (GT)**; iv) the lists of tasks extracted from the GT as the **lists of tasks extracted from generated texts (GTL)** (see Fig. 4).

A. Quality Assessment of M2T Transformation

1) OT2GT.: In this subsection we compare complexity, structure, as well as contextual and non-contextual similarity of texts generated by LLMs and by utilizing a traditional hybrid approach. The following LLMs are used: text-davinci-003 (GPT3), gpt-3.5-turbo (GPT3.5), and gpt-4 (GPT4) from openai.org [13], as well as Llama-7b-chat-hf [14] and Llama-70b-chat-hf [15] from hugging-face. See the examples of generated texts in Fig. 5.

In general, according to the Flesch–Kincaid readability tests (i.e., tests designed to indicate how difficult a passage in

---

[11]https://huggingface.co/datasets/patriziobellan/PET

[12]https://github.com/com-pot-93/m2t-trans/tree/main/pet_models, last access: 2024-04-2

[13]https://platform.openai.com/docs/models

[14]https://huggingface.co/meta-llama/Llama-2-7b-chat-hf
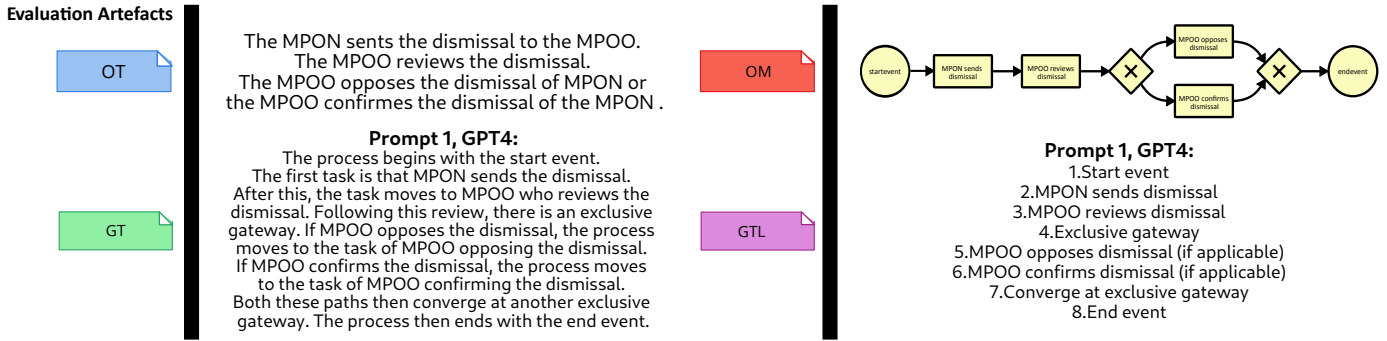
[15]https://huggingface.co/meta-llama/Llama-2-70b-chat

Fig. 4: Examples of Evaluation Artefacts

English is to understand [46][16]), both Flesch scores (FKGL≃7 and FRE≃68) indicate that the generated texts are simple enough to be successfully read by 80% of readers in the USA. In addition, we can see that all *LLM-generated texts* have the same level of *complexity* as the original texts, created by humans (see Tab. I, II). However, this might suggest that texts in the selected dataset are overly simplistic and may not accurately reflect reality: texts incorporating domain-specific knowledge and terminology could be more intricate in real-world scenarios.

Comparing the *structure* of original texts to the LLM-generated texts, the generated texts feature more sentences and a similar number of words, contributing to a more structured appearance (also shown in Tab. I, II).

The LLM-generated texts achieve up to 85% and 81% *non-contextual text similarity* (TS-NC) by GPT and Llama models respectively, suggesting that during text generation, LLMs employ similar words and concepts as humans do. Taking into account *contextual text similarity*(TS-C), which considers not only the similarity of words themselves but also the context, a maximum of 89% (GPT) and 87% (Llama) similarity was reached (again, see Tab. I, II).

Considering non-contextual, contextual text similarity and number of existing references to the model elements it could be seen that mostly the texts generated from GV achieve better results than texts generated out of the MER models. However, the difference is not significant (again, see Tab. I, II). Such behavior can be explained by noting that Graphviz is an older tool, and as a result, it might be more commonly found in the training data. In addition, "Prompt 2" seems to be more successful as "Prompt 1".

When comparing different LLMs, GPT3 creates shorter texts than other models (i.e., around 25% less sentences comparing to the original texts and 50% shorter than the texts generated by other LLMs on average). However, despite the fact that GPT3 texts contain less characters and sentences, the non-contextual text similarity between texts generated by GPT3 and the original text is only 2%-points smaller than

[16]Recent research highlights shortcomings of Flesch-Kincaid scores, indicating that it should not be used as an optimization target [47]. Here, we merely use this score as a rough indicator of complexity and do not argue for maximizing it.

for texts generated with GPT3.5 and GPT4 and 3%-points higher comparing with the texts generated by Llama models on average. This might suggest that almost 50% of the content within the text descriptions generated by other LLMs is *not relevant for the process content* (i.e., if it were removed from the process description, we would still be able to obtain the same model). At the same time, it might indicate that human-written texts are closer to the texts generated with GPT3.5, GPT4 and Llama models than to those, generated with GPT-3.

Taking into account prompt specific results, up to 46% of all texts generated from MER and "Prompt 1" contain references to the model elements and up to 29% by utilizing GV representation. When examining texts generated from MER by GPT3.5 and GPT4, references to specific modeling elements are present in 100% of the examples.

Using "Prompt 2" it is determined that only utilizing GPT3.5 in 2 out of 7 generated texts from MER and 1 out of 7 from GV references are still present (see #MR in Tab. I, II) and using other GPT models texts are reference-free. Considering "Prompt 2" and LLama models only texts generated from GV by Llama-70 are reference-free. In all other cases 1 out of 7 texts contains references to the types of model elements.

Despite the fact that none of the prompts are able to generate texts without mentioning model elements and their types, "Prompt 2" can be considered as sufficient, as 91% of all texts are reference-free in comparison to 63% with the "Prompt 1".

Generally, utilizing GPT3 and GPT4 yields to better results and Llama-70 appears to have a higher performance.

In comparison to LLM generated texts, the *traditionally generated texts* produce different results, as depicted in Tab. III.

None of the traditionally generated texts (independent of template and traversal algorithm) yields as good results as the LLMs. Mainly, due to the usage of templates that correspond to element types in the modeling language, traditional approaches produce language that is rigid, unnatural, and stale. We suspect that such correspondence to the modeling language can impact the perception of process description by various stakeholders without modeling knowledge and experience (we have to quantify potential positive and negative effects in a

TABLE I: Basic evaluation of generated text from original model with GPT3, GPT3.5 and GPT4, where FKGL - Flesch-Kincaid Grade Level, FRE - Flesch Reading Ease, #S - Num. of Sentences, #W - Num. of Words and #WS- Num. of Words per Sentence, TS - Text Similarity and #MR - Num. of generated texts containing references to the modeling elements

GR	prompt	LLM	FKGL	FRE	#S	#W	#WS	#MR↓	TS-NC↑	TS-C↑
	original text	human	7.00	67.25	5.00	79.00	14.62	0		
Prompt 1	MER	GPT3	6.80	71.95	4.00	51.00	12.75	0	0.7	0.89
		GPT3.5	6.40	71.75	8.00	110.00	13.75	7	0.69	0.78
		GPT4	6.60	73.17	9.00	107.00	12.17	7	0.75	0.79
		avg.	6.60	72.29	7.00	89.33	12.89	4.67	0.71	0.82
	GV	GPT3	6.60	73.58	5.00	68.00	14.00	0	0.75	0.87
		GPT3.5	7.30	70.84	6.00	89.00	14.83	3	0.76	0.84
		GPT4	6.50	73.78	7.00	91.00	14.80	1	0.79	0.83
		avg.	6.80	72.73	6.00	82.67	14.54	1.33	0.77	0.85
	avg.	all	6.70	72.51	6.50	86.00	13.72	3.00	0.74	0.83
Prompt 2	MER	GPT3	7.60	65.73	2.00	46.00	15.50	0	0.69	0.87
		GPT3.5	5.60	73.34	8.00	94.00	11.75	2	0.63	0.85
		GPT4	6.60	71.41	8.00	83.00	11.00	0	0.80	0.85
		avg.	6.60	70.16	6.00	74.33	12.75	0.67	0.71	0.86
	GV	GPT3	7.60	64.71	4.00	42.00	14.50	0	0.73	0.88
		GPT3.5	6.90	67.45	7.00	101.00	14.43	1	0.72	0.82
		GPT4	7.60	65.62	7.00	82.00	13.67	0	0.81	0.86
		avg.	7.37	65.93	6.00	75.00	14.20	0.33	0.75	0.85
	avg.	all	6.98	68.04	6.00	74.67	13.48	0.50	0.73	0.86

TABLE II: Basic evaluation of generated text from original model with Llama7 and Llama70; see abbreviations in the Table I

GR	prompt	LLM	FKGL	FRE	#S	#W	#WS	#MR↓	TS-NC↑	TS-C↑
	original text	human	7.00	67.25	5.00	79.00	14.62	0		
Prompt 1	MER	Llama-7	7.40	66.13	5.00	100.00	15.43	2	0.60	0.78
		Llama-70	6.40	73.78	4.00	62.00	13.30	0	0.75	0.87
		avg.	6.90	69.96	4.50	81.00	14.37	1.00	0.68	0.83
	GV	Llama-7	9.60	58.82	5.00	94.00	20.86	3	0.67	0.84
		Llama-70	5.80	70.39	12.00	140.00	10.83	3	0.72	0.84
		avg.	7.70	64.61	8.50	117.00	15.84	3.00	0.70	0.84
	avg.	all	7.30	67.28	6.50	99.00	15.11	2.00	0.69	0.83
Prompt 2	MER	Llama-7	8.30	61.56	6.00	82.00	15.33	1	0.55	0.77
		Llama-70	8.20	59.70	5.00	70.00	14.00	1	0.73	0.85
		avg.	8.25	60.63	5.50	76.00	14.67	1.00	0.64	0.81
	GV	Llama-7	9.50	57.98	6.00	97.00	17.67	1	0.65	0.76
		Llama-70	7.90	64.91	9.00	115.00	13.00	0	0.85	0.87
		avg.	8.70	61.44	7.50	106.00	15.34	0.50	0.75	0.82
	avg.	all	8.48	61.04	6.50	91.00	15.00	0.75	0.70	0.81

TABLE III: Basic evaluation of generated text from original model with the traditional approaches; see abbreviations in the Table I

Approach	FKGL	FRE	#S	#W	#WS	#MR↓	TS-NC↑	TS-C↑
original text	7.00	67.25	5.00	79.00	14.62	0		
LLMs	7.26	67.83	6.35	86.20	14.18	1.60	0.72	0.84
reduced	7.60	71.04	11.00	207.00	18.56	7	0.6	0.6
extended	7.50	71.24	12.00	214.00	18.14	7	0.62	0.57

separate study). As can be seen in Tab. III the *readability* score together with the number of words per sentence slightly increased on average. At the same time, the number of sentences doubled and the number of words per process description increased by 2.5 times on average, but both *contextual and non-contextual text similarity* decreased by 25%- and 11%-points comparing to LLM-generated process descriptions on average.

2) OM2OT+GT.: To confirm the appropriateness of the LLM-generated texts, it is crucial to evaluate not only the text-to-text similarity and the readability of the generated texts but also the relationships between the model and both the original and generated text. To find out the extent to which the model covers the content of the original text and how well the generated text covers the content of provided model, we use following metrics: (a) text similarity (TS-C) (i.e., comparing the text with all the tasks in the model), and (b) overlap between the text and the model i.e., how many sentences in the text align with tasks in the model (O-OM - only in original model, O-OT - only in original text).

As depicted in Table IV, the texts generated by LLMs achieve a text similarity score ranging from 76% to 86%. This result can be considered significant, as the similarity score between the original text and the model is 83%. The texts

	OT	The MPON sends the dismissal to the MPOO. The MPOO reviews the dismissal. The MPOO opposes the dismissal of MPON or the MPOO confirms the dismissal of the MPON.

Extended	First, *a task with the id a1567 and the label* "MPON sends the dismissal" occurs. Then *a task with the id a1569 and the label* "MPOO reviews the dismissal" occurs. Afterwards, this is followed *by an exclusive decision with 2 branches. This decision will be furthermore refered to as D1. The first branch task of D1* is executed if the condition is "". *The only entry in the first branch of D1 is a task with the id a1573 and the label* "MPOO opposes the dismissal". *The second branch of D1 is executed* if the condition is "". The only entry *in the second branch of D1 is a task with the id a1575 and the label* "MPOO confirmes the dismissal". At this point *all branches of D1 are finished*, and the process ends.	

	Prompt 1	*Prompt 2*
GPT3	MPON sends the dismissal which is then reviewed by MPOO. MPOO can either oppose the dismissal or confirm it. If MPOO opposes the dismissal, the process ends there. If MPOO confirms the dismissal, the process is completed.	MPON sends the dismissal, which is then reviewed by MPOO. Depending on the outcome, MPOO either opposes or confirms the dismissal, and the process is completed.
GPT3.5	*The process starts with the "startevent"* where MPON sends the dismissal. Then, *the task moves* to MPOO who reviews the dismissal. After that, *there is an exclusive gateway where two paths* can be taken. If MPOO opposes the dismissal, *the task moves to another task.* If MPOO confirms the dismissal, *the task moves directly to the next exclusive gateway.* From there, if *the task goes to the next task, it again reaches an exclusive gateway.* Finally, *the process ends with the "endevent".*	*The process starts with an event* where MPON sends the dismissal. Then, MPOO reviews the dismissal. After that, there is a decision point *represented by an exclusive gateway.* If MPOO opposes the dismissal, *the process goes to a task.* If MPOO confirms the dismissal, *it also goes to a task. Both paths converge at another exclusive gateway.* From there, *the process either goes to an end event or returns to a task. Finally*, the process ends at an end event.
GPT4	*The process begins with the start event. The first task is that* MPON sends the dismissal. After this, *the task moves to* MPOO who reviews the dismissal. Following this review, *there is an exclusive gateway.* If MPOO opposes the dismissal, *the process moves to the task of* MPOO opposing the dismissal. If MPOO confirms the dismissal, *the process moves to the task of* MPOO confirming the dismissal. *Both these paths then converge at another exclusive gateway. The process then ends with the end event.*	The process begins with MPON sending the dismissal. Following this, MPOO reviews the dismissal. After the review, there are two possible outcomes. Either MPOO opposes the dismissal or MPOO confirms the dismissal. Regardless of the outcome, the process concludes at the end.
Llama7	The process begins when MPON sends a dismissal notice. Then, MPOO reviews the notice and makes a decision whether to oppose or confirm the dismissal. If MPOO opposes the dismissal, the process ends. If MPOO confirms the dismissal, the process continues until it reaches its conclusion.	The MPON sends a dismissal notice to the MPOO *(step 1)*. The MPOO then reviews the dismissal *(step 2)*. If the MPOO opposes the dismissal, the process ends *(step 3)*; otherwise, the MPOO confirms the dismissal *(step 6)*. Once the dismissal is confirmed, the process ends *(step 7)*.
Llama70	The MPON sends a dismissal, which is then reviewed by the MPOO. The MPOO can either oppose or confirm the dismissal. If opposed, the process ends. If confirmed, the MPOO confirms the dismissal and the process ends.	The MPON *(Managerial Personnel Oversight Committee)* sends a dismissal request to the MPOO *(Management Personnel Oversight Officer)*. The MPOO reviews the request and decides whether to oppose or confirm the dismissal. If opposed, the MPOO sends the request back to the MPON. If confirmed, the MPOO sends the confirmation to the MPON. Finally, the process ends.

Legend	▮	**process content irrelevant information**
	▮	**information added during M2T transformation**

Fig. 5: Examples of Generated Texts utilizing Diverse Approaches

TABLE IV: OM2OT+GT Evaluation

	original	reduced	extended	GPT3	GPT3.5	GPT4	Llama7	Llama70
TS-C↑	0.83	0.59	0.57	0.86	0.80	0.81	0.76	0.84
O-OM↓	0.07	0.09	0.11	0.11	0.02	0.01	0.14	0.06
O-OT↓	0.23	0.49	0.54	0.13	0.35	0.35	0.40	0.38

generated utilizing traditional approaches achieve a similarity score of 58% only.

The majority of the model tasks are successfully aligned with the sentences in the texts, with only 8% of all tasks not matching with the sentences in the generated texts on average (see O-OM in IV). The best alignment is achieved with the texts generated by GPT models.

A high percentage of non-aligned sentences (see O-OT in IV) suggests the presence of the *process content irrelevant information* in the texts: up to 50% for texts generated by traditional approaches and up to 40% for LLM-generated texts in comparison to 23% for the human-generated process descriptions. Such results are consistent with the findings from the previous section (see Sect. V-A1).

Such outputs can indicate that *the information loss* for *process content relevant information* during M2T transformation is not significant.

Adjusting the granularity of the sentences can impact both,

the text and the model overlap. The usage of smaller sentences in the text makes it easier to align the model tasks with them. However, if the text contains a lot of specific information that cannot be covered by a model, high granularity will lead to higher misalignment between the text and the model.

3) OM2GTL.: To investigate the information loss (i.e., how much information was lost during M2T transformation) more precisely, we will extract tasks from generated texts and compare them to the original models. Task extraction is performed solely with GPT4, for both, LLM-generated and traditionally generated texts. The prompt for the tasks extraction can be found here[7].

All subsequent discussion points are based on Tab. V.To conduct a comparison between the original models and the extracted tasks from the generated texts, we first examine **the number of tasks**. Only models derived from texts generated using traditional approaches have similar number of tasks as the original model. Original models contain on average of 6.57 tasks, whereas the average number of tasks extracted from traditional approaches is 7.43 for reduced and 7.14 for extended approaches. The next closest to the original value was obtained by GPT3 and is equal to 7.93 tasks on average. Extracting tasks from texts generated by all other LLMs resulted in an average finding of approximately 30% more tasks in comparison to the original models.

TABLE V: OM2GM: Average number of tasks in original models and tasks extracted from generated texts

		GPT3	GPT3.5	GPT4	L7	L70
P1	Mer	8.00	10.86	9.29	9.14	9.14
	GV	8.43	9.71	9.57	8.71	7.71
	avg.	8.21	10.29	9.43	8.93	8.43
P2	MER	7.57	8.86	8.14	9.14	10.14
	GV	7.71	9.57	9.00	9.57	11.71
	avg.	7.64	9.21	8.57	9.36	10.93
avg.		7.93	9.75	9.00	9.14	9.68

When moving on to the contextual task similarity (TS-C) (see Tab. VI), an average similarity of 88% is observed. The models produced from texts generated by traditional approaches achieve the highest scores of 96% on average. However, models created from LLM-generated texts demonstrate commendable values, reaching up to 91% with GPT-3 and GPT-4. Llama models perform slightly worse than GPT models (2%-points on average) reaching up to 87% contextual text similarity.

Such results can be considered as acceptable, as the level of similarity between the original text and the original model is 83% (see Tab. IV). However, to estimate whether the information loss or excess occurs during the M2T transformation we look at the model-to-tasks overlap.

Considering model-to-tasks overlap, on average, only 2% of all tasks presented in the original model are not aligned with the tasks extracted from generated texts (see O-OO in Tab. VI). The best results are, again, achieved with the text generated by traditional approaches, where information loss strives for zero (again, see O-OO in Tab. VI). When using text

generated by LLMs, only up to 3% of information from the original models was lost for both representations (1% using GV and 3% using MER). GPT models perform better than Llama models. Generally, there is no great difference between selected GPT models. Referring to Llama models, Llama70 performs notably better than Llama7.

We can also see, that there is no significant difference between "Prompt 1"and "Prompt 2" and it is evident that there is no noteworthy distinction between MER and GV representations.

The low percentage of the tasks in the original model, that were not aligned to the tasks from the generated text (see O-OO in Tab. VI) indicates a low level of information loss for *process content relevant information* during M2T transformation.

The lower O-OG value between the tasks extracted from original models and texts generated by traditional approaches in comparison to LLM-generated texts (see Tab. VI) may signify that traditional approaches either tend to reduce the amount of generated content or hold it in a more structured way, whereas LLMs tend to generate some additional content based on their foreknowledge and advanced text generation capabilities.

In addition, up to 49% of all tasks extracted from LLM-generated texts are not aligned with the tasks in the original model (see O-OG in Tab. VI). It might demonstrate that new tasks are added during task extraction. The data in Tab. V can also confirm such behavior, where the number of extracted tasks increases in comparison to the original model.

This indicates that, during M2T transformation, new content is inserted into the process description by LLMs, which can lead to future errors and inconsistencies. Currently, we cannot definitively say whether newly added tasks are just misclassified as the tasks, are just hallucinations of the LLMs, or are a product of analysis and reasoning and are supposed to substitute those missing in the original models.

B. Threats to Validity

Caution should be exercised when interpreting obtained results, due to the nature of the dataset, selected models, and the limited sample size. The selected examples are simple and straightforward. The employed models were created based on annotations primarily utilized for process extraction evaluation, thus possessing an artificial nature.

The presented results capture the average picture and may not fully encapsulate the entire scope of the dataset. Omitting to consider results for individual documents in addition to average values might obscure variations within the data.

One limitation of our approach is tied to the quality of the BPMN artifacts provided by the stakeholders. The more modeling errors, or semantic errors are contained in the models, the harder it will be to extract useful information. Also, the range of BPMN constructs investigated in this work is limited (see Sect. V) and does not contain, for example, most events, pools (for modeling organizational roles) and specialized gateways (e.g., event-based gateway). Based on

TABLE VI: OM2GM Evaluation

Metric	GR	traditional		prompt 1					prompt 2					avg.
		red.	ext.	GPT3	GPT3.5	GPT4	L-7	L-70	GPT3	GPT3.5	GPT4	L-7	L-70	
TS-C↑	MER	0.96	0.96	0.90	0.79	0.87	0.82	0.87	0.91	0.82	0.91	0.85	0.87	0.88
	GV			0.90	0.85	0.90	0.82	0.87	0.90	0.82	0.89	0.83	0.87	0.88
O-OO↓	MER	0.00	0.00	0.00	0.02	0.02	0.04	0.04	0.02	0.02	0.02	0.11	0.00	0.03
	GV			0.00	0.00	0.00	0.04	0.00	0.00	0.00	0.00	0.02	0.00	0.01
O-OG↓	MER	0.08	0.12	0.21	0.45	0.34	0.42	0.33	0.21	0.39	0.26	0.39	0.44	0.30
	GV			0.25	0.37	0.39	0.43	0.26	0.19	0.37	0.32	0.49	0.46	0.31

the aforementioned limitations, of course it is a reasonable assumption, that LLMs may sometimes drop requirements, or even hallucinate additional requirements (signs of this can be seen in the evaluation in Sect. V-A3).

Furthermore, in this paper, the evaluation of information loss only involved a single round (i.e., OM → GT → GTL). In scenarios where multiple M2T and T2M transformations occur, the resulting outcomes could potentially vary significantly.

Finally, the quality assessment of the performed M2T transformation focuses on completeness, considering both the structure of the generated texts and the degree to which they cover information from the original texts and models. However, it is important to note that for now we cannot make any assumptions about the correctness of the generated texts, specifically regarding whether the control flow presented in the original model is accurately depicted.

VI. CONCLUSIONS AND DISCUSSION

In this work, we propose two scenarios in which the integration of LLM based M2T transformation can yield advantages for stakeholders in requirements engineering processes. In particular, we focus on extracting textual process descriptions from process models as ① a component of input documentation for the requirements engineer gathered during requirements elicitation and ② one of the resulting documents after requirements specification, particularly in the context of process-, service-, and component-oriented requirements engineering (see also Fig. 1).

We propose and evaluate two graphical representations that can substitute standard XML serialized BPMN models, as well as two LLM prompts to achieve successful M2T transformation. In addition, we propose and discuss a set of KPIs to quantify the results of M2T evaluations, i.e., readability score, text similarity, model-to-text, and model-to-tasks overlap.

The evaluation is conducted by converting LLM-generated models back to texts, extracting the tasks from these texts, and comparing the original with the generated artifacts. Regarding **RQ1** we design and supply two prompts for achieving M2T with LLMs (see Sect. IV). Regarding **RQ2** we present a hybrid (non-AI) M2T approach for generating texts, which allows us to compare the LLM results with more traditional methods and a list of KPIs to assess the difference (see Sect. IV). Finally, to tackle **RQ3** we discuss the level of information loss which occurs during M2T, based on a set of KPIs (see Sect. V).

Based on the performed evaluation (see Sect.V), we observe that using "Prompt 2" allows us to decrease the amount of generated text that lacks process content value. However, there isn't a significant difference between "Prompt 1"and "Prompt 2" based on selected KPIs. Similarly, there's no notable distinction between MER and GV representations.

When it comes to M2T generation, LLM-based approaches exhibit better readability scores and demonstrate superior performance, achieving higher similarity between the original and generated text, as well as between the model and generated text in comparison to traditional hybrid approaches (see Tab. I, II). Even though LLM-based approaches produce better texts according to the selected KPIs (see Tab. III), texts generated by hybrid approaches seem to be better suited for reverse T2M transformation, as they can guarantee the correctness of the generated process description for a particular model (i.e., no additional information or hallucinations), and can offer full control over the process, which is not possible when employing LLMs (see Tab. V).

A combination of both traditional and LLM-based approaches can offer significant benefits leveraging the strengths of both methodologies, i.e., the precision provided by traditional methods to ensure the correctness of created process descriptions and the naturalness and fluency of the text enhanced by LLM-based approaches.

The correlation between the completeness of a generated model and quality and structure of its textual description underscores that LLMs produce human-like text using predefined patterns but may lack true comprehension. We are able to perform a M2T transformation using both traditional and LLM-based approaches, with minimal information loss (2% on average). However, it is important to mention that introduced KPIs concentrate only on completeness of generated models and texts. At this point, we cannot make any assumptions about the correctness of the texts (i.e., whether the generated texts accurately represent the correct control flow introduced in original models).

Future research will focus on evaluating and improving the correctness of the generated models and texts, to automatically eliminate inconsistencies, and further improve and enlarge the datasets to provide a stable foundation for developing conversational modeling.

References

[1] J. L. González and J. S. Díaz, "Business process-driven requirements engineering : a goal-based approach," 2007.

[2] H. van der Aa, J. Carmona, H. Leopold, J. Mendling, and L. Padró, "Challenges and opportunities of applying natural language processing in business process management," in *Computational Linguistics*, pp. 2791–2801, 2018.

[3] R. Škrinjar, V. Vukšić, and M. Štemberger, "Adoption of business process orientation practices: Slovenian and croatian survey," *Business Systems Research Journal*, vol. 1, no. 1-2, pp. 5–19, 2012.

[4] A. Przybylek, "A business-oriented approach to requirements elicitation," in *2014 9th International Conference on Evaluation of Novel Approaches to Software Engineering (ENASE)*, pp. 1–12, 2014.

[5] D. Beverungen, "Exploring the interplay of the design and emergence of business processes as organizational routines," *Bus. Inf. Syst. Eng.*, vol. 6, no. 4, pp. 191–202, 2014.

[6] M. Dumas, M. L. Rosa, J. Mendling, and H. A. Reijers, *Fundamentals of Business Process Management*. Springer, 2013.

[7] W. M. van der Aalst, "Business process management: A comprehensive survey," *International Scholarly Research Notices*, vol. 2013, pp. 1–37, 2013.

[8] T. Kampik, "Process-oriented requirements engineering." https://www.signavio.com/post/process-oriented-requirements-engineering/, 2017. Last accessed: 2024-03-24.

[9] K. Wiegers and J. Beatty, *Software Requirements*. Best practices, Microsoft Press, 2013.

[10] P. Bellan, M. Dragoni, and C. Ghidini, "Extracting business process entities and relations from text using pre-trained language models and in-context learning," in *Enterprise Design, Operations, and Computing*, pp. 182–199, 09 2022.

[11] N. Klievtsova, J. Benzin, T. Kampik, J. Mangler, and S. Rinderle-Ma, "Conversational process modelling: State of the art, applications, and implications in practice," in *Business Process Management Forum - BPM 2023 Forum, Utrecht, The Netherlands, September 11-15, 2023, Proceedings* (C. D. Francescomarino, A. Burattin, C. Janiesch, and S. W. Sadiq, eds.), vol. 490 of *Lecture Notes in Business Information Processing*, pp. 319–336, Springer, 2023.

[12] M. Grohs, L. Abb, N. Elsayed, and J. Rehse, "Large language models can accomplish business process management tasks," *CoRR*, vol. abs/2307.09923, 2023.

[13] H. van der Aa, J. Carmona, H. Leopold, J. Mendling, and L. Padró, "Challenges and opportunities of applying natural language processing in business process management," in *Computational Linguistics*, pp. 2791–2801, 2018.

[14] P. Harmon and J. Garcia, "Bptrends report: The state of business process management: 2020," 08 2020.

[15] P. Bellan, C. Ghidini, M. Dragoni, S. P. Ponzetto, and H. van der Aa, "Process extraction from natural language text: the pet dataset and annotation guidelines," in *Workshop on Natural Language for Artificial Intelligence*, 2022.

[16] N. Kahani, M. Bagherzadeh, J. R. Cordy, J. Dingel, and D. Varró, "Survey and classification of model transformation tools," *Softw. Syst. Model.*, vol. 18, no. 4, pp. 2361–2397, 2019.

[17] A. Ghose, G. Koliadis, and A. Chueng, "Process discovery from model and text artefacts," in *Services Computing Workshops*, pp. 167–174, 2007.

[18] T. Yue, L. C. Briand, and Y. Labiche, "An automated approach to transform use cases into activity diagrams," in *Modelling Foundations and Appl.*, pp. 337–353, 2010.

[19] F. Friedrich, "Automated generation of business process models from natural language input," Master's thesis, School OF Business and Economics of the Humbold-University zu Berlin, 2010. Master's Thesis.

[20] L. Ackermann, J. Neuberger, and S. Jablonski, "Data-driven annotation of textual process descriptions based on formal meaning representations," in *Advanced Information Systems Engineering*, pp. 75–90, 2021.

[21] C. Qian, L. Wen, A. Kumar, L. Lin, L. Lin, Z. Zong, S. Li, and J. Wang, "An approach for process model extraction by multi-grained text classification," in *Advanced Information Systems Engineering*, pp. 268–282, 2020.

[22] H. Leopold, J. Mendling, and A. Polyvyanyy, "Generating natural language texts from business process models," in *Advanced Inf. Syst. Engineering*, pp. 64–79, 2012.

[23] E. Reiter and R. Dale, "Building applied natural language generation systems," *Natural Language Engineering*, vol. 3, 03 2002.

[24] B. Aysolmaz, H. Leopold, H. A. Reijers, and O. Demirörs, "A semi-automated approach for generating natural language requirements documents based on business process models," *Inf. Softw. Technol.*, vol. 93, pp. 14–29, 2018.

[25] H. Leopold, J. Mendling, and A. Polyvyanyy, "Supporting process model validation through natural language generation," *IEEE Trans. Software Eng.*, vol. 40, no. 8, pp. 818–840, 2014.

[26] H. Saiedian and R. Dale, "Requirements engineering: making the connection between the software developer and customer," *Information and Software Technology*, vol. 42, no. 6, pp. 419–428, 2000.

[27] S. Maalem and N. Zarour, "Challenge of validation in requirements engineering," *J. Innov. Digit. Ecosyst.*, vol. 3, no. 1, pp. 15–21, 2016.

[28] X. Franch, C. Palomares, C. Quer, P. Chatzipetrou, and T. Gorschek, "The state-of-practice in requirements specification: an extended interview study at 12 companies," *Requir. Eng.*, vol. 28, no. 3, pp. 377–409, 2023.

[29] K. Pohl, *Requirements Engineering: Fundamentals, Principles, and Techniques*. Springer Publishing Company, Incorporated, 1st ed., 2010.

[30] L. Kof, "Requirements analysis: Concept extraction and translation of textual specifications to executable models," in *Natural Language Processing and Information Systems, 14th International Conference on Applications of Natural Language to Information Systems, NLDB 2009, Saarbrücken, Germany, June 24-26, 2009. Revised Papers* (H. Horacek, E. Métais, R. Muñoz, and M. Wolska, eds.), vol. 5723 of *Lecture Notes in Computer Science*, pp. 79–90, Springer, 2009.

[31] V. Shukla, D. Pandey, and R. Shree, "Requirements engineering: A survey," 2015.

[32] J. Solís-Martínez, J. P. Espada, B. C. Pelayo G-Bustelo, and J. M. C. Lovelle, "BPMN MUSIM: Approach to improve the domain expert's efficiency in business processes modeling for the generation of specific software applications," *Expert Systems with Applications*, vol. 41, no. 4, Part 2, pp. 1864–1874, 2014.

[33] Y. Odeh, "BPMN in engineering software requirements: An introductory brief guide," in *Proceedings of the 9th International Conference on Information Management and Engineering, Barcelona, Spain, October 09 - 11, 2017*, pp. 11–16, ACM, 2017.

[34] F. M. Santoro, M. R. Borges, and J. A. Pino, "Acquiring knowledge on business processes from stakeholders' stories," *Advanced Eng. Informatics*, vol. 24, no. 2, pp. 138–148, 2010.

[35] R. E. Mayer, "Multimedia learning," vol. 41 of *Psychology of Learning and Motivation*, pp. 85–139, Academic Press, 2002.

[36] M. Ali, M. Fromm, K. Thellmann, R. Rutmann, M. Lübbering, J. Leveling, K. Klug, J. Ebert, N. Doll, J. S. Buschhoff, *et al.*, "Tokenizer choice for llm training: Negligible or crucial?," *arXiv preprint arXiv:2310.08754*, 2023.

[37] D. Whatley, M. Goldman, and R. C. Miller, "Snapdown: A text-based snapshot diagram language for programming education," in *Visual Languages and Human-Centric Comp.*, pp. 1–9, 2021.

[38] N. Klievtsova, J.-V. Benzin, T. Kampik, J. Mangler, and S. Rinderle-Ma, "Conversational process modeling: Can generative ai empower domain experts in creating and redesigning process models?," *CoRR*, vol. abs/2304.11065, 2024.

[39] K. Zhu, J. Wang, J. Zhou, Z. Wang, H. Chen, Y. Wang, L. Yang, W. Ye, N. Z. Gong, Y. Zhang, and X. Xie, "Promptbench: Towards evaluating the robustness of large language models on adversarial prompts," *CoRR*, vol. abs/2306.04528, 2023.

[40] J. White, Q. Fu, S. Hays, M. Sandborn, C. Olea, H. Gilbert, A. Elnashar, J. Spencer-Smith, and D. C. Schmidt, "A prompt pattern catalog to enhance prompt engineering with ChatGPT," *CoRR*, vol. abs/2302.11382, 2023.

[41] J. Vanhatalo, H. Völzer, and J. Koehler, "The refined process structure tree," *Data Knowl. Eng.*, vol. 68, no. 9, pp. 793–818, 2009.

[42] J. Mangler and S. Rinderle-Ma, "Cloud process execution engine: architecture and interfaces," *arXiv preprint arXiv:2208.12214*, 2022.

[43] D. Chandrasekaran and V. Mago, "Evolution of Semantic Similarity—A Survey," *ACM Computing Surveys*, vol. 54, no. 2, pp. 41:1–41:37, 2021.

[44] P. Bellan, M. Dragoni, and C. Ghidini, "A qualitative analysis of the state of the art in process extraction from text," in *DP@AI*IA*, pp. 19–30, 2020.

[45] X. Gu, K. M. Yoo, and S. Lee, "Response generation with context-aware prompt learning," *CoRR*, vol. abs/2111.02643, 2021.

[46] R. Flesch, "A new readability yardstick.," *Journal of applied psychology*, vol. 32, no. 3, p. 221, 1948.

[47] T. Tanprasert and D. Kauchak, "Flesch-kincaid is not a text simplification evaluation metric," in *Natural Language Generation, Evaluation, and Metrics*, pp. 1–14, 2021.