Thesis Blocking Notice

This practical transfer paper

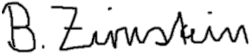…………………………………………………………………………………… (“**Thesis**”)

contains confidential information of the SAP group.

This Thesis must only be made available to the members of the examination board of Berlin School of Economics and Law and must not be published, reproduced or disclosed to any other third party – neither in full or part – for a period of 5 years after completion of the Thesis examination process without explicit prior written approval by SAP SE.

Date: January 09, 2024

Name of Student: Bruno Zirnstein

Address of Student: Rhinstr. 105c, 10315 Berlin, Germany

Signature:

Non-Disclosure Declaration

I, ……………………………………………………………, the undersigned,

acting as an examiner of the
practical transfer paper……………………………………………………… ("**Thesis**")

written by ..Bruno Zirnstein……………………………………………………… ("**Student**"")

enrolled at ..Berlin School of Economics and Law…………………….. ("**University**")

understand that the Thesis written by the Student contains SAP Confidential Information (as hereinafter defined).

"**SAP Confidential Information**" shall mean all information which SAP protects against unrestricted disclosure to others, furnished by SAP to the Student in the context of Thesis preparation and examination, including the Thesis, provided to me by the Student during the Thesis preparation and examination.

SAP Confidential Information shall not include:
a. Information that was already publicly available before it was disclosed to me or has become publicly available by no wrongful act or omission on my part;
b. Information that was already in my possession, through an authorized third party or by other legitimate means without confidentiality obligation at the time of disclosure,
c. Information developed by me independently of SAP without breach of confidentiality obligations;
d. Information which SAP has explicitly designated in writing as being non-confidential.

I hereby undertake, with respect to SAP Confidential Information:
a. not to use SAP Confidential Information disclosed during the Thesis preparation and examination for any purpose other than assessing and evaluating the Thesis;
b. not to reproduce SAP Confidential Information in any form except as required for assessing and evaluating the Thesis;
c. not to disclose it to any third party, without the prior written consent of SAP; and
d. to take all reasonable steps to keep all SAP Confidential Information strictly confidential.

Such obligation to not disclose the Thesis to any third party does not include the disclosure of the Thesis towards employees of the University, who must conduct the Thesis with regards to the examination regulations of the University. I assure that these employees acknowledge the confidentiality of the Thesis and their own obligation to keep this Thesis confidential. „**Reasonable steps**" mean those steps I take to protect my own similar proprietary and confidential information, which shall not be less than a reasonable standard of care.
This Declaration will be governed by the laws of Germany without reference to conflict of laws principles.

Date:

Name:

Address:

Signature:

Practical Transfer Report

Extraction of BPMN process models from unstructured textual descriptions

Bruno Zirnstein

presented on January 8, 2024
at the
Berlin School of Economics and Law

SAP SE
Dietmar-Hopp-Allee 16
D - 69190 Walldorf
Telefon: +49 6227 7-47474
Fax: +49 6227 7-57575

Abstract

This report discusses the extraction of BPMN process models from unstructured textual descriptions by explaining the inherent problems and presenting three distinct approaches to the task. These approaches, leveraging the natural language understanding capabilities of large language models, are evaluated for their ability to create a specific format of BPMN diagram from text-based data. The findings show that the "Direct Translation" and "Intermediate Graph Representation" approaches present significant challenges and limitations. Contrarily, the "Generating Traces & Process Mining" approach proves noteworthy, successfully solving the essential problems of the task while also being easy to implement and offering scope for further development. It therefore is an effective proposition for developing a proof-of-concept.

Table of Contents

Glossary

Event log is a recorded sequence of events or activities related to individual process instances. It typically consists of a CaseID, an activity, and a timestamp. It provides a detailed record of activities, resources involved, and outcomes, serving as a primary data source for process discovery, analysis, and improvement.

Process discovery algorithm is a chronological sequence of activities representing one instance of a process execution. It aids in identifying patterns for improving business processes.

SAP is a German multinational software corporation that provides businesses with the necessary tools to manage various operations and customer relations. The company's primary product is its enterprise resource planning (ERP) software, which allows organizations to manage business operations across procurement, manufacturing, service, sales, finance, and HR departments. In addition, SAP also offers cloud-based data services, machine learning, business process management (BPM), and analytics, among other services.

SAP Signavio is an SAP sub-brand, providing a suite of business process management tools designed to streamline business processes, reduce operational complexity, and ensure compliance. It allows businesses to model, analyze, optimize, and execute business process workflows. This platform helps organizations to understand and improve their business processes, make data-driven decisions, and translate business strategy into operational reality.

Signavio workspace is a designated virtual area within the Signavio software platform where users can create, manage, and collaborate on business process models and related projects.

Trace is a chronological sequence of activities or events recorded from the initiation to the termination of an executed instance of a process. Each trace is an instance of a process sequence, representing the path taken through the process. It often serves as a valuable source of information for process analysis, process mining, and performance evaluation. The collected set of traces forms an event log.

Acronyms

BPM Business Process Management

BPMN Business Process Modeling Notation

BTP Business Technology Platform

GPT Generative Pretrained Transformer

ID Identifier

JSON JavaScript Object Notation

KPI Key Performance Indicator

LLM Large Language Model

NLP Natural Language Processing

POC Proof of Concept

UI User Interface

List of Figures

1 Motivation

Large companies usually have structured information about their processes in the format of process models and other artifacts. A large amount of documents containing process-relevant information is however found in an unstructured format such as PDF containing a mixture of diagrams, tables, and free text. A series of qualitative interviews conducted with large enterprises in the DACH region suggests an order of magnitude of 1k to 100k process-relevant documents per company. That's why there is great interest in research and for businesses to develop an approach that can leverage this hidden value by automatically extracting relevant process information into business process management (BPM) software. Such an approach contains multiple components, among others converting textual process descriptions to standardized BPMN 2.0 diagrams, which is the topic of this work. The rapid improvements of LLMs in the past year have given the topic renewed momentum. Moreover, there is generally more interest in the intersection topic of LLMs and BPM [10].

2 Text-to-Model Problem Statement

In this chapter, the main challenges of the task are explained, starting with an overview of the overall proposed pipeline, followed by a detailed look at the conversion step.

2.1 Extraction Pipeline

In this approach, businesses upload their documents containing process-relevant information. Given these documents, their content is extracted, and all text sections, containing process-relevant information need to be detected and extracted (step 1 in Figure 1). It should also be considered to cluster the information per process as there will occur information related to multiple processes in the documents. The process description - respectively a collection of text related to a particular process - can be converted to a BPMN 2.0 diagram (step 2 in Figure 1), comprehending all relevant information. Then, the generated diagram is rendered in the UI for the user to approve it and edit if necessary (step 3 in Figure 1). As the last step, the diagram is saved in the user's Signavio workspace (step 4 in Figure 1).

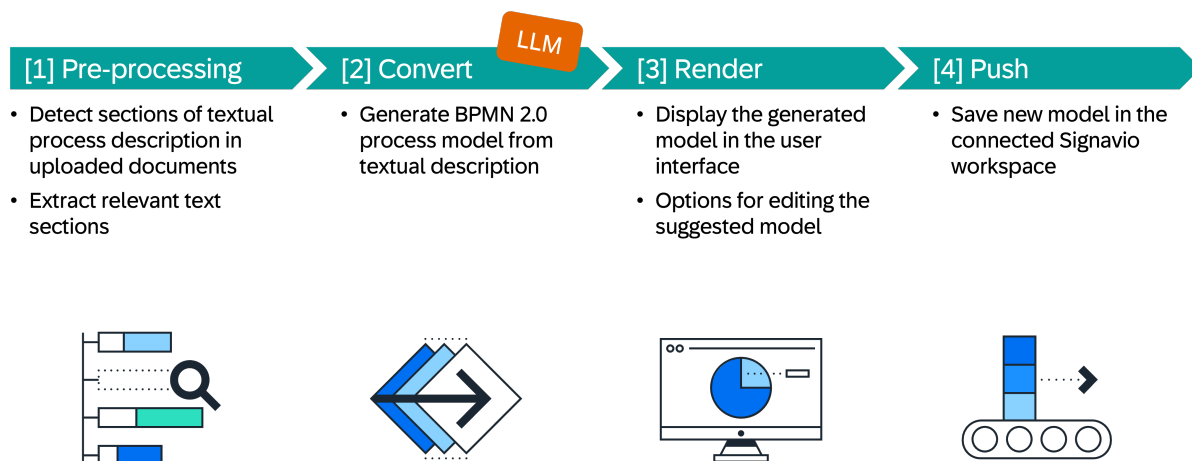| [1] Pre-processing	[2] Convert	[3] Render	[4] Push
• Detect sections of textual process description in uploaded documents • Extract relevant text sections	• Generate BPMN 2.0 process model from textual description	• Display the generated model in the user interface • Options for editing the suggested model	• Save new model in the connected Signavio workspace

Figure 1: Text-to-Model Pipeline Overview

2.2 Text-to-Model Conversion Step

Zooming in on the conversion step of the pipeline - the Text-to-Model transformation - the task can be divided into multiple problems:

- Activity extraction

- Control flow extraction

- Additional information extraction:

  - Pools and lanes

  - Documents

  - IT-systems

  - Comments

  - (Message flow)

- SAP Signavio-compliant output format

- Implementation effort (non-functional requirement)

Activities & Control Flow   First, the essential parts of a BPMN 2.0 diagram are activities and the control flow. For the activity extraction, all tasks or subprocesses that are performed in the process need to be identified. Subsequently, an accurate and comprehensive title coherent with the BPMN 2.0 guidelines [14] has to be determined for each activity. For the control flow extraction, the sequence of those activities has to be determined. The control flow can also be split up into multiple branches and joined back together using gateways when different or parallel activity sequences are executed.

Additional Information   Moreover, the diagram should contain additional information to enhance the information value and detail on certain activities. This feature is not essential for a functioning transformation but would drastically increase the value of the

generated diagram. The extraction of the message flow won't be addressed in this work, because it is not of high importance for SAP Signavio users.

Output Format   Lastly, to save the process model in the connected Signavio workspace, the output must be code for a BPMN 2.0 diagram, complying with a certain SAP Signavio-specific format (see Figure 7 for an example). Therefore, the result of the conversion step cannot be any other format, which might be easier to achieve.

Implementation Effort   As an additional (non-functional) requirement the implementation effort has to be considered. As the context of this work is an innovation project, the resources for testing and implementing approaches are very limited. The primary objective of the project is the development of a proof of concept and the demonstration of feasibility. Therefore, it is crucial to keep the effort to a minimum.

3 Related work

Scientific interest and work on the task of transforming natural language into a formal process representation has existed for over 10 years [2]. Various approaches from the NLP area were investigated to solve this problem [12]. However, with the advancements of large language model (LLM) capabilities, the NLP area was reformed and new approaches have emerged. Still, not many papers investigating these new approaches were published.

Klievtsova et al. explore the topic of chatbot-assisted conversational process modeling [11]. Among other aspects, their work includes the extraction of tasks and the control flow from a process description using LLMs and a proposed set of KPIs for comparing generated with ground-truth models in an evaluation step. Other work investigates the extraction of business process elements from text [3] and the transformation of textual process information to intermediate formats [9].

4 Approaches to Text-to-Model

In this chapter, three approaches using the unique natural language understanding capabilities of LLMs are presented and assessed.

4.1 Direct Translation

Concept The most straightforward approach is to directly transform the given process description into SAP Signavio-compliant BPMN 2.0 JSON code[1] with a single request to an LLM. This approach is efficient in terms of implementation effort and runtime.

Results To test this approach, GPT-4 was one-shot prompted to generate such SAP Signavio-compliant JSON code given a process description. However, the results were not satisfying as the generated code was not in the specified format. The desired format is very complex as it contains a lot of details about the diagram like specific IDs and explicit coordinates for each graphical element of the diagram (see Figure 7). Therefore, the LLM seems to have trouble with generating compliant code. The approach is not further investigated as it fails to meet a key requirement.

4.2 Intermediate Graph Representation

Concept Another approach is to transform the process description into another (simpler) intermediate format, such as a graph representation, which is transformed into SAP Signavio-compliant code in an extra step. This might work better than the first approach because the LLM only needs to understand the simpler intermediate representation, for example, the format of "Mermaid.js"[4] code (example in Figures 5 and 6).

Results Testing the first step of this approach (see the prompt in Figure 8), GPT-4 was able to generate valid code for a Mermaid.js graph, with elements similar to BPMN

---

[1]This JSON code is how BPMN 2.0 diagrams can be represented at SAP Signavio and used in its software. It therefore is the desired output format.

2.0 (see Figure 9 and Figure 10 for an example input and output of the first step). As Figure 10 demonstrates, there are some flaws in the generated diagram. For example, there are two duplicate sequence flows that could be joined back together using a gateway and there are also missing end nodes. To summarize, the first step requires further optimization, and the second conversion step from the intermediate format to the SAP Signavio-compliant code is required. It might be a viable approach, but it requires too much effort and therefore won't be further investigated for now.

4.3 Generating Traces & Process Mining

Concept Looking at the generated Mermaid.js graph in Figure 10, the generated diagram isn't a comprehensive process representation but rather displays all possible activity sequences. With this in mind, the idea of the third approach is to generate a unique set of traces from the process description and use a process discovery algorithm to extract a process model from that.

In the first step, GPT-4 is prompted to extract and output a unique set of traces given the process description. The output is a list of traces (see Figure 12 for an example), which are then used as an (artificial) event log and fed into the "Split Miner" [1] algorithm that extracts the process model and returns it as compliant code. The Split Miner algorithm is a state-of-the-art process discovery algorithm from 2019 that is implemented in an existing SAP Signavio service. Therefore, the approach is little effort and always returns SAP Signavio-compliant JSON code for a BPMN 2.0 diagram.

Results Testing this approach (see the prompt for trace extraction in Figure 11), GPT-4 reliably extracts a coherent and unique set of traces; the Split Miner service outputs a SAP Signavio-compliant diagram (see an exemplary result in Figure 2).

For various process descriptions, with similar structure, syntax, and complexity to the one shown in Figure 13, the output diagram always contains all referenced activities and the correct control flow, represented by activities, exclusive gateways, a start, and an end node.
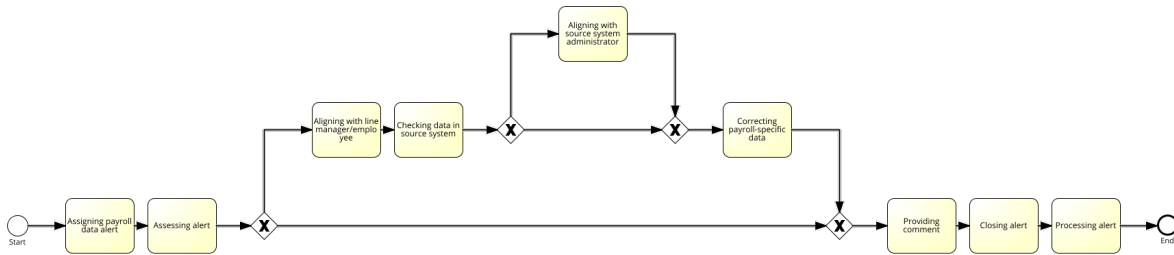
Figure 2: Generated SAP Signavio BPMN 2.0 diagram using the Generating Traces & Process
Mining approach

See the process description used as input in Figure 13 and the generated traces (artificial event
log) in Figure 14.

For process descriptions that are more ambiguous, implicit, and complex (see example
in Figure 15), the current implementation of the approach struggles to extract complex
control flows perfectly. Partially, the assumption of implicit knowledge is missing. How-
ever, that doesn't mean it can't deal with more difficult process descriptions, but the
approach has to be fine-tuned, e.g. by optimizing prompts and covering certain edge
cases.

5 Evaluation

To systematically assess the performance of any conversion approach, its generated diagram has to be evaluated for various process descriptions of different complexities. The evaluation can be done by an expert in process modeling, who manually compares the generated diagram with the original process description in terms of extracted activities, control flow, and additional information.

If there was a dataset of process descriptions with matching BPMN 2.0 diagrams given, another possibility is to compare the diagram generated from the process description with the matching diagram from the dataset. This *round-trip* evaluation can also be performed automatically. Therefore, a set of formal metrics to measure the conformance of the two diagrams need to be defined [6, 15, 16]. This automatic round-trip evaluation could be more accurate by reducing manual errors and biases and would allow different approaches to be formally evaluated as well as each part of an approach to be optimized against an objective score. However, it also requires additional preliminary effort and an appropriate dataset, which is not trivial. For example, many different diagrams are valid ground-truth diagrams for one process description. This especially applies to complex and ambiguous descriptions assuming much implicit knowledge [7].

As the purpose of this work is to investigate approaches that can solve the crucial problems of the described task in the first place, that's how approaches are evaluated. Any further evaluation techniques are not the topic of this work.

6 Prototype Implementation

The Mermaid.js and Generating Traces & Process Mining approaches were implemented as prototypes, showcasing the functionality and the current progress to other internal stakeholders. Both prototypes were built as a web application with Python 3.11 [17] and the open-source library Streamlit [5]. For using GPT-4, both prototypes use SAP's BTP LLM Proxy Service, SAP's service for LLM usage that uses Microsoft's Azure OpenAI Service [13] in the background. The Mermaid.js approach prototype can be run locally (see screenshot of the prototype in Figure 3). As the Generating Traces & Process Mining approach is more promising, this prototype was made available internally. It is deployed on SAP's Business Technology Platform (BTP), hosted by Cloud Foundry (see screenshot of the prototype in Figure 4).
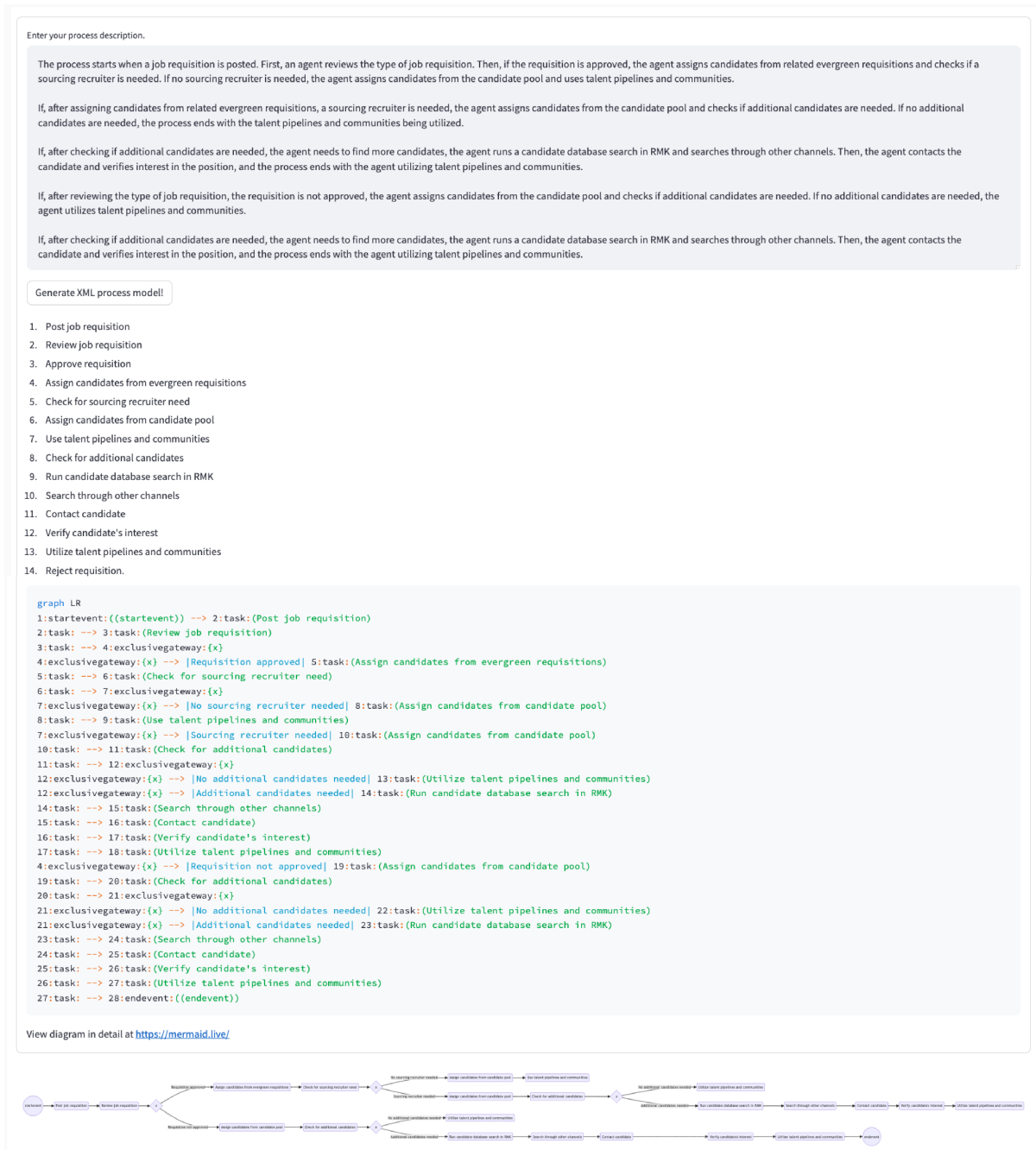
Figure 3: Screenshot of the Mermaid.js approach prototype

See the outputted graph also in Figure 10.

Figure 4: Screenshot of the Generating Traces & Process Mining approach prototype
See the outputted diagram in Figure 2.

7 Discussion

Summary   Aggregating all results, the approaches Direct Translation and Intermediate Graph Representation are not suitable for the current POC project. However, the Generating Traces & Process Mining approach solves all the crucial problems of the task. It extracts the activities and the control flow and always outputs SAP Signavio-compliant BPMN 2.0 JSON code. The approach is also very efficient in terms of implementation effort as all its functionality is handled by two existing services (GPT-4 and the SAP Signavio Split Miner service). Meanwhile, it does not extract additional information as described in the problem statement and still struggles with more difficult process descriptions. But, the approach also leaves room for further development and improvement (see Future Work). It therefore is a viable solution for the development of a POC.

Future Work   Some next steps regarding the Generating Traces & Process Mining approach that can be the subject of future work are:

- Add a post-processing step to enhance the diagram with additional information using another LLM request, given the process description and generated diagram code

- Gather more (diverse) process descriptions to systematically examine edge cases and malfunctions

- Implement an automated round-trip evaluation after defining a set of formal metrics

- Optimize the used prompts and assess the application of prompt chaining

- Experiment with human-in-the-loop approaches:

    - Suggest multiple possible process variants and allow the user to select

    - Allow the user to edit extracted traces, following conversational modeling [11]

The work, presented in this report will be continued and advanced.

8 Conclusion

In conclusion, the task of extracting BPMN process models from unstructured textual descriptions was investigated, motivated by the need of large companies to leverage their hidden process knowledge stored in internal documents. Three different approaches using LLMs were presented and assessed, namely Direct Translation, Intermediate Graph Representation, and Generating Traces & Process Mining.

The first two approaches were not feasible for the current proof of concept project, as they required too much effort and failed to produce a compliant output format. The Generating Traces & Process Mining approach, on the other hand, was able to solve the crucial problems of the task, such as extracting the activities and the control flow, always outputting Signavio-compliant JSON code and being very efficient in terms of implementation effort. However, the approach does not extract additional information and still struggles with more difficult process descriptions, but it also leaves room for further development and improvement.

Bibliography

[1] A. Augusto, R. Conforti, M. Dumas, M. La Rosa, and A. Polyvyanyy. Split miner: automated discovery of accurate and simple business process models from event logs. *Knowledge and Information Systems*, 59, 05 2019. doi: 10.1007/s10115-018-1214-x.

[2] P. Bellan, M. Dragoni, and C. Ghidini. Process extraction from text: state of the art and challenges for the future. *CoRR*, abs/2110.03754, 2021. URL `https://arxiv.org/abs/2110.03754`.

[3] P. Bellan, M. Dragoni, and C. Ghidini. *Extracting Business Process Entities and Relations from Text Using Pre-trained Language Models and In-Context Learning*, pages 182–199. 09 2022. ISBN 978-3-031-17603-6. doi: 10.1007/978-3-031-17604-3_11.

[4] Contributors to Mermaid. Mermaid: Generate diagrams from markdown-like text, 2014. URL `https://mermaid.js.org/`.

[5] Contributors to Streamlit. A faster way to build and share data apps., 2023. URL `https://github.com/streamlit/streamlit`.

[6] S. Dunzer, M. Stierle, M. Matzner, and S. Baier. Conformance checking: A state-of-the-art literature review. In *Proceedings of the 11th International Conference on Subject-Oriented Business Process Management*, S-BPM ONE '19, New York, NY, USA, 2019. Association for Computing Machinery. ISBN 9781450362504. doi: 10.1145/3329007.3329014. URL `https://doi.org/10.1145/3329007.3329014`.

[7] M. Franceschetti, R. Seiger, H. A. López, A. Burattin, L. García-Bañuelos, and B. Weber. A characterisation of ambiguity in bpm. In J. P. A. Almeida, J. Borbinha, G. Guizzardi, S. Link, and J. Zdravkovic, editors, *Conceptual Modeling*, pages 277–295, Cham, 2023. Springer Nature Switzerland. ISBN 978-3-031-47262-6.

[8] F. Friedrich, J. Mendling, and F. Puhlmann. Process model generation from natural language text. In H. Mouratidis and C. Rolland, editors, *Advanced Information Systems Engineering*, pages 482–496, Berlin, Heidelberg, 2011. Springer Berlin Heidelberg. ISBN 978-3-642-21640-4.

[9] M. Grohs, L. Abb, N. Elsayed, and J.-R. Rehse. Large language models can accomplish business process management tasks, 2023.

[10] T. Kampik, C. Warmuth, A. Rebmann, R. Agam, L. N. P. Egger, A. Gerber, J. Hoffart, J. Kolk, P. Herzig, G. Decker, H. van der Aa, A. Polyvyanyy, S. Rinderle-Ma, I. Weber, and M. Weidlich. Large process models: Business process management in the age of generative ai, 2023.

[11] N. Klievtsova, J.-V. Benzin, T. Kampik, J. Mangler, and S. Rinderle-Ma. Conversational process modelling: State of the art, applications, and implications in practice, 2023.

[12] B. Maqbool, F. Azam, M. W. Anwar, W. H. Butt, J. Zeb, I. Zafar, A. K. Nazir, and Z. Umair. A comprehensive investigation of bpmn models generation from textual requirements—techniques, tools and trends. In K. J. Kim and N. Baek, editors, *Information Science and Applications 2018*, pages 543–557, Singapore, 2019. Springer Singapore. ISBN 978-981-13-1056-0.

[13] Microsoft Azure. Azure openai service. `https://azure.microsoft.com/de-de/products/ai-services/openai-service`, 2024. Accessed: 2024-01-05.

[14] OMG. Business Process Model and Notation (BPMN), Version 2.0, January 2011. URL `https://www.omg.org/spec/BPMN/2.0/PDF`.

[15] P. Pietsch and S. Wenzel. Comparison of bpmn2 diagrams. In J. Mendling and M. Weidlich, editors, *Business Process Model and Notation*, pages 83–97, Berlin, Heidelberg, 2012. Springer Berlin Heidelberg. ISBN 978-3-642-33155-8.

[16] A. Suliman. Automatic multi-perspective conformance checking for bpmn models, 07 2018. URL `https://khazna.ku.ac.ae/files/6825449/file`.

[17] G. Van Rossum and F. L. Drake. *Python 3 Reference Manual*. CreateSpace, Scotts Valley, CA, 2009. ISBN 1441412697.

Appendix

```
graph LR
1:startevent:((startevent)) ---> 2:task:(Sit on the couch)
2:task: ---> 3:task:(Watch TV)
3:task: ---> 4:exclusivegateway:{x}
4:exclusivegateway:{x} ---> |Feeling hungry| 5:task:(Order a pizza)
4:exclusivegateway:{x} ---> |Not feeling hungry| 6:task:(Watch a movie)
5:task: ---> 7:endevent:((endevent))
6:task: ---> 7:endevent:
```

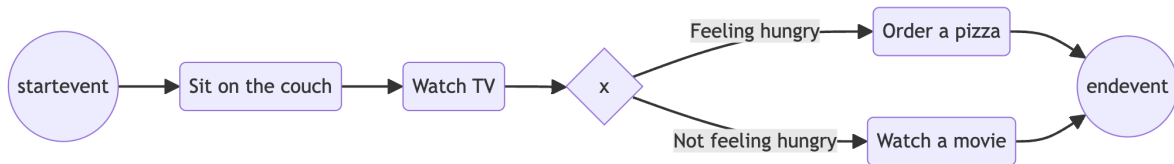Figure 5: Example Mermaid.js code for the graph in Figure 6

Figure 6: Example Mermaid.js graph for code in Figure 5

```json
1  {
2    "resourceId": "canvas",
3    "properties": {
4      "language": "English",
5      "targetnamespace": "http://www.signavio.com",
6      "expressionlanguage": "http://www.w3.org/TR/XPath",
7      "typelanguage": "http://www.w3.org/2001/XMLSchema",
8      "orientation": "horizontal"
9    },
10   "stencil": { "id": "BPMNDiagram" },
11   "childShapes": [
12     {
13       "resourceId": "sid-0BBA5180-9C3B-4DD7-83F3-8ACCDAAD2FAA",
14       "properties": {
15         "name": "Customer",
16         "boundaryvisible": true,
17         "adhocordering": "Parallel",
18         "minimum": 2,
19         "maximum": 2,
20         "bgcolor": "#ffffff",
21         "processid": "sid-7FAC670C-61CF-48EF-9492-E461272E84F1",
22         "bordercolor": "#000000"
23       },
24       "stencil": { "id": "Pool" },
25       "childShapes": [
26         {
27           "resourceId": "sid-2800EB56-5B34-4752-BFB3-6F5248BFDCCB",
28           "properties": { "bordercolor": "#000000" },
29           "stencil": { "id": "Lane" },
30           "childShapes": [
31             {
32               "resourceId": "sid-ECBFB2DA-1F4D-430B-AE6E-64BB245825DB",
33               "properties": {
34                 "name": "Start shopping tour",
35                 "dataoutputassociations_catchevents": "",
36                 "bgcolor": "#ffffff",
37                 "bordercolor": "#000000",
38                 "applyincalc": true
39               },
40               "stencil": { "id": "StartNoneEvent" },
41               "outgoing": [
42                 { "resourceId": "sid-00F35150-C1FC-4201-B3A8-41076724E7B5" }
43               ],
44               "bounds": {
45                 "lowerRight": {
46                   "x": 90.31555240012699,
47                   "y": 139.99999489254466
48                 },
49                 "upperLeft": { "x": 60.31555240012699, "y": 109.99999489254466 }
50               },
51             },
```

Figure 7: Extract of SAP Signavio-compliant BPMN 2.0 JSON Code

The first 50 lines of the JSON code are shown. A contained ID and the included coordinates of an element are highlighted. In the following code (not shown), more BPMN 2.0 elements follow as part of the "childShapes" array in line 25 as well as an array of associations.

Process description: {process_description}
Rules for mermaid js flowcharts:
The graph must use the LR (Left to Right) direction.
Each mermaid js node must have the following structure:
id:type:shape and text
    id − it is a unique identifier. Integer from 1 to n. Each node has a unique
        identifier
    type − defines the type of the element regarding to BPMN 2.0 notation.
        possible types are: start event, end event, task, exclusive gateway and parallel
           gateway.
        Based on the type of the node following shapes and texts are to be used:
        startevent: ((startevent))   i.e., id:startevent:((startevent))
        endevent: ((endevent))   i.e., id:endevent:((endevent))
        task: (task label)   i.e., id:task:(task label)
        exclusivegateway: {{x}}   i.e., id:exclusivegateway:{{x}}
        parallelgateway: {{AND}}   i.e., id:exclusivegateway:{{AND}}
All nodes that have occurred more than once should have following structure: id:type: (i
    .e., 2:task:) by futher occurrence.
It is strictly prohibited to use only id (i.e. 2) as a reference.
    all elements are connected with each other with the help of the direction.
        direction: −−>
    if there are some conditions or annotations it is necessary to use text on links (i.
      e., edge labels)
        edge label: |condition or annotation|
    edge label is always located between 2 nodes: id:exclusivegateway:{{x}} −−> |
      condition or annotation|id:task:(task label).
You can represent a decision point in the process where it branches into multiple paths
    by using either an 'exclusive gateway' or a 'parallel gateway'. This is known as '
    splitting' the control flow. Likewise, there could be points in the process where
    multiple paths merge back into one. This is known as 'joining' the control flow,
    which can also be done using either an 'exclusive gateway' or a 'parallel gateway'.
When rejoining branched paths, it's essential to use the same type of gateway that was
    used to create the branch initially. For instance, if an exclusive gateway was used
    to split the flow, use an exclusive gateway to merge it back.
Ensure that there are no duplicate sequences of tasks in your flowchart. If the same
    sequence of tasks happens rergardless of the path taken after a decision point,
    merge these paths back together using a gateway before proceeding with these tasks.
    This method avoids redundancy and enhances accuracy in your process representation.
Consider the following example of a process description and the generated mermaid.js
    graph.
Process description:
I sit at home in the evening.
If I am hungry, I am going to order a pizza and pay. If not, I am watching a movie.
Finally, I will go to bed and sleep.
Mermaid.js graph:
```mermaid
graph LR
1:startevent:((startevent)) −−> 2:task:(Sitting at home)
2:task: −−> 3:exclusivegateway:{{x}}
3:exclusivegateway:{{x}} −−> |Feeling hungry| 4:task:(Ordering a pizza)
4:task: −−> 5:task:(Paying for pizza)
3:exclusivegateway:{{x}} −−> |Not feeling hungry| 6:task:(Watching a movie)
5:task: −−> 7:exclusivegateway:{{x}}
6:task: −−> 7:exclusivegateway:
7:exclusivegateway: −−> 8:task:(Going to bed)
8:task: −−> 9:task:(Sleeping)
9:task: −−> 10:endevent:((endevent))
```
Activities: Considering the provided process description, create the list of activities
    (each 3−5 words maximum) one by one without any additional information. Strictly
    avoid duplicates. Do not output it.
Considering the provided process description, a set of custom rules and the generated
    list of activities, create a valid mermaid.js graph.

Figure 8: Prompt for generating a Mermaid.js graph

The process starts when a job requisition is posted. First, an agent reviews the type of job requisition. Then, if the requisition is approved, the agent assigns candidates from related evergreen requisitions and checks if a sourcing recruiter is needed. If no sourcing recruiter is needed, the agent assigns candidates from the candidate pool and uses talent pipelines and communities.

If, after assigning candidates from related evergreen requisitions, a sourcing recruiter is needed, the agent assigns candidates from the candidate pool and checks if additional candidates are needed. If no additional candidates are needed, the process ends with the talent pipelines and communities being utilized.

If, after checking if additional candidates are needed, the agent needs to find more candidates, the agent runs a candidate database search in RMK and searches through other channels. Then, the agent contacts the candidate and verifies interest in the position, and the process ends with the agent utilizing talent pipelines and communities.

If, after reviewing the type of job requisition, the requisition is not approved, the agent assigns candidates from the candidate pool and checks if additional candidates are needed. If no additional candidates are needed, the agent utilizes talent pipelines and communities.

If, after checking if additional candidates are needed, the agent needs to find more candidates, the agent runs a candidate database search in RMK and searches through other channels. Then, the agent contacts the candidate and verifies interest in the position, and the process ends with the agent utilizing talent pipelines and communities.

Figure 9: Example process description

Used as input to evaluate the Mermaid.js approach (see section 4.2). See the generated output in Figure 10.
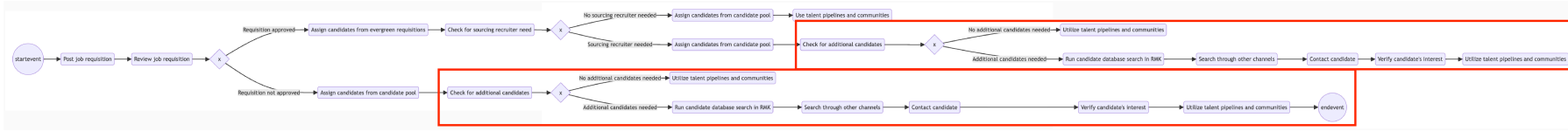
Figure 10: Generated Mermaid.js graph

Output of the Mermaid.js approach (see section 4.2) for the process description in Figure 9. Two duplicate sequence flows are highlighted.

You will be given a textual description of a business process. You should analyze this
    assignment and extract all potential process variants embedded in the narrative. An
    activity is a discrete, identifiable step in a business process. Your objective is
    to assign a short, distinct title for each activity, focusing only on real tasks
    that are executed, not regarding descriptions of decisions or associations.
The process variant is a potential path through the business process and should be
    represented as an array. Each string in this array stands for an activity or task in
     that variant — you should list them in the sequence that they appear in that
    version.
Your final output should be an array filled with different process variants you have
    identified.

To help you better understand what we're asking for, consider this example:

Given process description: "The business sequence begins with the acceptance of a
    purchase request from a customer. The sequence then diverts into two possible routes
    : (1) instantly proceeding with the order or (2) holding the order until inventory
    is replenished. The two routes then merge back into the task of packing the order
    for shipment. Finally, the sequence concludes with delivering the order to the
    consumer."

Expected output:
[
    [
        "Accepting purchase request",
        "Instantly processing order",
        "Order packing",
        "Order delivery"
    ],
    [
        "Accepting purchase request",
        "Holding order",
        "Order packing",
        "Order delivery"
    ]
]

Keeping this illustration in mind, extract all possible process variants from the
    following business process description:
{process_description}

Figure 11: Prompt for extracting a unique set of traces from a process description

```
[
  [
    "Making interest payment",
    "Deciding payment method",
    "Posting bank payment to general ledger"
  ],
  [
    "Making interest payment",
    "Deciding payment method",
    "Displaying payment schedule",
    "Generating payment request",
    "Processing payment request",
    "Posting manual payment to general ledger"
  ]
]
```

Figure 12: Example of a unique set of traces.

```
The process starts when a payroll data alert is assigned. First, the agent assesses the
    alert. Then, if the alert does not require a data change, the agent provides a
    comment and closes the alert. The process ends with the alert having been processed.

If, after assessing the alert, the alert requires data change, the agent aligns with the
    line manager and/or employee, checks if the data needs to be updated in the source
    system, and checks if the data needs to be updated in the source system. If the data
    does not need to be updated in the source system, the agent corrects the payroll−
    specific data, provides a comment, and closes the alert. Then, the process ends with
    the alert having been processed.

If, after checking if the data needs to be updated in the source system, the data does
    need to be updated, the agent aligns with the source system administrator, corrects
    the payroll−specific data, provides a comment, and closes the alert. Then, the
    process ends with the payroll data alert having been processed.
```

Figure 13: Tested input process description for diagram in Figure 2.

```
[
  [
    "Assigning payroll data alert",
    "Assessing alert",
    "Providing comment",
    "Closing alert",
    "Processing alert"
  ],
  [
    "Assigning payroll data alert",
    "Assessing alert",
    "Aligning with line manager/employee",
    "Checking data in source system",
    "Correcting payroll−specific data",
    "Providing comment",
    "Closing alert",
    "Processing alert"
  ],
  [
    "Assigning payroll data alert",
    "Assessing alert",
    "Aligning with line manager/employee",
    "Checking data in source system",
    "Aligning with source system administrator",
    "Correcting payroll−specific data",
    "Providing comment",
    "Closing alert",
    "Processing alert"
  ]
]
```

Figure 14: Generated set of traces (artificial event log) for the input process description in Figure 13.

Blizzard creates a cool online tool for creating characters for their new WoW expansion. When creating a World of Warcraft character, you can start doing two things: While you are setting up your account, you can already come up with good character names. The setup of your account starts with checking whether you have a battle.net account. If you do not have one yet, you enter the account information and click the link you receive in the confirmation mail. As soon as you have a battle.net account, you can check if you have an active WoW subscription. If not, you can select the payment method. If you choose credit card, enter your credit card information. If you choose your bank account, enter your IBAN and BIC numbers. After that you can log into the game and select realm, race and class of your character. Until now, you should have come up with some good names. You enter them one by one until a name is still available. You get a confirmation, and some selfies of your character, as soon as a expansion is released you get another message.

Figure 15: Example of a more complex process description

Declaration of Authorship

I hereby declare that the thesis submitted is my own unaided work. All direct or indirect sources used are acknowledged as references. I am aware that the thesis in digital form can be examined for the use of unauthorized aid and in order to determine whether the thesis as a whole or parts incorporated in it may be deemed as plagiarism. For the comparison of my work with existing sources, I agree that it shall be entered in a database where it shall also remain after examination, to enable comparison with future theses submitted. Further rights of reproduction and usage, however, are not granted here.

This paper was not previously presented to another examination board and has not been published.

_____   _____
City, date   Bruno Zirnstein