Revisiting Business Process Analysis through the lens of Large Language Models: Prompting experiments with BPMN process serializations

Damaris Dolha[a,] , Ana-Maria Ghiran[a] and Robert Andrei Buchmann[a]

Babeș-Bolyai University, Faculty of Economics and Business Administration, 58-60 T. Mihali St.,
Cluj-Napoca 400591, Romania

E-mails:
[a] *damarisdolha@gmail.com,*
[b] *anamaria.ghiran@econ.ubbcluj.ro,*
[c] *robert.buchmann@econ.ubbcluj.ro*

Abstract. Recent position papers have proposed that the traditional BPM (Business Process Management) lifecycle must be revisited considering recent generative AI advances, specifically by investigating how LLMs (Large Language Models) can assist various lifecycle phases. Inspired by that call to action, this paper reports on a series of experiments on how ChatGPT can be used to query the content of BPMN diagrams as potential support for the Analysis phase of the BPM lifecycle. We are particularly interested in how BPMN content - typically available in enterprises that adopted the BPM lifecycle - should be exposed to LLM services. We comparatively experiment with diagrams provided as XML serializations (standard-based) or as RDF serializations (tool-specific) - i.e., a comparison between a standard serialization characterized by intricate cross-referencing to compensate for the XML hierarchical structure and the "semantic graph" view of RDF that is open-ended in terms of semantic annotation and can be serialized as statements that resemble natural language. Quality of answers is assessed using the RAGAs framework.

Keywords: BPMN, Generative AI, RDF, XML, Bee-Up, Signavio, RAGAs

1. Introduction

The paper reports on continuous experimentation with how selected Large Language Models (LLM) services interpret Business Process Model and Notation (BPMN) models exposed in different formats – as standard XML serializations or as non-standard tool-specific RDF graphs. This work can inform how we engage with an AI-powered version of the Business Process Management (BPM) lifecycle, leveraging augmentations made possible by LLM services for certain phases of the lifecycle.

BPMN remains the dominant standard for documenting processes, predominantly relying on XML for diagram serializations – e.g., XPDL, BPEL-WS, BPMN XML, ADOXML. A large diversity of process-aware systems or engines [7] have been built on XML schemas to parse process descriptions. However, in the context of semantics-driven engineering as formulated by [5], the requirement of process understandability becomes just as relevant as process instantiation and execution; or, the instantiation must be agilely tailored to a semantically rich context, possibly available in the form of knowledge graphs. Towards such needs, the introduction of RDF (Resource Description Framework) encoding of BPMN content offers new semantics-based capabilities. However, such representations are not widely adopted and are tool-specific – examples are the Bee-Up modeling tool [13] and various demonstrators reported by the literature [3]. Our research explores the use of XML and RDF serializations of BPMN with generative Artificial Intelligence (GenAI), complemented by multi-modal capabilities of visually interpreting the diagrams. The experiments harness the Large Language Models (LLMs) provided by OpenAI - by means of targeted prompts we probe various aspects of selected BPMN exemplars, to see how RDF compares with XML in facilitating "artificial understanding" of processes, despite it not providing a standard vocabulary such as those available to all XML formats. Although the integration of graph databases as process storage and BPMN knowledge structures is still in experimental stages [3,24], we highlight the potential of RDF to facilitate semantically rich dialogue for process analysis with LLMs, beyond the "BPMN as closed world data structure" treatment offered by standard XML exports.

This research aligns with a current stream of investigation into the capability of AI to engage with conceptual models [10]. It also furthers the investigation into how LLM services can understand and process serialized diagrammatic models, building upon initial prompting strategies we have reported in a conference scope [1]. The comparative analysis in this version also extends across a more diverse set of scenarios, thereby expanding the findings representativeness and insights.

We focus on process serializations, rather than images, because most BPMS (business process management systems) or BPA (business process automation) platforms and services rely on serializations for process repositories and model interchange, as they require deterministic interoperability or execution. The introduction of image recognition capabilities by the GPT-4 Open-AI model significantly enhances the multi-modal interaction possibilities [12], enabling a more holistic approach to understanding and interacting with BPMN models. However visualization cannot fully grasp a process description – many details relevant to process analysis do not manifest on a visual level (e.g., data attributes, links between diagrams).

The paper is organized as follows: in Section 2, we establish the problem scope in the context of the BPM lifecycle based on our reading of recent literature. Next, Section 3 summarizes the experimental setup and explains the structural and syntactical differences between XML and RDF. The core findings of our experimental study are reported in Sections 4.1 and 4.2; the first analysis a full complex example, the latter focuses on minimalist BPMN patterns that are left unexplicit to force ChatGPT to reason independently of the textual labelling found in diagrams. Concluding this paper, Section 5 summarizes the findings and maps out directions for future exploration.

2. Large Language Models and the BPM lifecycle

Our investigation employs LLMs for the semantic analysis of procedural knowledge, which is quintessential to the BPM lifecycle. This has traditionally relied on process queries, which are traditionally based on graph queries [3,24], formal languages [2] or visual grammars [11]. Now, they must be revisited in relation to LLMs radically departing from the conventions of pre-GPT process analysis tools. Recent works [25] draw attention to how these models can redefine how we think about the phases of the BPM lifecycle: in the *process identification* phase, LLMs cut through the clutter of unstructured data: they do not just find information, they can gather workflow knowledge - moving into *process discovery*, the influence of GenAI can enhance process mining frameworks. Traditionally tethered to XML event logs, RDF can push graph-based process mining [18]. When it comes to *querying processes*, XML frameworks and XPath are now over-hauled by multi-modal AI such as GPT-4 [17], with their computer vision capabilities; yet, reliance on computer vision has its limits and must be complemented by a semantic serialization layer to also capture non-visual aspects that remain semantically relevant – e.g., links between different models (e.g., RACI responsibilities on task level) or data attributes (e.g., task costs). This is where the current report focuses. In the *redesign* phase, AI can advocate changes, using code generation on serializations to apply workflow updates. As those updates are implemented, LLMs enrich user interactions with detailed explanations, shifting emphasis from static work-flow sequences to dynamic conversational choreographies. During the *monitoring* phase, LLMs are not confined to data display, they can interpret and analyze data. The current report focusses on the process querying and interpretation, crucial stages in process analysis, that require means of question answering and reasoning on the process semantics.

As AI pushes the boundaries of Business Process Management, recent technological proposals change the way processes can be analyzed. The BPMN2KG initiative [3] illustrates the conversion of BPMN 2.0 models into knowledge graphs and marks a different view on the instantiation of process models. Similarly, the conversion tool of [24] allowed BPMN XML formats to morph into Neo4J labeled property graphs (LPG), providing an alternative graph representation that adheres to the BPMN 2.0 vocabulary. The work of [11] has ventured into the application of LLMs for textual analyses within BPM. This study tests the effectiveness of LLMs, particularly GPT-4, in deriving both imperative and declarative process models from text, demonstrating a substantial advance regarding the role of AI in process querying, as further supported by findings in [21]. Analyzing the effectiveness of ChatGPT in generating and deciphering diverse conceptual models [10] suggested certain operational nuances, while another evaluation [6] looked into prompt engineering for business process management. The integration of GPT-3.5 Turbo with Lean Six Sigma 4.0 methodologies [23] unveils the potential for upgrading customer service and decision-making processes in real time. Moreover, the intersection of GPT technology with Robotic Process Automation (RPA) is scrutinized for potential risks and opportunities, especially concerning security and compliance issues [19]. Addressing the intrinsic challenges that come with LLMs, forward-thinking strategies have been proposed that involve merging LLMs with knowledge graphs [26]. This innovative approach is particularly promising when these graphs are designed around diagrammatic procedural knowledge effectively serialized as semantic networks.

3. Experimental Setup

At the forefront of our study is Bee-Up 1.7 [13], a core component of the OMiLAB Digital Innovation environment [20], known for its role in enhancing both standard-based and domain-specific models with rich semantics, allowing the exploration of inter-links (semantic bridges) among a diversity of modeling languages – e.g., BPMN, UML, DMN, EPC and Petri Nets. SAP Signavio [22] was used as a representative for tools providing the standard BPMN 2.0 XML serialization. Both tools provide image exports as PNG and other image formats.

3.1. Serialization comparison

The difference between the XML and RDF serializations is explained here based on the sample diagram in Figure 1, which shows a minimal subprocess linked to a main process with pool and lanes containment, as well as labelled connectors outgoing from a gateway.

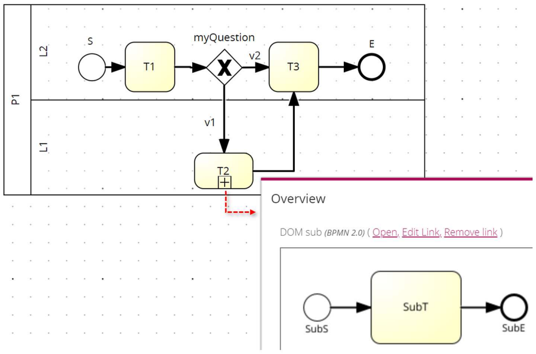
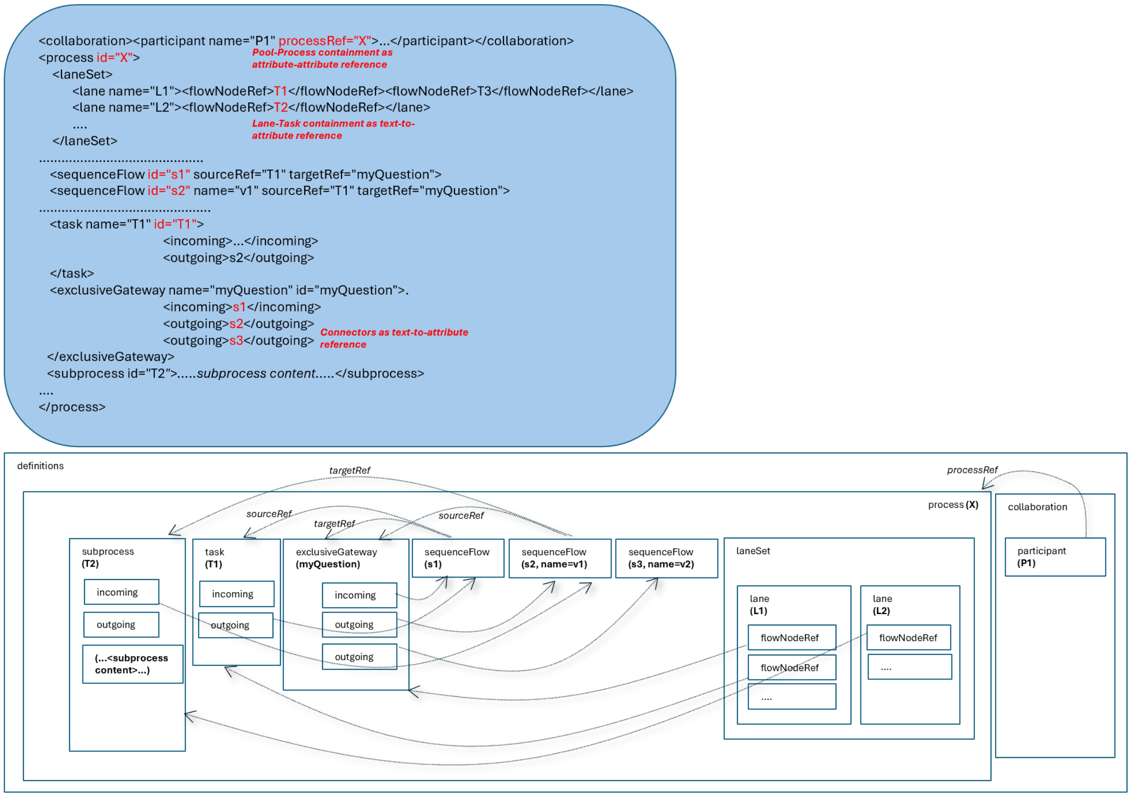


Fig. 1. BPMN exemplar comprising a link between a main process and a minimalist subprocess.

Figure 2 provides an overview of the hierarchical, DOM-based structured available in the standard serialization. However, because the hierarchical decomposition does not reflect the visual directed graph structure, an intricate network of cross-references across XML tags preserves all relevant relationships. Some of these are attribute-attribute matches (e.g., to specify containment between a pool and the process inside it), others are attribute-tag matches (e.g., to specify containment between a lane and its flow elements but also visual connectors referencing their connector heads), one-to-one (e.g., an arrow can have one starting point and one ending point) or one-to-many matches (e.g., a gateway has multiple outgoing connectors). Only a few relationships are expressed by the implicit parent-child XML nesting that would be familiar to a basic parser reading such content sequentially: a process containing lanes, a subprocess containing its contents. XML parsers can navigate such complex cross-references as prescribed by the governing XML schema, but process interpretation by a linear token-based parser reading it as textual content requires frequent back and forth jumps based on heterogeneous matching rules.



```
<collaboration><participant name="P1" processRef="X">...</participant></collaboration>
<process id="X">
  <laneSet>
      <lane name="L1"><flowNodeRef>T1</flowNodeRef><flowNodeRef>T3</flowNodeRef></lane>
      <lane name="L2"><flowNodeRef>T2</flowNodeRef></lane>
      ....
  </laneSet>
............................................
  <sequenceFlow id="s1" sourceRef="T1" targetRef="myQuestion">
  <sequenceFlow id="s2" name="v1" sourceRef="T1" targetRef="myQuestion">
.............................................
  <task name="T1" id="T1">
                     <incoming>...</incoming>
                     <outgoing>s2</outgoing>
  </task>
  <exclusiveGateway name="myQuestion" id="myQuestion">.
                     <incoming>s1</incoming>
                     <outgoing>s2</outgoing>
                     <outgoing>s3</outgoing>
 </exclusiveGateway>
 <subprocess id="T2">.....subprocess content.....</subprocess>
....
</process>
```

Pool-Process containment as attribute-attribute reference

Lane-Task containment as text-to-attribute reference

Connectors as text-to-attribute reference

Fig. 2. Insights in the DOM-based structure of the standard BPMN XML serialization (based on the exemplar in Figure 1).

Comparatively, Figure 3 showcases the graph structure of an RDF serialization, derived from transformation patterns initially formulated in [4]. The graph edges represent visual connectors, hyperlinks (e.g., links to subprocesses), containment relationships (to lanes, pools), any data attributes editable in the tool (e.g., simulation attributes) and also open-ended properties that can be attached as semantic annotations to any diagrammatic element (not used in this example). Some visual connectors to emulate the "property graph" approach are necessary when graph edges need to have their own properties – see the outgoing arrows from the XOR gateway, which need to be labelled differently. The connector is therefore reified to hold any attributes that are distinctively set for one instance of that connector. SPARQL rules and filters can be used to conveniently query connectors in the simple, non-annotated form, or in the reified, annotated form, depending on the process query needs. Subprocesses are isolated as separate named graphs, however linked within the same RDF dataset. Based on such patterns, an RDF export is available in the Bee-Up modeling tool, with some terminological aspects not detailed here (namespaces). On the top of the figure, TriG/Turtle statements show the process as it is serialized and grouped by the diagram graph where the statements belong. Many other attributes can be exported, not visible here as they are not relevant to semantically-oriented process queries (e.g., visual position) and are also filtered out by our component that delivers such graphs to ChatGPT.

```
:MainProcessModel {
                    :P1 a :Pool.
                    :L1 a :Lane; :isInside  :P1. :L2 a :Lane; :isInside :P1.
                    ...
                    :T1 a :Task; :isInside :L2; :followedBy :myQuestion.
                    :myQuestion a :XORGateway; :isInside :L2; :followedBy :T2, :T3.
                    :x a :ReifiedConnector; :from :myQuestion; :to :T2; :label "v1".
                    :T2 a :Subprocess; :referencedSubprocess  :SubProcessModel.
                    ...
                    }

:SubProcessModel {.........subprocess content..........}
```

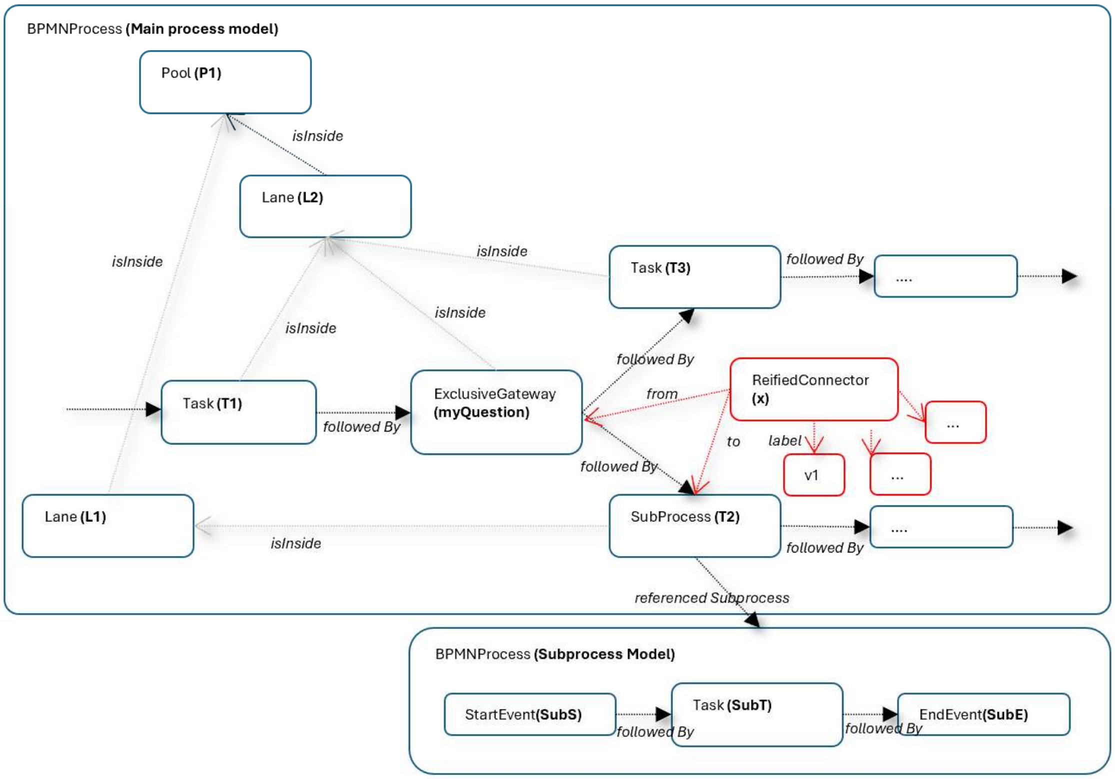


Fig. 3. Insights in the graph structure of the RDF serialization (based on the exemplar in Figure 1).

3.2. Model exemplars used in experiments

We focused our experiments on two kinds of BPMN models: (1) a realistic model of a main process linked to a subprocess, as illustrated in Figure 4; (2) a set of minimalist process patterns (Figures 5-6) that are labelled non-explicitly to allow us to probe process structure understanding while avoiding any business narrative that may be generated from scenarios unrelated to the actual process exemplars.

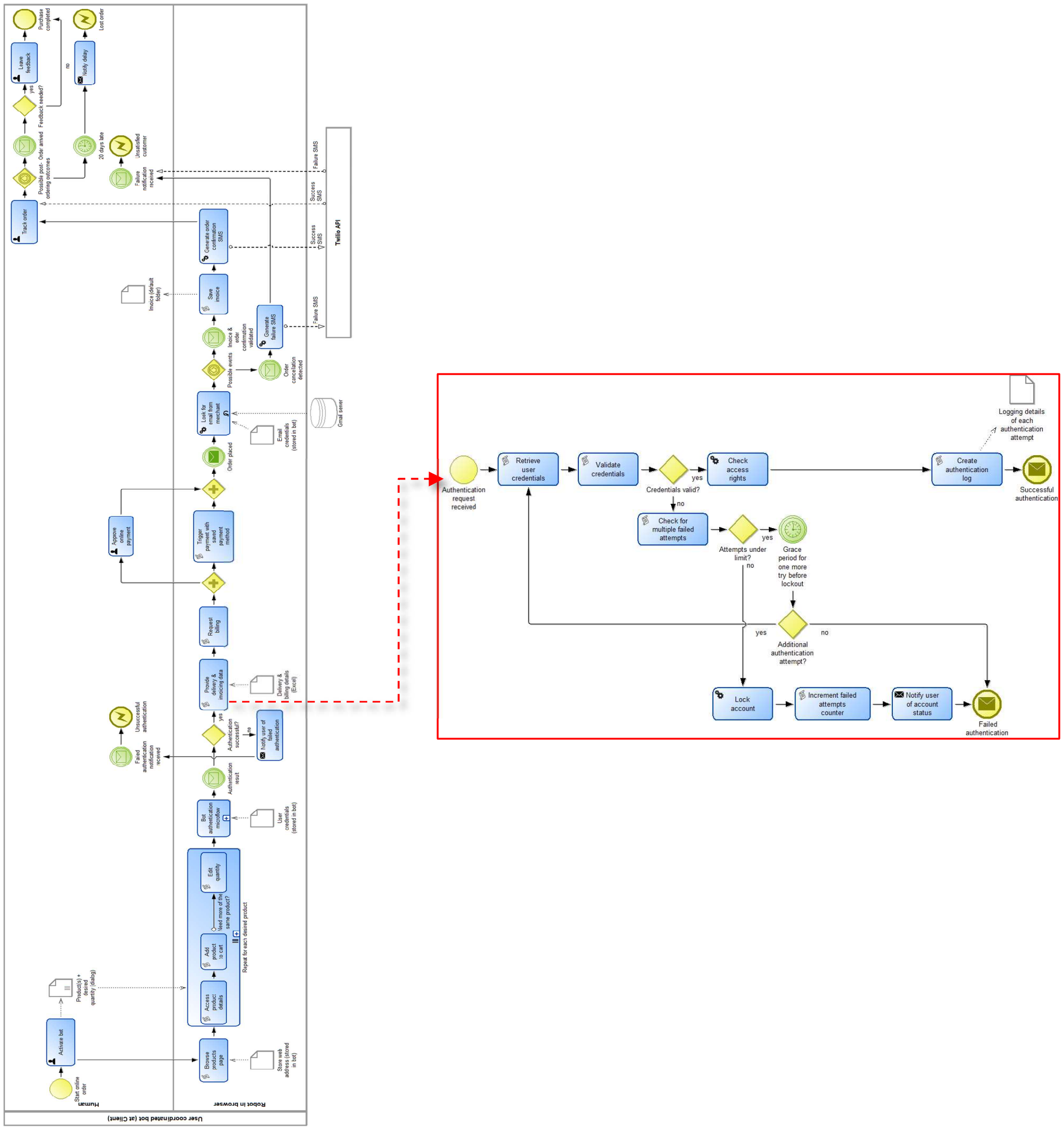
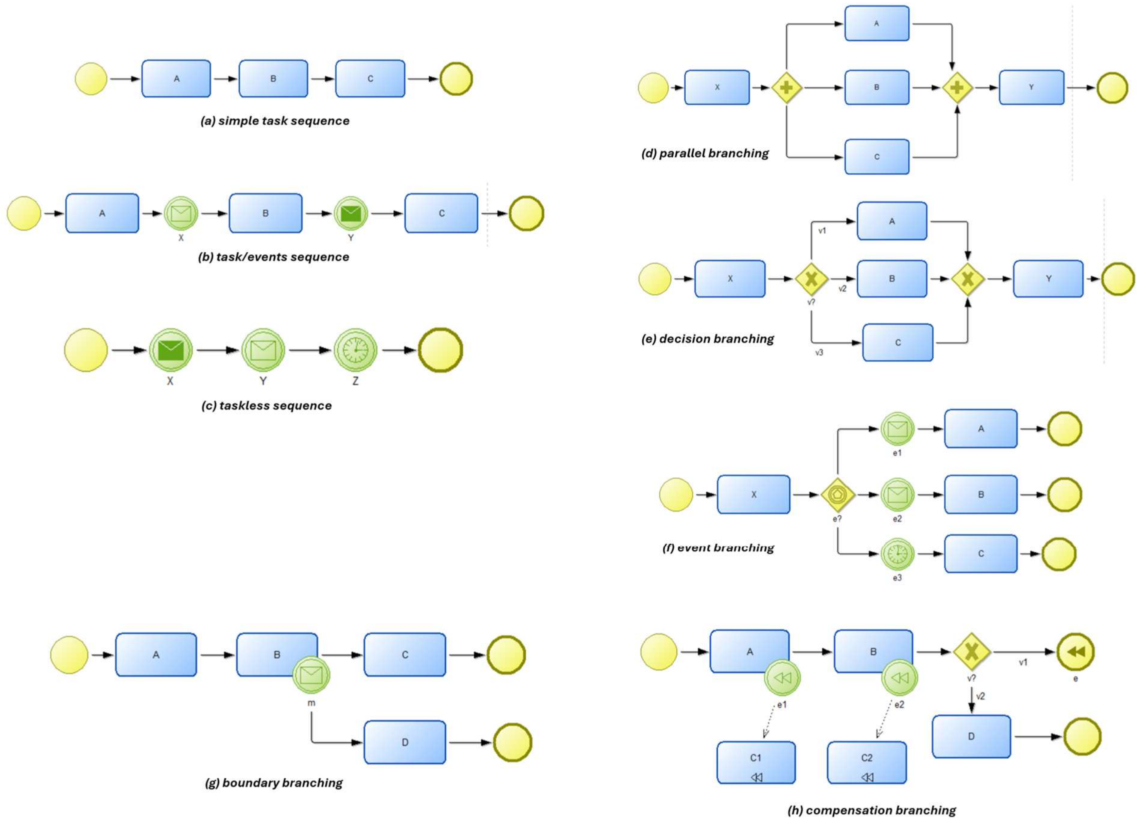

Authentication request received
Retrieve user credentials
Validate credentials
Credentials valid?
yes
no
Check access rights
Create authentication log
Logging details of each authentication attempt
Successful authentication
Check for multiple failed attempts
Attempts under limit?
yes
no
Grace period for one more try before lockout
Additional authentication attempt?
yes
no
Lock account
Increment failed attempts counter
Notify user of account status
Failed authentication

Fig. 4. BPMN diagrams depicting the main BPMN process and the "Bot authentication microflow" subprocess (Bee-Up 1.7).

We only showcase in Figures 4-6 the Bee-Up variants, having equivalent elements to the Signavio variants.

The realistic process model in Figure 4 depicts the logic of an RPA bot planned to mimic the human actions for on-line shopping, as well as the data requirements for the bot to accomplish such a task and the "human in the loop" interactions – including credentials needed to perform some authentication steps on behalf of the human.

The examples in Figures 5-6 follow a different strategy: only generic labels are visible, forcing the LLMs to look into the process structures instead of extrapolating on scenario narratives inspired by labels. This also helps us assess the structural and flow-based reasoning abilities that may manifest, driven by the different types of connectors and the semantics they carry – sequence flows, message flows, data associations.



(a) simple task sequence

(b) task/events sequence

(c) taskless sequence

(d) parallel branching

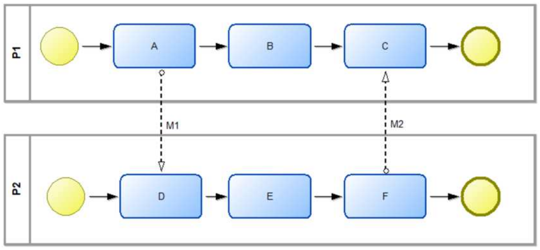
(e) decision branching

(f) event branching

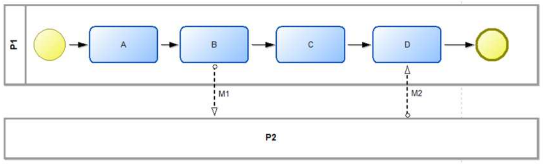
(g) boundary branching

(h) compensation branching

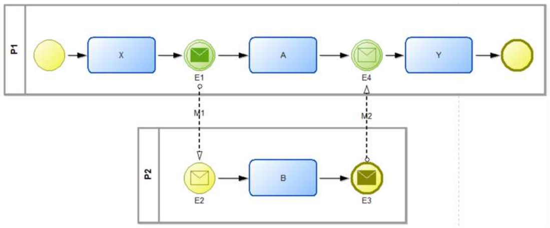
Fig. 5. Minimalist process modeling patterns in BPMN.



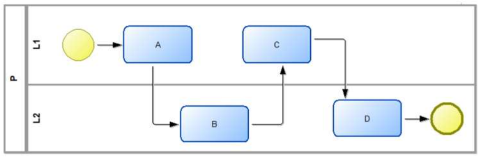
(i) task collaboration



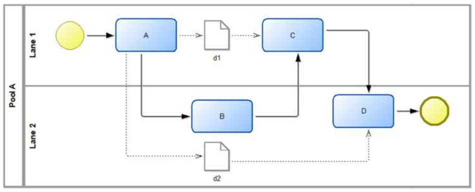
(j) collapsed pool collaboration



(k) event collaboration



(l) lane coordination



(m) lane coordination with data

Fig. 6. Collaborative and coordinated process patterns in BPMN.

The key feature leveraged in Bee-Up is the RDF export, which builds on previously introduced diagram-to-graph transformation patterns [4]. To parse and filter the RDF outputs as semantic graphs, we turned to Ontotext GraphDB 10.5 [15] after stripping away irrelevant attributes from RDF metadata. Ontotext GraphDB expands the standard SPARQL querying capabilities with OpenAI-oriented functions such as *gpt:ask()* [14] that engages directly with the GPT-4 model while exposing to it a convenient subgraph extracted from the RDF repository. The structure of the utilized query is shown below, filtering specific types of nodes and edges that are relevant for the process description (and not for its visualization):

```
#prefixes removed for concision
# <…> are placeholders for relevant graph edges or node types, # as well as
the user prompt

  SELECT ?answer ?rdfSer WHERE {
  SELECT (helper:rdf(helper:tuple(?x, ?prop, ?o)) AS ?rdf)
  WHERE {
            ?x a <RDF_class_identifier>; ?prop ?o.
            FILTER (?prop IN (<property1>, <property2>, <property3>,
…))
```

```
        }
  BIND(helper:serializeRDF(?rdf) as ?rdfSer)
  ?answer gpt:ask ("<user_prompt>" ?rdf)}
```

Another enabler of our analytical approach is **BPMN Analyst**, a custom GPT service using ChatGPT Plus [16], specifically engineered to dissect and interpret BPMN XML serializations from any BPMN imported files, through a series of trivial and non-trivial prompts. This strategic application of tailored AI tools enhances our capacity to analyze and understand BPMN content via LLM services.

4. Experimental Outcomes and Evaluation

We examine responses produced by GPT-4, during April-July 2024, to a variety of prompts probing its BPMN-based process analysis capabilities based on multi-modal content – the RDF representation, the XML standard serialization and exported images. In the first part, we refer by CASE I to the RDF variant and by CASE II to SAP Signavio exports. This approach is not intended to highlight limitations within any specific tool, but to explore the differences and potential insights that different serialization formats may reveal.

In the preliminary stages, the tools deployed in the aforementioned cases demonstrated adeptness in responding to straightforward inquiries, such as identifying participants within processes, elucidating the sequence of steps in the processes etc. These inquiries, predominantly aimed at recognizing basic BPMN elements established a baseline of competence.

We utilized the Retrieval Augmented Generation Assessment (RAGAs) framework [8] to compare the answers generated by the LLM, focusing on four key metrics and a reference ground truth: *faithfulness, answer relevancy, answer correctness* and *answer similarity* [9]. Starting with *Faithfulness*, this metric assesses the extent to which the claims made in a generated answer are supported by the information present in the BPMN model; an answer is considered faithful if it accurately reflects the processes and information modeled in the BPMN context without introducing inaccuracies or unsupported details. *Answer Relevance* measures how well the generated answer addresses the question posed, namely the prompt for each experiment. *Answer Correctness* evaluates the accuracy of the generated answer in relation to the ground truth. This metric looks at both the semantic and factual similarity between the response and the ground truth. High correctness indicates that the answer not only addresses the question correctly, but also aligns closely with the details within the BPMN model. *Answer Semantic Similarity* focuses on how closely the meaning and context of the response align with what was expected in the ground truth.

The XML process serialization offers a standard vocabulary, not only a standard structure, XML being traditionally a data interchange format. The RDF export of BPMN is only standard in the structural sense – as a directed semantic graph standard – and not as a process description vocabulary, for which intuitive tool-specific terms are generated by Bee-Up - derived from the concept labelling prescribed on metamodel level. Therefore, LLMs treat the RDF variant as a semantic network that happens to contain edges whose labels describe workflows, while the XML variant is recognized upfront as a standard interchange format widely available in training corpora and Web content.

The *contexts* array within our dataset is drawn from files provided either in TriG serialization format for RDF or BPMN format for XML, depending on the type of data representation needed. This distinction is crucial, as it ensures that our evaluations remain faithful to the actual context of our experiments, reflecting the true nature of the processes being analyzed. *ground_truth* in this dataset is equally significant, because it serves as the gold standard against which the generated answers are compared. Our findings are presented in the tables below.

4.1. Prompts on the explicit realistic example

Table 1

Prompt 1. Enumerate all activities after 'Add product to cart' action in the main process.

CASE I		**CASE II**	
Response generated by Ontotext GraphDB 10.5 using OpenAI's GPT-4 model on RDF formatted data		Response generated by *BPMN Analyst* on imported BPMN file	
1. Edit quantity 2. Provide delivery & invoicing data 3. Request billing 4. Trigger payment with saved payment method 5. Approve online payment 6. Look for email from merchant 7. Save invoice 8. Generate order confirmation SMS 9. Notify delay 10. Track order 11. Leave feedback 12. Generate failure SMS 13. Notify user of failed authentication 14. Activate bot		After the "Add product to cart" action in the main process, the next activity is "Edit quantity." If you need information on further activities or any other details from the process, feel free to ask!	
Ground truth	All the activities after "Add product to cart" action in the main process are: "Edit quantity", "Provide delivery & invoicing data" or "Notify user of failed authentication" (if applicable), "Request billing", "Trigger payment with saved payment method" and "Approve online payment", "Look for email from merchant", "Save invoice" or "Generate failure SMS" (if needed), "Generate order confirmation SMS", "Track order", "Leave feedback" or "Notify delay" (if applicable).		
RAGAs metrics			
Faithfulness	1.0	**Faithfulness**	0.926067
Answer Relevancy	0.778516	**Answer Relevancy**	0.339887
Answer Correctness	0.952734	**Answer Correctness**	0.5
Answer Similarity	0.90771	**Answer Similarity**	0.897952

The distinction between CASE I (RDF-formatted data) and CASE II (XML-formatted data) highlights significant differences in data handling and response generation. CASE I offers a rich, detailed response, leveraging the ability of RDF to use semantic relationships for a deep understanding of the process flow. This is evidenced by a perfect *faithfulness* score of 1.0, indicating that the generated answer can be accurately inferred from the provided BPMN process. However, its *answer relevancy* score of 0.778516, while lower than might be expected given the comprehensive response, suggests that the answer includes unnecessary and inaccurate details, such as the task "Activate bot", overshooting the needs of the prompt. Lastly, the high level of *answer correctness*, with a score of 0.952734, indicates a close alignment between the response and the ground truth. Conversely, CASE II presents a more streamlined response, focusing strictly on the immediate next step of the process. This approach, characteristic of the structured, tag-based processing of XML, captures data directly without the semantic depth of RDF. Although its *faithfulness* score is also high (at 0.926067), it indicates minor deviations from the complete process flow possibly expected in the response. The notably lower *answer relevancy* score of 0.339887 reflects a lack of comprehensive coverage of all subsequent activities, while the *answer correctness* of only 0.5 reveals that the response, while partially accurate, falls short in representing the depth and complexity of the BPMN process as required. The higher *similarity* score for CASE I (0.90771) suggests that it aligns more closely with the ground truth.

Table 2

Prompt 2. What happens if the authentication is not successful?

CASE I		**CASE II**	
If the authentication is not successful, the user will be notified of the failed authentication. This is represented by the task "Notify user of failed authentication". This notification will likely prompt the user to attempt again, possibly resetting their password if they have forgotten it. The exact steps taken will depend on the specific protocols of the authentication system being used.		If the authentication is not successful, the process flows to a "sendTask" where the user is notified of the failed authentication. This task explicitly handles the communication of the unsuccessful authentication result to the user.	
Ground truth	If the authentication is not successful, the user is notified of the failed authentication.		
RAGAs metrics			
Faithfulness	0.6	**Faithfulness**	0.0
Answer Relevancy	0.911374	**Answer Relevancy**	1.0
Answer Correctness	0.448622	**Answer Correctness**	0.529958
Answer Similarity	0.937347	**Answer Similarity**	0.919833

Faithfulness is a critical metric that measures the factual consistency of the response against the provided context. In the first case, the RDF-formatted data provided a structured context that aligns relatively well with the response, resulting in a *faithfulness* score of 0.6. This score indicates that while the response is largely aligned with the RDF context, it includes speculative steps like password reset, which are not part of the modeled process, thus reducing the score. For the second case, despite the correctness of the response, its *faithfulness* score was surprisingly low, at 0.0. The response might be correct in practical terms, but if it deviates in any way from the parsed context (e.g., due to incomplete or misinterpreted XML data), the faithfulness score would reflect this misalignment. In terms of *answer relevancy*, the response in CASE I is mostly relevant, with minor penalties for the additional, speculative details. Meanwhile, the perfect relevancy score in CASE II reflects the effectiveness of the response in addressing the core issue without introducing irrelevant details. Focusing on the next metric, namely *answer correctness*, the scores in both cases reflect responses that are nearly half correct. This is due to details that are not part of the modeled process. Concerning *answer similarity*, both CASE I and CASE II responses are semantically close to the ground truth, as indicated by their high answer similarity scores, although CASE I, with its detailed and contextual information, achieves a slightly higher score, demonstrating the benefits of RDF-formatted data in providing comprehensive responses.

Table 3

Prompt 3. What are the costs within the current authentication subprocess?

CASE I		**CASE II**
The costs within the current authentication subprocess are as follows: 1. Retrieve user credentials: $0.01 2. Validate credentials: $0.02 3. Check access rights: $0.03 4. Create authentication log: $0.01 5. Check for multiple failed attempts: $0.01 6. Lock account: $0.03 7. Increment failed attempts counter: $0.01 8. Notify user of account status: $0.02 9. Notify user of failed authentication: $0.02 Please note that these costs are likely representative of time, resources, and computational costs associated with each step in the subprocess.		The current authentication subprocess does not have any explicit cost-related metadata defined within the provided BPMN file. If you need detailed cost information, it might be necessary to check the process documentation or other relevant resources. If you have other specific queries about this subprocess or need to verify other elements, feel free to ask.
Ground truth	The costs within the current authentication subprocess are: Retrieve user credentials: $0.01, Validate credentials: $0.02, Check access rights: $0.03, Create authentication log: $0.01, Check for multiple failed attempts: $0.01, Lock account: $0.03, Increment failed attempts counter: $0.01, Notify user of account status: $0.02.	
RAGAs metrics		

Faithfulness	1.0	**Faithfulness**	0.0
Answer Relevancy	1.0	**Answer Relevancy**	0.0
Answer Correctness	0.749073	**Answer Correctness**	0.199727
Answer Similarity	0.996273	**Answer Similarity**	0.798452

This is based on costing attributes being annotated to the diagram elements. The RDF-based response (CASE I) has a perfect score in terms of *faithfulness* and provides an exhaustive breakdown of costs associated with each task in the evaluated subprocess, demonstrating explicit labeling and encoding of cost metadata alongside process elements. In comparison, the XML standard (CASE II), with a score of 0.0 with regard to this metric, adheres to a more semantically restricted, closed-world XML schema and does not connect it to attributes attached as arbitrary metadata or to Signavio dictionary assets (e.g., RACI links to organization roles) – although these can be exported separately and again linked by inter-tag key references or manually edited according to the generic metadata tags (with metaKey, metaValue attributes). *Answer relevancy* scores 0.0 in CASE II, proof that XML understanding seems to stick to generative content based on the standard XML schema, avoiding tool-specific extension; in contrast, even if the RDF export is tool-specific from the grounds up, it seems to be more organically navigated as a freely expandable network of labelled associations (with a perfect score for *answer relevancy*). This is somewhat ironical – as RDF has been often seen as a metadata standard rather than a graph database format – but for LLMs, it may be more semantically integrative than actual metadata based on intricate cross-references between keys. The relatively high *correctness* score of 0.749073 in CASE I indicates that while the response is mostly correct, it includes a minor assumption, namely the additional information on speculative nature of cost estimations, which affect the alignment between the generated answer and the ground truth. The lack of completeness and specificity in addressing the query results in a significantly lower *correctness* score (0.199727) for CASE II. The discrepancy between the scores related to the *answer similarity* RAGAs metric is not as drastic as it is regarding *answer correctness*; yet, CASE I stands out with a nearly perfect score of 0.996273, directly matching the ground truth.

4.2. Prompts on the Non-Explicit Minimalist Models

Building on the experiments outlined in Section 4.1, we now shift our focus to exploring minimalist, non-explicit patterns. Using the same framework and metrics, we examine the patterns illustrated in Figures 5 and 6 to uncover how these streamlined models – represented in RDF and XML – perform under a diversity of queries that include various perspectives, such as temporal understanding, participant involvement, interactions and associations.

Table 4

Prompt 4. List the tasks in the order described by the process.

CASE I		**CASE II**	
Response generated by Ontotext GraphDB 10.5 using OpenAI’s GPT-4 model on RDF formatted data		Response generated by *BPMN Analyst* on imported BPMN file	
1. Task A, 2. Task B, 3. Task C		The tasks in the order described by the process are: 1. Task A, 2. Task B, 3. Task C	
Ground truth	1. Task A, 2. Task B, 3. Task C		
RAGAs metrics			
Faithfulness	0.0	**Faithfulness**	0.0
Answer Relevancy	0.860502	**Answer Relevancy**	0.965969
Answer Correctness	1.0	**Answer Correctness**	0.539539

Answer Similarity	1.0	**Answer Similarity**	0.958105

Table 5

Prompt 5. Identify all the tasks that occur after event X.

CASE I		CASE II	
The tasks that occur after event X are task B and task C.		The tasks that occur after event X are: Task B, Task C	
Ground truth	The tasks that occur after event X are task B and task C.		
RAGAs metrics			
Faithfulness	0.5	**Faithfulness**	1.0
Answer Relevancy	0.960992	**Answer Relevancy**	0.962644
Answer Correctness	1.0	**Answer Correctness**	0.622142
Answer Similarity	1.0	**Answer Similarity**	0.988569

Table 6

Prompt 6. Identify the elements that are parallel with task C.

CASE I		CASE II	
The elements that are parallel with task C are task A and task B.		The tasks that are parallel with Task C are: Task A, Task B	
Ground truth	The elements that are parallel with task C are task A and task B.		
RAGAs metrics			
Faithfulness	0.0	**Faithfulness**	1.0
Answer Relevancy	0.939789	**Answer Relevancy**	0.93837
Answer Correctness	1.0	**Answer Correctness**	0.98921
Answer Similarity	1.0	**Answer Similarity**	0.956842

The generated responses to queries regarding simple task sequences (Figure 5-a), sequences with tasks and events (Figure 5-b) and decision branching patterns (Figure 5-e), though differing stylistically, are correct in both cases, dealing well with experiments focusing on temporal order. However, the RDF-based responses excel in *correctness* and *semantic similarity*, suggesting RDF's enabling role to infer relationships within the given patterns, albeit with occasional lower *faithfulness* scores than the XML case. The open-ended, context-rich nature of RDF allows for comprehensive and semantically rich responses, though it may occasionally introduce additional information that negatively impacts *faithfulness* and, at times, *answer relevancy*, as seen in Tables 7, 8, and 9. In contrast, the structured format of XML ensures high *faithfulness*, with generated responses that adhere strictly to the defined schema. The XML response in CASE II achieved a higher *answer relevancy* score (0.965969) compared to the score obtained by the RDF response in CASE I (0.860502), which is mostly attributable to the fact that the generated answer in CASE II closely mirrored the structure and wording of the ground truth.

Table 7

Prompt 7. Identify the events that occur before event Z and their types.

CASE I	CASE II
The events that occur before event Z are event X and event Y. Both of them are of type 'Intermediate Event BPMN'.	It appears that there are no preceding events directly connected to event Z in the provided BPMN file. This could mean that event Z might be an isolated event or connected in a way not captured by the typical sequence flow

		mechanism. If you have any specific elements or further details you'd like to check, please let me know!	
Ground truth	The events that occur before event Z are event X and event Y, both of type 'Intermediate Event BPMN'.		
RAGAs metrics			
Faithfulness	0.0	**Faithfulness**	0.0
Answer Relevancy	0.951354	**Answer Relevancy**	0.0
Answer Correctness	0.848971	**Answer Correctness**	0.64194
Answer Similarity	0.995917	**Answer Similarity**	0.853476

The direct pertinence to the prompt of the generated response in CASE I is evidenced by a high *relevancy* score of 0.951354, outshining CASE II, which fails to address the requirements of the given prompt. Moreover, it is noteworthy that the XML format in CASE II effectively handles certain subordinating conjunctions, such as "after", but struggles with "before". This discrepancy highlights a limitation in the ability of this format to capture and convey temporal dependencies that are crucial for depicting the sequence of the process (refer to Figure 5-c). While XML can represent straightforward sequential flows, it appears less capable of interpreting reverse or preceding relationships, leading to lower scores for *answer relevancy, answer correctness* and *answer similarity*.

Table 8

Prompt 8. Shortly describe the workflow after task X.

CASE I		**CASE II**	
After task X, the workflow goes through a non-exclusive parallel gateway. This gateway splits the path into three branches, each leading to a separate task: Task A, Task B, and Task C. These tasks are executed concurrently. Once all three tasks are completed, they converge at another non-exclusive gateway. The workflow then proceeds to Task Y, and finally to an end event.		After the task named 'X', the workflow proceeds to a parallel gateway. Here is a brief description of the workflow: 1. **Task 'X'** (ID: sid-6DBF436A-75C9-48D1-AB51-FFE0D462CDDE): This is the starting task for the described sequence. 2. **Parallel Gateway** (ID: sid-7C3529D3-C3CE-48FE-AF56-DC99EFC4F252): The next step after completing Task 'X' is reaching a parallel gateway. This gateway will allow the process to branch into multiple parallel paths. To provide a more detailed description of the subsequent tasks or events after the parallel gateway, please specify whether you need information on a particular branch or all branches emanating from this gateway.	
Ground truth	After task X, the workflow reaches a non-exclusive parallel gateway, which splits the path into three branches, each leading to a separate task: task A, task B and task C. These tasks are executed in parallel. Once all three tasks are completed, they converge at another non-exclusive gateway. The workflow then proceeds to task Y and after its execution, the process ends at the end event.		
RAGAs metrics			
Faithfulness	1.0	**Faithfulness**	0.2
Answer Relevancy	0.902193	**Answer Relevancy**	0.856114
Answer Correctness	0.866111	**Answer Correctness**	0.605165
Answer Similarity	0.993858	**Answer Similarity**	0.920659

Table 9

Prompt 9. What happens if the process follows path v1?

CASE I	**CASE II**
If the process follows path v1, it will move from the Exclusive Gateway to task A. After task A is completed, it will	If the process follows path v1, it leads to a task with the following details: **ID:** sid-D5F0E9D6-336C-420F-9638-

proceed to the ending select one gateway and then on to task Y. Finally, it will end with the End Event.		DAFABAD9D54A, **Name:** A, **Type:** Task. This task represents the next step in the process when the path v1 is followed. If you need further details about this task or subsequent steps, please let me know.	
Ground truth	If the process follows path v1, it leads to task A. After task A is completed, the gateway closes, continuing with task Y. After the completion of task Y, the process ends at the end event.		
RAGAs metrics			
Faithfulness	0.666667	**Faithfulness**	0.666667
Answer Relevancy	0.948102	**Answer Relevancy**	0.86871
Answer Correctness	0.689055	**Answer Correctness**	0.595897
Answer Similarity	0.956219	**Answer Similarity**	0.883587

In both scenarios – parallel branching by an AND gateway (Figure 5-d) and decision branching by an XOR gateway (Figure 5-e) – the responses of the GPT model exhibit noticeable differences, nuancing the importance of the flexibility and context-rich structure of RDF in providing detailed and holistic BPMN process interpretations, particularly where understanding the entire process flow is essential. CASE II in both Table 11 and Table 12 showcases a tendency of GPT-4 to stop analyzing the rest of the process after identifying the element in the immediate vicinity of the one mentioned in the prompt, limitation which is reflected in lower RAGAs scores compared to the ones in CASE I. The more rigid, structured schema of the XML-formatted data limits the ability of the GPT model to interpret and infer beyond explicitly defined elements.

Table 10

Prompt 10. What is the decision point based on?

CASE I	CASE II
The decision point is usually based on certain conditions or events. In this specific scenario, it appears that the decision point, denoted as "e?", is an event-based Gateway type, which means the flow of process will go towards different paths depending on which event occurs first (e1, e2, or e3). The decision is not instantaneously made but is contingent on the occurrence of these events.	The analysis of the sequence flows in the BPMN file reveals an event-based gateway as a critical decision point. Here are the relevant details: **Decision Point: Event-Based Gateway** (sid-31E2E7EE-1F2C-4118-9EAA-F9BCB418D2B1). **Flows from the Decision Point:** 1. **Flow ID**: sid-425B9781-1B88-420C-92A1-307DAEAC812B: **Source**: sid-31E2E7EE-1F2C-4118-9EAA-F9BCB418D2B1 (Event-Based Gateway), **Target**: sid-FC7D8C8F-2C6A-4501-BE05-3C35F68ACB6A (Intermediate Catch Event). 2. **Flow ID**: sid-F5909E5E-358F-4A2B-A6D4-D2C264B64D8E: **Source**: sid-31E2E7EE-1F2C-4118-9EAA-F9BCB418D2B1 (Event-Based Gateway), **Target**: sid-578C0A06-2EA9-4790-9A8D-CA2516563D38 (Intermediate Catch Event). 3. **Flow ID**: sid-14A0AE19-7B86-4B75-A84A-48C548090F17: **Source**: sid-31E2E7EE-1F2C-4118-9EAA-F9BCB418D2B1 (Event-Based Gateway), **Target**: sid-FAFB5087-D586-41C2-892F-3C43C0105B6B (Intermediate Catch Event). **Context:** An event-based gateway waits for a specific event to occur before continuing the process flow. Each of these intermediate catch events following the gateway likely represents different events or conditions that need to be met for the process to proceed. **Next Steps:** Depending on which intermediate catch event is triggered, the process will flow to the respective task. Here are the subsequent tasks based on the intermediate catch events: **Intermediate Catch Event** (sid-FC7D8C8F-2C6A-4501-BE05-

		3C35F68ACB6A): **Next Task**: sid-D709E105-9CA5-4226-A461-373030A6E182. **Intermediate Catch Event** (sid-578C0A06-2EA9-4790-9A8D-CA2516563D38): **Next Task**: sid-F79FAE96-C2C6-435E-8CA6-C3B0D0658382. **Intermediate Catch Event** (sid-FAFB5087-D586-41C2-892F-3C43C0105B6B): **Next Task**: sid-80BF32BA-0974-4F28-8C53-B282AE82D24D. If you need further analysis on any specific path or additional details about other elements, please let me know!	
Ground truth	The decision point is based on the occurrence of one of the intermediate events labeled “e1”, “e2” and “e3”.		
RAGAs metrics			
Faithfulness	0.5	**Faithfulness**	1.0
Answer Relevancy	0.808789	**Answer Relevancy**	0.824274
Answer Correctness	0.770946	**Answer Correctness**	0.210059
Answer Similarity	0.901965	**Answer Similarity**	0.840237

Although the generated answer in CASE I includes inferred details (not explicitly present in the model depicted in Figure 5-f) and does not maintain such a strict factual consistency with the elements explicitly defined in its structure – such as CASE II, supported by a high *faithfulness* score – it denotes greater contextual alignment and semantic richness in comparison to the response illustrated in CASE II. Interestingly, this time, CASE II went beyond the immediate detected elements after the gateway and identified future tasks linked to the intermediate events. This demonstrates the potential of XML to detect subsequent elements, even though it often lacks the holistic contextual integration provided by RDF.

Table 11

Prompt 11. How does the boundary message event m affect the workflow of the current process?

CASE I		**CASE II**	
The boundary message event "m" is of type interrupting and catching, meaning that it listens for the arrival of a specific message while the current process activity (Task B) is taking place. If the message event "m" is triggered during the execution of Task B, it interrupts the current execution flow of Task B. This causes a transition to another task (Task D), effectively altering the normal sequence of the workflow. In this case, the workflow diverts from the sequence of tasks B, C to tasks D and end event, bypassing task C completely. This boundary event provides a way of dealing with exceptional conditions or alternative paths in the process.		The message boundary event "m" is attached to Task B (ID: sid-D17CFFA6-F193-4DF8-A45C-A13784EA0566). When the message event "m" is triggered, it follows the outgoing sequence flow (ID: sid-13F6B017-C153-4BA9-B63B-6BBB7461546C) to Task D (ID: sid-8C1B1F14-ED67-4C9D-80DE-3839FC3C52ED). To summarize: **Boundary Event "m"** is attached to **Task B**. When **Boundary Event "m"** is triggered, the workflow moves to **Task D**. This event causes an interruption in Task B and redirects the workflow to Task D. If you need further details or have additional questions about this process, please let me know!	
Ground truth	The boundary message event m affects the workflow of the current process in the following manner: if message m arrives during the execution of task B, its execution is interrupted and the process flow proceeds towards the next step the boundary message event is connected to, which is task D. After task D is completed, the process ends at the end event that task D connects to.		
RAGAs metrics			
Faithfulness	1.0	**Faithfulness**	0.0
Answer Relevancy	0.88403	**Answer Relevancy**	0.890075
Answer Correctness	0.85832	**Answer Correctness**	0.771313
Answer Similarity	0.96269	**Answer Similarity**	0.942431

Table 11 encompasses an example where the response generated by GPT-4 in CASE I completely matches the explicit details provided in the BPMN model, proven by the perfect *faithfulness* score of

1.0. Moreover, the accurate interpretation of the elements and relationships of the process (refer to Figure 5-g) resulted in generally high *relevancy, correctness* and *similarity* scores. Conversely, CASE II demonstrated a notable limitation in *faithfulness* and scored lower than the first case regarding *answer relevancy* and *similarity*, due to its fragmented explanation.

Table 12

Prompt 12. Shortly describe the steps within the compensatory process in the current workflow.

<table>
<tr><th colspan="2">CASE I</th><th colspan="2">CASE II</th></tr>
<tr><td colspan="2">The compensatory process in the current workflow involves several steps: 1. Event 'e' begins the process as a form of compensation, which means it is a mechanism to undo or alter the effects of a previously executed task. 2. This event will first check if task 'A' can be interrupted. If yes, event 'e1' is triggered which is a catching intermediate event. This form of event captures the occurrence of a specific situation during the process execution. 3. After 'e1' is triggered, tasks 'B' and 'D' are executed which are not aimed for compensation and their execution can be interrupted. 4. After these tasks are executed, the process will check if task 'C1' can be interrupted. This task is part of the compensatory measures. If task 'C1' can be interrupted, then event 'e2' is triggered. 5. At the end of 'e2', task 'C2' which is another task aimed for compensation is triggered. 6. Finally, if either of the compensatory tasks 'C1' or 'C2' are not successfully executed, or if the whole compensatory process cannot continue for any reason, the 'End Event (BPMN)-35338' will be triggered. This event indicates that the process are stopped without any compensation happening. Therefore, the compensatory process in this workflow involves some catching events and tasks which could potentially be interrupted, and some tasks specifically designed for compensation. If the process is successful, compensation will occur, if not, the workflow will be ended without compensation.</td><td colspan="2">Here are the steps within the current workflow: 1. <b>Start Event</b> (ID: sid-4190BF19-E66E-40C0-BD19-CF11FD70DC17): The process begins here. 2. <b>Task A</b> (ID: sid-B0E20E2C-17F7-4236-AD66-35A5B50A3C4F): Regular task in the process. 3. <b>Task B</b> (ID: sid-0ED0FA2F-267E-453C-B225-D12D734087AB): Another task in the process. 4. <b>Task D</b> (ID: sid-74F6E813-4960-41EB-83E6-B839614F77D9): Subsequent task in the process. 5. <b>End Event</b> (ID: sid-A73CB847-9F1C-4A93-B3F0-672872DD5A57): Regular end event of the process. Additionally, there are boundary events: 6. <b>Boundary Event e2</b> (ID: sid-5A50CCAC-1031-48F2-8771-5658DE8856CA): Attached to a task, handling exceptions or interruptions. 7. <b>Boundary Event e1</b> (ID: sid-2A8C1759-B88D-4690-9F48-7A1CA900D4F7): Another boundary event handling exceptions. Compensatory tasks identified: 8. <b>Task C1</b> (ID: sid-6BB7BB10-C45F-473F-9AC3-18A1445778FE): This task is marked as a compensatory task. 9. <b>Task C2</b> (ID: sid-D34A8755-F7C0-4F40-8058-014DC08956E4): Another compensatory task. 10. <b>End Event e</b> (ID: sid-7C94CE9B-DEF8-406B-9226-7C8F568FFDD5): End event of a compensatory process. The compensatory process steps include Task C1 and Task C2, which are explicitly marked for compensation. These tasks are executed to undo the effects of previously completed tasks if an error or a specific event triggers the need for compensation.</td></tr>
<tr><td><b>Ground truth</b></td><td colspan="3">The steps within the compensatory process in the current workflow are: 1. The process ends with a compensation end event e, signaling the need to revert the effects of tasks successfully completed earlier in the process, to which compensation boundary events are attached. 2. Compensation boundary event e2 invokes compensation activity C2, which undoes task B. 3. Compensation boundary event e1 triggers compensation activity C1, which undoes task A. 4. Once tasks C1 and C2, respectively, are completed, the process ends.</td></tr>
<tr><th colspan="4">RAGAs metrics</th></tr>
<tr><td><b>Faithfulness</b></td><td>0.764706</td><td><b>Faithfulness</b></td><td>1.0</td></tr>
<tr><td><b>Answer Relevancy</b></td><td>0.947457</td><td><b>Answer Relevancy</b></td><td>0.88715</td></tr>
<tr><td><b>Answer Correctness</b></td><td>0.235811</td><td><b>Answer Correctness</b></td><td>0.233919</td></tr>
<tr><td><b>Answer Similarity</b></td><td>0.943244</td><td><b>Answer Similarity</b></td><td>0.935677</td></tr>
</table>

CASE II failed to recognize the initial step in the compensatory process triggered by the compensation end event “e” in Figure 5-h. This omission underlines a critical gap in understanding the flow of the process, signifying a significant constraint in capturing contextual relationships, ultimately resulting in an incomplete description and lower *correctness* score. Conversely, CASE I provided a more comprehensive interpretation of the BPMN elements, correctly capturing the role of the compensation end event and the subsequent steps, achieving higher overall RAGAs scores. Nevertheless, once again, the generated response, although comprehensive, accurate and contextually rich, includes procedural logic that goes beyond the factual information presented in the structured context, impacting its factual consistency rating (*faithfulness* score).

Table 13

Prompt 13. What is the execution of task C conditioned by?

CASE I		**CASE II**	
The execution of task C is conditioned by Task F. According to the given flow, Task F sends a message (M2) to Task C, suggesting that Task C cannot be executed until this message has been received from Task F.		The analysis did not reveal any explicit conditions directly leading to "Task C" in the provided BPMN file. This implies that "Task C" may not be conditioned by any specified conditions within sequence flows or gateways in the XML structure provided. It might be triggered unconditionally or through implicit process logic not detailed in the file. If you have any specific areas or elements you want to explore further, please let me know!	
Ground truth	The execution of task C is conditioned by the arrival of message M2 produced by task F after its execution.		
RAGAs metrics			
Faithfulness	0.0	**Faithfulness**	1.0
Answer Relevancy	0.86535	**Answer Relevancy**	0.0
Answer Correctness	0.73872	**Answer Correctness**	0.203697
Answer Similarity	0.954881	**Answer Similarity**	0.814786

According to the *relevancy, correctness* and *similarity* scores highlighted in Table 13, the RDF formatted data in CASE I allows for more detailed and contextually rich responses, with respect to the collaborative processes illustrated in Figure 6-i, compared to CASE II, where the response of *BPMN Analyst* reveals a significant limitation in its ability to generalize the query and accurately identify the message flow between Task F and Task C, even after multiple attempts.

Table 14

Prompt 14. How does the participant “P1” interact with other participants of the process?

CASE I		**CASE II**	
From the provided data, participant "P1" interacts with other participants by sending a message "M1" from task BPMN-23834-B to participant "P2". Also, participant "P1" receives a message "M2" from participant "P2" to task BPMN-23840-D.		It seems that there are no direct message flows involving participant "P1" with other participants in the provided BPMN file. This suggests that "P1" does not have explicit interactions (via message flows) with other participants as per the current model. If you need further analysis or insights into another part of the process or a different aspect, please let me know!	
Ground truth	The participant “P1” interacts with other participants of the process by sending the message “M1” to the participant “P2” and receiving the message “M2” from the participant “P2”.		
RAGAs metrics			
Faithfulness	0.0	**Faithfulness**	0.0
Answer Relevancy	0.960106	**Answer Relevancy**	0.848579
Answer Correctness	0.662243	**Answer Correctness**	0.215372
Answer Similarity	0.934687	**Answer Similarity**	0.861489

The inherent capability of RDF to link entities through semantic relationships facilitates a deeper understanding of the process interactions illustrated in Figure 6-j – as demonstrated by the overall higher RAGAs scores of CASE I in comparison to CASE II – even when they are not explicitly defined by the standard sequence flows or gateways. Meanwhile, relying on XML-formatted data, *BPMN Analyst*, using GPT-4, struggled to generalize and infer interactions beyond what is directly specified in the BPMN file, becoming challenging to capture inter-participant interactions (hence the *answer correctness* score of only 0.215372).

Table 15

Prompt 15. Identify all the elements of type event that occur before E3, in the order depicted in the process.

CASE I		**CASE II**	
The events that occur before E3, in the order depicted in the process, are: Start Event (BPMN)-35079, E1, E2.		The analysis indicates that the only event occurring before "E3" is: **Start Event**: sid-87CD25C7-AA75-4667-AF37-755E03DC9681 (E2). This suggests that "E2" is the preceding event in the sequence flow leading to "E3." If you need further details or have additional queries about the process, feel free to ask!	
Ground truth	The elements of type event that occur before E3, in the order depicted in the process are Start Event, E1 and E2.		
RAGAs metrics			
Faithfulness	1.0	**Faithfulness**	0.5
Answer Relevancy	0.937978	**Answer Relevancy**	0.882107
Answer Correctness	0.733765	**Answer Correctness**	0.52821
Answer Similarity	0.935061	**Answer Similarity**	0.912838

In the scenario involving collaboration between events (see Figure 6-k), it is worth mentioning that CASE II did not struggle with the subordinating conjunction “before” as significantly as it did in previous scenarios (such as Figure 5-c). This is evident from its ability to identify at least one preceding event, which marks a relative improvement in handling temporal sequencing terms. However, this instance still reflects an inherent limitation: specifically, CASE II tends to halt its analysis after detecting an element in the vicinity of the prompt-specified element, failing to continue its examination of the entire process. In contrast, CASE I demonstrates a more exhaustive approach, correctly identifying all the relevant events in the precise order depicted by the process. Mapping complex relationships and process flows more effectively ensures a higher degree of accuracy and completeness in the responses, as evidenced by all the RAGAs scores of the metrics used for this analysis.

Table 16

Prompt 16. What does the start of the process executed by P2 depend on?

CASE I		**CASE II**	
The start of the process executed by P2 depends on Event E1 in P1 which triggers M1. This in turn activates the start event E2 in P2 which starts the process execution.		The start of the process executed by P2 is indicated by the Start Event with the ID sid-87CD25C7-AA75-4667-AF37-755E03DC9681. This Start Event does not have any dependencies, meaning it initiates the process independently without requiring any preceding events or conditions.	
Ground truth	The start of the process executed by P2 depends on the arrival of message M1 sent by P1.		
RAGAs metrics			
Faithfulness	0.0	**Faithfulness**	0.333333
Answer Relevancy	0.989297	**Answer Relevancy**	0.894388
Answer Correctness	0.832791	**Answer Correctness**	0.463756
Answer Similarity	0.93118	**Answer Similarity**	0.855022

Analyzing the results from Table 16, although CASE II successfully identified the start event from the process involving participant P2, it fails to recognize the inter-participant message flow and dependencies, unlike CASE I. The response in CASE II suggests that the start event in P2 initiates independently, without any preceding events or conditions, which contradicts the collaborative nature of the processes shown in Figure 6-k. The inability to capture the collaborative dependencies indicates a limitation in the linear and less context-aware data structure of XML, showing a significant drop in *answer correctness* (0.463756).

Table 17

Prompt 17. Which role is in charge of each specific task?

CASE I		**CASE II**	
The role in charge of specific task would be: Task A is under the responsibility of Role L1 Task B is under the responsibility of Role L2 Task C is under the responsibility of Role L1 Task D is under the responsibility of Role L2		It seems that roles might be defined in connection with lanes or participants, which might not have been captured directly within task elements. Given this, the task roles can be inferred from their respective lanes or the process participants.	
Ground truth	The role in charge of each specific task is: role L1 is in charge of task A and task C, while role L2 is in charge of task B and task D.		
RAGAs metrics			
Faithfulness	0.0	**Faithfulness**	0.666667
Answer Relevancy	0.915295	**Answer Relevancy**	0.850039
Answer Correctness	0.865048	**Answer Correctness**	0.641801
Answer Similarity	0.960044	**Answer Similarity**	0.85292

Although lower than the RAGAs scores in CASE I – since CASE I includes a response that accurately listed the roles responsible for each task – it is important to understand why CASE II received generally high RAGAs scores, as well, in spite of failing to deliver a correct response. The metric for *faithfulness* is relatively high at 0.666667, indicating that the generated answer, although not directly answering the prompt, is factually consistent with the provided context (the BPMN file). The response correctly identifies that roles might be defined in connection with lanes or participants in the BPMN model (see Figure 6-l), which is accurate in the context of the structural conventions of BPMN. Scoring 0.850039 for *answer relevancy* suggests that the response is still pertinent to the general question about task roles: the GPT-generated response correctly notes that roles might be inferred from lanes or participants. The *correctness* score of 0.641801 demonstrates an understanding of BPMN conventions, even though it does not directly match the ground truth. The *similarity* score of 0.85292 indicates the use of terminology and concepts that are related to BPMN role assignments, which are somewhat aligned with the expected content of the answer.

Table 18

Prompt 18. What is the execution of task C conditioned by?

CASE I	**CASE II**
The execution of task C is conditioned by the data object "d1".	The execution of "Task C" is not conditioned by any specific conditions or expressions on the incoming sequence flows in the provided BPMN file. This indicates that "Task C" is triggered by its preceding activities without any explicit conditional logic governing its initiation. If you need further analysis or have additional questions regarding the BPMN file, feel free to ask!

Ground truth	The execution of task C is conditioned by data object d1 as input.		
RAGAs metrics			
Faithfulness	1.0	**Faithfulness**	1.0
Answer Relevancy	0.967062	**Answer Relevancy**	0.908293
Answer Correctness	0.747609	**Answer Correctness**	0.20961
Answer Similarity	0.990435	**Answer Similarity**	0.83844

Despite failing to deliver a relevant response, as proven by the lower score of 0.20961 for *correctness* in CASE II – indicating that the response did not address the specific query about the data object “d1” (see Figure 6-m) – the high *faithfulness, relevancy* and *similarity* scores for the second case can be attributed to factors similar to those identified in Table 17. While the issues there were related to message flows, in Table 18, the XML-formatted data failed to capture association relationships, such as those involving data objects; instead, the focus was on standard sequence flows.

Table 19

Prompt 19. Identify the tasks that use d2 as input.

CASE I		**CASE II**	
The task that uses d2 as input is "D".		There are no tasks that use the data object "d2" as an input in the provided BPMN file. This means "d2" is not associated with any tasks through data input associations. If you need further assistance or have more questions regarding the BPMN file, feel free to ask!	
Ground truth	The task that uses d2 as input is D.		
RAGAs metrics			
Faithfulness	1.0	**Faithfulness**	0.0
Answer Relevancy	0.962767	**Answer Relevancy**	0.862306
Answer Correctness	0.997953	**Answer Correctness**	0.587001
Answer Similarity	0.997953	**Answer Similarity**	0.848002

This comparison highlights the superior ability of RDF in querying processes where nuanced relationships and metadata play a crucial role (refer to Figure 6-m). The semantic richness of RDF ensures that detailed associations, such as those between tasks and data objects, are effectively captured and interpreted, with CASE I obtaining a perfect *faithfulness* and near-perfect *correctness* scores. On the other hand, CASE II failed to recognize the association between the data object d2 and task D, which can be attributed to the more rigid structure of XML and its focus on sequence flows, which often overlooks the detailed data input associations that RDF can handle.

5. Conclusions

Through the comparative use of RDF-encoded semantic graphs and XML-encoded diagrams, the experiments delineate nuanced treatments for BPMN models subjected to certain LLM services as procedural knowledge to be interpreted. Our findings, substantiated by the scores obtained using the RAGAs framework, indicate that RDF exports provide a more open-ended and context-aware approach to process interpretation, compared to the standard XML export which appears to be treated as a closed-world data structure, even though the RDF version employs a non-standard, tool-specific, process description vocabulary.

The study has inherent limitations due to the fast evolution and stochastic nature of LLM services – not only variations between different versions, but also between work sessions are noticeable. Therefore, this work is not intended to be an evaluation of capabilities of a certain LLM product, but more of a proposition of an interpretation and analysis protocol towards an augmented version of the BPM lifecycle as proposed by [25].

Future work will be invested in further exploration of the possible synergies between the BPM lifecycle and LLM services, specifically on prompting strategies that can generate process serializations from narrative instructions in a way that balances visual clarity, expressive labelling and non-trivial structural complexity.

References

[1] Dolha, D., Buchmann, R. A. (2024).Generative AI for BPMN Process Analysis: Experiments with Multi-modal Process Representations. In Proceedings of BIR 2024, LNBIP 529, pp. 19-35, Springer, https://doi.org/10.1007/978-3-031-71333-0_2

[2] BOC GmbH (2024), ADOxx AQL query language. https://www.adoxx.org/live/adoxx-query-language-aql.

[3] Bachhofner, S., Kiesling, E., Revoredo, K., Waibel, P. and Polleres, A. (2022). Automated Process Knowledge Graph Construction from BPMN Models. Database and Expert Systems Applications. 13426, pp. 32-47. Springer, Cham. doi:http://dx.doi.org/10.1007/978-3-031-12423-5_3.

[4] Buchmann, R. A. and Karagiannis, D. (2015). Pattern-based Transformation of Diagrammatic Conceptual Models for Semantic Enrichment in the Web of Data. Knowledge-Based and Intelligent Information & Engineering Systems 19th Annual Conference (KES-2015). 60, pp. 150-159. Procedia Computer Science. doi:http://dx.doi.org/10.1016/j.procs.2015.08.114.

[5] Buchmann, R., Eder, J., Fill, H. G., Frank, U., Karagiannis, D., Laurenzi, E., Mylopoulos, J., Plexousakis, D., Santos, M. Y. (2024): Large language models: Expectations for semantics-driven systems engineering, Data & Knowledge Engineering 152, article 10234, doi: https://doi.org/10.1016/j.datak.2024.102324.

[6] Busch, K. R. (2023). Just Tell Me: Prompt Engineering in Business Process Management. Enterprise, Business-Process and Information Systems Modeling (BPMDS EMMSAD 2023). Lecture Notes in Business Information Processing. 479, pp. 3-11. Springer, Cham. doi:https://doi.org/10.1007/978-3-031-34241-7_1.

[7] Camunda (2024), What is BPMN? Business Process Model and Notation (n.d.). https://camunda.com/bpmn/.

[8] Es, S., James, J., Espinosa-Anke, L. and Schockaert, S. (2024). RAGAs: Automated Evaluation of Retrieval Augmented Generation. Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics: System Demonstrations (pp. 150-158). Association for Computational Linguistics. Retrieved from https://aclanthology.org/2024.eacl-demo.16.

[9] Exploding Gradients (2023), Ragas metrics. https://docs.ragas.io/en/v0.0.17/concepts/metrics/index.html.

[10] Fill, H., Fettke, P. and Köpke, J. (2023). Conceptual Modeling and Large Language Models: Impressions From First Experiments With ChatGPT. Enterprise Modelling and Information Systems Architectures (EMISAJ), 18, 1-15. doi:https://doi.org/10.18417/emisa.18.3.

[11] Grohs, M., Abb, L., Elsayed, N. and Rehse, J. (2023). Large Language Models can accomplish Business Process Management Tasks. Business Process Management Workshops. BPM 2023. Lecture Notes in Business Information Processing. 492, pp. 453-465. Springer, Cham. doi:https://doi.org/10.1007/978-3-031-50974-2_34.

[12] Guntur, B. H. (2024, April). Automating Data Flow Diagram Generation from User Stories Using Large Language Models. 7th Workshop on Natural Language Processing for Requirements Engineering. Retrieved from https://hal.science/hal-04525925/.

[13] OMILAB NPO (2024), Bee-Up For Education. https://bee-up.omilab.org/activities/bee-up/.

[14] Ontotext. (2024). Querying OpenAI GPT Models: https://graphdb.ontotext.com/documentation/10.5/gpt-queries.html#configuring.

[15] Ontotext. (2024b). What is GraphDB?: https://graphdb.ontotext.com/documentation/10.5/.

[16] OpenAI (2023) Introducing ChatGPT Plus. https://openai.com/index/chatgpt-plus/.

[17] OpenAI. (2023b). GPT-4. https://openai.com/research/gpt-4.

[18] Jalali, A. (2021). Graph-Based Process Mining. International Conference on Process Mining (pp. 273-285). Springer, Cham. doi:https://doi.org/10.1007/978-3-030-72693-5_21.

[19] Jasińska, K., Lewicz, M. and Rostalski, M. (2023). Digitization of the enterprise - prospects for process automation with using RPA and GPT integration. 27th International Conference on Knowledge Based and Intelligent Information and Engineering Sytems. 225, pp. 3243-3254. Procedia Computer Science. doi:https://doi.org/10.1016/j.procs.2023.10.318.

[20] Karagiannis, D., Buchmann, R. A. and Utz, W. (2022). The OMiLAB Digital Innovation environment: Agile conceptual models to bridge business value with Digital and Physical Twins for Product-Service Systems development. Computers in Industry, 138. doi:https://doi.org/10.1016/j.compind.2022.103631.

[21] Polyvyanyy, A. (2021). Process Querying: Methods, Techniques, and Applications. Process Querying Methods (pp. 511-524). Springer, Cham. doi:https://doi.org/10.1007/978-3-030-92875-9_18.

[22] SAP (2024), SAP Signavio Process Transformation Suite, Academic Edition. Retrieved from SAP Signavio: https://www.signavio.com/academic-and-research-alliances/.

[23] Shahin, M., Chen, F., Hosseinzadeh, A., Maghanaki, M. and Eghbalian, A. (2024). A Novel Approach to Voice of Customer Extraction using GPT-3.5 Turbo: Linking Advanced NLP and Lean Six Sigma 4.0. (A. Nee, K. Cheng, D. Russell, M. Shunmugam and I. Lazoğlu, Eds.) The International Journal of Advanced Manufacturing Technology, 131, 3615-3630. doi:http://dx.doi.org/10.21203/rs.3.rs-3246823/v1.

[24] Uifălean, Ș., Ghiran, A. and Buchmann, R. (2023). Employing Graph Databases for Business Process Management and Representation. Advances in Information Systems Development (pp. 73-92). Springer, Cham. doi:http://dx.doi.org/10.1007/978-3-031-32418-5_5.

[25] Vidgof, M., Bachhofner, S. and Mendling, J. (2023). Large Language Models for Business Process Management: Opportunities and Challenges. Business Process Management Forum. Springer Nature Switzerland. doi:https://doi.org/10.48550/arXiv.2307.09923.

[26] Yang, L., Chen, H., Li, Z., Ding, X. and Wu, X. (2024). Give Us the Facts: Enhancing Large Language Models with Knowledge Graphs for Fact-aware Language Modeling. IEEE Transactions on Knowledge and Data Engineering, 36, 3091-3110. Retrieved from https://api.semanticscholar.org/CorpusID:259203671.