Towards a Natural Language Conversational Interface for Process Mining

Luciana Barbieri[1], Edmundo Roberto Mauro Madeira[1], Kleber Stroeh[2], and Wil M. P. van der Aalst[3,4]

[1] Institute of Computing, University of Campinas (Unicamp), Brazil
{luciana.barbieri,edmundo}@ic.unicamp.br
[2] Everflow Process Mining, Brazil
kleber.stroeh@everflow.ai
[3] Fraunhofer Institute for Applied Information Technology FIT, Germany
[4] RWTH Aachen University, Germany
wvdaalst@pads.rwth-aachen.de

Abstract. Despite all the recent advances in process mining, making it accessible to non-technical users remains a challenge. In order to democratize this technology and make process mining ubiquitous, we propose a conversational interface that allows non-technical professionals to retrieve relevant information about their processes and operations by simply asking questions in their own language. In this work, we propose a reference architecture to support a conversational, process mining oriented interface to existing process mining tools. We combine classic natural language processing techniques (such as entity recognition and semantic parsing) with an abstract logical representation for process mining queries. We also provide a compilation of real natural language questions (aiming to form a dataset of that sort) and an implementation of the architecture that interfaces to an existing commercial tool: Everflow. Last but not least, we analyze the performance of this implementation and point out directions for future work.

Keywords: Process Mining · Process Querying · Natural Language Interface.

1 Introduction

Process Mining (PM) aims to discover, monitor and enhance processes using information extracted from event logs [2]. There exist mature academic and commercial process mining techniques and tools that provide analyses over event log data. The use of these tools, however, requires knowledge of the technology itself and is mostly done by technical teams (process analysts, data scientists and alike).

To make process mining more ubiquitous, i.e., accessible on a daily basis by non-technical teams, we propose a natural language conversational interface. Business level and operations teams, for example, can take great benefit from the insights produced by process mining tools when accessed through such an intuitive conversational interface.

In spite of recent advances in Natural Language Processing (NLP), understanding the semantics of a natural language question and translating it to a correct corresponding logical query is still a challenging task. Problems such as ambiguity (same natural language expression having multiple interpretations) and variability (many different expressions having the same meaning) are yet difficult to handle. Context awareness brings yet another level of complexity to the task, as the meaning of a natural language question may depend on previous questions and responses.

The main objective of this ongoing research is to propose, implement and evaluate an architecture for a process mining natural language conversational interface that takes questions in natural language and translates them to logical queries that can be run against existing process mining tools. The contributions presented in this paper are:

- Introduce a reference architecture for a process mining natural language conversational interface
- Propose an abstract logical representation for process mining queries that is, on the one hand, independent of the underlying concrete process mining tool and, on the other, mappable to its API calls
- An initial collection and categorization of natural language process mining questions aiming to create a public dataset
- A proof of concept of the proposed architecture, including integration to a commercial tool (Everflow Process Mining[5]) through its (RESTful) API

The remainder of this paper is organized as follows. Section 2 reviews related work. Section 3 introduces the proposed architecture. Section 4 describes the PM question dataset under construction. Section 5 presents the conducted proof of concept. Section 6 concludes this paper and points out future work and directions.

2 Related Work

Natural Language Interfaces to Databases From the many existing NLP applications, the ones that are mostly related to this research are the so called Natural Language Interfaces to Databases (NLIDB). The main objective of NLIDB is to enable users who are not familiar with complex query languages such as SQL to easily formulate queries over information stored in databases using natural language.

Even though NLIDB is not a new research topic, recent advances in natural language processing have raised its importance and popularity during the last decade [3]. Current methods differ in the use of natural language itself (from queries restrictedly written according to specific grammatical constraints to full

[5] https://everflow.ai/

natural language sentences), as well as in the technical approaches used to parse and convert them to a machine-readable format such as SQL or SPARQL. Most common parsing techniques are based on rule matching or machine learning. In either case, the types of queries that can be handled by the system are limited either by the set of rules, in the first case, or by the training data in the second.

While most of the existing NLIDB methods are designed to handle queries over any domain (metadata and/or domain ontologies are usually taken as input to map domain terminology to database entities), using specific process mining domain knowledge yields context to the design of a potentially more robust natural language interface.

Natural Language Processing Applications in Business Process Management and Process Mining One of the most important applications of NLP techniques to the Business Process Management (BPM) domain is the extraction of process models from natural language text [4]. Other existing applications of NLP to BPM include the automatic generation of textual descriptions from process models [6] and the comparison of process models to textual descriptions [9]. In [1], the authors discuss future challenges for NLP applications in the BPM field, including the use of conversational systems to support the execution of business processes.

Most related to our research is the work presented in [5], where the authors propose a method to answer natural language queries over process automation event logs. The method extends the ATHENA NLIDB system [8] to translate natural language queries to queries over process execution data (event logs) stored in Elasticsearch.

Existing process mining techniques and tools can provide automatic analysis over event log data, which can be used to answer high-level user questions. To the best of our knowledge, this is the first research work aiming to automatically understand and answer natural language questions over process mining data and analyses.

3 Proposed Method

Our proposed method can be best described by the architecture depicted in Figure 1. In broad terms, it can be viewed as a pipeline moving from top to bottom. The input is a question in regular natural language (in our case, English). Questions can be provided as text or speech - planned future work includes an Automatic Speech Recognition module, which will provide “speech-to-text” functionality.

To close the pipeline, we envision Response Generation and Text to Speech modules to provide a conversational response to the user. In the scope of this work, this response was simplified and corresponds to a piece of information directly derived from the call to the PM Tool’s API. The following sections detail the modules responsible for understanding the input natural language question and mapping it to an API call.

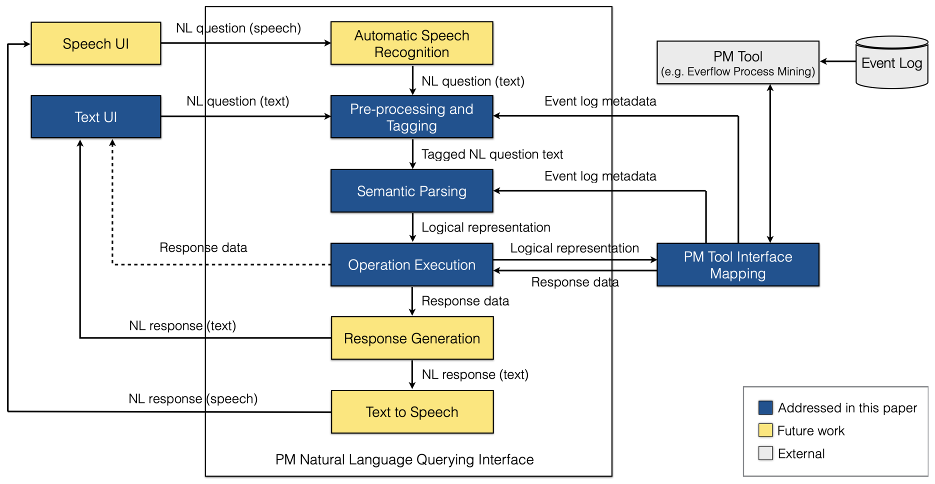


Fig. 1. Process Mining Natural Language Querying Interface Architecture Overview

3.1 Pre-processing and Tagging

The input text passes initially through a Pre-processing and Tagging step, where the following processing occurs:

- Tokenization, which is the splitting of text into tokens. Separation is based on whitespaces, punctuation marks and apostrophes, among others.
- Part-of-Speech (POS) Tagging, which performs morphological analysis over the text, marking tokens with tags such as PRON (pronoun), ADJ (adjective) and VERB.
- Dependency Parsing, which provides semi-syntactic analysis and marks tokens with grammatical structure information and dependency relations between them.
- Lemmatization, which finds the base (non-inflected) form of words.
- Entity Recognition, which identifies and tags real-world entities in the text, as detailed below.

Entity Recognition identifies general entities from pre-defined categories, such as names of people and organizations, geographic locations, time and quantities, among others. In addition to that, a natural language interface for process mining must be able to recognize the process mining entities present in sentences. Terms such as event, case, activity, resource and variant (along with its synonyms) must be recognized and tagged appropriately. The resulting tags are a crucial input for the next task in the processing pipeline (semantic parsing). Figure 2 depicts the process mining data model that underlies the recognition of such terms. Although this model is based in [2], one should notice that, for the purpose of this work, the term "event" refers to both event and activity instance.

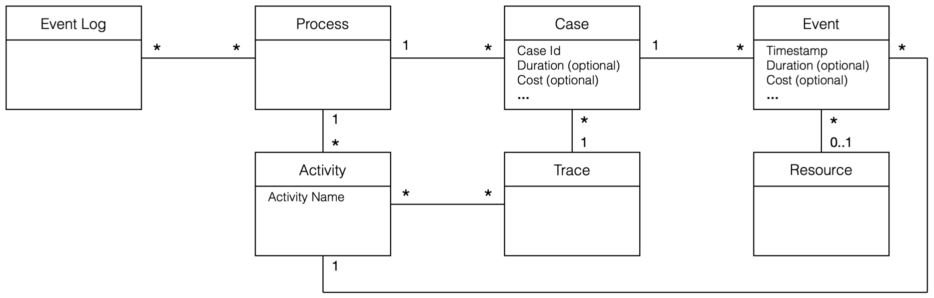


Fig. 2. Process Mining Data Model Underlying Entity Recognition

Besides dealing with general process mining terms, the system must be able to recognize domain-specific terms. This includes the names of non-standard attributes present in the event log along with possible categorical values, among others. To be able to recognize such terms, this module uses event log metadata (names, types and possible values) of these attributes. As the proposed natural language interface does not deal directly with the event log, the PM Tool Interface Mapping layer takes the responsibility of interfacing with the PM Tool to gather these metadata. Figure 3 shows examples of questions tagged with recognized entities. Notice that “Hailey Lee” and “Chicago” are categorical attribute values gathered from event log metadata and used to tag these terms during entity recognition.

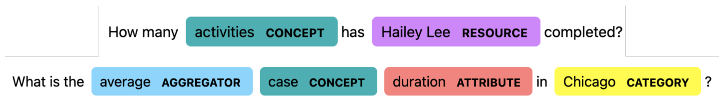

How many activities CONCEPT has Hailey Lee RESOURCE completed?
What is the average AGGREGATOR case CONCEPT duration ATTRIBUTE in Chicago CATEGORY ?

Fig. 3. Entity Recognition Examples

3.2 Semantic Parsing

Semantic parsing aims to understand the meaning of a natural language sentence and map it to a logical (machine-readable) representation. The most common methods used for semantic parsing are rule-based and neural approaches. While rule-based methods are usually more appropriate to build domain specific systems targeted to understand a finite set of sentences, neural systems are more suitable to handle complex scenarios at the cost of requiring large training corpora. Logical representations usually take the form of lambda calculus, query languages such as SQL and SPARQL or executable programs, among others.

Rule Matching As, to the best of our knowledge, there is no process mining question dataset that could be annotated and used to train traditional machine learning or neural models, we have initially adopted a rule matching approach for semantic parsing. Besides requiring no training data, the method has the advantage of achieving high accuracy in answering predictable questions.

In our proof of concept, we used the spaCy open-source natural language processing library[6]. Its Rule Matcher component allows the definition of rules that match sequences of tokens. Rules are based on tags filled in the previous steps in the pipeline (part-of-speech tags, dependency parsing results, entity recognition labels), together with actual words or expressions (in our case, words or expressions used to express the sort of process mining relationship/analysis being queried). Figure 4 illustrates the matching of the question "What activities have been assigned to Hailey Lee?" to a rule pattern.

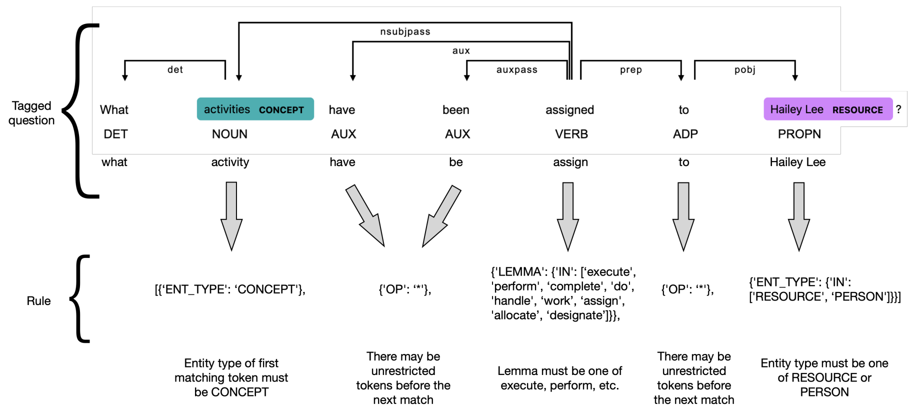

Tagged question
nsubjpass
aux
det
auxpass
prep
pobj
What DET what
activities CONCEPT NOUN activity
have AUX have
been AUX be
assigned VERB assign
to ADP to
Hailey Lee RESOURCE PROPN Hailey Lee
?
Rule
[{'ENT_TYPE': 'CONCEPT'},
{'OP': '*'},
{'LEMMA': {'IN': ['execute', 'perform', 'complete', 'do', 'handle', 'work', 'assign', 'allocate', 'designate']}},
{'OP': '*'},
{'ENT_TYPE': {'IN': ['RESOURCE', 'PERSON']}}]
Entity type of first matching token must be CONCEPT
There may be unrestricted tokens before the next match
Lemma must be one of execute, perform, etc.
There may be unrestricted tokens before the next match
Entity type must be one of RESOURCE or PERSON

Fig. 4. Rule matching example

In this case, the matched pattern leads the system to the conclusion that the user wants the list of activity instances associated to a particular resource (Hailey Lee).

Logical Representation After the semantics of a question is understood (i.e. after it matches a rule), it must be converted to a corresponding logical (PM tool independent) representation. The Question Decomposition Meaning Representation (QDMR) proposed in [11] and inspired by SQL has been used for this purpose with some extensions.

In QDMR questions are represented by a sequence of steps where each step corresponds to an operator. Each operator (except for `select`) is applied to the results of a previous step in the sequence. Additional parameters may be given to logical operators depending on the entities (concepts, attributes, aggregations, etc.) recognized in the natural language question. Table 1 presents the most

[6] https://spacy.io/

relevant QDMR operators used in this research work to compose the logical representation of PM queries. For the complete set, please refer to [11].

Table 1. Some QDMR operators used for PM question logical representation

Operator	**Description**	**Example**	**Logical Form**
`select`	Return all instances of the given concept.	Show me all cases.	`select case`

`filter`	Return the referenced instances for which the given condition holds.	Show me all cases from Chicago.	`select case`
			`filter city Chicago #1`

`project`	Return the given attribute/relation for the referenced instances.	How long does each process instance take to execute?	`select case`
			`project duration #1`

`aggregate`	Apply the given aggregation to the referenced values.	What is the average case duration?	`select case`
			`project duration #1`
			`aggregate average #2`

`group`	Apply the given aggregation to each subset of values corresponding to each key	What is the average cost of each activity?	`select event`
			`project cost #1`
			`project activity #1`
			`group average #2 #3`

`superlative`	Return the referenced instances for which the given value is the highest/lowest.	What was the slowest case?	`select case`
			`project duration #1`
			`superlative max #1 #2`

Notice that hash tags are used to refer to the results of a previous logical operation in the sequence, which may be a set of event or case instances or their attribute values. For example, in the following sequence, `#1` refers to the results of `select case` , which are all case instances and `#2` refers to the values of the `duration` attribute for `#1` .

```
select case
project duration #1
aggregate average #2
```

The original set of QDMR operators was extended by this work to allow querying the behavioral aspects of process execution. Inspired by and initially based on the set of predicates defined by the Process Query Language (PQL) [7], the `predicate` operator was introduced to logically represent questions over behavioral relations between executed activities. Supported predicates can be applied over cases or traces and are presented in Table 2.

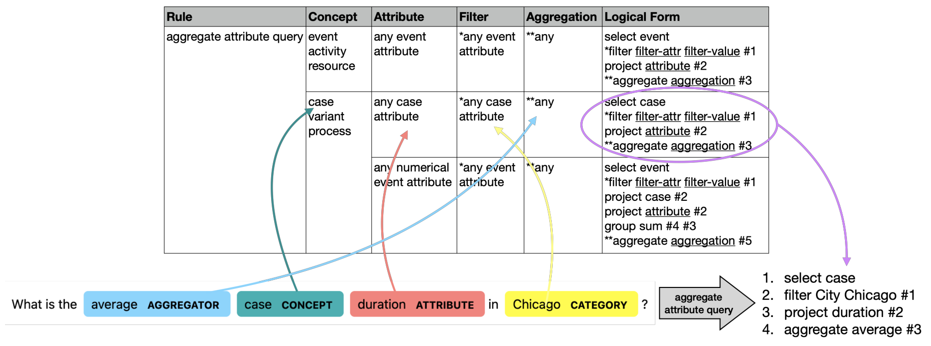
Rule to Logical Representation Mapping As one of the architectural goals of the proposed method is to allow integration to any Process Mining tool, it makes as few assumptions as possible on how the integrated Process Mining tool models the event log data. As a result, a minimal process mining data model based in the XES standard event log format [10] drives the mapping of matched rules to logical representation. Some of the entities tagged and handled as concepts during entity recognition and rule matching (activity, resource, trace) are, at this point, mapped to attributes of event and case, which are the only

Table 2. Predicates used for PM question logical representation

Predicate	Parameters	Description
occurs	activity	Return the referenced cases or traces that execute the given activity.
cooccur	activity1, activity2	Return the referenced cases or traces that execute both activity1 and activity2 or none.
conflict	activity1, activity2	Return the referenced cases or traces that execute either activity1, activity2 or none.
causal	activity1, activity2	Return the referenced cases or traces where any occurrence of activity1 precedes any occurrence of activity2.
concurrent	activity1, activity2	Return the referenced cases or traces where some occurrence of activity1 occurs at the same time as some occurrence of activity2.
activity-count	-	Return the number of activities executed by each referenced case or trace, including repetitions.
distinct-activity-count	-	Return the number of distinct activities executed by each referenced case or trace.
occurence-count	activity	Return the number of times the given activity is executed for each referenced case or trace.

selectable concepts (assuming that processes are queried one at a time). Non-standard attributes contained in the event log are mapped based on the metadata obtained from the PM tool.

Once a rule fires, a corresponding logical representation must be put together. This depends not only on what rule has been matched, but also on the entities (concepts, attributes, etc.) recognized in the sentence. As an example, Figure 5 depicts the possible logical representations to be created when the "aggregate attribute query" rule is matched.



Rule	Concept	Attribute	Filter	Aggregation	Logical Form
aggregate attribute query	event activity resource	any event attribute	*any event attribute	**any	select event *filter filter-attr filter-value #1 project attribute #2 **aggregate aggregation #3
	case variant process	any case attribute	*any case attribute	**any	select case *filter filter-attr filter-value #1 project attribute #2 **aggregate aggregation #3
		any numerical event attribute	*any event attribute	**any	select event *filter filter-attr filter-value #1 project case #2 project attribute #2 group sum #4 #3 **aggregate aggregation #5

1. select case
2. filter City Chicago #1
3. project duration #2
4. aggregate average #3

Fig. 5. Logical forms for attribute query rules

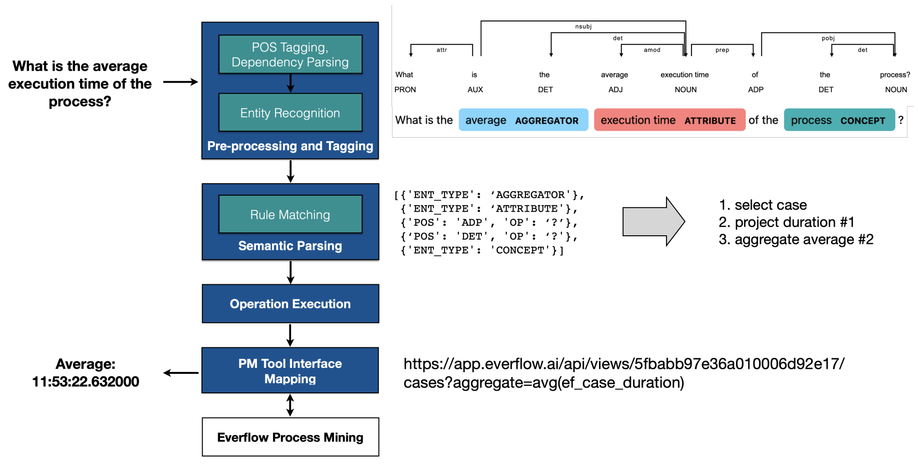
The matched rule indexes the first column in the table, while the entities tagged in the sentence index the next four (concept, attribute, filter and aggregation). The last column corresponds to the logical representation that will be used to drive the calls to the PM Tool API detailed in the following subsection. Asterisks indicate optional entities and the corresponding logical operations that are added to the sequence when they are present. The complete set of correspondences between rules and logical representations is available at `https://ic.unicamp.br/~luciana.barbieri/ruletological.pdf` .

3.3 PM Tool Interface Mapping

The final step in the question processing pipeline is to map the logical representation of the query into a real API call provided by a process mining tool.

In this work, we integrated the architecture into Everflow's RESTful API. This API presents endpoints that mimic process mining main concepts and naturally maps into the PM data model used to create logical representations.

Using Everflow's "cases" and "events" endpoints, altogether with their associated parameters (such as "filter" and "aggregate"), it is straightforward to map the logical representation into actual API calls. Figure 6 illustrates the end-to-end mapping of a natural language question to a final API call.



```
[{'ENT_TYPE': 'AGGREGATOR'},
 {'ENT_TYPE': 'ATTRIBUTE'},
 {'POS': 'ADP', 'OP': '?'},
 {'POS': 'DET', 'OP': '?'},
 {'ENT_TYPE': 'CONCEPT'}]
```

```
https://app.everflow.ai/api/views/5fbabb97e36a010006d92e17/
cases?aggregate=avg(ef_case_duration)
```

Fig. 6. End to end mapping of question to API call

The integration of a new PM tool currently requires a different instantiation of the PM Tool Interface Mapping component. Planned future work includes the definition of a standard API to replace this component and allow PM tools to easily integrate our natural language conversational interface.

4 Sample Questions

An initial set of natural language questions was collected from graduate students with beginner to intermediate level of expertise in Process Mining, resulting in 250 general (not specific to any existing event log) questions originally written in Portuguese. Free translation was performed by 3 volunteers resulting in 794 questions in English (multiple translations were done by the volunteers for some of the questions).

Questions were then categorized into 4 groups: event log data questions (questions over case/event instances, attributes and counts), process model questions (model structure, behavior and quality), process mining analysis questions (conformance checking, bottleneck analysis, transitions, root cause analysis, social network, etc.) and advanced analysis questions (prediction, prescription and simulation). Table 3 summarizes these categories.

Variability in both language and contents of questions can be improved in the future by collecting them directly in English and eliminating the translation step. Nonetheless, as no public dataset of PM questions is currently available, the samples collected so far played an important role in setting the ground for this research and being the initial input for building the rules used for semantic parsing. The complete set of 794 questions, with their corresponding classifications, is available at `https://ic.unicamp.br/~luciana.barbieri/pmquestions.csv` .

Table 3. Question categories

Category	**# Samples**	**Example**
Event log data	327	Which activity has the highest mean resolution time?
Process model	107	What are the possible start and end activities in my log?
Analysis	240	What are the most frequent non-conformances in my process?
Advanced analysis	120	What is the predicted completion time for case X?

5 Proof of Concept

In order to verify the applicability of the proposed method, we implemented a subset of the architecture (blue colored components) presented in Figure 1. For this proof of concept, we used the spaCy open-source natural language processing library, targeting questions from the "Event log data" category of the original collected set. The library's Rule Matcher component was used and fed with 34 semantic rules covering event log attribute querying, instance querying and counting, aggregations and superlatives ("most", "least"), among others.

In order to test the implementation, the set of questions from the "Event log data" category was further refined by removing compound questions (questions containing multiple embedded subquestions) and time-bound questions (questions containing time-related filters as in "What is the average duration of cases completed in the first quarter of 2020?"), as these constructions were not covered by the implemented set of semantic rules. This led to a testing dataset of 203 questions.

This testing set was executed against a Work Force Management based event log that was uploaded into the Everflow Process Mining tool. However, any process mining event log could be used, as the collected questions are not context-specific (not bounded to any particular event log).

From the 203 testing questions, 163 (80.3%) were correctly answered, 22 (10.8%) were not responded because the system was not able to match any semantic rule, and 18 (8.9%) were incorrectly answered, because they fired the wrong semantic rule or because they were falsely tagged during the Preprocessing and Tagging phase.

Examples of successfully answered questions are “What is the most common flow of activities?”, “Which activity has the highest mean resolution time?” and “What are the 10% slower cases?” Unmatched questions include “What resources execute what activities?” and “Which resources are the most agile in task execution?”. Likewise, examples of questions that fired the wrong semantic rule are “What resources take the longest to execute activity A?” and “What are the resources assigned to the fastest cases?”. Actually, all these failed tests illustrate the shortcomings of a rule-based approach, where the final result is sensitively connected to the rules in use. This means that they could be fixed by a more crafted, possibly longer, rule set, which is hard to achieve and difficult to maintain.

On the other hand, properly answered questions such as “What is the average execution time of the process in Chicago?” illustrate the ability of the system to use terms that are specific to the event log. In this example, “Chicago” is a value under the attribute “City” in the event log, and could be used in the question due to the capacity to handle metadata coming from the process mining tool. This question was, of course, not present in the original testing dataset.

Overall, in spite of the limited size and variation of the testing questions and rules, the 80.3% accuracy seems promising as a first result. As expected, a rule-based approach has limitations in treating questions that stray too much away from the structures implemented in the rules. In general, this method presents high precision, but low generalization.

6 Conclusions and Future Work

Implementing the proposed reference architecture and testing it against the aforementioned sample question dataset has led to some interesting conclusions. Rule-based semantic parsing was an appropriate choice for bootstrapping a natural language interface for PM as no training data set of any kind or size is currently available to train any supervised or semi-supervised machine learning technique.

Furthermore, as the PM general ontology is small (few entities and relations), it was possible to answer questions for a selected, pre-defined, set with high accuracy using a relatively small number of rules. However, this approach does come with limitations. Rule-based semantic parsing does not generalize well, with new rules being required for most new/unpredicted questions.

In order to overcome this generalization limitation and to evolve the study towards a fully functional architecture, we envision the following future work:

– Use machine learning for semantic parsing by using the developed rule matching parser to create an annotated training dataset.

– Increase our experiment by working with questions in other categories (process model, analysis, advanced analysis).
– Extend the response mechanism to include natural language response generation, making responses more natural and user-friendly.
– Extend the training dataset and make it public. This implies collecting additional questions, if possible, directly in English and associated with a selected event log, so that questions can be more context-based and closer to what real business users would ask in a specific domain.

Acknowledgements We would like to thank Coordenação de Aperfeiçoamento de Pessoal de Nível Superior – Brasil (CAPES) – Finance Code 001, for providing the financial support for this work. Likewise, we would like to thank Professors Sarajane Marques Peres and Marcelo Fantinato for helping us collect questions from their students for our dataset.

References

1. van der Aa, H., Carmona Vargas, J., Leopold, H., Mendling, J., Padró, L.: Challenges and opportunities of applying natural language processing in business process management. In: International Conference on Computational Linguistics. pp. 2791–2801. Association for Computational Linguistics (2018)
2. van der Aalst, W.M.P.: Process Mining: Data Science in Action. Springer (2016)
3. Affolter, K., Stockinger, K., Bernstein, A.: A comparative survey of recent natural language interfaces for databases. The VLDB Journal **28**(5), 793–819 (2019)
4. Friedrich, F., Mendling, J., Puhlmann, F.: Process model generation from natural language text. In: Advanced Information Systems Engineering. pp. 482–496. Springer Berlin Heidelberg (2011)
5. Han, X., Hu, L., Sen, J., Dang, Y., Gao, B., Isahagian, V., Lei, C., Efthymiou, V., Özcan, F., Quamar, A., Huang, Z., Muthusamy, V.: Bootstrapping natural language querying on process automation data. In: 2020 IEEE International Conference on Services Computing (SCC). pp. 170–177 (2020)
6. Leopold, H., Mendling, J., Polyvyanyy, A.: Generating natural language texts from business process models. In: Advanced Information Systems Engineering. pp. 64–79. Springer Berlin Heidelberg (2012)
7. Polyvyanyy, A., ter Hofstede, A.H., La Rosa, M., Ouyang, C., Pika, A.: Process query language: Design, implementation, and evaluation. arXiv preprint arXiv:1909.09543 (2019)
8. Saha, D., Floratou, A., Sankaranarayanan, K., Minhas, U.F., Mittal, A.R., Özcan, F.: Athena: an ontology-driven system for natural language querying over relational data stores. Proceedings of the VLDB Endowment **9**(12), 1209–1220 (2016)
9. Sànchez-Ferreres, J., Carmona, J., Padró, L.: Aligning textual and graphical descriptions of processes through ilp techniques. In: Advanced Information Systems Engineering. pp. 413–427. Springer International Publishing (2017)
10. Verbeek, H.M.W., Buijs, J.C.A.M., van Dongen, B.F., van der Aalst, W.M.P.: Xes, xesame, and prom 6. In: Information Systems Evolution. vol. 72, pp. 60–75. Springer Berlin Heidelberg (2010)
11. Wolfson, T., Geva, M., Gupta, A., Gardner, M., Goldberg, Y., Deutch, D., Berant, J.: Break it down: A question understanding benchmark. Transactions of the Association for Computational Linguistics **8**, 183–198 (2020)