Integrating Textual and Model-based Process Descriptions for Comprehensive Process Search

Henrik Leopold[1], Han van der Aa[1], Fabian Pittke[2], Manuel Raffel[2], Jan Mendling[2], and Hajo A. Reijers[1]

[1] Vrije Universiteit Amsterdam
De Boelelaan 1081, 1081 HV Amsterdam, The Netherlands
h.leopold|j.h.vander.aa|h.a.reijers@vu.nl
[2] WU Vienna Welthandelsplatz 1, A-1020 Vienna, Austria
fabian.pittke|jan.mendling|manuel.raffel@wu.ac.at

Abstract. Documenting business processes using process models is common practice in many organizations. However, not all process information is best captured in process models. Hence, many organizations complement these models with textual descriptions that specify additional details. The problem with this supplementary use of textual descriptions is that existing techniques for automatically searching process repositories are limited to process models. They are not capable of taking the information from textual descriptions into account and, therefore, provide incomplete search results. In this paper, we address this problem and propose a technique that is capable of searching textual as well as model-based process descriptions. It automatically extracts process information from both descriptions types and stores it in a unified data format. An evaluation with a large Austrian bank demonstrates that the additional consideration of textual descriptions allows us to identify more relevant processes from a repository.

1 Introduction

Business process models have proven to be an effective means for the visualization and improvement of complex organizational operations [7]. However, not all process-related information is available in the form of process models. On the one hand, because the creation of process models is a time-consuming endeavor that requires considerable resources [13]. On the other hand, because not all process information is best captured as a process model [1]. In particular, *work instructions* that describe tasks at a high level of detail are often documented in the form of textual descriptions, as this format is more suitable for specifying a high number of details [4]. As a result, process repositories in practice do not only consist of process models, but often also contain textual process descriptions. These are linked to individual activities of the process models in order to specify the detailed action items behind them.

The problem of this supplementary use of textual descriptions in process repositories is that automatic analysis techniques designed for process mod-

els, such as weakness identification [5], service identification [20], or compliance queries [3], might provide incomplete results. Suppose a company aims to increase the share of digital communication, then it can query its process repository to find all processes that still include paper-based communication. The query results, however, will be limited to the process models that indicate the use of paper-based communication already in their activity text labels. Process models that describe the process at a higher level of abstraction, but link to textual descriptions revealing that this process is indeed associated with paper-based communication, will be ignored. Currently, there is no technique available that provides the possibility to search textual and model-based process descriptions in an integrated fashion. One explanation for the absence of such a technique might be the challenges that are associated with it. Among others, it requires the definition of an integrated data format that is able to represent both textual and model-based process descriptions.

Against this background, we use this paper to propose a technique that can search both text and model-based process descriptions. It combines natural language analysis techniques in a novel way and transforms textual as well as model-based process descriptions into a unified data format. By integrating technology from the semantic web domain, we facilitate the possibility of performing comprehensive search operations on this data format.

The remainder of this paper is organized as follows. Section 2 introduces the problem of searching textual and model-based process descriptions and discusses related work. Section 3 then introduces our proposed technique on a conceptual level. Section 4 presents the results of an evaluation with a large Austrian bank. Finally, Section 5 concludes the paper and provides an outlook on future research.

2 Background

This section introduces the background of our research. First, we illustrate the problem of searching textual and model-based process descriptions. Then, we reflect on related techniques that are currently available.

2.1 Motivating Example

In order to illustrate the importance of textual descriptions in the context of a process search, let us take a look at the implications of only taking process models into account. To this end, consider the example shown in Figure 1. It shows a simple process model created using the Business Process Modeling and Notaton (BPMN) and a small complementary text from a bank. We can see that the business process is triggered by the request to open a new bank account. Subsequently, the credit history of the customer is evaluated. The outcome of this evaluation can be either positive or negative. In case of a negative credit evaluation, the customer is rejected. If the credit history evaluated as positive, a new bank account is opened. Finally, the request is closed. In addition to the BPMN process model, there is complementary text. It further specifies the

details of the activity *"Opening of new bank account for customer"*. Among others, it describes that the opening of a bank account is associated with a mail-based information exchange with the customer.

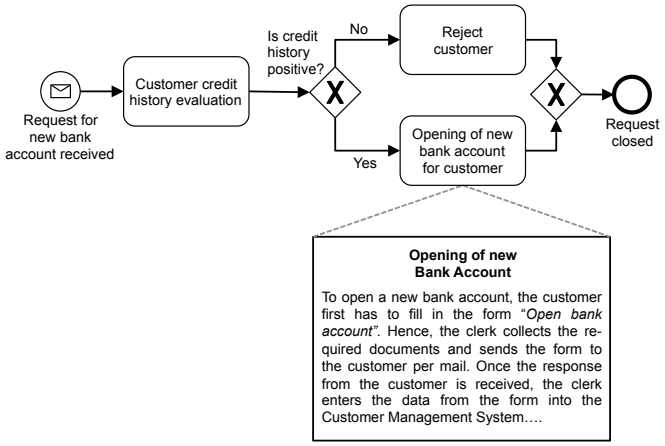

Fig. 1. Exemplary process model with complementary natural language description

Assume this process model is part of the process repository of an organization. If this organization was interested in all business processes that involve an interaction with a customer, automated search techniques would have no difficulties to identify the depicted process. That is, because the customer is explicitly mentioned in the activity labels, e.g. in *"Customer credit history evaluation"*. However, suppose the organization aims at improving its operations by replacing all mail-based correspondence with an electronic alternative. In this case, an automated search on the process model would not identify any potential for improvement. That is because the activities of the process model do not contain any words that might be associated with the activities of mailing or sending. Only the description attached to the activity *"Opening of new bank account"* explicitly refers to sending a form per mail.

This example illustrates the advantage of performing search operations that cover textual as well model-based process descriptions. As a respective technique is currently missing, it is our goal to define such a technique in this paper.

2.2 Related Work

The work from this paper relates to two major streams of research: process model search techniques and process model analysis techniques that employ Natural Language Processing (NLP) technology.

Techniques for *process model search* can be divided into two main groups. The first group consists of techniques focusing on *structure*. They compare query

and process model with respect to behavioral properties, for instance, whether two activities occur in a particular order. Among others, such structural querying techniques have been defined based on temporal logical [2], the weak ordering formalism [16], and on indexing [28, 14]. The limitation of these structural techniques is that they typically assume that semantically identical activities have identical or similar labels. The second group of search techniques focuses on the *content from text labels* and tries to overcome this problem. Among others, they employ NLP techniques to identify similar models based on their activity labels. Notable examples for such techniques have been defined in [3] and [27], where the authors use dictionaries and language modeling to retrieve semantically similar models.

NLP techniques are also often applied in the context of *process model analysis*. For instance, they are used to assure linguistic quality aspects of process models such as naming conventions [22, 19]. Other application scenarios include the generation of process models from natural language texts and vice versa [11, 17] and the detection of overlapping behavior of two process models [9, 21].

Despite the important role of NLP technology for process model search and analysis, a conceptual solution for an integrated search technique is still missing. To develop such a technique, we need to define a data format that allows us to store the information extracted from both process description types in a unified way. Based on such a format, we can then perform search operations covering model-based as well as textual process descriptions.

3 Conceptual Approach

In this section, we introduce our approach for comprehensive process search by integrating textual and model-based process descriptions. We give an overview of the architecture of our approach. We then describe the unified format we use to integrate textual and model-based content. Afterwards, we show how to parse and transform textual and model-based descriptions into this unified format. Finally, we illustrate how the use of the unified data format supports comprehensive process search.

3.1 Overview

The main idea of our architecture is that the differing input sources of textual and model-based process descriptions must be stored in a unified way. Hence, two parsing components first extract the relevant information from the two input sources and then store it in the unified data store. Once the data store has been populated with all available process descriptions, it can be used to search processes. To this end, a user interface provides the possibility to specify queries in a user-friendly manner. Figure 2 illustrates our architecture graphically.

In the subsequent sections, we describe this architecture in detail. Because of the predominant role of the unified data store, we begin with the specification of the unified format.

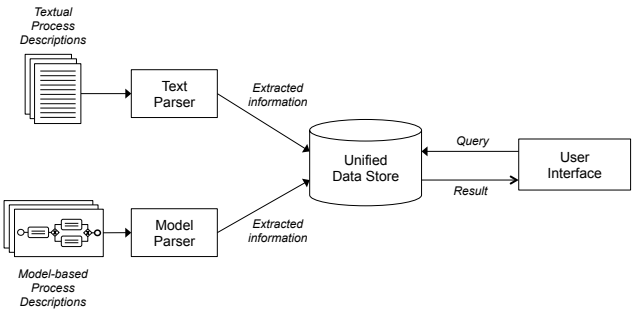

Fig. 2. Exemplary process model with complementary natural language description

3.2 A Unified Format for Integrating Textual and Model-based Process Descriptions

To define a unified format for textual and model-based process descriptions, it is important to understand how each of these descriptions conveys the semantics of the business process it describes. In essence, textual process descriptions describe business processes by using sequences of proper natural language sentences structured into sections, subsections, and paragraphs. Process models, by contrast, also consist of graphical representations of modeling constructs such as activities, events, and gateways. An important share of the semantics of process models is, however, defined by the natural language labels that are attached to the activities [18]. These labels, however, do not necessarily represent proper sentences. As examples, consider the activity labels *"Opening of bank account for customer"* or *"Customer credit history evaluation"* from the BPMN model in Figure 1. Therefore, a unified format must provide the possibility to store the essential information distilled from activity labels as well as proper natural language sentences.

According to [25], every activity label can be characterized by three components: an action, a business object on which the action is performed, and an optional additional information fragment that is providing further details. As an example, consider the activity label *"Opening of new bank account for customer"*. This activity consists of the action *"to open"*, the business object *"new bank account"*, and the additional information fragment *"for customer"*. In a proper natural language sentence, we can identify respective counterparts. Consider the sentence *"The clerk opens a new bank account for the customer"*. A grammatical analysis would reveal that this sentence contains the predicate *"opens"*, the object *"bank account"*, the subject *"clerk"*, and the adverbial *"for the customer"*. This example illustrates that the predicate corresponds to the action, the object to the business object, and the adverbial to the additional information fragment. A subject refers to the role executing the activity, which is typically specified outside the activity label.

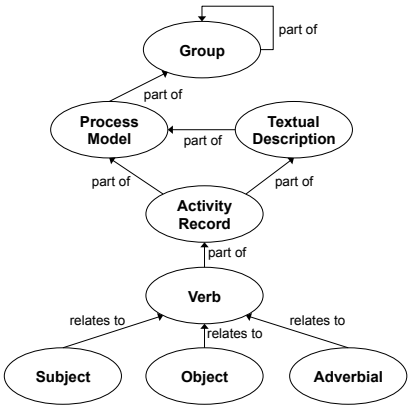

Fig. 3. Overview of Unified Data Model

Based on these insights, we specify a format that stores the language content from sentences and activity labels in a unified way. Figure 3 illustrates this format. The core of this format is a so-called activity record, which might be part of a process model or a textual description. Each activity record may consist of one or more verbs, depending on the grammatical structure of the activity label or the sentence it refers to. Each verb relates to a subject, an object, and an adverbial. Note that each of these entities might be empty if the corresponding activity label or sentence does not contain this information. As indicated by the relations between the activity record, the process model, and the textual description, each process model and each textual description may contain several activity records. Moreover, a process model may consist of several textual descriptions. As process models in industry are typically organized in hierarchical process architectures [24], our format supports the organization of process models into groups and sub groups.

Practically, we implement this unified data format by building on the Resource Description Framework (RDF), an XML-based specification developed by the World Wide Web Consortium (W3C)[1]. RDF describes data in the form of triples that consist of two entities and a relation between them. As an example, consider the relation *"relates to"* between a object and an verb in our unified data format. A possible RDF triple for this relation would be (*"customer"*, *"relates to"*, *"reject"*). Similarly, all other relations from the unified data format can be represented as RDF triples. The advantage of storing data in the RDF format is that it can be easily and effectively accessed and queried [6]. Hence, it greatly contributes to our goal of providing a technique for integrated search.

In the subsequent sections, we describe how textual process descriptions as well as process models can be automatically transformed into this unified data format.

[1] http://www.w3.org/RDF

3.3 Parsing Textual Process Descriptions

The text parser component takes a textual description as input and automatically extracts the process information required for the unified data format. This procedure consist of three subsequent steps:

1. *Linguistic analysis of sentences:* The first step concerns the identification of the grammatical entities such as subject, object, predicate, and adverbial for each sentence. What is more, we determine the relations between these entities, e.g. which verb relates to which object. To illustrate the required steps and the associated challenges, consider the sentence *"Hence, the clerk collects the required documents and sends the form to the customer per mail."* from Figure 1. It contains one subject (*"clerk"*), two predicates (*"collects"* and *"sends"*), two objects (*"documents"* and *"form"*), and two adverbials (*"to the customer"* and *"per mail"*). Furthermore, it is important to note the relations between these entities. The predicate *"collects"* relates to the object *"documents"*, whereas the predicate *"sends"* relates to the object *"form"*. To automatically determine these grammatical entities and their relations, we again make use of the Stanford Parser. Besides the recognition of sentence borders, the Stanford Parser is also capable of producing word dependencies [8]. As an example, consider the following two object-related dependencies that the Stanford Parser generates for the considered example sentence:

   ```
   dobj(collects-5, documents-8)
   dobj(sends-10, form-12)
   ```

   These so-called direct-object dependencies (*dobj*) specify which words the Stanford Parser considers to be objects and which predicates relate to them. Thus, the first dependency tells us that *"documents"* (position 8 in the sentence) is an object that relates to the predicate *"collects"* (position 5 in the sentence). Analogously, *"form"* is an object that relates to the predicate *"sends"*. To make use of these generated dependencies, we developed an algorithm that automatically analyzes the Stanford Parser output and extracts the grammatical entities as well as their relations. Our component builds on the knowledge about existing dependencies and the consistent structure of these dependencies (name of the dependency followed by brackets that include two entities and their position). As a result, we are able to automatically obtain a set of grammatical entities and their relations from any given natural language sentence.

2. *Normalization of sentence components:* The words in sentences often do not occur in their base forms, i.e., verbs are not only used as infinitives; nouns are not always provided as singular nouns. This becomes a problem when entities are compared in the context of a search operation. For instance, *"send"*, *"sends"*, and *"sent"* all refer to the same base verb. However, an automated string comparison would indicate that these words differ from each other. To deal with such cases, we use the lexical database WordNet

[26] to convert all words into their base form, i.e., predicates into infinitive verbs and subject as well as objects into singular nouns. As a result, the predicates *"sends"* and *"sent"* are both transformed into *"send"*.

3. *Transformation of sentence components into RDF:* Once the entities have successfully been extracted and transformed into their base forms, the information is stored in the RDF format. To demonstrate this step, again consider the sentence *"Hence, the clerk collects the required documents and sends the form to the customer per mail."* from Figure 1. For each predicate of the sentence, we create a *verb - activity record* RDF triple in order to capture the relation between the predicates and the sentence. The sentence is then represented by an activity record. Suppose this activity record has the identification number 2, then the respective RDF triples look as follows:

   ```
   (collect, part of, ActivityRecord2)
   (send, part of, ActivityRecord2)
   ```

   In addition, we need to link the subjects, objects, and adverbials to the respective verbs:

   ```
   (clerk, relates to, collect)
   (clerk, relates to, send)
   (document, relates to, collect)
   (form, relates to, send)
   (to customer, relates to, send)
   (per mail, relates to, send)
   ```

3.4 Parsing Model-based Process Descriptions

This component expects a set of process models as input and automatically extracts the information required for the unified data format. Similar to the parsing of textual process descriptions, it consists of three subsequent steps:

1. *Linguistic analysis of activity labels:* The linguistic analysis aims at properly deriving the activity components from the labels of the input model set. As discussed earlier in this section, the main challenge is to automatically detect the varying grammatical structures, even if the activity label does not contain a proper verb. As an example, consider the activity *"Customer credit history evaluation"* from Figure 1. For this label it is necessary to automatically recognize that *"evaluation"* represents the action and *"customer credit history"* the business object. To properly derive these components from activity labels, we employ the label analysis technique introduced by [22]. It takes an activity label as input and respectively returns the comprised action(s), business object(s), and additional information fragment(s).

2. *Normalization of activity label components:* Similar to proper natural language sentences, activities often contain inflected words, i.e., verbs occurring in the third person form or nouns used in the plural form. What is more, actions may even represent nouns (e.g., *"evaluation"* in *"Customer credit history evaluation"*). As pointed out for sentences, this has notable implications if two components are compared in the context of a search operation. Hence, we apply the lexical database WordNet [26] also on activity labels to convert all actions into infinitive verbs and all nouns into singular nouns. As a result, the action of the activity label *"Customer credit history evaluation"* is accordingly transformed into *"evaluate"*.

3. *Transformation of activity label components into RDF:* The storage of the extracted and normalized components as RDF works analogously to the storage of the sentence entities. We demonstrate this step using the activity *"Opening of new bank account for customer"*. As a result of applying the previous steps, we identified the action *"open"*, the object *"bank account"*, and the addition *"for customer"*. Suppose the resulting activity record has the identification number 3, then the RDF triples look as follows:

   ```
   (open, part of, ActivityRecord3)
   (bank account, relates to, open)
   (for customer, relates to, open)
   ```

   The example triples illustrate that the action is linked to the activity by using the *verb - activity record* triple. The business object and the addition are respectively associated with the verb by using the *object-verb* and the *adverbial-verb* relation.

In the next section, we show how we query the extracted data from the unified data format.

3.5 Querying the Unified Data Store

In order to query the extracted RDF triples, we use SPARQL (Simple Protocol and RDF Query Language). In essence, SPARQL is similar to SQL (Structured Query Language), the most popular language to query data from relational databases), but is specifically designed to query RDF data. As an example, consider the SPARQL query in Figure 4, which retrieves all process models and textual process descriptions that contain an activity record relating to the verb *"send"* and the object *"form"*.

The example from Figure 4 shows that a SPARQL query has the basic structure of an SQL query, i.e., it follows the *select - from - where* pattern. Before the actual query, however, it is required to define where the data model definition can be found (line 1). As SPARQL is designed for the semantic web, this is done via a Unified Resource Identifier (URI). In this example, for illustration purposes, we use the URI *http://www.processsearch.com/Property/*. After the definition

of this prefix, the actual query starts. Line 2 specifies that we are interested in all process names of process models that fulfill the requirements stated as RDF triples in the block below. Using the variable *?verb*, we define that there must be a *Verb* according to our data model that carries the label *"send"* (lines 5 and 6). Moreover, we use the variable *?object* to define that *?verb* must be related to an *Object* that carries the label *"form"* (lines 8-10). Finally, we define that we are only interested in process models that contain an activity record that relates to an entity *Verb* as specified in *?verb*.

```
1  PREFIX ps: <http://www.processsearch.com/Property/>
2  SELECT ?processName
3  WHERE
4  {
5          ?verb ps:Type "Verb" .
6          ?verb ps:Label "send" .
7
8          ?object ps:RelatesTo ?verb .
9          ?object ps:Type "Object" .
10         ?object ps:Label "form" .
11
12         ?verb ps:PartOf ?activityRecord .
13         ?activityRecord ps:PartOf ?processModel .
14         ?processModel ps:Label ?processName .
15 }
```

Fig. 4. Exemplary SPARQL query to retrieve data from the RDF database

This exemplary query illustrates that an RDF-based unified data store can be easily queried for information we are interested in. To provide the users of our technique with an intuitive feature to search, we implemented a graphical interface in which users can specify the verbs, objects, subjects, and adverbials of the process descriptions they would like to retrieve. The input from the graphical user interface is then automatically inserted into a SPARQL query as provided above. As a result, the user can perform any search based on these four components and does not have to deal with any technical details.

4 Evaluation

In this section, we evaluate our technique with the process repository of an Austrian bank. Our goal is to demonstrate that the additional consideration of textual descriptions yields more comprehensive search results than the sole consideration of process models. We first discuss the setup of our evaluation experiment. Then, we introduce the process model collection we use. Afterwards, we explain the prototypical implementation of our technique. Finally, we present and discuss the results.

4.1 Setup

For the evaluation of our approach, we collaborated with a large Austrian bank. The Business Process Management department of this bank was struggling with two search scenarios that are of particular relevance to the work presented in this paper.

1. *Search for media disruptions:* Media disruptions occur when the information-carrying medium is changed, for example, when a clerk enters the data from a physical letter into an information system. Because media disruptions are often associated with errors, our evaluation partner had a considerable interest in identifying media disruptions in their process landscape. To design an exemplary query, we built on the insights from [5]. In a study on weakness patterns, they found that media disruptions are mainly indicated by the actions *"print"* and *"scan"* as well as the activity *"Enter data"*.
2. *Search for manual activities:* Manual activities are inevitable in most business processes. However, as automation is often associated with saving costs, the identification of automation candidates represents a key task in business process improvement [23]. Thus, our evaluation partner was also interested in identifying which automation potential their process repository exhibits. According to [5], manual activities are typically indicated by the actions *"document"*, *"record"*, and *"calculate"* as well by combinations of the actions *"verify"* and *"archive"* with the business objects *"document"* and *"information"*.

We use queries based on the weakness patterns discussed above to demonstrate the capabilities of our approach and to show the importance of taking text-based process descriptions into consideration.

4.2 Data

The process repository of our evaluation partner consists of 1,667 Event-driven Process Chains (EPCs). The process models cover various aspects of the banking business including the opening of accounts, the management and selling of financial products, as well as customer relationship management. On average, the process models contain 6.5 activities per model. The smallest model contains 1 activity, whereas the largest contains 181 activities. In addition to the process models, the repository contains 119 textual process descriptions in the PDF format. The textual descriptions complement the process models and mainly concern the area of credit management. Due to existing overlaps between the process models in the repository, a single textual description can be referred to by multiple process models. The size of these complementary process descriptions ranges from 119 to 60,558 words. Most of the description are rather long, resulting in an average size of 13,130 words. The language of both the process model elements and textual process descriptions is German.

4.3 Implementation

We implemented the approach defined in Section 3 as a Java prototype. To be able to deal with German process models and process descriptions, we integrated the German package of the Stanford Parser [15], the German component of the label analysis technique from [22], and a German implementation of WordNet called GermaNet [12]. In addition to these techniques, we use the Apache PDFBox to process PDF files and the import functionality from [10] to process different process model formats. Finally, to store the extracted RDF triples, we use the Apache Jena component TDB[2], which is a database optimized for RDF storage and querying.

4.4 Results

The results for both search scenarios are illustrated in Figure 5. The figure shows the aggregated total for the search scenarios and the disaggregated number of retrieved processes for each weakness pattern (e.g., *Verb = "print"* or *Verb = "document"*). The light grey bars indicate the number of processes we retrieved from searching the model-based process descriptions. The dark grey bars indicate the number of processes we retrieved from searching the model-based as well as the textual process descriptions. The results illustrate that our proposition holds: we retrieve additional relevant process models if we also take the textual process descriptions accompanying the models into account. Interestingly, that does not only hold for the total of both search scenarios, but also for each of the weakness patterns as, for instance, for the verb *"print"* in the media disruption scenario or for the verb *"document"* in the manual activity search scenario. Altogether, the number of processes that are retrieved from the process repository increases from 83 to 151 for the media disruption scenario (an increase of 81.9 %) and from 213 to 359 (an increase of 68.5 %) for the manual activity scenario[3].

A detailed analysis of the results revealed that there is no overlap between the processes retrieved from the model-based and the textual descriptions in neither of the search scenarios. This shows that the details of some processes are fully described by process models, while the details of others are only captured in the accompanying textual descriptions. This again highlights the importance of considering both types of process descriptions.

The practical relevance of our technique is further demonstrated by the way it was perceived by our evaluation partner. The bank considered our technique to be highly useful for the analyses they are conducting and decided to integrate it with their ARIS platform. They set up a script that updates the database behind our technique on a daily basis. In this way, search operations can be conducted in an efficient way.

[2] https://jena.apache.org/

[3] Note that because a single textual description can be referred to by several process models, the identification of one relevant textual document may yield multiple relevant process models. This explains why the increase in the number of retrieved processes might be even higher than the total number of textual process descriptions.

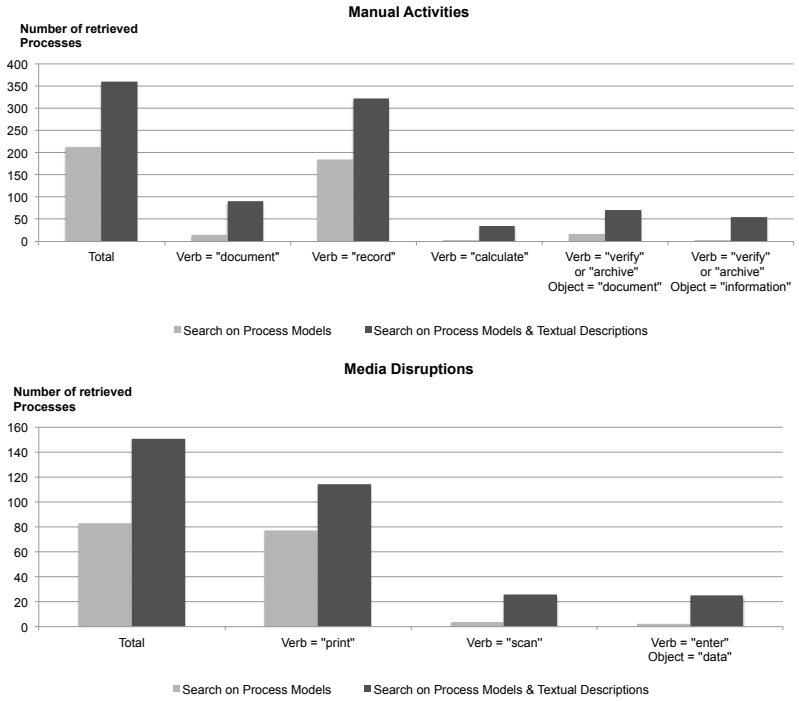

Fig. 5. Results for media disruptions search

While these results are promising, they also have to be discussed in the light of some limitations. First, it is important to note that the investigated process repository is not statistically representative. That is, process repositories from other companies may consist of more or also fewer textual descriptions than the one we investigated. However, our technique does not rely on any specifics we encountered in the repository of our evaluation partner. Thus, we are confident that our technique will perform comparably on other collections. Second, it should be noted that our technique cannot guarantee that all relevant information is identified. One reason is that the user has to define proper key words. Another reason is that our technique can only find information that is explicitly documented in one of the addressed description types.

5 Conclusion

In this paper, we introduced a comprehensive search technique that allows the user to identify information in textual as well as in model-based process descriptions. The technique combines natural language analysis tools in a novel way and builds on the transformation of textual and model-based process descriptions into a unified data format. We implemented the technique as a Java

prototype that stores the extracted data in an RDF database and provides the user with a graphical interface to specify queries. An evaluation with a large bank showed that our solution can be successfully applied in industry and that that the additional consideration of textual process descriptions indeed increases the number of identified processes.

From a research perspective, the proposed technique provides the foundations for integrating textual and model-based information. To the best of our knowledge, we are the first to define an integrated data format that allows to combine the process information from these two process description types. Hence, our technique can improve existing process search techniques and may help to increase their scope. From a practical perspective, our technique helps organization to perform more comprehensive search operations. As demonstrated in the evaluation, textual sources may contain equally relevant information about processes as model-based descriptions.

In future work, we plan to extend our approach with respect to structural process properties. To this end, we aim at integrating behavioral aspects from process models into the data format. In addition, we plan to define a technique that is capable of extracting such behavioral aspects from textual process descriptions.

References

1. Aa, van der, H., Leopold, H., Mannhardt, F., Reijers, H.A.: On the fragmentation of process information: Challenges, solutions, and outlook. In: BPMDS'15 Working Conference (2015)
2. Awad, A., Decker, G., Weske, M.: Efficient compliance checking using bpmn-q and temporal logic. In: Business Process Management, pp. 326–341. Springer (2008)
3. Awad, A., Polyvyanyy, A., Weske, M.: Semantic querying of business process models. In: Enterprise Distributed Object Computing Conference, 2008. EDOC'08. 12th International IEEE. pp. 85–94. IEEE (2008)
4. Baier, T., Mendling, J.: Bridging abstraction layers in process mining by automated matching of events and activities. In: Business Process Management, pp. 17–32. Springer (2013)
5. Becker, J., Bergener, P., Räckers, M., Weiß, B., Winkelmann, A.: Pattern-based semi-automatic analysis of weaknesses in semantic business process models in the banking sector (2010)
6. Candan, K.S., Liu, H., Suvarna, R.: Resource description framework: Metadata and its applications. SIGKDD Explor. Newsl. 3(1), 6–19 (Jul 2001)
7. Davies, I., Green, P., Rosemann, M., Indulska, M., Gallo, S.: How do practitioners use conceptual modeling in practice? Data & Knowledge Engineering 58(3), 358–380 (2006)
8. De Marneffe, M.C., Manning, C.D.: The stanford typed dependencies representation. In: Coling 2008: Proceedings of the workshop on Cross-Framework and Cross-Domain Parser Evaluation. pp. 1–8. Association for Computational Linguistics (2008)
9. Dijkman, R.M., Dumas, M., García-Bañuelos, L.: Graph matching algorithms for business process model similarity search. In: Proceedings of the 7th International

Conference on Business Process Management. pp. 48–63. BPM '09, Springer-Verlag, Berlin, Heidelberg (2009)
10. Eid-Sabbagh, R.H., Kunze, M., Meyer, A., Weske, M.: A platform for research on process model collections. In: Mendling, J., Weidlich, M. (eds.) Business Process Model and Notation, Lecture Notes in Business Information Processing, vol. 125, pp. 8–22. Springer Berlin Heidelberg (2012)
11. Friedrich, F., Mendling, J., Puhlmann, F.: Process Model Generation from Natural Language Text. In: Proceedings of the 23rd international conference on Advanced Information Systems Engineering. LNCS, vol. 6741, pp. 482–496. Springer (2011)
12. Hamp, B., Feldweg, H.: Germanet - a lexical-semantic net for german. In: Proceedings of ACL workshop Automatic Information Extraction and Building of Lexical Semantic Resources for NLP Applications. pp. 9–15 (1997)
13. Indulska, M., Green, P., Recker, J., Rosemann, M.: Business process modeling: Perceived benefits. In: Conceptual Modeling-ER 2009, pp. 458–471. Springer (2009)
14. Jin, T., Wang, J., Wu, N., La Rosa, M., Ter Hofstede, A.H.: Efficient and accurate retrieval of business process models through indexing. In: On the Move to Meaningful Internet Systems: OTM 2010, pp. 402–409. Springer (2010)
15. Klein, D., Manning, C.D.: Accurate Unlexicalized Parsing. 41st Meeting of the Association for Computational Linguistics pp. 423–430 (2003)
16. Kunze, M., Weidlich, M., Weske, M.: Behavioral similarity–a proper metric. In: Business Process Management, pp. 166–181. Springer (2011)
17. Leopold, H., Mendling, J., Polyvyanyy, A.: Supporting process model validation through natural language generation. Software Engineering, IEEE Transactions on 40(8), 818–840 (2014)
18. Leopold, H.: Natural Language in Business Process Models: Theoretical Foundations, Techniques, and Applications, LNBIP, vol. 168. Springer (2013)
19. Leopold, H., Eid-Sabbagh, R.H., Mendling, J., Azevedo, L.G., Baião, F.A.: Detection of naming convention violations in process models for different languages. Decision Support Systems 56(0), 310–325 (12 2013)
20. Leopold, H., Mendling, J.: Automatic derivation of service candidates from business process model repositories. In: Business Information Systems. pp. 84–95 (2012)
21. Leopold, H., Niepert, M., Weidlich, M., Mendling, J., Dijkman, R.M., Stuckenschmidt, H.: Probabilistic optimization of semantic process model matching. In: BPM. pp. 319–334 (2012)
22. Leopold, H., Smirnov, S., Mendling, J.: On the refactoring of activity labels in business process models. Information Systems 37(5), 443–459 (2012)
23. Limam Mansar, S., Reijers, H.A.: Best practices in business process redesign: use and impact. Business Process Management Journal 13(2), 193–213 (2007)
24. Malinova, M., Leopold, H., Mendling, J.: An empirical investigation on the design of process architectures. In: Wirtschaftsinformatik (2013)
25. Mendling, J., Reijers, H.A., Recker, J.: Activity Labeling in Process Modeling: Empirical Insights and Recommendations. Information Systems 35(4), 467–482 (2010)
26. Miller, G., Fellbaum, C.: WordNet: An Electronic Lexical Database. MIT Press, Cambridge, MA (1998)
27. Qiao, M., Akkiraju, R., Rembert, A.J.: Towards efficient business process clustering and retrieval: combining language modeling and structure matching. In: Business Process Management, pp. 199–214. Springer (2011)
28. Yan, Z., Dijkman, R., Grefen, P.: Fast business process similarity search. Distributed and Parallel Databases 30(2), 105–144 (2012)