# Business Process Search within Unstructured Repositories

Maya Lincoln and Avi Wasser
*ProcessWeb, Haifa, Israel*

Keywords: Business Process Search, Process Ontology, Business Process Repositories, Natural Language Processing, Semantic Search, Operational Search.

Abstract: In recent years, researchers have become increasingly interested in developing frameworks and tools for searching business process model repositories. While research on searching structured repositories has been extensive, little attention was dedicated to searching business process content within unstructured repositories, such as the Web. We demonstrate why current search technologies are not useful for extracting process content from the Web, and explain the core reasons for the deficiency. We then present a framework for overcoming this material weakness, and discuss possible applications for realizing the suggested method.



## 1 INTRODUCTION

Business Process Models (BPMs) are considered an important mine of organizational knowledge, and therefore are a major source for searching and retrieving operational and enterprise related data.

Researchers have become increasingly interested in developing methods and tools for retrieving information from business process repositories (Awad, 2008; Lincoln and Gal, 2011; Wasser et al., 2006). While research on searching structured repositories has been extensive, little or no attention was dedicated to searching business process content within unstructured repositories, such as the Web. Such repositories are constantly becoming more extensive, and are accessible to a wide user population through search engines.

Two common methods for retrieving information from a repository are querying and searching. Querying is aimed at retrieving information using a structured query language. The significance of querying business processes has been acknowledged by BPMI that launched a Business Process Query Language (BPQL) initiative. Searching, on the other hand, allows information retrieval using keywords or natural language and was shown to be an effective method for non-experts.

Research in the field of business process retrieval has mainly focused on semantics and structural similarity analysis techniques (Awad, 2008; Markovic et al., 2008; Beeri et al., 2008; Karni et al., 2014). Using these frameworks one can retrieve process models that either contain semantically related components (e.g. activity names with a specified keyword) or match a requested graph structure: e.g. that presents a sequence of activities. While these methods can be applied on structured process repositories, it is practically impossible to apply them on the unstructured Web.

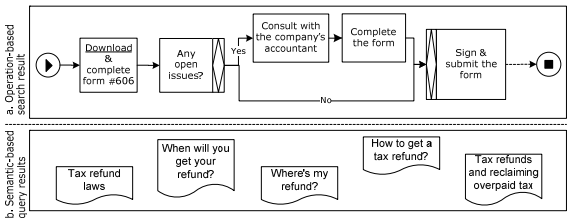


Figure 1: An example of search results for "how to claim a tax refund."

In order to illustrate why semantic search is not adequate for process retrieval from unstructured repositories, we will present a motivating example, as follows. Consider an employee interested in finding out "how to claim a tax refund." An expected outcome of this retrieval request would be a process model that represents the order of activities that one should follow in order to achieve the required process goal, as illustrated in Fig. 1a. The benefit of such a retrieval framework is that the result is ready for execution. Without any preliminary knowledge of the underlying repository structure, the user can receive a full-fledged process

Lincoln M. and Wasser A..
Business Process Search within Unstructured Repositories.
DOI: 10.5220/0005169404670474
In *Proceedings of the International Conference on Knowledge Engineering and Ontology Development* (KEOD-2014), pages 467-474
ISBN: 978-989-758-049-9

model.

The retrieval output is related to the search phrase in operational terms. For example, Fig. 1a provides a segment that is not similar semantically to the search phrase text. Specifically, all three search phrase terms ("Claim", "Tax" and "Refund") are not represented by any of its activities. Such "how-to" questions are hard to fulfill using common query languages due to the complex logic that is embedded within such questions (Beeri et al., 2008) and especially without specific knowledge on process structure and activity naming. Therefore, using querying techniques on the Web, would yield a list of data items (e.g. Web pages, or media items) with semantically similar titles, as illustrated in Fig. 1b. Such outcome does not tell the user "how-to" fulfill the process goal in a structured and operational manner.

In this work we present a framework that aims to overcome this material weakness. The suggested retrieval framework is based on operational, and not on semantic similarities. This is done by processing the unstructured Web content, into a structured process repository. Then, the operational business logic is extracted from the generated repository through the analysis of process components. The proposed framework dynamically extracts process models according to the ad-hoc requests as expressed in the user's search phrase.

This work proposes an innovative method for searching business process models within the Web, while making use of the how-to knowledge that is encoded in this valuable repository. The following contributions are presented: (a) automatic construction of processes from unstructured, non-BPM, repositories; (b) generic support to an operation-based search of business process models within the Web; and (c) capability to generate ad-hoc process model results from natural language or graph-based Web search.

The rest of the paper is organized as follows: we present related work in Section 2, positioning our work with respect to previous research. In Section 3 we present the major shortcomings of current Web search engines in extracting process data. Section 4 formulates the search problem and describes our framework for searching business processes within the Web. We discuss future research elaborations and conclude in Section 5.

## 2 RELATED WORK

Related works include query and search techniques in BPM. Works such as Shao et al. (2009) and Awad (2007) query business process repositories to extract process model (graph) segments. Such methods require prior knowledge of the structure of the process repository and the exact notation that is used to express it. Therefore, they are not adequate for search on the Web that should work well even without prior knowledge regarding the process repository.

Keyword search on general tree or graph data structures can also be applied to process repositories (Hristidis et al., 2003; Guo et al., 2003; He et al., 2007). Some works extend the tree and graph keyword search methods to support a more intuitive interface for the user by enabling searches based on natural language (Katz et al., 2010). The retrieved information in both keyword and natural language search methods is in the form of single process model components such as activities and roles that are semantically similar to the searched phrase. These techniques are merely relevant to process search on the Web, since in this case (a) users are seeking to receive a complete process; and (b) the expected process result is usually not related semantically to the search phrase, but rather operationally. The work of Lincoln and Gal (2011) extends the above line of works. This work supports the retrieval of complete process segments by applying dynamic segmentation of the process repository. The search result is a compendium of data (a segment of a business process model) related to the operational meaning of the searched text. Nevertheless, as all other works, this method relies also on a process-structured database, and cannot work "as is" on an unstructured repository, such as the Web.

Another line of work focuses on automatic construction of process data ontologies. The work of Belhajjame and Brambilla (2009) proposes a query-by-example approach that relies on ontological description of business processes, activities, and their relationships. The work of Lincoln and Gal (2011) automatically extracts and uses the operational layer (the "how-to") and the business rules encapsulated in a process repository. Such automatic ontology extraction techniques are important for analyzing data encapsulated in the Web. Nevertheless, the current research literature is based solely on process-flow structured repositories and not on unstructured repositories such as the Web.

# 3 KEY DEFICIENCIES OF SEMANTIC BUSINESS PROCESS SEARCH

Current Web searches are based on keyword queries and semantic similarity lookups. This makes data extraction relatively easy and simple for users. Nevertheless, and as demonstrated in Section 1, it is practically impossible to extract processes from the Web using current semantic search engine technology. This is an inherent material weakness that in our opinion presents a significant barrier for the evolution of Web usability.

The main search engines (e.g. Google, Microsoft Bing, Ask, and others) are still at experimental, initial phases of enabling Web process-searches. For example, recent R&D efforts of Google yield lists of "how to" instructions for very limited process sets. For instance, when searching in Google "how to issue an invoice," a set of related documents and media is retrieved, without any process-formatted results. However, in some cases, we identified initial attempts to retrieve instruction-based results for certain "how to" queries. These results are presented before the standard Google search results, within a dedicated frame, in a list-format, which is a first step in aiming to retrieve and present process-flow formats. This presentation is still at a preliminary phase as Google requests users' feedback regarding the quality of these instruction lists.

Besides these attempts to present somewhat process-driven results, we note that standard search results for "how-to" or "process-driven" queries are very limited, again due to the aforementioned material weakness of semantic search. The work of Wasser and Lincoln (2014) presents examples of process-search scenarios using the top four search engines (Google, Bing, Yahoo and Ask). The results demonstrated that current search technologies cannot provide adequate results. The sample searches included not only business process but also personal process queries as the authors believe that the size and amount of data presented in the Web will also extend the scope of process related searches beyond the domain of BPM.

Clearly, these results encompass a large, unstructured, set of data with a low level of usability. Results are not presented in standardized process notations- nor in basic flowchart formations. Practically, for the end-user, it is not possible to deduct an actual process from search results. It is not clear what the required steps are and what is the order of activities for achieving the process goal. It is also not possible assess the quality and relevance of the suggested results from an operational viewpoint- as ranking is based on semantics and not on operational characteristics of a process. Hence, these samples along with the elaborated example presented in section 1 demonstrate current process-search deficiencies and the need for an alternative framework that will support such Web searches.

The work of Wasser and Lincoln (2014) estimates the demand for such a solution for process-based queries. According to this work, As of July 2014, "how-to" related searches are conducted over 91.4 million times per year. This amount emphasizes the need for process search, and is expected to grow significantly when it will be possible to retrieve proper, process-format, results from these searches.

# 4 FRAMEWORK SPECIFICATIONS FOR WEB PROCESS SEARCHES

In this Section we describe the main expectations from a framework for process search within unstructured data repositories, such as the Web. So far, this paper has focused on analyzing deficiencies of current search method, and this section is aimed at highlighting the high-level steps and requirements required to fulfil an adequate framework.

The main target of the framework is to make processes accessible and usable for everybody through simple Web searches. The framework should enable searches on "how to do things" using a simple query language for expert as well as non-expert users - using their natural language terminology for describing the process goal. While current Web searches are based on keywords and semantic similarity, the suggested high-level retrieval framework is based on operational similarity.

The suggested framework includes an input, an output and five processing steps (see illustration in Fig. 2). To discuss the main concepts, we first present the required definitions, and based on these definitions we then describe the main requirements for each step, as follows.

## 4.1 Definitions

A Business Process Model is a directed graph, $M = (V, E)$, where $V$ is a set of nodes and

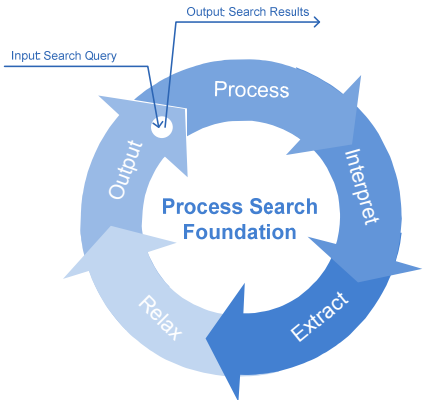


Figure 2: A high-level framework for process search on the Web.

$E \subseteq V \times V$ is a set of edges. Each node $v$ is associated with a set of (attribute,value) pairs, denoted $A(v)$. Given an attribute $a$, we use $a(v)$ to denote the set of values d s.t. $(a,d) \in A(v)$. An example of a Business Process Model (BPM) related attributes and values is presented in Table 1. Each edge is associated with a label, denoted $l(v)$. We use $\varepsilon$ to denote the empty label.

Table 1: An example of BPM related attributes and values for a node, $v$.

| Attribute ( $a$ ) | Description | Value ( $d$ ) |
|---|---|---|
| Type | Describes the type of data that the node represents. Each node posses exactly one Type attribute | Activity, decision, control, role, event (including the start and end events) |
| Text | Describes an action when type=activity/decision/control, and an event when type=event. Each node posses exactly one Text attribute | Action: "Perform project period close," "Open a new accounting period" Event: "Project termination," "New accounting period started" |
| Role | Describes the executing party of the node's action (relevant for type=activity/decision/control). Each node posses exactly one Role attribute | Accounts receivable clerk, System administrator, Human resource manager |
| Document | Represents a document involved in the action's execution (relevant for type=activity/decision/control). Each node can be related to 0-n Documents | Customers list, pricing list, an invoice |

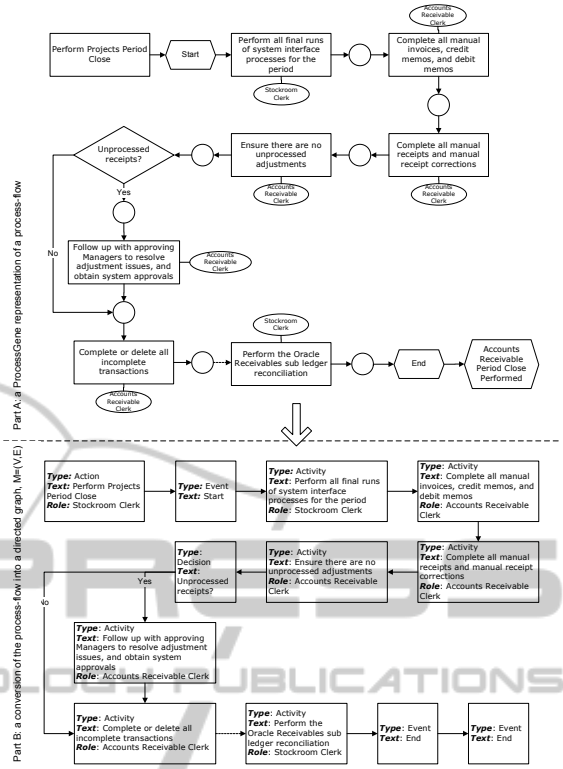


Figure 3: Conversion of a process flow diagram (Part a) into the directed graph (Part b).

Fig. 3 presents a simple flowchart diagram of a business process (Part a) and its conversion into the directed graph, $M=(V,E)$, using the above component definitions (Part b).

## 4.2 Input

The framework is aimed at enabling users to define "how to" questions such as: "How can I assure regulatory compliance in sales processes?," "What non-discrimination measures are being taken in HR processes?," and "How can I initiate a purchase order without a purchase requisition?." To do that, it is required to select an adequate query language.

When searching for the most suitable query language we should take into account two user types: (a) the simple, common user that has no knowledge in BPM; and (b) an expert user that is familiar with BPM. As for the simple user - natural language is the easiest way for phrasing his search goal. This allows him to phrase the question in his own words. For the expert user, the query language should also be intuitive but will enable further specifications that will support more specific searches. Therefore, the query language should enable querying the structural level as well (namely:

a "structure-aware query"). Taking these issues into account, we came to the following conclusions: (a) the query language for expert users should be based on keywords and natural language for searching the BPM content layer and optionally - labels for involving also structural constraints; (b) the combination of labels and keywords should be based on the simple flow diagram principles.

For that, two language syntaxes should be defined:

1. A graphical syntax – enabling the user to express queries using flow diagram graphical components (e.g., activities, roles, edges)
2. A textual syntax – enabling the user to express queries using keywords/natural language phrases within optional flow diagram patterns

More formally, the structure-aware query language can be defined as follows. A structure-aware query is a directed graph, $Q'=(V',E')$, where $V'$ is a set of nodes and $E'\subseteq V'\times V'$ is a set of edges. Each node $v'$ is associated with a set of (attribute, regular-expression) pairs, denoted $A'(v')$. The regular expressions refer to the attribute values in *M*'s nodes. An example for an (attribute, regular-expression) pair is: (Text, *Financ*), which means that the description related to this node contains the string: "Financ". Given an attribute $a'$, we use $a'(v')$ to denote the set of regular expressions re s.t. $(a',re)\in A'(v')$. Finally, each edge is associated with a binary function, $f$, denoted $f(v')$, that returns a Boolean value. This function describes the relationship between two nodes. Examples of $f(v')$ are:

1. A decision that leads to a report-related event.
2. Two activities that are not connected by any path to each other.
3. Two nodes with certain attributes that are connected by a limited path length (see illustration in Fig. 4). According to this example, the Boolean function associated with the edge that connects the two nodes will returns *True* only if the graph path between the nodes contains no more than two nodes. In addition, the first node is a control action, executed by an accountant, and the second node represents a decision that involves a receipt-like text.

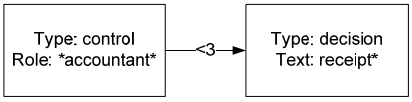


Figure 4: An example of a structure-aware query in which two activities are connected by no more than two steps.

## 4.3 Process

The initial phase of the search process aims at processing the unstructured Web data into a structured process repository. Technically, it will not be feasible, nor efficient, to decompose the entire Web data into a BPM structure "on-demand" for every search query. Therefore, this conversion should be executed after each Web content change, similarly to the search engine's indexing and data mapping processes.

First, it is required to identify process "blocks" and process elements within Web pages. We will elaborate on this part in future works. Second, it will be required to decompose textual phrases related to the identified process elements (e.g. activities) into meaningful process ontologies - in order to further analyze their meaning and build structural process taxonomies automatically. This should be done by utilizing NLP methods, as well as by deploying process textual-decomposition models. Third, based on the extracted process structures and the generated process ontologies, it is required to generate unique process data taxonomies that will represent the Web encapsulated know-how, and will further assist in processing the search query. We will elaborate and present examples of such taxonomies in future work.

The outcome of this phase is (a) a collection of process-flows in the format of directed graphs; and (b) process taxonomies that further enable the processing of search queries on the generated graphs.

## 4.4 Interpret

This phase is required only for natural language queries. At this stage it is required to interpret the natural language phrase into process-aware notations. This is done automatically, and serves as a basis for machine processing of the search query.

## 4.5 Extract

At this phase it is required to extract related process segments from the structured repository that fulfil the search goal, and combine them into process-format search results. Note that a naïve solution at this phase would be to examine all possible process model segments within the repository. Nevertheless, such algorithm can be highly inefficient. Therefore, as a preliminary step, it is first required to reduce the set of segments by selecting only relevant candidates.

For natural language queries such extraction and

reduction process can be performed using state of the art operational search methods. For a structure-aware query, both goals will be achieved by performing a graph search according to the following definition. A query result is a mapping of $Q$ to $M$, in which:

$$\text{For each } v' \in V'(Q),\ f(\mu(v')) = T \quad (1)$$

$$\text{For each } (v', re) \in E'(Q),\ f(\mu(v'), \mu(re)) = T \quad (2)$$

Where $T$ symbolizes the Boolean value *True*.

## 4.6 Relax

In cases where the search phrase retrieves only few or no results, it is required to apply search phrase relaxation rules to optimize the search-result range. The location of each rule in the list will represent its relative priority. For natural language (NL) queries, the relaxation rules should be carried out on the process ontology model, generated at the former "Interpret" phase. Examples for such relaxations are presented as follows:

1. Convert each ontology component (e.g. action, activity, object, attribute) into their synonyms
2. Replace the action with an action located at a higher or lower hierarchal level in the ontology model
3. Replace the object with an object located at a higher or lower hierarchal level in the ontology model
4. Replace the action with a more advanced or prior action at the ontology model
5. Replace the object with a more advanced or prior object at the ontology model

In case of a structure-aware query, the relaxation rules can be elaborated to express also structure-based logics. Examples of such relaxation rules are listed in Table 1 below, where "E" symbolizes an edge-related relaxation rule and "N" a node-related one.

The relaxation process ends at the earliest of either (a) reaching the expected amount of results; or (b) implementing all relaxation rules.

## 4.7 Output

The goal of this phase is to output a ranked list of full-fledged process models. This phase includes three main steps, as follows (see illustration in Fig. 5).

Table 1: An example of structure-aware query relaxation rules.

| # | Type | Example |
|---|---|---|
| 1 | E | Enlarge the limit regarding the connecting path's length between two nodes |
| 2 | E | Enable a long path length between two nodes, instead of a "no connecting path" constraint |
| 3 | N | Replace "Activity" with "Decision" in all "Type" related regular expressions, and vice versa |
| 4 | N | Replace "Activity" with "Control" in all "Type" related regular expressions, and vice versa |
| 5 | N | Replace "Control" with "Decision" in all "Type" related regular expressions, and vice versa |
| 6 | N | Add a wild-card at the beginning and at the end of each Text related constraint |
| 7 | N | Remove all query components related to "Role" |
| 8 | N | Remove all query components related to "Type" |

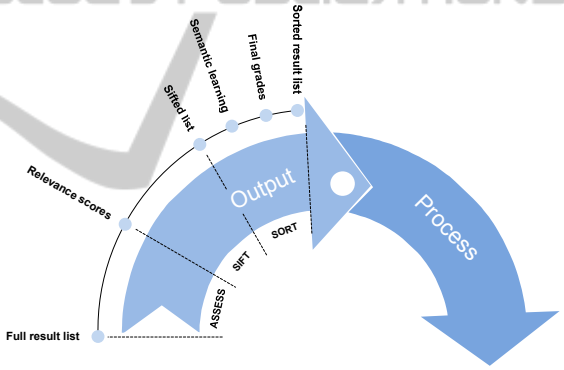


Figure 5: The main steps comprising the "Output preparation" phase.

### 4.7.1 Assess

The framework aims at returning search results ranked by their relative importance, so that users will be able to examine them more efficiently and effectively. Therefore, this phase's goal is to assess the relevance of result candidates, retrieved at the previous phases, to the search request. In case of an NL-based search, it is required to calculate the proximity of each process result to the process ontology model that represents the search query. To do that it is possible to use proximity assessment methods. In case of a structure-aware query it is required to calculate the similarity between two graphs: each result vs. the query graph. It is possible to calculate this similarity using one of the state-of-

the-art methods for assessing similarity between process models, e.g. the works of Dongen et al. (2008), Ehrig et al. (2007), and van der Aalst et al. (2006). On top of this score, in case the advanced structure-aware query also allows the user to specify importance weights on each edge, it is possible to add an additional score that reflects the weights of the matched edges. Note that it will not be required to handle results that were produced based on relaxation rules differently, since they will naturally be panelled by the similarity calculation methods.

#### 4.7.2 Sift

At this phase several thresholds should be used to determine the inclusion of each result candidate in the final result list. Examples for non-inclusion rules may be as follows: (a) very short results may be excluded if most results are much longer; or (b) exclusion of results in which the action flow does not match any action sequence in the action sequence model.

#### 4.7.3 Sort

Eventually, it is required to sort the list of search results according to their similarity score, as calculated during the above "Assess" stage. As an advanced proposition, it will also be recommended to apply learning capabilities that will opt to improve the ranking quality for each specific user. An example of such learning mechanism is presented in the work of Wasser and Lincoln (2012). The learning mechanism in that work analyzes, in real-time, the linguistic relationships between process ontology models and adjusts them according to previous human inputs. As part of the search process it will be possible to collect such inputs from previous searches and user-specific result selections. The learning mechanism can increase the effectiveness of the method.

## 5 CONCLUSIONS

We presented a framework for searching process models within the Web. The framework aims to overcome the shortcomings of existing search technologies within unstructured repositories. The proposed framework provides a starting point that can already be applied in real-life scenarios, yet several research issues remain open- to be addressed in future research. We mention three such extensions here. First, formalizing the framework into a detailed executable method. Second, extending the models of process logic for determining the ranking of extracted results. Third, extending the set of relaxation rules. It is hoped that by expanding search and query capabilities of processes within the Web, users will be able to extract operational knowledge more simply and efficiently.

## REFERENCES


Awad, A., 2007. BPMN-Q: A Language to Query Business Processes. In EMISA, volume 119, pages 115128.

Awad, A., Polyvyanyy, A., Weske, M., 2008. Semantic querying of business process models. In *12th International IEEE Enterprise Distributed Object Computing Conference*, pages 8594. IEEE.

Beeri, C., Eyal, A., Kamenkovich, S., Milo, T., 2008. Querying business processes with BP-QL. Information Systems, 33(6):477507.

Belhajjame, K., Brambilla, M., 2009. Ontology-based description and discovery of business processes. Enterprise, Business-Process and Information Systems Modeling, pages 8598.

Ehrig, M., Koschmider, A., Oberweis, A., 2007. Measuring similarity between semantic business process models. In *Proceedings of the fourth Asia-Pacific conference on Conceptual modelling - Volume 67*, APCCM '07, pages 7180, Darlinghurst, Australia, Australia. Australian Computer Society, Inc.

Guo, L., Shao, F., Botev, C., Shanmugasundaram, J., 2003. XRANK: Ranked keyword search over XML documents. In *Proceedings of the 2003 ACM SIGMOD international conference on Management of data*, pages 1627. ACM.

He, H., Wang, H., Yang, J., Yu, P.S., 2007. BLINKS: ranked keyword searches on graphs. In *Proceedings of the 2007 ACM SIGMOD international conference on Management of data*, pages 305316. ACM.

Hristidis, V., Papakonstantinou, Y., Balmin, A., 2003. Keyword proximity search on XML graphs.

Karni, R., Wasser, A., Lincoln, M., 2014. Content analysis of business processes. *International journal of e-business development*.

Katz, B., Lin, J., Quan, D., 2010. Natural language annotations for the Semantic Web. On the Move to Meaningful Internet Systems 2002: CoopIS, DOA, and ODBASE, pages 13171331.

Leopold, H., Smirnov, S., Mendling, J., 2010. Refactoring of process model activity labels. In Natural Language Processing and Information Systems, pages 268276. *Springer*.

Lincoln, M., Gal, A., 2011. Searching business process repositories using operational similarity. On the Move to Meaningful Internet Systems: OTM, pages 219.

Lincoln, M., Golani, M., Gal, A., 2010. Machine-assisted design of business process models using descriptor

space analysis. Business Process Management, pages 128144.

Markovic, I., Pereira, A. C., Stojanovic, N., 2008. A framework for querying in business process modelling. In *Proceedings of the Multikonferenz Wirtschaftsinformatik (MKWI)*, Munchen, Germany.

Shao, Q., Sun, P., Chen, Y., 2009. WISE: a workflow information search engine. In ICDE'09. *IEEE 25th International Conference on*, pages 14911494. IEEE.

van der Aalst, W., de Medeiros, A., Weijters, A., 2006. Process equivalence: Comparing two process models based on observed behaviour. In Business Process Management, pages 129144. *Springer Berlin / Heidelberg*.

van Dongen, B. F., Dijkman, R. M., Mendling, J., 2008. Measuring similarity between business process models. In Advanced Information Systems Engineering, *20th Int. Conference, CAiSE 2008*, Montpellier, France, pages 450464. Springer.

Wasser, A., Lincoln, M., 2012. Semantic machine learning for business process content generation. In On the Move to Meaningful Internet Systems: OTM 2012, pages 7491. *Springer*.

Wasser, A., Lincoln, M., 2014. Key Deficiencies of Semantic Business Process Search. COOPIS workshops: INBAST.

Wasser, A., Lincoln, M., Karni, R., 2006. ProcessGene querya tool for querying the content layer of business process models. In *Demo Session of the 4th International Conference on Business Process Management*, pages 18.