Automated Process Model Annotation Support: Building Blocks and Parameters

Michael Fellmann[1], Felix Oehmgen[1]

[1] Institute of Computer Science, University of Rostock, Rostock, Germany
{michael.fellmann, felix.oehmgen}@uni-rostock.de

Abstract. In business process modeling, semi-formal models typically rely on natural language used to express the labels of model elements. This can easily lead to ambiguities and misinterpretations. To mitigate this issue, the combination of process models with formal ontologies or predefined vocabularies has often been suggested. A cornerstone of such suggestions is to annotate elements from process models with ontologies or predefined vocabularies. Although annotation is suggested in such works, past and current approaches rarely discuss building blocks, parameters and strategies for automating the tedious and error-prone manual task. In this paper, we hence first describe the nature of the annotation task. We then identify building blocks and parameters for automated systems and describe an implementation of an annotation system we used to conduct first empirical studies on the effect of parameters. The paper at hand in sum presents design options and parameters for (semi-) automatically linking semi-formal process models with more formal knowledge representations. It hence may be a source of inspiration for further explorations and experiments on that topic.

keywords: Business Process, Semantic Annotation, Automatic Matching.

1 Introduction

In business process modeling, semi-formal modeling languages such as BPMN are used to specify which activities occur in which order within business processes. Whereas the order of the activities is specified using constructs of the respective modeling language, the individual semantics of a model element such as “Check order” is bound to natural language. However, if models have to be interpreted by machines, e.g. for offering modeling support, querying on a semantic level [20] or content analysis, a more formal, machine processable semantics of modeling elements is required [1]. More use cases that would be possible if an automated annotation could be realized are described in more detail in [18]. In the past, several approaches tried to formalize the semantics of individual model elements by annotating elements of ontologies or other predefined vocabularies that to some degree formally specify the semantics of a model element. However, such approaches suffer from a major limitation: Annotation is a highly manual and tedious task. The user has to select suitable elements of an ontology by browsing the ontology or doing a keyword-based search in the labels of the ontology. Even if the system is capable of presenting some annotation suggestions, e.g. based on lexical similarity of labels, the user has to make sure that annotations match the appropriate context in the process model by inspecting the structure of the ontology that typically is organized in a hierarchy. For example, if the ontology contains two activities labelled with "Accept invitation", it is important whether this activity is part of the hiring process (where the applicant accepts e.g. a job interview) or the planning process for business trips (where the employee accepts an invitation of a business partner). In other words, the semantic context of an element that is to be annotated must be considered. Since only a very limited number of highly automated context-sensitive approaches for process model annotation is available so far (see [18] for an overview on current and past annotation approaches, [19] for an implementation using Markov Logic), this contribution is meant to facilitate developing, comparing and optimizing such approaches. To bootstrap systematic research in this direction, we describe building blocks and parameters (in short: design options) for automated annotation. With this, interest in a very promising research topic should be raised; both in regard to scientific outcome as well as practical usefulness (for use cases, see e.g. [18]).

The remainder is structured as follows. In Section 2, the annotation task is described and three major building blocks for semantic annotation are identified. In Section 3, these building blocks along with their parameters are described in more detail. In Section 4, first considerations and results for/of an empirical analysis are given. In Section 5, related work is discussed and in Section 6 the article is concluded.

2 Description of the Annotation Task

2.1 Fundamental Characteristics of the Annotation Task

Semantic annotation as investigated in this paper means linking process model tasks (e.g. a task such as "Check order") with elements of an ontology or vocabulary such as "Order checking"). We denote these elements as "concepts". In regard to the characteristics of the ontology or vocabulary used for annotation, we assume that it is structured in a hierarchical way, that semantics of the hierarchy is "part-of" and that there is a partial ordering between siblings in the hierarchy. This assumption seems to be justified when considering major examples of vocabularies or ontologies such as the PCF (Process Classification Framework), a publicly available collection of approx. thousand enterprise activities which is also available industry-specific versions [2]. Another example is the MIT Process Handbook [3], a large collection of enterprise knowledge integrated into an ontology where activities are also ordered in a part-of-hierarchy.

2.2 Deriving Building Blocks for IT-Support by Observing Human Annotators

In order to understand which building blocks are required for an automated annotation approach, it is helpful to observe and interview human annotators about their strategy. We did so by observing and interviewing students who manually annotated business process models as a part of a tutorial. Process models were specified in the BPMN language and annotated with elements of the PCF (Process Classification Framework) taxonomy. 50 undergraduate students with good knowledge in process modelling participated in small groups in the exercise in the years 2012-2014 and annotated 23 models in a group effort. Since this empirical work is not in the center of the article at hand, we only roughly report the insights we gained. A recurring pattern that has been observed both directly and by interviewing the students has been that annotation roughly followed a 3-step procedure: *First*, keyword search was performed to search for relevant elements of the PCF taxonomy. *Second*, in case that multiple relevant elements of the taxonomy were found, the context of these elements was considered and items of the taxonomy were preferred that better correspond to the overall topic of the process. For example, if the topic of the process was Human Resources (HR), participants preferred activities belonging to the category "6. Human Resources" of the PCF taxonomy. *Third*, in a last step, the selection of an item for annotation was reviewed considering the annotation of preceding and following model elements to verify that it is meaningful and fits the process context. In this step, the partial ordering of the activity taxonomy was taken into consideration meaning that if activities in the taxonomy appeared to occur in a meaningful order (e.g. check order, approve order, execute order), participants strived to not violate that order in the annotations. In this step, also activities that are on a similar hierarchy level (i.e. that are not more specific or detailed) than those selected for the surrounding model elements have been preferred, if possible. In sum, roughly three steps were executed: (1) retrieve annotation candidates by lexical matching, (2) put annotation candidates into context and select the most meaningful and (3) optimize annotation in regard to the annotations of surrounding model elements in terms of order and hierarchy level. These three steps inspire corresponding building blocks of an automated annotation approach which we refer to as *element annotation*, *context detection* and *annotation fitting*. They are described in the following along with adjustment parameters.

3 Building Blocks and Parameters

3.1 Element Annotation

For annotating process model activities, relevant activity concepts in the taxonomy have to be found. It is thus necessary to match model element labels against activity concepts from the vocabulary as it is illustrated in Fig. 1. To match process labels against vocabulary concepts, we basically need a similarity function $sim_{ac}()$ that returns the similarity between a process activity $a \in A$ and an activity concept $c \in C$ between 0 and 1.

$$sim_{ac}(a, c) \in [0,1] \tag{1}$$

Using this function, a set of *annotation candidates* M (for "metadata") can be computed containing process elements a that match to vocabulary concepts c with a matching value $s \in [0,1]$ being above a similarity threshold thr_{sim} and that occurs between a minimum level l_{hmin} (to exclude root node) and maximum level l_{hmax} (to prevent too fine-grained annotations) hierarchical position in the taxonomy. The hierarchical position for a concept $c \in C$ is given by the function $h(c)$.

$$M = \{ (a, c, s) \mid a \in A \wedge c \in C \ \wedge \ s \geq thr_{sim} \ \wedge \ l_{hmin} \leq h(c) \leq \ l_{hmax} \ \} \qquad \textbf{(2)}$$

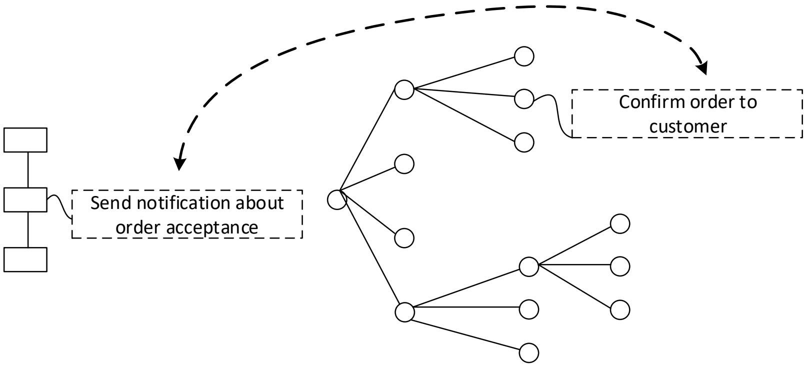

Confirm order to customer
Send notification about order acceptance

Fig. 1. Element annotation

3.2 Identification of Context

If the business topic such as e.g. "Human Resources" of a process model is known, then this knowledge can be leveraged to improve the annotation result. To do so, it could be used to discriminate between activity concepts with a comparable lexical matching value that are candidates for annotation. Hence it is required to detect the general topic of a model which we call *category* in the following. A category $d \in D$ (for "domain") can be interpreted as activity concepts that are sub-concepts of the taxonomy root, i.e. $D = \{d | d \in C \bigwedge (d, root) \in H\}$ with H being the set of hierarchy relations of the concepts in the taxonomy. In the simplest form, a category may be specified for the whole model by the user. If that is not possible, a category for the whole model may be derived in an automated way. However, unfortunately there may be models with multiple categories (i.e. multiple topics in one model such as HR and financial planning) and hence it is not clear which category is dominating the model. Such an example is illustrated by Fig. 2. In order to cope with the possibility of multi-category models, the model needs to be partitioned into fragments $f \in F$ containing subsets of activities $f \subseteq A$ referring to the same category (with a default-subset f_d for parts of the model that cannot be assigned to a category):

$$A = \cup_{i=1}^{N} f_i \cup f_d \qquad \text{where} \ \ f_i \cap f_j = \emptyset \, , i \neq j \qquad \textbf{(3)}$$

The function $d(f)$ returns the category $d \in D$ for a given fragment f and function $f(a)$ returns the corresponding fragment $f \in F$ for a given activity $a \in A$. Each fragment is associated to exactly one category, i.e. $\forall f \in F \, : \, \exists d \in D \ \wedge \ |d(f)| = 1$. Likewise, each activity should be contained in exactly one fragment, i.e. $\forall a \in A \, : \, \exists f \in F \ \wedge \ |f(a)| = 1$.

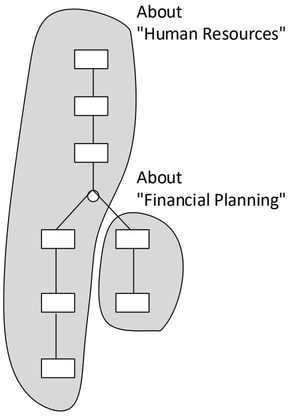


Fig. 2. Context in a process model

The task of partitioning the model into fragments that are associated to a category is dependent on two parameters. First, a minimum size s_{min} of a fragment has to be specified in order to prevent partitioning the model in activity fragments containing single activities and hence losing the usefulness of categories to discriminate between candidate activity concepts. Second, a lower threshold value for the minimum lexical matching value ls_{min} has to be set specifying the minimum average lexical similarity value between all activities contained in a fragment and sub-concepts in the category $d \in D$ that may be assigned to the fragment. Hence a splitting function $split()$ takes the process model and these two parameters as input and generates a set F of process fragments as output.

$$split(P, s_{min}, ls_{min}) = F \qquad (4)$$

In order to use categories associated to fragments to augment element annotation, the implementation of the similarity function $sim_{ac}(\mathrm{a}, \mathrm{c})$ has to be extended. Such an extended similarity function will give a higher similarity value if the category given by $d\big(f(a)\big)$, i.e. the category associated to the fragment an activity belongs to, matches the category of c. In order to detect the latter, a function $d'(c)$ which returns the category $d \in D$ for a given activity concept $c \in C$ is defined. In order to control the influence of category matches, i.e. if $d\big(f(a)\big) = d'(c)$, a weight w_{cat} is added to the refined similarity function $sim'_{ac}()$.

$$sim'_{ac}(a, c, w_{cat}) \in [0,1] \qquad (5)$$

3.3 Annotation Fitting

In order to increase the semantic quality of the annotation, annotations may be “fitted”. This in essence means to choose between activities that receive comparable matching values in the element annotation step according to notion of *betweeness* and differences in the hierarchy level. Fig. 3 illustrates this with a small example of three activities that are symbolized on the left side. The discrimination problem is considered for the middle activity. Comparably well matching activity concepts are depicted as grey shaded small circles on the right side. According to the notion of betweeness, the bottom most activity concept c₃ can be neglected since it is not in the area of preferred annotation candidates (surrounded by a dotted line). This is due to the fact that it is not between the already selected best matching activity concepts for the previous and following process activity which are illustrated as solid black filled circles. Further, according to the principle of preferring a similar hierarchical level, c₂ can also be skipped. Hence amongst similar lexical matches, c₁ is superior to c₂ and c₃ and hence c₁ is selected for annotation.

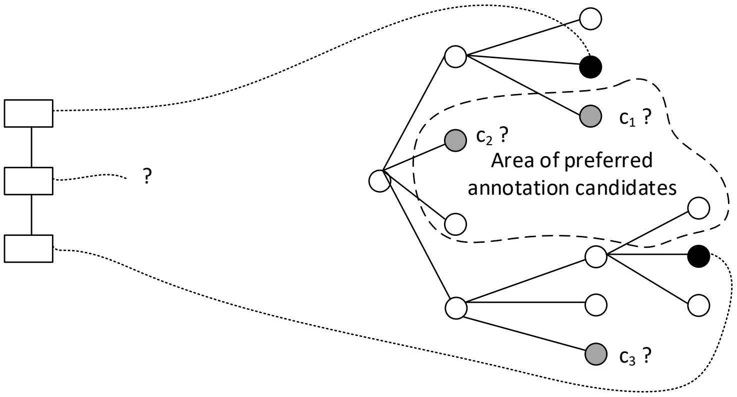


Fig. 3. Selection of possible activity concepts

The procedure introduced so far to select activity concepts for annotation roughly resemble to interpolation routines in common image manipulation software where the color of a pixel is calculated according to its neighbors (e.g. Gaussian Filter). We stick to that analogy and call this procedure of fitting *semantic interpolation*. Analogously, the radius of interpolation may be more than just one pixel which in our case is one preceding and one following activity. The radius may be extended to all preceding and following elements which in literature is also referred to as the *corona* of a process model element. In this way, the size of the corona may range from 1 to n, i.e. from all preceding and following activities reachable via one step or n steps in the process graph. Beyond the radius parameter for semantic interpolation, it is important that a fitting function is able to adjust the influence of the lexical matching in relation to the influence of hierarchy. Taking this into account, we define a function $fit()$ as follows:

$$fit(M, r, w_{lex}, w_{tax}) = M^{FIT} \quad \textbf{(6)}$$

where M are the annotation candidates (cf. formula 2), r is the radius used in semantic interpolation, w_{lex} the weight of the lexical matching result and w_{tax} is the weight of the hierarchy match. The latter refers to how the difference of the hierarchy levels of two activity concepts $c \in C$ given by $|h(c_i) - h(c_j)|$ with $i \neq j$ affects the semantic interpolation. The function produces a fitted annotation set $M^{FIT} \subseteq M$ with just one annotation per process activity, i.e. $|A| = |M^{FIT}|$.

3.4 Overview of Possible Configurations

The building blocks specified in in the previous sections may be combined leading to different configurations of the overall automatic annotation system. The following Table 1 describes these configurations. The first column provides a number for each configuration. The next three columns indicate if the building blocks (B1-B3) are used that are specified in the previous sections. The next column *Configuration Description* provides a short name (in bold) and description of this configuration variant. The next column *Configuration Parameters* provides a list of accumulated configuration parameters resulting from the different building blocks described in the previous sections.

Table 1. Building blocks and parameters

#	B1	B2	B3	Configuration Description	Configuration Parameters
1	✓			**Element matching** Element matching using lexical matching strategies provides a simple approach useful e.g. to provide a human user with suggestions for annotation.	– thr_{sim} similarity threshold – l_{hmin} min. hierarchy level of activity concepts – l_{hmax} max. hierar. level of activity concepts
2	✓	✓		**Element matching with category information** Element matching is augmented with category information so that annotation better reflects the business context of the process model. It hence reduces off-topic annotations.	Parameters of configuration variant 1 plus the following additional parameters: – s_{min} minimum size of a fragment – ls_{min} minimum average lexical similarity value – w_{cat} weight of category matches
3	✓		✓	**Element matching with semantic interpolation** Element matching is augmented with semantic interpolation so that the annotation better reflects the order and granularity of activities represented in the vocabulary. It hence provides a more “smooth” and standard-oriented annotation.	Parameters of configuration variant 1 plus the following additional parameters: – r radius of semantic interpolation – w_{lex} weight of lexical matching in interpol. – w_{tax} weight of taxon. matching in interpolation
4	✓	✓	✓	**Element matching with category information and sematic interpolation** This configuration combines #2 and #3 and hence provides the most comprehensive annotation approach that has the highest potential to imitate human annotation behavior.	Parameters of configuration variant 1-3 leading to an overall set of 9 parameters.

The configurations and parameters described in Table 1 may be used in the development, comparison and optimization of different implementation strategies and hence support a systematically evaluation of automated annotation approaches.

4 Preliminary Analysis and Insights

To gain first insights regarding the implementation of the building blocks introduced so far, a project was set up to create a simple algorithm. The goal of this algorithm is to match labels of process model activities to a similar or equal counterpart in a standardized framework. The standardized annotations are provided by the Process Classification Framework (PCF). It consists of twelve main hierarchies which are structured into sub-hierarchies of four levels. The hierarchies are in order of increased detail: *Process category*, *process group*, *process* and *activity*. The first five Process Categories contain operating process while the other describe management and support processes. The models were created by students in a non-related task. The labels were manually matched to the best corresponding concept in PCF thus creating a gold standard. This standard is the basis on testing the accuracy of the algorithm and offers valuable information by analyzing the characteristics of correct matches.

4.2 Simple Algorithm Structure

The algorithm is designed to apply a sequence of techniques that each individually affect the outcome. The workflow below shows each step the algorithm takes in order to find the best match for a label. The round-edged rectangles represent techniques. The sharp-edged rectangles indicate the resource data as well as the resulting outcome.

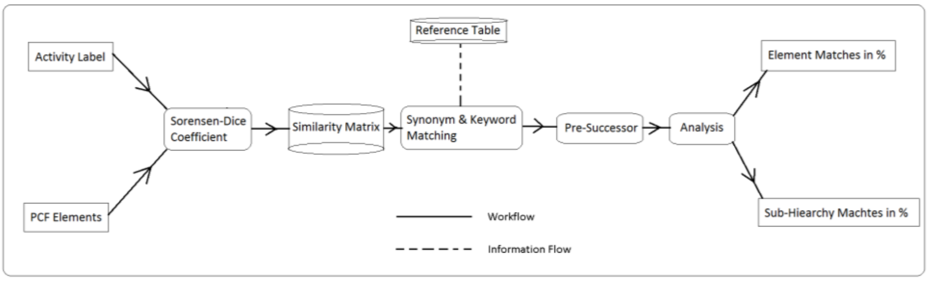


Fig. 4. Implemented procedure of annotation

In the following, we describe how the implementation reflect the three building blocks introduced in the previous sections.

Element annotation. The process starts with the resources holding the information schemas of in this case a model and the PCF. For convenience the labels of the individual model elements are called *activity labels* and the annotations in the PCF are called *PCF elements*. Each step aims at deriving information about the activity label on different levels. The first method compares solely the characters of two strings. The result is an editing distance showing lexical similarity. In this algorithm, the method used is called the Sorensen-Dice Coefficient. The result of this comparison between activity labels and PCF elements are cross products stored in a *similarity matrix*. This matrix is the core of the algorithm holding a similarity value for all possible pairs thus enabling an analysis for the best match. As stated in Section 3.1, the scores range between 0 and

1 with 1 being a 100% match. The following steps aim at modifying this value to single out the best match. We did not restrict the similarity threshold thr_{sim}, the min. hierarchy level l_{hmin} was set to 1 and the max hierarchy level $t_{hmax}$was set to unbounded.

Context detection. To identify the context of a model helps to narrow down the list of possible matches according to their domain. For this analysis the theory in Section 3.2 is taken and applied on the test case, although at the moment we identify the context of a process model only globally, i.e. not partitioning the model in fragments (parameters s_{min} and ls_{min} hence are irrelevant). The PCF is already classified into 12 domain specific hierarchies with a total of 4 layers. The hierarchy level *process group* (layer 2) thereby serves as the reference hierarchy. To match an activity label to the correct process group means in this case that the results are narrowed down to a list of max 60 PCF elements. The method uses information extracted by an external tool to derive certain words from the activity labels called *synonyms* as well as from the process groups called *keywords* which indicate domain affiliations. Keywords are words that show representative value for a process group in the PCF list. A word suits a keyword if it is present more than 3 times in a sub hierarchy (Fig. 9). This application roughly corresponds to building block 2 from Section 3.4. The matching process compares words by lexical matching. Semantic matching does not take place yet.

The same procedure takes place to create the synonyms list. The synonyms are generated by analyzing the activity labels for representative words. Since a gold standard is provided, a list of representative words on the activity label side towards process groups can be extracted (Fig. 10). The further calculation is the same as with the keywords. The synonyms list is however furthermore in need of constant updating for application on other models. The list created in this case is solely based on the gold standard and still requires verification for other models. Both lists are a preliminary result and are stored as a *reference table*. Matching activity labels to this list enables a context-related comparison and highlights matches on a more abstract level. Controlling the influence of the these matches corresponds to parameter w_{cat}.

Annotation fitting. The last step implies on a theoretical level (cf. Section 3.2) analyzing the predecessors and successors (i.e. the semantic context) of an activity label in the respective model. The content of models can be affiliated to certain process groups. The aim of this step is to punish and reward similarity scores by manipulating the overall score of a model in a process group. This behavior encourages scores of certain process groups to increase more in order to narrow down the highest similarity scores onto few process groups. For instance this step should increase the scores in process group 2.1 and 2.2 for model "1" above all other process groups to reward similarity scores in this sub-matrix (Fig. 8). This pattern of models showing an affinity to certain process groups was discovered by analyzing the gold standard but the algorithm further assumes that the results of the previous steps create the same pattern. The algorithm therefore manipulates the calculated scores. In terms of parameters, radius r is unbounded and weights w_{tax} and w_{lex} are implicitly set to 0.

After all methods are processed, an analysis is run to extract the best matches from the similarity matrix with the highest similarity score. Thus two scores are calculated showing the percentage of correct activity label to PCF element match and the percentage of correct activity label to process group match (for the sake of brevity, *process group* is also simply called *hierarchy* in Fig. 5-10).

4.3 Test and Results

The basic function of the algorithm is to create a matrix containing measures of similarity ranging between 0 and 1 across all possible matches and to manipulate these measures following each step. Each step can be adjusted on the amount of influence on the similarity measure. The testing phase was conducted in order to determine the effectiveness of each step at varying influence. The test set includes 430 annotations derived of 33 independent models. The base result for comparison consists solely on lexical matching. The results of the test is displayed in three graphs (Fig. 5, 6, 7) found in the appendix. In total synonym matching achieved the best results and has a big effect on matching a label to the correct PCF element. It manages to increase the percentage of correct matches from 12% to 18% for direct matches and from 25% to 57% for hierarchy matches, i.e. where at least the process group (level 2) is detected correctly. The other steps show little to no improvement.

The analysis and test are overall a first try at matching model labels. A test of the algorithm with combined influence of all steps is still missing and keyword and synonym matching step is heavily based on the characteristics of the gold standard. Furthermore steps like semantic matching that have not been implemented yet. The analysis could however confirm that semantic patterns such as the overall topic of a process model can be detected and that the PCF can in fact be leveraged as a valuable standard framework. The analysis moreover showed promising results concerning the creation and application of a synonym list.

Comparing our results to the method proposed by Leopold et al. [19], this method achieves about 76% correctness for annotating all activities with a concept from the correct main category and 44% of correctness for the process group level. In the light of these numbers (especially correctness on the process group level for which we achieve 57%), our approach seems promising. However, it requires a manually created synonym list. A detailed comparison is left open for future work.

5 Related Work

Most approaches developed so far only suggest manual annotation (cf. an overview [18]). For example, [4] describe a mapping relation of a BPEL4WS process to an OWL-S ontology as well as relations between concepts from the OWL-S profile ontology to domain ontologies; [5–7] develops an approach for adding properties of model elements or establishing relations to separate annotation models. A model for semantically annotating business process models is devised in [8]. There are however some works considering the annotation support by tools. For example, Bögl et al [9] describe a semantic linkage of Event-driven Process Chain (EPC) functions and events to ontology instances supported by a lexicon (WordNet), term extraction and stemming. Similarly, annotation approaches for BPMN models with ontologies have been developed [10–12] and partly supported using various lexical analysis techniques. Also, the annotation of process models with other domain specific ontologies such as the SCOR model for supply chain management has been explored [13] as well as annotations of process models with goal models [14].

However, the only approach that we are aware that considers context information (e.g. in the form of preceding or following annotations) when calculating an annotation suggestion is that of Leopold et al. [19]. The approach makes use of a Markov Logic-based formalization and considers automated annotation as an optimization problem. Further, in the field of execution-level (i.e. runtime) processes, the structure and lifecycle of involved objects in the process is considered [15–17].

6 Discussion and Conclusion

As of today, annotation of process models is rarely automated. Also, rarely prototypes are shown. Regarding the semantics of annotation, context information is (apart from [19]) almost never used [18]. This is a surprising research gap that exists even today – after almost one decade of research on semantic technologies applied to BPM that started with simple process model annotation proposals like [1]. Therefore, a research opportunity lies in developing (semi-)automated annotation approaches in order to first leverage existing standards such as PCF and second to make use of the wealth of semantic technologies (e.g. for search and matching of models on the semantic level) when process models could automatically be annotated. In this paper, we first have described the nature of the annotation task and how humans perform it. We then identify building blocks and parameters for automated systems that imitate human annotation behavior. We then conduct first empirical studies on the effect of parameters. It turned out that context information such as the topic of a process model is indeed very important for an automated annotation approach. All in all, this contribution aims to inspire more research on methods in (semi-)automatic approaches capable of linking semi-formal process models with more formal knowledge representations. With this, new use cases are possible as described in [18] shifting the automated interpretation of process models to a new and more semantic level.

Literature

1. Thomas, O., Fellmann, M.: Semantic Process Modeling - Design and Implementation of an Ontology-Based Representation of Business Processes. Bus. Inf. Syst. Eng. 1, 438–451 (2009).
2. APQC: Process Classification Framework (PCF), Version 5.2.0. (2010).
3. Malone, T.W., Crowston, K., Herman, G.A.: Organizing Business Knowledge: The MIT Process Handbook. The MIT Press (2003).
4. Aslam, M.A., Auer, S., Shen, J., Herrmann, M.: Expressing Business Process Models as OWL-S Ontologies. In: Eder, J. and Dustdar, S. (eds.) Business Process Management Workshops. pp. 400–415. Springer Berlin Heidelberg (2006).
5. Fill, H.-G.: Using Semantically Annotated Models for Supporting Business Process Benchmarking. In: Grabis, J. and Kirikova, M. (eds.) Perspectives in Business Informatics Research. pp. 29–43. Springer Berlin Heidelberg (2011).

6. Fill, H.-G., Schremser, D., Karagiannis, D.: A Generic Approach for the Semantic Annotation of Conceptual Models Using a Service-Oriented Architecture. Int. J. Knowl. Manag. 9, (2013).

7. Fill, H.-G.: On the Social Network Based Semantic Annotation of Conceptual Models. In: Buchmann, R., Kifor, C.V., and Yu, J. (eds.) Knowledge Science, Engineering and Management. pp. 138–149. Springer International Publishing (2014).

8. Mturi, E., Johannesson, P.: A context-based process semantic annotation model for a process model repository. Bus. Process Manag. J. 19, 404–430 (2013).

9. Bögl, A., Schrefl, M., Pomberger, G., Weber, N.: Semantic Annotation of EPC Models in Engineering Domains to Facilitate an Automated Identification of Common Modelling Practices. In: Filipe, J. and Cordeiro, J. (eds.) Enterprise Information Systems. pp. 155–171. Springer Berlin Heidelberg (2009).

10. Francescomarino, C.D., Tonella, P.: Supporting Ontology-Based Semantic Annotation of Business Processes with Automated Suggestions. In: Halpin, T., Krogstie, J., Nurcan, S., Proper, E., Schmidt, R., Soffer, P., and Ukor, R. (eds.) Enterprise, Business-Process and Information Systems Modeling. pp. 211–223. Springer Berlin Heidelberg (2009).

11. Di Francescomarino, C., Tonella, P.: Supporting Ontology-Based Semantic Annotation of Business Processes with Automated Suggestions: Int. J. Inf. Syst. Model. Des. 1, 59–84 (2010).

12. Rospocher, M., Francescomarino, C.D., Ghidini, C., Serafini, L., Tonella, P.: Collaborative Specification of Semantically Annotated Business Processes. In: Rinderle-Ma, S., Sadiq, S., and Leymann, F. (eds.) Business Process Management Workshops. pp. 305–317. Springer Berlin Heidelberg (2010).

13. Wang, X., Li, N., Cai, H., Xu, B.: An Ontological Approach for Semantic Annotation of Supply Chain Process Models. In: Meersman, R., Dillon, T., and Herrero, P. (eds.) On the Move to Meaningful Internet Systems: OTM 2010. pp. 540–554. Springer Berlin Heidelberg (2010).

14. Lin, Y.: Semantic Annotation for Process Models: Facilitating Process Knowledge Management via Semantic Interoperability. Department of Computer and Information Science Norwegian University of Science and Technology, Trondheim, Norway (2008).

15. Born, M., Dörr, F., Weber, I.: User-Friendly Semantic Annotation in Business Process Modeling. In: Weske, M., Hacid, M.-S., and Godart, C. (eds.) Web Information Systems Engineering – WISE 2007 Workshops. pp. 260–271. Springer Berlin Heidelberg (2007).

16. Born, M., Hoffmann, J., Kaczmarek, T., Kowalkiewicz, M., Markovic, I., Scicluna, J., Weber, I., Zhou, X.: Semantic Annotation and Composition of Business Processes with Maestro. In: Bechhofer, S., Hauswirth, M., Hoffmann, J., and Koubarakis, M. (eds.) The Semantic Web: Research and Applications. pp. 772–776. Springer Berlin Heidelberg (2008).

17. Born, M., Hoffmann, J., Kaczmarek, T., Kowalkiewicz, M., Markovic, I., Scicluna, J., Weber, I., Zhou, X.: Supporting Execution-Level Business Process Modeling with Semantic Technologies. In: Zhou, X., Yokota, H., Deng, K., and Liu, Q. (eds.) Database Systems for Advanced Applications. pp. 759–763. Springer Berlin Heidelberg (2009).

18. Fellmann, M.: Towards Automated Process Model Annotation with Activity Taxonomies: Use Cases and State of the Art. In: Abramowicz, W. (ed.) Business Information Systems. pp. 74–90. Springer, Cham (2017).

19. Leopold, H., Meilicke, C., Fellmann, M., Pittke, F., Stuckenschmidt, H., Mendling, J.: Towards the Automated Annotation of Process Models. In: Zdravkovic, J., Kirikova, M., and Johannesson, P. (eds.) Advanced Information Systems Engineering. pp. 401–416. Springer International Publishing (2015).

20. Fellmann, M., Thomas, O.: Process Model Verification with SemQuu. In: Nüttgens, M., Thomas, O., and Weber, B. (eds.) Enterprise Modelling and Information Systems Architectures (EMISA 2011), Hamburg, Germany. pp. 231–236. Köllen, Bonn (2011).

Appendix

Selected results from the implementation (Fig. 5-7). Annotation quality (precision, y-axis) is shown in relation to various parameter values (x-axis). *Direct Hit* means correct annotation in regard to the gold standard. *Hierarchy* means correct annotation at the process group level (level 2) of the Process Classification Framework.

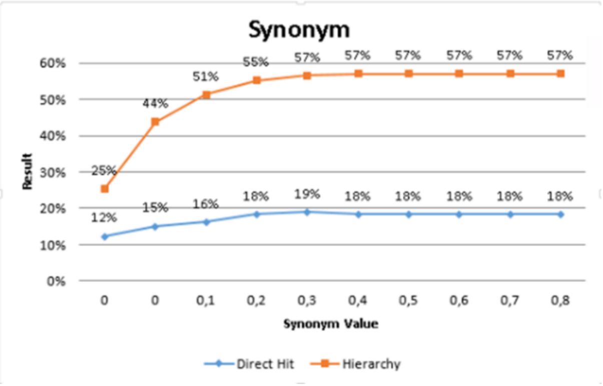


Fig. 5. Results for Synonym Matching

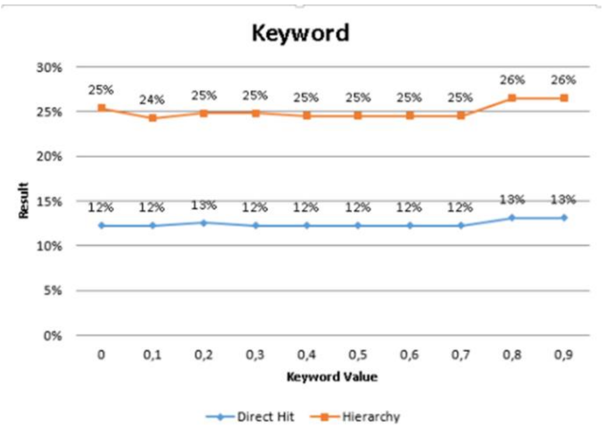


Fig. 6. Results for keyword Matching

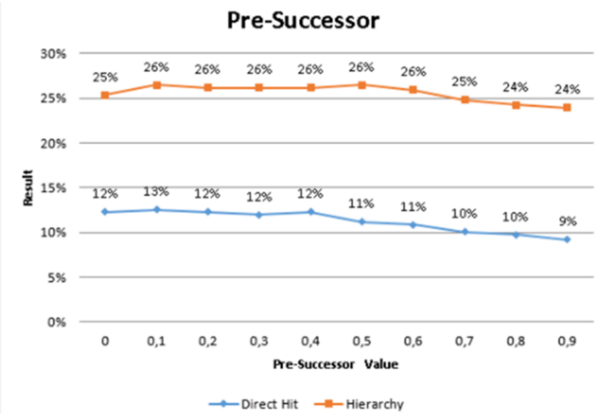
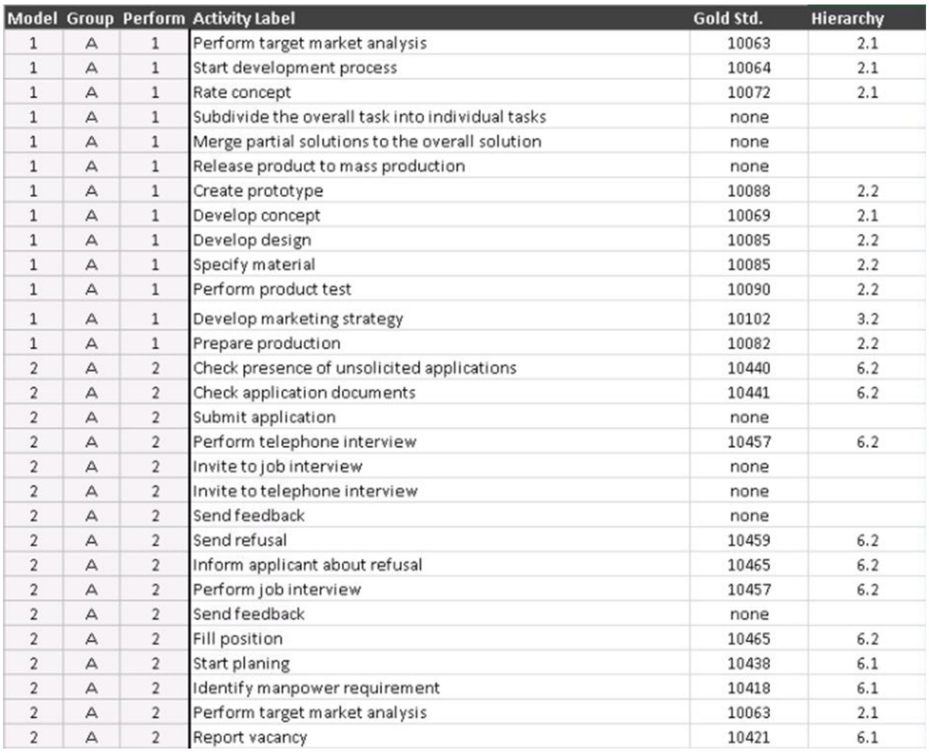
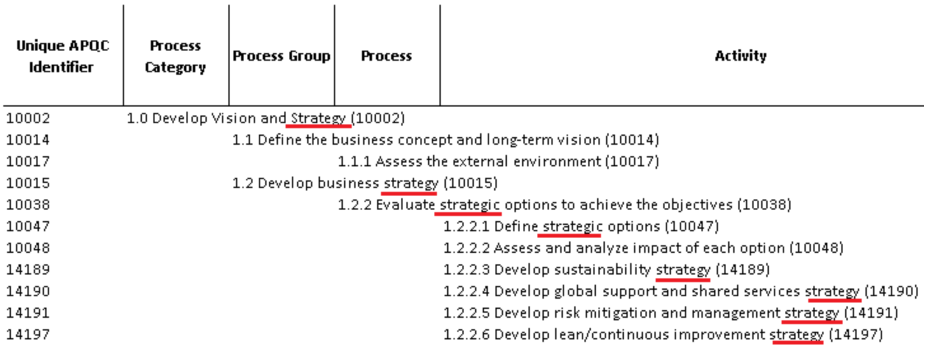


Fig. 7. Results for Pre-Successor



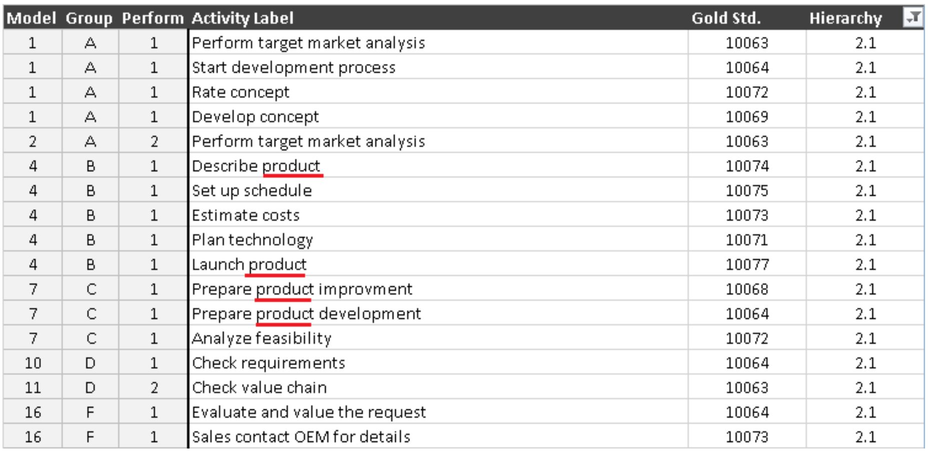
Model	Group	Perform	Activity Label	Gold Std.	Hierarchy
1	A	1	Perform target market analysis	10063	2.1
1	A	1	Start development process	10064	2.1
1	A	1	Rate concept	10072	2.1
1	A	1	Subdivide the overall task into individual tasks	none	
1	A	1	Merge partial solutions to the overall solution	none	
1	A	1	Release product to mass production	none	
1	A	1	Create prototype	10088	2.2
1	A	1	Develop concept	10069	2.1
1	A	1	Develop design	10085	2.2
1	A	1	Specify material	10085	2.2
1	A	1	Perform product test	10090	2.2
1	A	1	Develop marketing strategy	10102	3.2
1	A	1	Prepare production	10082	2.2
2	A	2	Check presence of unsolicited applications	10440	6.2
2	A	2	Check application documents	10441	6.2
2	A	2	Submit application	none	
2	A	2	Perform telephone interview	10457	6.2
2	A	2	Invite to job interview	none	
2	A	2	Invite to telephone interview	none	
2	A	2	Send feedback	none	
2	A	2	Send refusal	10459	6.2
2	A	2	Inform applicant about refusal	10465	6.2
2	A	2	Perform job interview	10457	6.2
2	A	2	Send feedback	none	
2	A	2	Fill position	10465	6.2
2	A	2	Start planing	10438	6.1
2	A	2	Identify manpower requirement	10418	6.1
2	A	2	Perform target market analysis	10063	2.1
2	A	2	Report vacancy	10421	6.1

Fig. 8. Demonstration of the affinity of process models of the gold standard to distinct categories on the process group level of the Process Classification Framework. The models are labeled by ascending numbers on the leftmost column. The gold standard is used to derive the label for the process group in the PCF (for the sake of brevity, this level is called “hierarchy” in the figures). Model 1 for instance shows strong affiliation to hierarchy 2.1 and 2.2.



Unique APQC Identifier	Process Category	Process Group	Process	Activity
10002	1.0 Develop Vision and Strategy (10002)			
10014		1.1 Define the business concept and long-term vision (10014)		
10017			1.1.1 Assess the external environment (10017)	
10015		1.2 Develop business strategy (10015)		
10038			1.2.2 Evaluate strategic options to achieve the objectives (10038)	
10047				1.2.2.1 Define strategic options (10047)
10048				1.2.2.2 Assess and analyze impact of each option (10048)
14189				1.2.2.3 Develop sustainability strategy (14189)
14190				1.2.2.4 Develop global support and shared services strategy (14190)
14191				1.2.2.5 Develop risk mitigation and management strategy (14191)
14197				1.2.2.6 Develop lean/continuous improvement strategy (14197)

Fig. 9. keyword example



Model	Group	Perform	Activity Label	Gold Std.	Hierarchy
1	A	1	Perform target market analysis	10063	2.1
1	A	1	Start development process	10064	2.1
1	A	1	Rate concept	10072	2.1
1	A	1	Develop concept	10069	2.1
2	A	2	Perform target market analysis	10063	2.1
4	B	1	Describe product	10074	2.1
4	B	1	Set up schedule	10075	2.1
4	B	1	Estimate costs	10073	2.1
4	B	1	Plan technology	10071	2.1
4	B	1	Launch product	10077	2.1
7	C	1	Prepare product improvment	10068	2.1
7	C	1	Prepare product development	10064	2.1
7	C	1	Analyze feasibility	10072	2.1
10	D	1	Check requirements	10064	2.1
11	D	2	Check value chain	10063	2.1
16	F	1	Evaluate and value the request	10064	2.1
16	F	1	Sales contact OEM for details	10073	2.1

Fig. 10. Synonym example