Michael Niemann, Melanie Siebenhaar, Julian Eckert, Ralf Steinmetz:
Process Model Analysis using Related Cluster Pairs. In: Hajo Reijers, Marcello La Rosa, Remco Dijkman: Proceedings of the 1st International Workshop on 'Process in the Large' (IW-PL 2010) (pre-proceedings) in connection with the 8th International Conference on Business Process Management (BPM 2010), Stevens Institute of Technology, Hoboken, NJ, USA, September 2010.

Process Model Analysis using Related Cluster Pairs

Michael Niemann, Melanie Siebenhaar, Julian Eckert, Ralf Steinmetz

KOM – Multimedia Communications Lab,
Department of Computer Science,
Technische Universität Darmstadt, Germany

Abstract. Due to changing market conditions and resulting flexibility requirements, the reference-conform implementation of processes in companies increasingly gains importance. The internal assessment of the realisation of reference processes (process conformance) is a resource-intensive task in terms of time and cost. The paper at hand presents a process model analysis method to address this issue using a combined structural and semantic comparison and analysis approach. The method provides decision support for process analysts concerning the adjustment of processes to reference processes in IT Governance contexts.

Key words: static process analysis, structural process analysis, semantic process matching. **Paper category:** Research in progress.

1 Introduction

In Business Process Management, reference models provide predefined solutions to a specific class of problems, e.g., the acceleration of business process implementation, or the harmonisation of internal processes. Process reference models, in particular, describe domain specific processes and generally provide an established and solid foundation for the analysis and improvement of internal processes. By describing dynamic aspects of an enterprise, e.g., activity sequences, organisational activities required to satisfy customer needs, control flow between activities, particular dependency constraints, etc., they help decreasing risk and provide beneficial clues for detection and improvement of weaknesses. [1][2]

IT Governance defines guidelines and reference processes in order to standardise processes in a company and IT departments, aiming at assuring conformance and simplification of control. Reference models in the field of IT Governance (also called best practice frameworks) are voluminous and have a large application scope (cp. [3], [4]). They consist of recommended general procedures, roles, responsibilities, and guidelines, combined with explicit process reference models [5]. Established frameworks, such as the IT Infrastructure Library (ITIL)[6], specify best practices as process or workflow models. As governance targets management processes rather than operative processes, processes and activities defined by IT Governance frameworks generally reside on a relatively high abstraction level.

Once introduced in companies, the adherence to reference models is diminished over time, e.g., by undocumented changes such as merging with new processes or process fragments, or natural human workflow evolution. In these cases, differences must be identified in retrospect, which mostly is a costly and time consuming procedure.

Reference models for governance purposes rather are *to-do-* or *check-lists* than control flow-oriented models and can be considered abstract models [4][3][6]. Comparing abstract processes is different: it is important to investigate *whether* (activity similarity), and *in what order* (activity permutation) composite, general activities are performed rather than *in what exact way*, e.g., the behaviour. When assessing processes with respect to such abstract process models, it is important for the process engineer to find general correspondences between process model parts. Even if possible, precise matchings are mostly not mandatory. Atomic activities as well as process behaviour are not of central importance in governance reference models – correspondence determination between process models and structure analysis become more important in this respect.

Commonly, process comparisons are performed by considering adequate notions of equivalence, e.g., bisimulation, trace or similar equivalences based on string-based, structural, and behavioural similarity metrics (cp. [7], [8]). For large models, those computations quickly become very complex. In particular, behavioural comparison approaches anticipate the comparability of process models, i.e., the existence of exact pairwise candidate assignments. Existing approaches for reference process analyses are often limited by the high computational complexity of the graph matching problem. Performing process comparison and analysis of process models deployed for governance purposes, i.e., the control and steering of IT systems, raises additional challenges that we address within this paper.

The paper at hand presents an analysis technique for process models identifying related activity groups in terms of structure and content (related cluster pairs). A *related cluster pair*, intuitively, consists of two groups of activities having one correspondent in the other process model, respectively. Generally, clusters abstract from the behaviour of the comprised activities (in terms of activity permutation and gateway conditions).

Using this technique, we are able to provide similarity values not only for entire processes, but also *cluster level* similarities. Additionally, by merging clusters, the technique allows the indication of the position of supplementary or missing activities (*location of differences*) and the indication of *activity order differences* (permutation). The currently realised approach computes similarities between activities of event-driven process chains (EPC) models (events and functions).

The remainder of this paper is structured as follows. In section 2, we introduce fundamental concepts used, in section 3 we explain the analysis approach in detail. After a comparison with related work in section 4, section 5 concludes the paper.

2 Basics

In this section, we introduce basic concepts and definitions such as event-driven process chains, similarity measures, and SESE regions.

2.1 Event-driven process chains

Event-driven process chains (EPC) are a method for the modelling of business processes, introduced within the scope of the Architecture of Integrated Information Systems (ARIS) [9]. The method of EPCs is widespread and its concepts can be easily transferred to other modeling approaches. An basic EPC can be defined as follows:

Definition 1 (Event-driven process chain). *An event-driven process chain represents a directed, connected graph G = (V,E). The set of vertices V consists of three disjoint sets of functions F, events E, and connectors C. The vertices are connected by arcs representing the control flow. Functions and events appear in an alternating sequence.*
Let I(v) and O(v) be the set of incoming and outgoing arcs for a given node $v \in V$, respectively. Then, $\exists e_S \in E$ with $|I(e_S)| = 0$ and $|O(e_S)| = 1$ denoted as start event e_S and $\exists e_E \in E$ with $|I(e_E)| = 1$ and $|O(e_E)| = 0$ denoted as end event e_E. $\forall f \in F$ and $e \in E \setminus \{e_S, e_E\}$: $|I(f)| = |I(e)| = |O(f)| = |O(e)| = 1$.
A connector $c \in C$ of type $t \in \{AND, OR, XOR\}$ represents a logical connection between functions and events. $\forall c_S \in C$ with $|I(c_S)| = 1$ and $|O(c_S)| \geq 1$, c_S is denoted as split connector and $\forall c_J \in C$ with $|I(c_J)| \geq 1$ and $|O(c_J)| = 1$, c_J is denoted as join connector. If one or more functions directly follow an event, the respective connector in between must be an AND connector. Finally, each $f \in F$ and each $e \in E$ is assigned a label.

2.2 Similarity measures

In order to determine similar function or event pairs, we apply two generally different similarity metrics: string-based and semantic similarity measures. Generally, three major classes of *string-based metrics* can be distinguished: edit-distance-based, token-based, and hybrid metrics [10]. *Edit-distance-based metrics* determine the minimal cost in terms of edit operations to transform a string S into a string T where edit operations are insertions, deletions, and substitutions of characters. *Token-based metrics* compare multi-word strings on token (i.e., word) level (instead of character level) and *hybrid metrics* combine character- and token-based methods. As representatives of token-based and hybrid metrics, the Jaccard (sim_{jac}) and the Monge Elkan metric (sim_{moe}), respectively, are defined as follows for the token sets A and B [10][11]:

$$sim_{jac}(A,B) = \frac{|A \cap B|}{|A \cup B|}; \qquad sim_{moe}(A,B) = \frac{1}{|A|}\sum_{i=1}^{|A|} \max_{j=1}^{|B|} sim(A_i, B_j)$$

The Monge Elkan metric maximises the similarity between the tokens of set A and all tokens of set B. The overall similarity equals the mean average of these maximum scores. As a *semantic similarity metric* distributional similarity is considered in our approach, allowing for the fact that different process designers may use different terms for the same activity. Two kinds of distributional similarity can be distinguished: *first order* and *second order* similarity. The former refers to words occurring in the same context, while the latter concerns words which occur in similar contexts. The corpus is tokenised and stopwords (frequent function words) are eliminated. The metric applies a context window size of ± 3 words. Moving the window over the corpus results in a set of dependency triples for a given word. A dependency triple is of the form (w, r, w'), where w represents the given word whose context is examined, w' is a word occurring in the context of w, and r refers to the relationship between w and w' (e.g., the relative position of w' with respect to w). To obtain the distributional first order similarity of two given words w_1 and w_2, a comparison of their dependency triples is performed using the following information theoretic measure suggested by Lin [12]:

$$sim_{Lin} = \frac{\sum_{(r,w')}(w_1, *_r, *_{w'}) + (w_2, *_r, *_{w'})}{\sum_{(r,w')}(w_1, *, *) + \sum_{(r,w')}(w_2, *, *)}$$

The measure is based on the assumption that the similarity between two words can be expressed as the amount of information contained within the dependency triples which are common to both words, divided by the amount of information contained in all the dependency triples of w_1 and w_2 that match the pattern $(w_1, *, *)$ and $(w_2, *, *)$, where $*$ is a wildcard for r and w', respectively.

2.3 SESE Regions

A Single-Entry Single-Exit (SESE) region, intuitively, represents an area within a graph that has a distinct entry edge and a distinct exit edge [13]. Inside nodes can only be reached from those outside by passing the entry edge and nodes outside can only be reached from inside by passing the exit edge.

Definition 2 (Canonical SESE Region). *For a given edge e, a canonical SESE region R (if it exists) is the smallest SESE region of which e is either the entry or the exit edge. Canonical SESE regions are either node disjoint or nested. [13]*

This definition emphasises that each edge e of a graph G does not necessarily have to be part of an enclosing edge pair of a SESE region. This is especially the case, if e resides inside a canonical region. Furthermore, canonical SESE regions represent a unique and node disjunctive decomposition of a graph-based process model. Additionally, SESE regions meet the condition of transitivity [13]. Given two SESE regions $S_1 = (a, b)$ and $S_2 = (b, c)$, their union also represents a SESE region S_3: $(a, b) \cup (b, c) = (a, c) = S_3$.

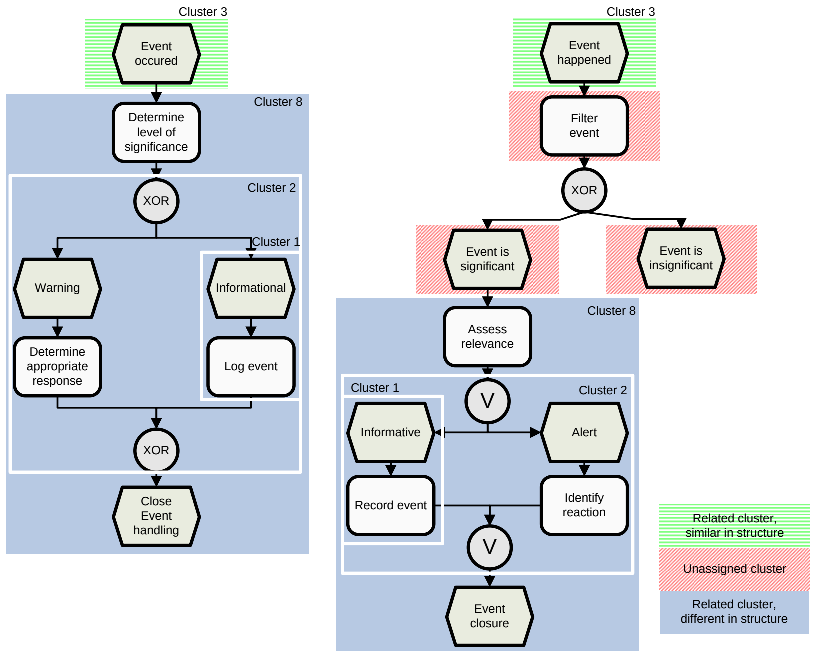

Fig. 1. Simplified Example from ITIL [6]

3 Related Cluster Analysis

The analysis technique presented in the following consists of two steps: *correspondences and cluster determination*, and *conditioned cluster merging*. Due to space limitations, we show a simplified example from ITIL [6] in Figure 1. The EPC model to the right shows a simplified excerpt from the reference process "Event Management Process" defined in the "Service Operation" book of ITIL. An *event*, in this context, "can be defined as any detectable or discernible occurence that has significance for the management of the IT Infrastructure or the delivery of IT services" [6]. The left model shows a simplified potential realisation of this process in an IT department.

3.1 Correspondences and cluster determination

Correspondences of process nodes, i.e., functions or events, are identified using a combined string-based and semantic similarity measure. For the nodes k_i of model 1 and m_j of model 2, the node similarity sim_{node} is computed as follows:

$$sim_{node}(k_i, m_j) = w_1 \cdot sim_{fos}(k_i, m_j) + w_2 \cdot sim_{moe}(k_i, m_j) + w_3 \cdot sim_{jac}(k_i, m_j)$$

We compute a similarity value per node pair, that consists of the weighted combination of two string-based metrics (sim_{moe} and sim_{jac}) and a semantic first order similarity measure (sim_{fos}, cf. Section 2.2). The weight for the syntactical metric part is 25% ($w_2 = w_3 = 0.125$), while we weight the semantic part with 75% (w_1). In the result shown in Figure 1, we see that, e.g., “filter event” and “log event” have not been assigned to each other, although they have a high syntactic similarity value (0.50). As well, “determine level of significance” and “assess relevance” have correctly been matched, although only 2 out of 6 words are synonyms. The utilised semantic metric identifies word similarities based on a Wikipedia corpus exceeding synonym relationship. The result is a list of relevant correspondences per node. For the exclusive identification of 1 : 1 - relationships, we solve the resulting assignment problem with one of the standard procedures [14][15].

In a further step, clusters and related cluster pairs are determined. A cluster C is defined as $C = (F, E)$, by the sets of functions F and events E it comprises. Intuitively, it is a SESE region with additional characteristics.

Definition 3 (Related Cluster Pair). *A related cluster pair is defined as a six-tuple* $(C^A, C^B, M^F, M^E, \mathrm{sim}^{\mathrm{R}}, t)$ *where* $C^A = (F^A, E^A)$ *and* $C^B = (F^B, E^B)$ *are clusters with function sets* F^A *and* F^B*, and event sets* E^A *and* E^B*.*
$M^F : F^A \rightarrow F^B$ *and* $M^E : E^A \rightarrow E^B$ *are isomorphisms such that* $\forall f \in F^A : \mathrm{sim}^{\mathrm{R}}(f, M^F(f)) \geq t$ *and* $\forall e \in E^A : \mathrm{sim}^{\mathrm{R}}(e, M^E(e)) \geq t$ *where* $\mathrm{sim}^{\mathrm{R}}$ *is a symmetric similarity function and* $t \in \mathbb{R}$ *with* $0 \leq t \leq 1$ *is a threshold.*

Note, that the set of vertices V of clusters in this case only refers to functions and events. Gateways as well as the ordering of activities are (explicitly) not considered here.

Each related node pair turns into a smallest possible related cluster pair (e.g., cluster 1). All unassigned nodes form unassigned clusters (e.g., node “filter event”, cp. Fig. 1). For all nodes of the left model, corresponding nodes have been assigned. In the right model, for the nodes “filter event”, “event is significant” and “event is insignificant” no correspondences were found.

3.2 Conditioned cluster merging

In the second step, adjacent clusters are merged in both models simultaneously – related cluster pairs as well as unassigned clusters. While the latter are just merged per model, the merging of the first ones is model-spanning and adheres to conditions. The first condition is to demand from an adjacent node B of node A in model 1, that its corresponding node B’ in model 2 is adjacent to A’, which is the correspondent to A. The second condition is, that the resulting node group must in turn be a cluster. This way, we aggregate sets of nodes to form larger related clusters.

This bottom up-process first merges nested SESE regions to form related clusters and then merges sequences of clusters, and is interrupted by stop conditions such as adjacent unassigned nodes. This way, the process models are transformed into sequences of largest possible related clusters.

During this computation, *cluster types* are determined and assigned to the related cluster pairs. In particular, we distinguish `BasicSEQ` (sequence), `XOR-SJ`, `OR-SJ`, `AND-SJ` (split-join, respectively), and `ITER` (loops). In Figure 1, the related cluster pair cluster 8 has the type `SEQ`, the inner cluster to the left is an `XOR-SJ`, while on the right, an `OR-SJ` has been identified.

We determine further cluster characteristics: `PERM` designates differences in node sequence in the cluster pair, `CHILDSTRUCTDIFF` and `PARENTSTRUCTDIFF` mark structural differences of related clusters referring to the resulting parent cluster or the child clusters, `NONBASIC` tags a related cluster pair that contains more complex than the above simple cluster types. In the example, characteristics of cluster 8 are `CHILDSTRUCDIFF` and `BasicSEQ`.

Related clusters are marked according to whether they are internally similar in structure or not (*cluster similarity level*). Generally, we distinguish two different cluster similarity levels: *content-related* cluster pairs contain corresponding elements with differing structure, while *structure-related* cluster pairs consist of clusters that are similar concerning their elements in terms of structure *and* content. In the example, the activities were aggregated to form the shown clusters, however, as outlined, the cluster types cluster 8 refers to are not similar. Those cluster pairs are marked as “content-related”, and *not* “structure-related”. In these cases, the final decision on whether the modeled activities actually *mean* the same must be left to human experts. Alternatively, formal behaviour investigations can be performed on clusters in order to identify further differences, or to compute change operations to map them.

The *related cluster pair similarity* refers to the similarity of two related clusters and represents the mean average of the similarity value of the node pairs contained in the respective clusters.

3.3 Discussion

Our approach identifies largest-possible related clusters and computes cluster types, structural internal cluster characteristics, as well as similarity levels (*content-related* vs. *structure-related*). Based on these pieces of information, a detailed report on the similarity of process models can be automatically generated, which are used for governance purposes, e.g., supporting process conformance checks. Generally, in our approach, we are able to consider two process models (parts) as similar, even if other similarity notions do not indicate a sufficient relation. Concerning process part similarity and process model similarity, the notion of related cluster pair similarity is different from existing similarity notions.

The computational complexity of solving the assignment problem is $O(n^3)$ [14]. This determines the complexity of the approach presented. The actual calculation of the correspondences matrix is cheaper: $O(nm)$, considering events and functions as input parameter of the first (n) and the second model (m).

In our approach, we combine label and semantic similarity. We weight the latter with 75% with respect to different labels actually describing the same activity. For semantic similarity we use a Wikipedia corpus-based approach. This way, we can identify word relations that exceed synonym-centred investigation.

We are able to find, e.g., the mapping of the word pair “hotel” and “accomodation”, not representing a synonym relationship.

So far, related work does not address the large-scope investigation of similar process regions. This approach represents an inexact investigation, marking regions similar, even if they differ in structure (e.g., gateways, conditions). However, this allows for a fast, yet effective investigation of large and complex processes, as it is often needed in application areas in IT Governance. Based on this analysis, detailed reports on process conformance can be computed, and change operations can be formulated, if required. Considering state-of-the-art governance reference processes as counterparts, detailed reports gain special importance, referring to expert conformance check reports.

4 Related Work

In this section, we discuss related work from the field of process analysis, comparable to the IT Governance context of the work at hand (cf. Tab. 1).

Andrews et al. present both a technique and prototype tool for visual graph comparison, which analyses similarities of given graphs and suggests a merged graph [16]. The resulting graph can be manually edited by the process engineer (e.g., replacing labels, changing node positions). The approach assumes the external provision of node similarities. Clearly, the emphasis lies on graphical graph layouting and presentation to the process engineer for final visual assessment.

Dijkman presents a technique to identify the differences between EPC process models. Besides the type of a difference, also the exact position of the differences can be determined [17]. For this, the difference typology presented in [18] is formalised. For the actual computation, the author makes use of formal semantics. Since the approach has exponential complexity, it requires repeated scoping of the process models. The approach processes EPCs with a small number of start events.

A further approach by Dijkman et al. [8] proposes the application of graph matching algorithms to the problem of ranking business process models in a given repository with respect to their similarity to a given process model. The four heuristics presented are based on the graph-edit-distance algorithm, which is NP-complete. To determine the similarity of the graph nodes, the node labels and their types are compared using string-edit-distance measures. The approach does not consider semantic similarity and does not indicate to the process engineer, where similarities and differences are located within the process models. In [19], the authors present several general basic approaches for process comparison, partly used in their later contributions.

Küster et al. introduce an approach for comparing different versions of one process model in the absence of a change log [20]. For the determination of differences, the authors make use of externally provided node correspondances, and SESE fragments. Differences and derived change operations are then grouped by associating them to the affected SESE fragments. Based on this, a hierarchical change log is composed, exploiting the nesting relationship of the SESE

Table 1. Overview of related work

Publication	**D**	**C**	**M**	**V**	**SB**	**IS**	**ES**	**BS**	**SR**	**SG**	**MR**
Andrews et al. [16]	×	–	×	×	–	–	–	–	–	–	×
Dijkman [17][18]	×	–	–	–	–	–	–	–	–	×	×
Dijkman et al. [8]	×	–	–	–	×	×	–	–	–	×	–
Küster et al. [20]	×	×	×	×	×	–	–	–	×	–	×
Melnik et al. [21]	–	–	–	–	×	–	–	–	–	–	–
Ehrig et al. [22][23]	–	–	–	–	×	×	×	–	–	–	–
Dijkman et al. [19]	×	–	–	–	×	×	–	×	–	×	–
Li et al. [24][25]	×	×	–	×	–	–	–	×	–	–	×
This approach	×	–	–	×	×	×	–	–	×	–	–

Abbrev.	*Meaning*	*Abbrev.*	*Meaning*
		SB	string-based similarity (labels)
D	differences determination	IS	implicit semantics (for label match.)
C	change suggestions	ES	explicit semantics (for label match.)
V	visualization	SR	structural similarity (SESE regions)
M	(semi-) autom. merging	SG	structural similarity (Graph-Edit Distance)
MR	manual assignment required	BS	behavioural similarity

fragments and their associated change operations. The change log can be used to resolve all or parts of the differences and to obtain a consolidated model. In general, this approach primarily considers different versions of the same process and does not account for models designed by different parties. Explicitly, a node correspondences matrix is required. Application across tool boundaries, i.e., an application area other than version comparison is not intended. In contrast, the approach at hand explicitly targets the analysis of general governance processes, modelled by different parties using different tools. It focuses on the identification of process regions of conformance and non-conformance, as well as on decision support for those regions where process conformance is initially unclear. In a governance context, the computation of change operations is not useful in every case – process conformance might be given, although the structure or the ordering of some activities might not be similar, respectively.

The graph matching algorithm presented by Melnik et al. in [21] performs a mapping between the corresponding nodes of two given graphs and can be applied to different scenarios with diverse data structures (e.g., matching of two data schemas in data warehousing applications). As pre-processing step, the two data structures to be compared are converted into directed labelled graphs. A similarity matrix constitutes the input for the next step, the so-called *similarity flooding*. This step represents an iterative fixpoint computation to determine the set of similar nodes. It is based on the assumption, that if two nodes are similar, their adjacent nodes are more likely to be similar, and thus, their similarity increases. For the determination of node similarities a simple string-based comparison is used. The computation results in a mapping between corresponding nodes. No differences are considered.

Ehrig et al. introduce a (semi-)automatic approach for the detection of similar process elements in business process models based on semantic information using ontologies [22]. To automatically compute similarities, the authors make use of a description of Petri net elements based on OWL-DL, the Pr/T net ontology, introduced in [23]. For their comparison, the authors apply text-based, implicit and explicit semantic similarity measures resulting in a combined similarity measure between concept instances. The similarity values of the concept instances are aggregated to an overall similarity of the two process models. Node similarities and process differences are not indicated.

Li et al. [24] develop an approach ("mining process variants") for identification of a generic process reference model for a given set of variants for integration into Process-Aware Information Systems (PAIS). They identify activities to be clustered as blocks based on an aggregated order matrix. The algorithm has a complexity of $O(n^3)$ and is validated using simulation on 7000 process models [25]. Referring to the blocks, they investigate the behaviour (ordering) of activities. As an activity assignment matrix is required as input, the central intention is different from the approach at hand.

5 Conclusion

In this paper, we presented an analysis technique for process models, computing similarities between activities as well as identifying related activity groups in terms of structure and content (related clusters pairs). A *related cluster* consists of a group of activities, all having one correspondent in the other process model, respectively. Generally, clusters abstract from the behaviour of the comprised activities. Using this technique, we are able to provide similarity values not only for entire processes, but also *cluster level similarities.* Additionally, by merging clusters, the technique determines the position of supplementary or missing activities (*location of differences*) and indicates *activity order differences.* During the computation, we identify largest-possible related clusters and compute cluster *types*, structural characteristics, such as identification of alternating sequences and complex cluster types, as well as cluster similarity levels (*content-related* vs. *structure-related*). Based on these information, detailed reports on the similarity of process models are generated. These are useful for governance purposes, e.g., supporting process conformance checks. The approach supports automated investigation of process models concerning the conformance to governance reference models.

The overall goal is to provide decision support for process owners on how to adjust processes in order to map reference processes in the fastest and cheapest possible way. The approach processes EPC models modeled by different parties using different tools (using the same data format) in $O(n^3)$ time. We realised our approach as a proof-of-concept prototype (ProMatch.KOM [26]). We are currently performing evaluations using 50 EPC models from the reference model "Handels-H" [27], currently showing 85% average accuracy and an F1-Measure of 92%. ProMatch.KOM has been implemented as plug-in for the process mining framework `ProM`[1].

As part of future work, we currently develop an IT Governance ontology for process annotation. This way, a more precise description and matching of

[1] `http://prom.win.tue.nl/tools/prom/`

processes and activities is possible, improving analysis quality. Further, we will address the analysis of complex EPCs by defining and computing a third level of cluster similarity, the “partly related cluster pair”. Clusters of this type combine nodes having correspondences with a minority of unassigned ones. In order to make our approach comparable in terms of evaluation results, we will also perform process model search on established test data sets.

Disclaimer

The project was funded by means of the German Federal Ministry of Economy and Technology under the promotional reference 01MQ07012. The authors take the responsibility for the contents.

This work is supported in part by the E-Finance Lab e.V., Frankfurt am Main, Germany (http://www.efinancelab.com).

References

1. J. Becker, M. Rosemann, and M. Kugeler, *Process Management.* Secaucus, NJ, USA: Springer-Verlag New York, Inc., 2003.
2. P. Fettke, P. Loos, and J. Zwicker, “Business Process Reference Models: Survey and Classification,” in *Business Process Management Workshops*, pp. 469–483, 2005.
3. IT Governance Institute (ITGI), *CobiT 4.1: Control Objectives for Information and Related Technology.* Rolling Meadows, IL: IT Governance Institute (ITGI), 2007. `http://www.itgi.org/cobit`.
4. K. Brand and H. Boonen, *IT Governance based on CObIT 4.1.* Van Haren Publishing, Zaltbommel, The Netherlands, 3. ed., 2007.
5. P. Weill and J. W. Ross, *IT Governance - How Top Performers Manage IT Decision Rights for Superiour Results.* Cambridge, MA: Harvard Business School Press, 2004.
6. Office of Governance Commerce (OGC), *ITIL v3: Information Technology Infrastructure Library Version 3*, vol. 1-5. 2007. `http://www.itil.org`.
7. A. K. Alves de Medeiros, W. M. P. van der Aalst, and A. J. M. M. Weijters, “Quantifying process equivalence based on observed behavior,” *Data Knowledge Engineering*, vol. 64, no. 1, pp. 55–74, 2008.
8. R. Dijkman, M. Dumas, and L. García-Bañuelos, “Graph Matching Algorithms for Business Process Model Similarity Search,” in *BPM ’09: Proceedings of the 7th International Conference on Business Process Management*, (Berlin, Heidelberg), pp. 48–63, 2009. Springer-Verlag.
9. A.-W. Scheer and K. Schneider, “ARIS - Architecture of Integrated Information Systems,” in *Handbook on Architectures of Information Systems* (P. Bernus, K. Mertins, and G. Schmidt, eds.), pp. 605–624, Springer Berlin Heidelberg, 2006.
10. W. W. Cohen, P. Ravikumar, and S. E. Fienberg, “A Comparison of String Distance Metrics for Name-Matching Tasks.,” in *Proceedings of IJCAI-03 Workshop on Information Integration*, pp. 73–78, August 2003.
11. A. Monge and C. Elkan, “The field matching problem: Algorithms and applications,” in *In Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, pp. 267–270, 1996.

12. D. Lin, "Automatic retrieval and clustering of similar words," in *Proceedings of the 17th international conference on Computational linguistics*, (Morristown, NJ, USA), pp. 768–774, Association for Computational Linguistics, 1998.
13. R. Johnson, D. Pearson, and K. Pingali, "The program structure tree: computing control regions in linear time," in *PLDI '94: Proceedings of the ACM SIGPLAN 1994 conference on Programming language design and implementation*, (New York, NY, USA), pp. 171–185, ACM, 1994.
14. D. Alevras, "Assignment and matching," in *Encyclopedia of Optimization*, pp. 106–108, Springer-Verlag, 2009.
15. G. A. Mills-Tettey, A. T. Stentz, and M. B. Dias, "The dynamic hungarian algorithm for the assignment problem with changing costs," Tech. Rep. CMU-RI-TR-07-27, Robotics Institute, Pittsburgh, PA, July 2007.
16. K. Andrews, M. Wohlfahrt, and G. Wurzinger, "Visual Graph Comparison," in *IV '09: Proceedings of the 2009 13th International Conference Information Visualisation*, (Washington, DC, USA), pp. 62–67, 2009. IEEE Computer Society.
17. R. Dijkman, "Diagnosing differences between business process models," in *BPM '08: Proceedings of the 6th International Conference on Business Process Management*, (Berlin, Heidelberg), pp. 261–277, 2008. Springer-Verlag.
18. R. Dijkman, "A Classification of Differences between Similar Business Processes," *Enterprise Distributed Object Computing Conference, IEEE International*, 2007.
19. R. M. Dijkman, M. Dumas, B. F. van Dongen, R. Käärik, and J. Mendling, "Similarity of business process models: Metrics and evaluation," tech. rep., BETA Research School, Eindhoven, The Netherlands, 2009.
20. J. M. Küster, C. Gerth, A. Förster, and G. Engels, "Detecting and resolving process model differences in the absence of a change log," in *BPM '08: Proceedings of the 6th International Conference on Business Process Management*, (Berlin, Heidelberg), pp. 244–260, 2008. Springer-Verlag.
21. S. Melnik, H. Garcia-Molina, and E. Rahm, "Similarity flooding: A versatile graph matching algorithm and its application to schema matching," in *ICDE '02: Proceedings of the 18th International Conference on Data Engineering*, (Washington, DC, USA), 2002. IEEE Computer Society.
22. M. Ehrig, A. Koschmider, and A. Oberweis, "Measuring similarity between semantic business process models," in *Proceedings of the Fourth APCCM 2007* (J. F. Roddick and A. Hinze, eds.), vol. 67, pp. 71–80, Australian Computer Science Communications, January 2007.
23. A. Koschmider and A. Oberweis, "Ontology based business process description," in *Proceedings of the CAiSE'05 WORKSHOPS* (J. C. E. Teniente, ed.), no. 2, (Porto/Portugal), pp. 321–333, June 2005.
24. C. Li, M. Reichert, and A. Wombacher, "Discovering reference process models by mining process variants," in *ICWS '08: Proceedings of the 2008 IEEE International Conference on Web Services*, (Washington, DC, USA), pp. 45–53, IEEE Computer Society, 2008.
25. C. Li, M. Reichert, and A. Wombacher, "Discovering Reference Models by Mining Process Variants Using a Heuristic Approach," *Lecture Notes in Computer Science*, vol. 5701, pp. 344–+, 2009.
26. M. Siebenhaar, M. Niemann, J. Eckert, and R. Steinmetz, "ProMatch.KOM: Tool Support for Process Model Analysis and Improvement," in *Demonstration Track of the 8th International Conference on Business Process Management (BPM 2010), Hoboken, New Jersey, USA (to appear)*, September 2010.
27. "Cubetto Toolset HTML Export - Referenzmodell Handel." `http://www.semture.de/downloads/cubetto/handels-h/index.html`. last accessed on: 2009-11-24.