Mining Resource Assignments and Teamwork Compositions from Process Logs

Stefan Schönig[1], Cristina Cabanillas[2], Claudio Di Ciccio[2], Stefan Jablonski[1], Jan Mendling[2]

[1]University of Bayreuth, Germany
{stefan.schoenig, stefan.jablonski}@uni-bayreuth.de

[2]Vienna University of Economics and Business, Austria
{cristina.cabanillas, claudio.di.ciccio, jan.mendling}@wu.ac.at

Abstract. Process mining aims at discovering processes by extracting knowledge from event logs. Such knowledge may refer to different business process perspectives. The organisational perspective deals, among other things, with the assignment of human resources to process activities. Information about the resources that are involved in process activities can be mined from event logs in order to discover resource assignment conditions. This is valuable for process analysis and redesign. Prior process mining approaches in this context present one of the following issues: (i) they are limited to discovering a restricted set of resource assignment conditions; (ii) they are not fully efficient; (iii) the discovered process models are difficult to read due to the high number of assignment conditions included; or (iv) they are limited by the assumption that only one resource is responsible for each process activity and hence, collaborative activities are disregarded. To overcome these issues, we present an integrated process mining framework that provides extensive support for the discovery of resource assignment and teamwork patterns.

1 Introduction

Business process management is a well accepted method for structuring the activities carried out in an organisation, analysing them for efficiency and effectiveness, and identifying potential for improvement [6]. Processes are not always explicitly defined, which calls for a means to discover the implicit rules according to which the processes are executed. Process mining provides different methods, among others, for automatically discovering processes by extracting knowledge from event logs in the form of a process model. Various algorithms are available to discover models capturing the control flow of a process [1]. For other perspectives, specifically involving human resources[1], only partial solutions for mining have been developed, even though resource information is not only important for performance but also for compliance analysis [3, 4]. There is a need to better support this organisational perspective by mining resource-related process aspects. This is acknowledged by different approaches that mine this perspective [10, 14]. Works in this area, however, focus on specific aspects of the organisational perspective such as role models, separation of duty or social networks. None of the prior works offers a comprehensive and integrated support for the well-established workflow resource patterns [11] and insights into the interplay between organisational and control-flow aspects [8], also known as cross-perspective patterns. In [13] we addressed this research gap by developing an integrated declarative process mining approach for the organisational perspective. It supports *(i)* all creation aspects of the workflow resource patterns and *(ii)* cross-organisational patterns. Figure 1 illustrates its subdivision into an event log pre-processing phase, a phase for integratedly mining resource assignments as well as cross-perspective patterns, and a model post-processing phase.

The mining of resources is usually restricted by the assumption that individual activities are performed by exactly one person. However, in domains like healthcare, software development, and knowledge-intensive processes in general, most of the activities are carried out collaboratively, such that several human resources are involved with working on a single activity. Domains in which collaborative work is frequent can greatly benefit from mining team compositions, which unveil the capabilities and organisational relations of the team members [5]. Such approaches for mining collaborative work in business processes are currently missing. In [12], we addressed this research gap by extending the proposed declarative process mining framework [13] towards the integration of collaborative activities. The approach comprises two steps: it first extracts the teams participating in a collaborative activity from an event log and then discovers the overall characteristics of the team members in terms of the skills, organisational roles, etc., that are present in the team; afterwards, a two-step post-processing phase derives the most informative team compositions including the distribution of the discovered characteristics among the team members.

The approches for mining individual resource assignments and teamwork have been evaluated in terms of *(i)* feasibility, by means of a proof-of-concept implementation; *(ii)* performance, by conducting simulation experiments; and *(iii)* applicability, by using the approaches on real-life event logs.

This paper is structured as follows: Section 2 introduces the target expressiveness of our mining framework as well as the data sources we use. Section 3 describes the target language we use to represent the

[1]We use the term "resource" to refer to "human resources".

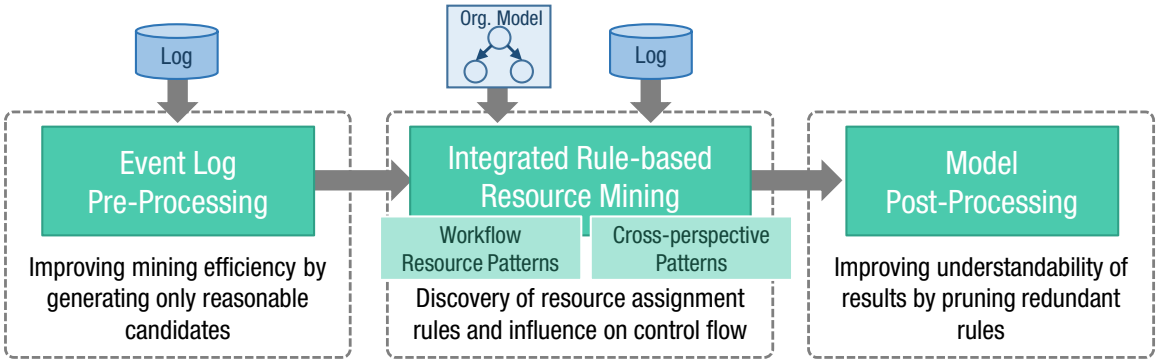

Figure 1: Efficient framework for discovering resource assignment rules for individual activities [13]

mining results. In Section 4 we describe our approach for discovering resource assignment patterns. Section 5 gives an overview of the teamwork mining approach and the paper is finally concluded in Section 6.

2 Fundamentals

The well-known workflow resource patterns [11] capture the various ways in which resources are represented and utilised in business processes. The creation patterns are of specific interest to our research since they describe different ways in which resources can be assigned to activities. These patterns, which will be referred to as *organisational patterns* from now on, include: *Direct Distribution*, or the ability to specify at design time the identity of the resource that will execute a task; *Role-Based Distribution*, or the ability to specify at design time that a task can only be executed by resources that have a given role; *Organisational Distribution*, or the ability to offer or allocate activity instances to resources based on their organisational position and their organisational relationship with other resources; *Separation of Duties*, or the ability to specify that two tasks must be allocated to different resources in a given process instance; *Case Handling*, or the ability to allocate all the activity instances within a given process instance to the same resource; *Retain Familiar* (a.k.a. *Binding of Duties*), or the ability to allocate an activity instance within a given process instance to the same resource that performed a preceding activity instance; *Capability-Based Distribution*, or the ability to offer or allocate instances of an activity to resources based on their specific capabilities; *Deferred Distribution*, or the ability to defer the specification of the identity of the resource that will execute a task until run time; and *History-Based Distribution*, or the ability to offer or allocate activity instances to resources based on their execution history; Note that the creation patterns *Authorisation* and *Automatic Execution* are not in the list because they are not directly related to resource assignment;

It has been identified that process control flows are intertwined with dependencies upon resource characteristics [8]. For instance, sometimes an activity must be executed eventually before another one for specific resources but not for others. As an example, resources with a certain role (e.g., trainees) must always perform a certain activity (e.g., double-check result) before they can continue. This might not be required for other roles (e.g., supervisors). We call this pattern *Role-Based Sequence*. A specific collection of such *cross-perspective patterns* capturing these situations has not been defined. Nonetheless, in general, they can be specified by combining the aforementioned organisational patterns with the control-flow patterns described in [2]. The *Resource-Based Response* pattern, e.g., describes that for a specific resource a certain activity has to eventually follow another activity. The organisational and the cross-perspective patterns constitute the set of patterns to be discovered by our framework.

Our mining approaches take as input *(i) an event log*, i.e., a machine-recorded file that reports on the execution of tasks during the enactment of the instances of a given process; and *(ii) organisational background knowledge*, i.e., prior knowledge about the roles, capabilities and the membership of resources to organisational units, among others.

3 Target Language

The modelling language to represent the discovered patterns must offer the possibility to define *(i)* expressive organisational patterns and *(ii)* cross-perspective patterns. Two different representational paradigms for process models can be distinguished: procedural models describe which activities can be executed next in a process; declarative models define by means of rules the execution constraints that the process has to satisfy [2]. Current procedural languages like BPMN put a strong emphasis on control flow and assume other perspectives to be specified separately. Cross-perspective patterns cannot be readily modelled [7]. Declarative process modelling does not limit the number of perspectives involved in the constraints defined.

However, a central shortcoming of existing languages like Declare [2] is that they are not provided with the capability to directly define the connection between the process behaviour and other perspectives. We will use DPIL [15] for modelling the output of the mining because it supports multiple perspectives including the behavioural and the organisational perspectives, as well as the interplay between them. DPIL is expressive enough to cover the workflow patterns [15]. Nonetheless, the concepts of our approach are generic so that other declarative languages could also be used as long as they provided support for the modelling of our target patterns.

In order to express organisational information, DPIL builds upon a generic organisational meta model. It comprises the following elements: *Identity* represents agents that can be directly assigned to activities, i.e., both human and non-human resources. *Group* represents abstract agents that may describe several identities as a whole, e.g., roles or groups. *Relation* represents relations (*RelationType*) that may exist between these elements. Thus, relations like "a person is the boss of another person" or "a person belongs to a certain department" can easily be modelled. In this context, relations are generally irreflexive. A relation is irreflexive if an element cannot be in relation to itself. The *supervisor* relation, e.g., is irreflexive, since a person cannot be their own supervisor. In addition, some relations may be transitive. A relation is transitive if whenever an individual i_1 is related to another individual i_2 with that relation, and i_2 is in turn related to a third individual i_3 with the same relation, then i_1 is also related to i_3. For instance, the *supervisor* and *delegate* relations are typically transitive because organisations are usually hierarchically structured.

DPIL provides a textual notation based on the use of *macros* to define reusable rules. For instance, the $sequence(a, b)$ macro states that the existence of a *start event* of task b implies the previous occurrence of a *complete event* of task a; and the $role(a, r)$ macro states that an activity a is assigned to a role r. Figure 2 shows an example of a process for trip management modelled with DPIL. It specifies that it is mandatory to approve a business trip before flight tickets can be booked. Moreover, it is necessary that the approval be carried out by a resource with the role *Professor*.

4 Analysing Resource Assignments

In this section we describe our approach to discover organisational and cross-perspective patterns. First, we describe how rule candidates are generated and checked. Then, we classify them according to support and confidence values. Finally, we present a catalogue of rule templates that covers the target expressiveness.

```
use group Professor

process BusinessTrip {
   task Book flight
   task Approve Application

   ensure role(Approve Application, Professor)
   ensure sequence(Approve Application, Book flight)
}
```

Figure 2: Process for trip management modelled with DPIL

4.1 Resource Assignment Rule Discovery

Declarative process modelling languages like DPIL are based on so-called *rule templates*. A rule template captures frequently needed relations and defines a particular type of rules. Unlike concrete rules, a rule template consists of placeholders, i.e., typed variables. A rule template is instantiated by providing concrete values for these placeholders. For instance, the model described in Fig. 2 makes use of two rule templates represented by the macros *sequence(T_1,T_2)* and *role(T,G)*. These templates comprise placeholders of type *Task* T as well as *Group* G. In all well-known declarative process mining approaches, rule templates are used for querying the provided event log to find solutions for the placeholders. A solution is any combination of concrete values for the placeholders that yields an actual rule satisfied in the event log. First, all possible rules need to be constructed by instantiating the given set of rule templates with all possible combinations of occurring process elements provided in the event log. For example, the sequence template consists of two placeholders of type $Task$. Assuming that $|T|$ different tasks occur in the event log, $|T|^2$ *rule candidates* are generated. The resulting candidates are subsequently checked w.r.t. the log. Checking rule candidates as described above provides for every candidate the number of instances, i.e., the traces in the event log where it holds. Based on these values it is possible to classify rules and to separate non-valid from valid ones.

Maggi et al. [9] adopted different metrics, specifically support (*supp*) and confidence (*conf*) for evaluating the relevance of rule candidates. Let $|\Phi|$ be number of traces in an event log Φ. Let $|\sigma_{nv}(r)|$ be the number of traces in which a rule $r : A \rightarrow B$ is satisfied. The support $supp(r)$ and confidence $conf(r)$ values of a rule r are defined as:

$$supp(r) := \frac{|\sigma_{nv}(r)|}{|\Phi|}, \quad conf(r) := \frac{supp(r)}{supp(A)}$$

We make use of the confidence value to classify a rule candidate r as a *valid* rule (i.e., satisfied in *almost all* traces) or a *non-valid* rule (i.e., violated in *most of* the recorded traces). Therefore, the threshold *minConf* is introduced to classify rule candidates. Candidates r with $conf(r) \geq minConf$ are classified as valid. All rule candidates r with

$conf(r) < minConf$ are non-valid rules and are not part of the resulting process model. Using the confidence values of rule candidates it is directly possible to generate a DPIL process model reflecting organisational and cross-perspecitve patterns.

4.2 Templates of Organisational Patterns

Since DPIL builds upon a flexible organisational meta model, it is possible to define rule templates that describe many aspects of the organisation. By instantiating these rule templates with all possible parameter combinations of defined resources, groups and relation types, it is possible to generate rule candidates that focus on the organisational perspective of the process to be analysed. These candidates can then be checked under consideration of the event log and the organisational model.

In the following we define rule templates and their macros for our target set of patterns. First of all, we distinguish between templates for organisational patterns and templates for cross-perspective patterns. The former are, in turn, divided into two groups based on the types and number of parameters: rule templates related to a single task and rule templates related to more than one task. We provide representative examples for each group of rule templates that cover frequently needed organisational information. Note that, besides the templates described next, further templates could be defined individually to cover the analyst's needs.

The first group includes rule templates that define organisational patterns referred to one process activity. The *Direct Distribution* pattern can be extracted with the *direct(T,I)* template. Given the free variables T and I and an event log with $|T|$ distinct tasks and $|I|$ distinct resources, there are $|T| \cdot |I|$ candidates to be checked.

```
direct(T,I) iff start(of T) implies start(of T by I)
```

The *Role-Based distribution* pattern can be extracted with the *role(T,G)* template. Here, rule candidates for every task and group combination are generated, i.e., $|T| \cdot |G|$ rule candidates need to be checked.

```
role(T,G) iff start(of T by :p) implies
relation(subject p predicate hasRole object G)
```

The *Capability-Based distribution* pattern can be extracted with the *capability(T,RT,G)* template. A capability is represented by a relation of an individual to a group, e.g., i_1 *hasDegree ComputerScience*. According to the placeholders, $|T| \cdot |RT| \cdot |G|$ candidates are generated.

```
capability(T, RT, G) iff start(of T by :p) implies
relation(subject p predicate RT object G)
```

The assignment of resources based on organisational positions of individuals, described by the *Organisation-Based Distribution* pattern, can be extracted with the *orgDistSingle(T,RT,G)* template. Here, $|T| \cdot |RT| \cdot |G|$ rules must be checked.

```
orgDistSingle(T, RT, G) iff start(of T by :p) implies
relation(subject p predicate RT object G)
```

The second group includes rule templates that define organisational patterns referred to several tasks. The *Separation of Duties* pattern can be extracted with the *separate(T_1,T_2)* template. For this template, $|T|^2$ candidates need to be checked.

```
separate(T1,T2) iff start(of T1 by :p) and start(of T2)
implies start(of T2 by not p)
```

The *Retain Familiar* pattern can be extracted with the *binding(T_1,T_2)* template. Similarly to the previous case, $|T|^2$ candidates need to be checked.

```
binding(T1,T2) iff start(of T1 by :p) and start(of T2)
implies start(of T2 by p)
```

The *Case Handling* pattern can be extracted with the *caseHandling* template. Here, $|T|$ candidates have to be checked.

```
caseHandling iff forall(task T start(of T) implies
start(of T by :p))
```

Resources can also be assigned to tasks according to their organisational relation with the performers of other process activities, e.g., an approval task might be assigned to people that can supervise the work done by the performers of a previous task. This is covered by the *Organisation-Based Distribution* pattern and can be extracted with the *orgDistMulti(T_1,T_2,RT)* template where variable RT specifies the type of relation between the two individuals involved. There exist $|T|^2 \cdot |RT|$ rule candidates.

```
orgDistMulti(T1,T2,RT) iff start(of T1 by :p1) and
start(of T2 by :p2) implies
relation(subject p1 predicate RT object p2)
```

A cross-perspective rule describes a temporal dependency or constraint between tasks but only applies for a certain set of identities, like in the following examples. Note, that other well-known control-flow patterns described in [2] can be defined in a similar way. The *Role-Based Sequence* pattern can be extracted with the *roleSequence(T_1,T_2,G)* template. Here, $|T|^2 \cdot |G|$ candidates need to be checked.

```
roleSequence(T1,T2,G) iff start(of T2 by :p at :t) and
relation(subject p predicate hasRole object G)
implies complete(of T1 at < t)
```

5 Mining Teamwork Patterns

In this section, we describe our approach to automatically discover team compositions from event logs. For preparing and representing the mining output, we rely again on the DPIL mining framework and the organisational rule templates. An example is depicted in Fig. 3. From the first mining step three different teams are extracted from an event log for a collaborative activity A. After analysing the teams against the existing background knowledge from the organisational model, the team characteristics *role Doctor*, *role Nurse* and *capability Blood Test* are discovered for these teams. Finally, the post-processing phase con-

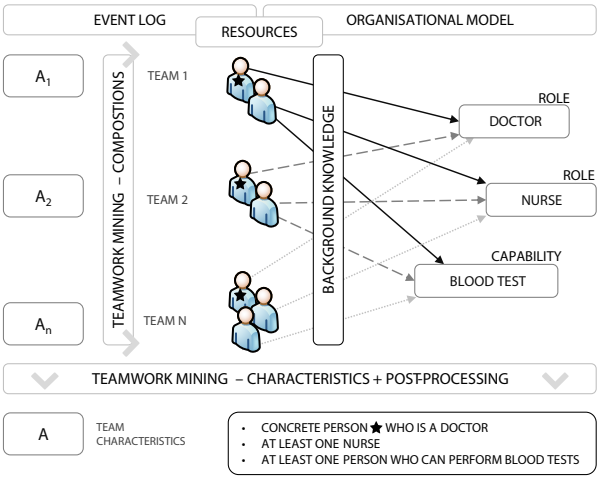

Figure 3: Approach for mining team compositions [12]

cludes that the teams performing that activity must always include a specific person $\star$ who, in turn, is a doctor; at least one nurse; and at least one person that can perform blood tests.

5.1 Extraction of Discriminative Teams and their Characteristics

The first step is to extract the different sets of participants that are needed to perform the different process instances. For every trace we extract the set of distinct individuals that are associated with at least one of its events. The result of the scan of an event log with n different traces is a set of distinct teams.

The set of frequent teams already provides valuable insights into the operation of a business process. However, it does not help managers to learn how they can compose teams in a potentially more effective way. Therefore, the next step of our approach focuses on the mining of frequent team characteristics. To this extent, we make again use of the DPIL mining framework and its concepts of rule templates with certain placeholders. In contrast to our previous approach, which mines for resource assignment rules for each single task, here we abstract from the tasks and search for assignment rules that refer to a complete sub-process. Hence, the parameter referring to a task in the rule templates can be omitted. This way, the *direct* template, e.g., results in the following definition:

```
direct(I) iff event() implies event(by I)
```

The approach is visualized in Fig. 4. It illustrates the complete mining procedure by example direct and role rule candidates and different team compositions. For example, this means that a *direct*(i_1) rule holds true for a certain process if individual i_1 performs at least one task in every instance of the collaborative activity. Therefore, mining teamwork aspects reduces the number of candidates for the *direct* template to n_I rules to be evaluated. Evaluating rule candidates as described above yields for every candidate the number of traces in the log where it holds. In order to judge the relevance of the rules, we adopt similarly to Section 4 the support and confidence thresholds concepts to evaluate the relevance of rule candidates.

5.2 Post-Processing

Extracting the characteristics of the team members is not sufficient to describe how the team is actually composed. In order to compose a new team it is necessary to know how the characteristics are distributed among the team members. For instance, the results so far only say that at least *someone* in every team has the extracted characteristics. However, it is not clear how many persons in a team have a certain characteristic and which of these characteristics are maybe fulfilled by one and the same person, i.e., *overlapping* characteristics. In the following, we briefly describe two post-processing steps to define more precisely the way in which a team is composed for a collaborative activity. Each of these steps introduces precision but also computational complexity.

In a first step we move from analysing at team level to analysing at resource level. Specifically, we count for each characteristic among those extracted for the teams the number of team members that fulfil it. The outcome is the minimum number of persons that fulfil each characteristic, i.e., a rule within a team. The result is more informative than the initial one because it adds cardinality to the extracted characteristics. However, it is done only at an individual level, i.e., for each single characteristic. Since one single person may have several of the characteristics discovered, in a last step we consider all the possible combinations of characteristics and we check each of them for every member of the team. The outcome is the set of overlapping rule sets with the number of resources that must have each set of characteristics within the team. This offers a more detailed view of the team composition.

6 Conclusion and Outlook

In this paper we presented an integrated process mining framework to efficiently discover resource assignment patterns as well as team attributes and composition patterns of collaborative activities in business processes. The framework builds upon a declarative process mining technique focusing on the process resource perspective. In [12, 13] we showed the applicability and success of the approach with real life event logs and case studies. In all application areas we showed that resource mining provides interesting insights in the way resources are involved, how teams are composed and how collaborative work is performed. In future work we want to apply our mining approaches on event logs from additional domains like hospital logs, and use the results for different purposes, e.g., for checking compliance rules with respect

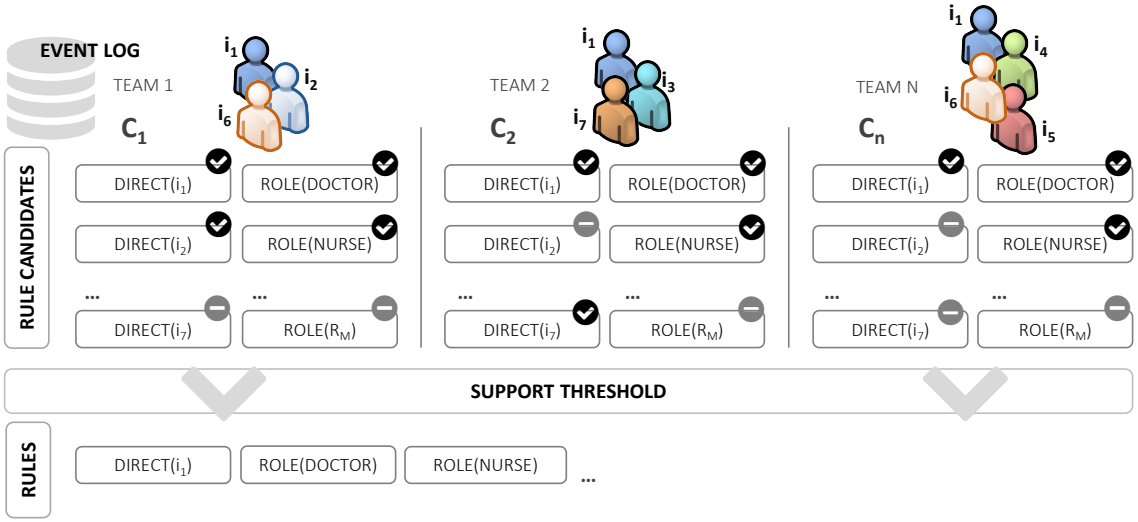

Figure 4: Checking of various organisational patterns in collection of resources [12]

to team compositions.

Acknowledgements

This research was funded by the Austrian Research Promotion Agency (FFG), grant 845638 (SHAPE).

References

[1] van der Aalst, W.: Process mining: discovery, conformance and enhancement of business processes. Springer-Verlag Berlin Heidelberg (2011)

[2] van der Aalst, W., Pesic, M., Schonenberg, H.: Declarative workflows: Balancing between flexibility and support. Computer Science - Research and Development **23**(2), 99–113 (2009)

[3] van der Aalst, W.M.P., Rosemann, M., Dumas, M.: Deadline-based escalation in process-aware information systems. Decision Support Systems **43**(2), 492–511 (2007)

[4] Cabanillas, C., Knuplesch, D., Resinas, M., Reichert, M., Mendling, J., Ruiz-Cortés, A.: RALph: A Graphical Notation for Resource Assignments in Business Processes. In: CAiSE, vol. 9097, pp. 53–68 (2015)

[5] Di Ciccio, C., Marrella, A., Russo, A.: Knowledge-intensive processes: Characteristics, requirements and analysis of contemporary approaches. J. Data Semantics **4**(1), 29–57 (2015)

[6] Dumas, M., Rosa, M.L., Mendling, J., Reijers, H.A.: Fundamentals of Business Process Management. Springer (2013)

[7] Jablonski, S.: MOBILE: A modular workflow model and architecture. In: Working Conference on Dynamic Modelling and Information Systems (1994)

[8] de Leoni, M., van der Aalst, W.M., Dees, M.: A general process mining framework for correlating, predicting and clustering dynamic behavior based on event logs. Information Systems **56**, 235–257 (2016)

[9] Maggi, F.M., Bose, J.C., van der Aalst, W.: Efficient Discovery of Understandable Declarative Process Models from Event Logs. In: CAiSE, pp. 270–285 (2012)

[10] Nakatumba, J., van der Aalst, W.: Analyzing resource behavior using process mining. In: BPM Workshops, pp. 69–80 (2010)

[11] Russell, N., van der Aalst, W.M.P., ter Hofstede, A.H.M., Edmond, D.: Workflow Resource Patterns: Identification, Representation and Tool Support. In: CAiSE, pp. 216–232 (2005)

[12] Schönig, S., Cabanillas, C., Di Ciccio, C., Jablonski, S., Mendling, J.: Mining Team Compositions for Collaborative Work in Business Processes. Software and Systems Modeling (2016)

[13] Schönig, S., Cabanillas, C., Jablonski, S., Mendling, J.: A Framework for Efficiently Mining the Organisational Perspective of Business Processes. Decision Support Systems **89**, 87–97 (2016)

[14] Song, M., Van der Aalst, W.M.: Towards comprehensive support for organizational mining. Decision Support Systems **46**(1), 300–317 (2008)

[15] Zeising, M., Schönig, S., Jablonski, S.: Towards a Common Platform for the Support of Routine and Agile Business Processes. In: Collaborative Computing: Networking, Applications and Worksharing, pp. 94–103 (2014)

This document is a pre-print copy of the manuscript
(Schönig et al. 2016)
published in **journal**.

References

Schönig, Stefan, Cristina Cabanillas, Claudio Di Ciccio, Stefan Jablonski, and Jan Mendling (2016). "Mining Resource Assignments and Teamwork Compositions from Process Logs". In: *Softwaretechnik-Trends* 36.4. URL: http://pi.informatik.uni-siegen.de/stt/36_4/03_Technische_Beitraege/Schoenig_et_al_2016.pdf.

BibTeX

```
@Article{         Schoenig.etal/STT2016:MiningResourceAssignmentsfromLogs,
  author        = {Sch{\"{o}}nig, Stefan and Cabanillas, Cristina and Di
                   Ciccio, Claudio and Jablonski, Stefan and Mendling, Jan},
  title         = {Mining Resource Assignments and Teamwork Compositions from
                   Process Logs},
  journal       = {Softwaretechnik-Trends},
  year          = {2016},
  volume        = {36},
  number        = {4},
  url           = {http://pi.informatik.uni-siegen.de/stt/36_4/03_Technische_Beitraege/Schoenig_et_al_2016.pdf}
}
```