Mining Expressive and Executable Resource-Aware Imperative Process Models

Cristina Cabanillas[1](✉), Stefan Schönig[2], Christian Sturm[2], and Jan Mendling[1]

[1] Vienna University of Economics and Business, Vienna, Austria
{cristina.cabanillas,jan.mendling}@wu.ac.at
[2] University of Bayreuth, Bayreuth, Germany
{stefan.schoenig,christian.sturm}@uni-bayreuth.de

Abstract. Process mining extracts relevant information on executed business processes from historical data stored in event logs. The data typically available include the activities executed, temporal information and the resources in charge of their execution. With such data, the functional, behavioural and organisational perspectives of a process can be discovered. Many existing process mining approaches are capable of generating representations involving the first two perspectives with all types of processes. The extraction of simple and complex resource assignment rules has also been tackled with declarative process models. However, it is noticeable that despite imperative notations like BPMN are mostly used for process modelling nowadays, the existing process mining approaches for enriching such models with resource assignments cannot discover rules like separation of duties and do not produce executable resource-aware process models. In this paper we present an approach for mining resource-aware imperative process models that uses an expressive resource assignment language (RALph) with the de-facto standard notation BPMN. The organisational perspective of the resulting models can be automatically analysed thanks to the formal semantics of RALph. The method has been implemented and tested with a real use case.

Keywords: Organisational mining · Process mining
RALph · Resource assignment · Resource mining

1 Introduction

Process mining extracts relevant information on executed business processes from historical data stored in event logs. The discovered process models can be used for subsequent process improvement or for compliance checking against reference models or regulations [1]. The richer the data in the event logs is, the more facets of the underlying processes can be discovered. Typical data stored

This work was funded by the Austrian Science Fund (FWF) - grant V 569-N31 (PRAIS).

in event logs include the activities executed, temporal information on activity executions and the employees of the organisation in charge of them. With such data, the functional (activities), behavioral (control flow) and organisational (human resources, or for short resources) perspectives of business processes can be discovered [2].

Most of the current support for process mining focuses on the two former perspectives and is capable of generating textual as well as graphical representations of the processes discovered [3,4]. The target of those approaches have been both routine (a.k.a. procedural) processes, usually modelled with imperative notations, such as Business Process Model and Notation (BPMN) [5]; and flexible processes, for which declarative notations, such as Declare [6], are preferred. Work on mining the organisational perspective resulting in declarative process models with expressive resource assignments specifying who is allowed to execute the process activities according to the roles, skills and a number of properties related to the resources and the process, has recently been done [7]. Approaches on resource mining with imperative output models have also been developed [8–10]. However, frequently used rules like separation of duties [11] cannot be discovered, and the graphical resource assignments defined are based on the BPMN swimlanes, which are not provided with semantics [5]. That constrains the expressiveness of the resulting models as well as their use to only documentation purposes and not for automatic execution or analysis. This gap is remarkable as BPMN is the de-facto standard notation for process modelling and is used in most of current Business Process Management Systems (BPMSs).

In this paper we introduce an approach for mining resource-aware imperative process models that uses a graphical notation for modelling resource assignments called RALph [12] together with BPMN. As a result, we obtain an executable specification of a business process that can contain a large variety of resource assignment rules, including those defined in the acknowledged creation patterns [11]. The respective RALphMiner has been implemented using an SQL-based mining technique [13], and it has been tested with a real use case from the university domain.

The remainder of the paper is structured as follows: Sect. 2 presents background information by describing the research problem and related work. Section 3 explains our process mining approach. Section 4 describes our implementation and the application results. Finally, Sect. 5 concludes the paper and gives an outlook of directions for potential future work.

2 Background

In this section, we discuss the background of our work. Section 2.1 describes the research problem that we address and Sect. 2.2 summarises previous work.

2.1 Research Problem

The process participants are the actual responsible for the correct operation of the business processes of an organisation. The specification of *who must do*

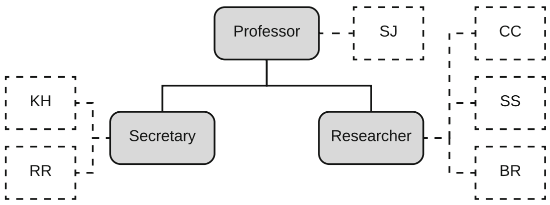

Fig. 1. Hierarchy of positions within a research group

what is usually done based on organisational information. The most common conditions or rules to assign resources to process activities have been collected in a subset of the acknowledged workflow resource patterns, specifically, the creation patterns [11]. They include the following *organisational patterns*: Direct (Dir), Role-based (Rb), Capability-based (Cb), History-based (Hb), Organisational (Org) and Deferred (Def) Distribution; Separation of Duties (SoD); Case Handling (CH); and Retain Familiar (RF), a.k.a. Binding of Duties.

Each organisation may be interested in a certain group of patterns for assigning resources to processes, depending on the organisational information available. Usual data include organisational units, positions, roles and characteristics of the specific people, such as their skills to undertake certain types of tasks. For instance, the research group (organisational unit) depicted in Fig. 1 is structured as a hierarchy of positions. The group is led by a professor (SJ) accountable for the work of two secretaries (KH, RR) and three researchers (BR, SS, CC). One of the most frequent activities related to research is the management of trips for attending conferences, giving invited research talks, and the like. In that process, a researcher first applies for a work trip, which must be approved by their immediate superior. Once approved, the applicant researcher is in charge of booking the accommodation required and of buying the respective transport tickets. Finally, all the documentation is stored by one of the secretaries in order to preserve it for potential future needs, e.g., internal audits.

Process executions are usually stored in event logs, i.e., machine-recorded files that report on the execution of tasks during the enactment of the instances of a given process. In an event log, every process instance corresponds to a sequence (*trace*) of recorded entries, namely, *events*. Each event is defined by a set of *attributes*. These attributes typically involve an explicit reference to the enacted task and to the operating resource [1]. For instance, the following excerpt of a business trip process event log encoded in the XES logging format [14] shows the recorded information of the *start event* of an instance of the activity *Apply for trip* performed by the resource *SS*.

```
<event>
<string key="org:resource" value="SS"/>
<date key="time:timestamp" value="2017-08-06T14:58:00.000+01:00"/>
<string key="concept:name" value="Apply for trip"/>
<string key="lifecycle:transition" value="complete"/>
</event>
```

However, as different activity instances could be executed by different resources, it is necessary to infer the actual resource assignment rules from the

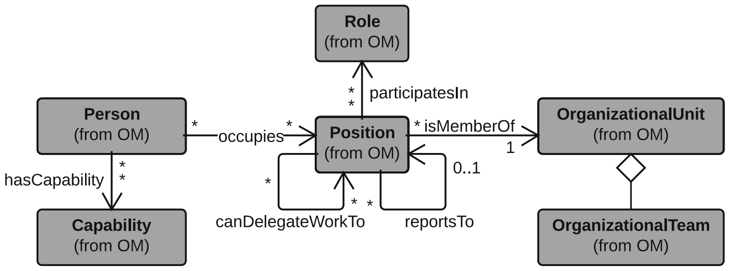

Fig. 2. Organisational metamodel used by RALph (taken from [11])

event log data by applying process mining on the organisational perspective. The organisational information is crucial for that purpose. For instance, in the previous example we should infer that the activity *Apply for trip* is performed by a resource with the position *Researcher*. The output resource assignment rules should be specified together with the functional and behavioural perspectives of the process in the resulting process model.

Therefore, the problem at hand has two inputs, namely, event logs and organisational information. We aim at mining the organisational perspective and defining imperative process models leveraging the fact that BPMN is the de-facto standard notation for process modelling. Furthermore, as BPMN models can be automatised, we want to have executable resource assignment rules, too. Finally, we target simple and complex resource assignment rules on the basis of the organisational patterns. Note that with BPMN, for instance, the last two points are not met since patterns like separation of duties cannot be specified with the BPMN swimlanes and these do not have executable semantics [5]. To address those issues, we can rely on a graphical notation for resource assignment called RALph [12]. The advantages of RALph include: (i) it is independent of any process modelling notation, (ii) it is expressive enough to provide support for all the organisational patterns, and (iii) it has a formal semantics provided by a semantic mapping to Resource Assignment Language (RAL) [15]. RAL is a textual notation whose semantics has been formally defined with description logics, which provides automated analysis power. That means that RALph enables not only the graphical representation of resource assignments, but also their translation to textual assignments as well as their automatic analysis at design time and at run time.

RALph assumes a hierarchical organisation compatible with the organisational metamodel depicted in Fig. 2, similarly to other previous approaches [11]. The notation consists of entities and connectors that enable the visual modelling of resource assignments in process models based on that metamodel. To exemplify the use of RALph, we have modelled some resource assignments of the example scenario in Fig. 3. In Fig. 3a, a *Position* entity is connected to an activity to indicate that a person with the position *Researcher* has to apply for a trip. In Fig. 3b, a RALph *hierarchy connector* is used to specify that the

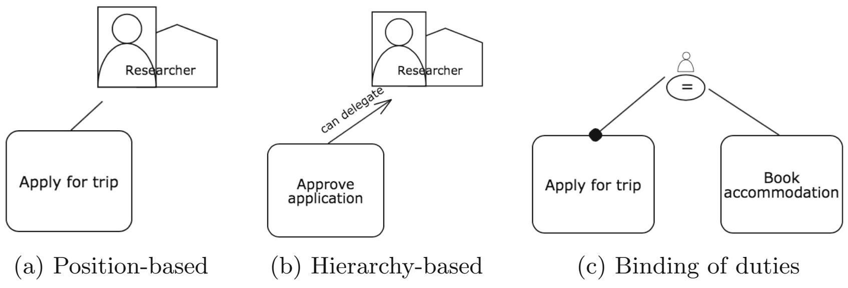

Fig. 3. Examples of RALph assignments with BPMN

Table 1. Approaches for organisational mining: ✓ supported; – not supported; (i) imperative; (d) declarative; (t) textual; (g) graphical; n/a not applicable

Approach	Organisational patterns									Process modelling	Resource assignment	Execution
	Dir	Rb	Def	SoD	CH	RF	Cb	Hb	Org			
[8]	✓	✓	–	–	–	–	✓	–	✓	Petri net (i)	SAR (t)	–
[9,22]	–	✓	–	–	–	–	–	–	–	n/a	n/a	n/a
[10]	–	✓	–	–	–	–	–	–	–	BPMN (i)	BPMN (g)	–
[7]	✓	✓	–	✓	✓	✓	✓	–	✓	DPIL (d)	DPIL (t)	✓

approval of the application must be done by someone who can delegate work to researchers, i.e., a researcher's superior. Finally, Fig. 3c depicts a binding of duties between two activities, meaning that activity *Book accommodation* has to be executed by the same person who performed activity *Apply for trip*. Note that if we had several assignment rules associated with one activity, the intersection of all of them (AND) would be used to find suitable resources. For more flexibility, RALph provides an *alternative connector* that enables the union of the resource assignment rules (OR). For a more detailed description of the RALph notation we refer to [12].

2.2 Related Work

In the last years, a number of techniques for mining the organisational perspective of a process have been developed [16]. Using input data from process event logs, several methods focus on extracting the organisational model and/or a social network [17] behind a business process, which show the characteristics and relationships among the process participants. There is also increasing interest in analysing resource behaviour and productivity [18] as well as the influence of resources on process performance [19–21].

However, the approaches that are most closely related to our research problem are those which address the discovery of organisational patterns with the aim of enriching a given process model with resource assignments [23] (cf. Table 1).

Among them, the so-called staff assignment mining approach [8] is able to extract several types of assignment rules based on decision tree learning. The identification of separation and binding of duties, among others, are not addressed. The output is an imperative process model (a Petri net, an Event-driven Process Chain (EPC) or a Heuristic net) and textual resource assignments written as Staff Assignment Rules (SAR). The approaches classified as role mining [9,10,22] share with each other the fact that they focus on organisational roles. Some of them address the identification of roles by analysing only the data in the event logs [22]. In this case, resource assignment is not an objective. Others aim at building a Role-Based Access Control (RBAC) model [24] that includes information about roles, permissions and role-based assignments to the process activities [9]. The assignments are part of the RBAC model and hence, they are decoupled from the process model. An explicit link to the process model is present in the approach introduced in [10], which uses the Handover of Roles (HooR) principle to enrich a given control-flow model with roles that cluster the process activities under the assumption that each resource has exactly one role. BPMN and its swimlanes [5] are used to show the outcome of the approach. This and some of the aforementioned methods have been integrated into the ProM tool suite[1]. None of the previous approaches covers the whole set of organisational patterns. The DpilMiner was developed to narrow that gap [7]. It implements a three-step framework that can mine not only most of the organisational patterns but also patterns that consider the control flow and the resources together. The output is a declarative process model with textual resource assignments defined with Declarative Process Intermediate Language (DPIL) [25]. The History-based Distribution pattern is not covered because DPIL does not support the definition of the respective resource assignments. On the other hand, the Deferred Distribution pattern can in no case be addressed by a mining approach as it relates to run time.

From the previous discussion we identify two major challenges related to resource mining with imperative process models: (i) the discovery of a large amount of organisational patterns (i.e., expressiveness), and (ii) an executable and user-friendly specification of the resulting resource-aware process models.

3 RALph Mining Method

In the following, we describe our approach for mining the organisational patterns and generating RALph-aware process models. First, in Sect. 3.1 we outline the fundamental concepts on which the method relies. Afterwards, in Sect. 3.2 we define a set of templates that we need in order to discover RALph assignment rules. In Sect. 3.3 we describe the metrics we use for discovering the rules. In Sect. 3.4 we explain and exemplify the discovery mechanism, based on Structured Query Language (SQL) queries. Finally, in Sect. 3.5 we delve into the order in which certain queries must be run and in how to refine the output RALph-aware process models obtained from the mining.

1 http://www.promtools.org/doku.php.

3.1 Fundamentals of Multi-perspective Process Mining

The discovery of the behavioural perspective (control flow) of imperative process models uses any kind of mining technique. However, the extension of such models to include or enhance further perspectives is often done through declarative constraints. That is the case of RALph, which declaratively adds resource assignment rules to provide more expressiveness to the organisational perspective. Therefore, we apply declarative mining to address the problem at hand. Declarative mining is based on the definition of *constraint templates*. Templates are patterns that define parameterised classes of properties, and constraints are their concrete instantiations. Constraint templates are used for querying the provided event log to find solutions for the placeholders. A solution, a.k.a. *constraint candidate*, is any combination of concrete values for the placeholders that yields a concrete rule that is satisfied in the event log. This approach has its roots in declarative process modelling notations, based on rules or constraints, especially in Declare [6]. For instance, a *response* constraint indicates that if activity A *occurs*, activity B must eventually *follow*. A template for this constraint parameterises the variable elements of the rule, in this case A and B, so that by replacing these placeholders with specific activities found in traces of an event log, it can be automatically identified which pairs of activities fulfil the constraint. For example, the *response constraint* is fully satisfied in the traces $\mathbf{t}_1 = \langle A, A, B, C\rangle$, $\mathbf{t}_2 = \langle B, B, C, D\rangle$, and $\mathbf{t}_3 = \langle A, B, C, B\rangle$, but not in $\mathbf{t}_4 = \langle A, B, A, C\rangle$ because, in this case, the second occurrence of A is not followed by a B. In $\mathbf{t}_2$, it is actually *vacuously satisfied* [26], i.e., in a trivial way, because A never occurs.

The semantics of the constraints and the templates can be formalised using formal logics, such as Linear Temporal Logic over finite traces (LTL$_f$) [27]. Declare has traditionally focused on the process functional and behavioural perspectives. The operators that have been typically used include, among others, the $\mathbf{F}$ and $\mathbf{G}$ LTL$_f$ future operators, where: $\mathbf{F}\psi_1$ means that ψ_1 holds sometime in the future, and $\mathbf{G}\psi_1$ means that ψ_1 holds forever in the future. The aforementioned *response* constraint is defined with LTL$_f$ as $\mathbf{G}(A \rightarrow \mathbf{F}B)$.

An *activation activity* of a constraint in a trace is an activity whose execution imposes, because of that constraint, some obligations on the execution of other activities (*target activities*) in the same trace. For example, in the *response* constraint A is an activation activity and B is a target activity, because the execution of A forces B to be executed. An activation of a constraint leads to a *fulfilment* or to a *violation*. Consider again $\mathbf{G}(A \rightarrow \mathbf{F}B)$. In the trace $\mathbf{t}_1$, the constraint is activated and fulfiled twice, whereas in trace $\mathbf{t}_3$, it is activated and fulfiled only once. Referring to the formal specification of constraints in LTL$_f$, *activation* ϕ_a is the sub-formula that lies on the left-hand side of the implication operator $\rightarrow$, whereas *target* ϕ_t is the formula that lies on its right-hand side.

The importance of multi-perspective dependencies led to the definition of a multi-perspective version of Declare (MP-Declare) [28], which is of interest to us since we aim at defining templates for constraints that relate to the process organisational perspective. Its semantics build on the notion of *payload* of an event, which is the set of attributes that define it (cf. Sect. 2.1).

$e(activity)$ identifies the occurrence of an event in order to distinguish it from the activity name. At the time of a certain event e, its attributes $x_1, \ldots, x_m$ have certain values. $p^e_{activity} = (val_{x1}, \ldots, val_{xn})$ represents its payload. To denote the projection of the payload $p^e_A = (x_1, \ldots, x_n)$ over attributes $x_1, \ldots, x_m$ with $m \leqslant n$, the shorthand notation $p^e_A[x_1, \ldots, x_m]$ is used. For instance, $p^e_{ApplyForTrip}[Resource] =$ SS is the projection of the attribute *Resource* in the event description shown in Sect. 2.1. Furthermore, the n-ples of attributes x_i are represented as $\vec{x}$.

Therefore, the templates in MP-Declare extend standard Declare with additional conditions on event attributes. Specifically, given the events $e(A)$ and $e(B)$ with payloads $p^e_A = (x_1, \ldots, x_n)$ and $p^e_B = (y_1, \ldots, y_n)$, the *activation condition* φ_a, the *correlation condition* φ_c, and the *target condition* φ_t are defined. The activation condition is part of the activation ϕ_a, whilst the correlation and target conditions are part of the target ϕ_t, according to their respective time of evaluation. The *activation* condition is a statement that must be valid when the activation occurs. In the case of the *response* template, the activation condition has the form $\varphi_a(x_1, \ldots, x_n)$, meaning that the proposition φ_a over $(x_1, \ldots, x_n)$ must hold true. The *correlation* condition is a statement that must be valid when the target occurs, and it relates the values of the attributes in the payloads of the activation and the target event. It has the form $\varphi_c(x_1, \ldots, x_m, y_1, \ldots, y_m)$ with $m \leqslant n$, where φ_c is a propositional formula on the variables of both the payload of $e(A)$ and the payload of $e(B)$. *Target* conditions exert limitations on the values of the attributes that are registered at the moment wherein the target activity occurs. They have the form $\varphi_t(y_1, \ldots, y_m)$ with $m \leqslant n$, where φ_t is a propositional formula involving variables in the payload of $e(B)$.

3.2 RALph Assignment Templates

Resource assignment modelling languages like RALph are declarative by nature. Therefore, in order to extract RALph-aware process models from event logs, we can rely on existing principles for declarative process mining.

RALph allows for the definition of constraints concerning the assignment of certain resources to activities. Consider a *Direct Assignment* constraint that reflects a constraint on activity a, demanding a, if executed, to be performed by a specific resource res. The respective template comprises placeholders of type *Activity A* as well as *Resource Res*. In Table 2 we provide all RALph constraint templates that should be discovered by our approach according to RALph's expressive power [12]. The table shows the constraint templates, the corresponding semantics in LTL$_f$ and the related payload, i.e., the event attribute that is considered when mining for a certain assignment constraint. In case of the *Direct Assignment* template we have to query the event log for constraints of the shape $\mathbf{G}(A \rightarrow (A \wedge \varphi_t(\vec{x})))$, where the target condition $\varphi_t(\vec{x})$ is of the form $p^e_A[Resource] = val$. To discover *Role-based*, *Capability-based*, *Position-based* and *Unit-based* assignment rules, we query for the same semantics as for *Direct Assignments* but we have to consider different payloads that refer to information

Table 2. Semantics of RALph assignment rules. (*) Resp. *canDelegateWorkTo*

Template	LTL$_f$ Semantics	Related payload cond.
Direct Assignment	$\mathbf{G}(A \rightarrow (A \wedge \varphi_t(\vec{x})))$	$p^{e}_{A}[Resource] = res$
Role-based Assignm.	$\mathbf{G}(A \rightarrow (A \wedge \varphi_t(\vec{x})))$	$p^{e}_{A}[Role] = r$
Pos.-based Assignm.	$\mathbf{G}(A \rightarrow (A \wedge \varphi_t(\vec{x})))$	$p^{e}_{A}[Position] = p$
Cap.-based Assignm.	$\mathbf{G}(A \rightarrow (A \wedge \varphi_t(\vec{x})))$	$p^{e}_{A}[Capability] = c$
Unit-based Assignm.	$\mathbf{G}(A \rightarrow (A \wedge \varphi_t(\vec{x})))$	$p^{e}_{A}[Unit] = u$
Negated Assignm.	$\mathbf{G}(A \rightarrow (A \wedge \varphi_t(\vec{x})))$	$p^{e}_{A}[Unit]! = u$
Binding of Duties	$\mathbf{G}(A \rightarrow \mathbf{G}(B \rightarrow (B \wedge \varphi_c(\vec{x}, \vec{y}))))$	$p^{e}_{A}[Res.] = p^{e}_{B}[Res.]$
Separation of Duties	$\mathbf{G}(A \rightarrow \mathbf{G}(B \rightarrow (B \wedge \varphi_c(\vec{x}, \vec{y}))))$	$p^{e}_{A}[Res.]! = p^{e}_{B}[Res.]$
Hierarchy-based Ass.	$\mathbf{G}(A \rightarrow \mathbf{G}(B \rightarrow (B \wedge \varphi_c(\vec{x}, \vec{y}))))$	$p^{e}_{A}[Res.]$ *reportsTo* $p^{e}_{B}[Res.]$(*)

stemming from the organisational model, e.g., $p^{e}_{A}[Position]$ to discover position-based assignments as described in the example scenario in Sect. 2.1. A *Binding of Duties* template $\mathbf{G}(A \rightarrow \mathbf{G}(B \rightarrow (B \wedge \varphi_c(\vec{x}, \vec{y}))))$ reflects constraints on activity a and b, demanding b, if executed, to be performed by the same resource as activity a. Here, we query the event log for correlation conditions $\varphi_c(\vec{x}, \vec{y})$ on the payloads of the events that correspond to both activities a and b with the specific condition that $p^{e}_{A}[Resource] = p^{e}_{B}[Resource]$.

For subsequent automated discovery, the analyst will select from the set of predefined constraint templates the ones to be discovered depending, for instance, on the type of organisational information available (e.g., only roles, roles and positions, etcetera).

3.3 Metrics for RALph Mining

Querying with constraint templates provides for every possible combination of concrete values for the placeholders in the templates the number of satisfactions in the event log. Based on the number of satisfactions, two metrics, *Support* and *Confidence*, are calculated, which express the probability of an assignment constraint to hold in the process. *Support* is the number of fulfilments of a constraint divided by the number of occurrences of the condition of a constraint. The *Confidence* metric scales the support by the fraction of traces in the log wherein the activation condition is satisfied. Constraints are considered valid if their *Support* and *Confidence* measures are above a user-defined threshold. For our approach we adopt the most recent definition by Di Ciccio et al. [3]. Here, we only consider the event-based support that is meant to be used for all constraints wherein both activation and target events occur.

As defined in [3], we denote the set of *events* in a *trace* $\mathbf{t}$ of an event log L that fulfil an LTL$_f$ formula ψ as $\models^{e}_{\mathbf{t}} (\psi)$. The set of all the *events* in *log* L that fulfil ψ are denoted as $\models^{e}_{L} (\psi)$. Given a resource assignment constraint Ξ comprising activation ϕ_a and target ϕ_t, we define the event-based support $\mathcal{S}^{e}_{L}$ and the event-based confidence $\mathcal{C}^{e}_{L}$ as follows:

$$\mathcal{S}_L^e = \frac{\sum_{i=1}^{|L|} \left| \models_{\mathbf{t}_i}^e (\varXi) \right|}{\left| \models_L^e (\phi_a) \right|} \qquad (1) \qquad\qquad \mathcal{C}_L^e = \frac{\mathcal{S}_L^e \times \left| \models_L^e (\phi_a) \right|}{|L|} \qquad (2)$$

3.4 Discovering RALph Assignment Rules with SQL

Our proposed RALph mining method builds on the SQL-based process discovery approach described in [13] because of its versatility towards customisation. With SQL queries it is possible to extract relevant process knowledge from event logs stored in a conventional relational database following the *RelationalXES (RXES)* architecture [29]. The database tables in our case include: (1) one event log table capturing the following event attributes: *EventID* (unique identifier for each recorded event), *TraceID* (unique identifier for the corresponding trace), *ActivityID* (name of the corresponding activity the event refers to), *Time* (date and time the event has occurred) as well as *Resource* (identifier of the performing resource); and (2) tables for the relationships in the organisational metamodel (cf. Fig. 2) storing the organisational information, which results in six tables: HasCapability (*Person*, *Capability*), Occupies (*Person*, *Position*), ReportsTo (*Position*, *Position*) - resp. CanDelegateWorkTo -, ParticipatesIn (*Position*, *Role*) and IsMemberOf (*Position*, *Unit*).

The mining technique discovers all constraints of a certain template under the consideration of two thresholds *minSupp* and *minConf* related to the metrics described in Sect. 3.3 by applying conventional database queries without any parsing or data conversion. As an example, we explain the SQL query to extract *Direct Assignment* constraints, i.e., the fact that a certain activity has been executed by a specific resource.

```
SELECT  ‘Direct Assignment’, A, l1.Resource, [Support], [Confidence]
FROM Log l1, [ActivityCombinations] c
WHERE l1.Activity = c.A
GROUP BY c.A, c.Resource
HAVING [Support] > minSupp AND [Confidence] > minConf
```

In the FROM clause the data source tables are joined together, i.e., the table of the analysed event log where every tuple depicts a single event and, if available, the tables of the *OrganisationalModel*. Furthermore, the clause contains a subquery *ActivityCombinations* that provides a table with the activity combinations that should be checked. Every source table gets an abbreviation assigned to be referable in other clauses, e.g., “l1” for the event log table or “c” for the combination table. The WHERE clause contains the different constraint expressions that have to hold for activities and their events, i.e., the constraint activation condition as well as its fulfilment requirements. After deriving the fulfilments, the tuples are grouped by the set of parameters of the constraint template in the GROUP BY clause. After grouping, the number of tuples corresponding to a certain parameter combination can be extracted using the SQL aggregate function COUNT(*). In addition, a subquery computes the number of occurrences of the condition of the constraint. This way, the *Support* value of each constraint can be derived. The *Confidence* of each parameter combination can be calculated in a similar way. The resulting values of both queries can then be filtered by user-defined thresholds. In the last step, the query output is selected in the SELECT clause, i.e., the parameter combination and its corresponding *Support* $\mathcal{S}_L^e$ and *Confidence* $\mathcal{C}_L^e$ values. The result set contains tuples for each parameter combination that fulfils the constraint under consideration of the given thresholds. The *Support* value is computed with the subquery below.

```
COUNT(*) /  (SELECT COUNT(*) FROM Log WHERE Activity = A)
```

We next explain the query to extract *Position-based Assignment* constraints. The FROM, WHERE and GROUP BY clauses of the query are as follows:

```
SELECT  'Position-based Assignment', A, l1.Unit, [Support], [Confidence]
FROM Log l1, Position p, [ActivityCombinations] c
WHERE l1.Activity = c.A AND a.Resource = u.Resource
GROUP BY c.A, p.Position
HAVING [Support] > minSupp AND [Confidence] > minConf
```

In addition to the event log and the activity combinations we also join the table with the resource-positions assignments according to the organisational model in the FROM clause. The query sums up all occurrences of events with respective resources and groups the occurrences w.r.t. the corresponding position given in the table *Occupies*.

This approach is followed to define SQL queries for all the types of resource assignments that we aim at discovering, in our case, those in Table 2. All the queries can be found in [30].

3.5 Alternative Connectors and Pruning

If with certain *minSupp* and *minConf* thresholds we do not extract any resource assignment rule for a process activity, it could be the case that several resource assignment rules are associated to it with lower frequencies. Consider, for instance, that for an activity *Apply for trip* we could not extract a valid *Position-based Assignment* rule since for no rule candidate $\mathcal{S}_L^e > minSupp$ with, e.g., $minSupp = 0.95$ holds. In this case, however, it could be possible to extract a *Position-based Assignment* rule for *Researcher* with $\mathcal{S}_L^e = 0.5$ and a *Capability-based Assignment* rule for *Can speak English* with $\mathcal{S}_L^e = 0.5$, respectively. This union is modelled with the RALph alternative connector to express that one of the two conditions suffices to find suitable resources. Therefore, alternative connectors are examined at the end of the mining procedure using lower support thresholds and combining the different extracted assignment rules.

The mining method extracts *all* the assignment rules related to each activity. However, when several rules are extracted for one single activity (AND), not all of them might be strictly necessary to understand the process. Specifically, some rules may be implied by *stronger* rules because they are less restrictive and do not provide added value to the current resource assignment expression of an activity. Those rules complicate the understandability of the discovered

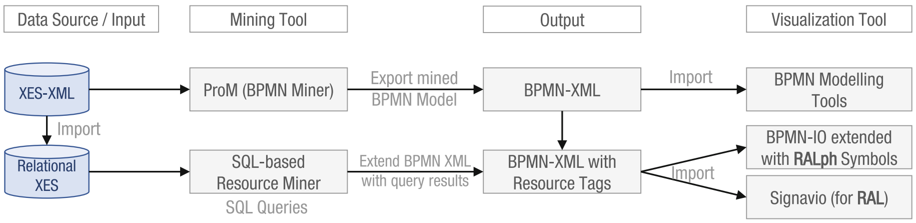

Fig. 4. Implementation architecture and mining procedure

models and hence, they are unnecessary. The work in [7] identifies two pruning approaches to eliminate unnecessary resource assignment rules: pruning based on organisational rule hierarchies (e.g., *position-based assignment* dominates *direct assignment*) and pruning based on transitive reduction (e.g., for binding of duties rules). The requirement for all pruning operations is that they do not change the meaning of the generated model. These post-processing methods can be applied to the approach at hand in a similar way in order to avoid overloading the output RALph-aware process models with unnecessary assignments that would, on the other hand, worsen their readability.

4 Implementation and Evaluation

The RALph mining approach has been implemented as a web-based process mining tool. The implemented architecture and used toolset are illustrated in Fig. 4. We aim at discovering RALph-aware process models and hence, two main elements must be discovered, namely, the definition of the process itself (i.e., the functional and behavioural perspectives) and the resource assignment rules for the process activities (i.e., the organisational perspective). As mentioned in Sect. 1, there is a number of approaches for discovering a business process. Implementations for many of them are available as plug-ins in the ProM framework. We use the *BPMN Miner* tool with a XES event log to extract a resource-unaware BPMN model. Afterwards, the resulting BPMN model is exported as an XML file according to the *BPMN-XML* specification [5]. We use the SQL mining approach described in Sect. 3.4 for extracting RALph assignment rules. Since this approach builds on the relational RXES event log representation, we first have to import the XES event log to relational database tables in RXES format as well as make the organisational information available as tables as explained in Sect. 3.4. We can then run the set of SQL queries required to extract RALph resource assignment rules. The resulting assignment rules are attached in the previous BPMN-XML file to the respective activity as specific resource tags. Here, we match activities from the given BPMN model and the extracted assignment rules based on activity identifiers given in the event log. The RALph-aware BPMN model is then visualised in the graphical BPMN diagram editor *bpmn.io*[2], which has been extended with the RALph symbols. For automatically

[2] BPMN Viewer and Editor, https://bpmn.io.

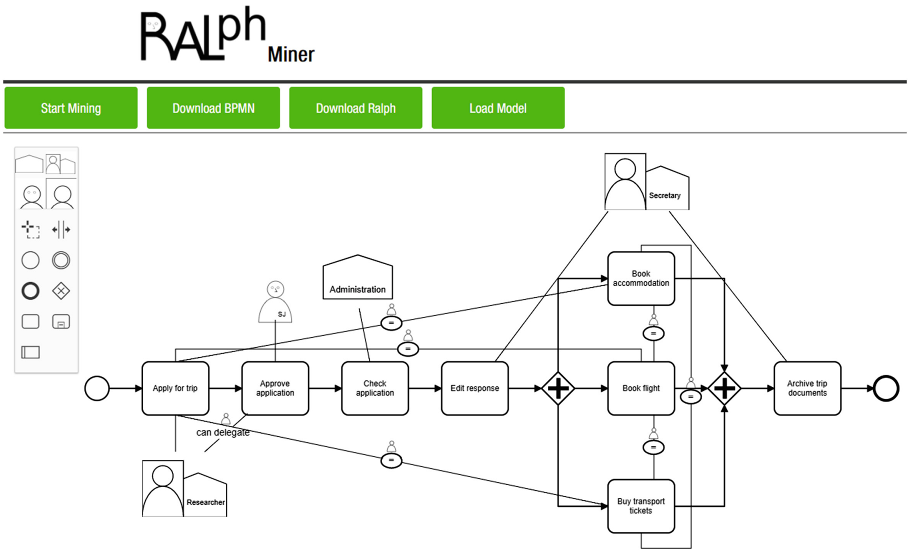

Fig. 5. User interface of the RALph miner with extracted assignment rules

arranging and layouting the RALph assignment symbols in the process diagram, we used a Java Script based implementation of the Sugiyama graph layout algorithm [31]. Additionally, the underyling formal RAL expressions can be imported and edited in BPMN editors like *Signavio*[3]. A plug-in is available to automatically analyse such RAL assignments so that the RAL-aware process model can be automatically executed [32].

As a proof of concept, we applied the described toolset to an event log of a university business trip management system. The log contains 2104 events of 8 different activities related to the application and the approval of university business trips as well as the management of accommodations and transfers, e.g., booking accommodations and transport tickets. The system has been used for 6 months by 11 employees of a research institute. The organisational model of the institute comprises 2 organisational units: Administration, with 2 employees; and Research Group, divided into 3 positions that include 6 researchers, 1 professor and 2 secretaries. Since the underlying mining technique is based on SQL queries, the performance of our approach is directly related to the corresponding mining approach introduced in [13]. On the given event log, we were able to execute all RALph resource assignment queries (cf. Table 2) in less than one second. The resulting BPMN model with the extracted RALph assignment rules is shown in Fig. 5. The screenshot also shows the extended *bpmn.io* modelling toolbox on the left-hand side. Note that the model has not been pruned as described in Sect. 3.5 since the implementation of that post-processing feature is still pending work. Therefore, the model contains some assignment rules that are irrelevant,

[3] https://www.signavio.com.

e.g., the direct assignment of entity *SJ* to *Approve Application* or the binding of duties rule between *Book accommodation* and *Buy transport tickets*.

5 Conclusions and Future Work

In this paper we have addressed a gap in process mining related to the extraction of expressive resource assignment conditions from event logs and the generation of executable resource-aware imperative process models as output models. We have legeraged existing methods and notations, in particular: an SQL-based process mining approach has been extended and the RALph notation has been used to enrich the resulting BPMN process models.

The limitations of the work presented here include: (i) the syntactic translation from LTL_f to SQL has been omitted in this paper due to space limitations but it has been performed as part of a follow-up contribution; and (ii) despite the organisational metamodel used in RALph has been extensively used in other approaches, in companies with a different structure some aspects of the organisation might not be captured and considered.

The implementation of the pruning step after the discovery of the RALph-aware process models is the next task to be performed. Afterwards, an extended evaluation of the quality of the approach will be conducted. Besides, we aim to investigate whether RALph could be used to add resource assignments to declarative process modelling notations for increasing expressiveness. Scalability in resource-aware process models is also an issue to be addressed in this context.

References

1. van der Aalst, W.M.P.: Process Mining - Discovery, Conformance and Enhancement of Business Processes. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-19345-3
2. Dumas, M., La Rosa, M., Mendling, J., Reijers, H.A.: Fundamentals of Business Process Management. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-33143-5
3. Di Ciccio, C., Mecella, M.: On the discovery of declarative control flows for artful processes. ACM Trans. Manag. Inf. Syst. **5**(4), 24:1–24:37 (2015)
4. Rovani, M., Maggi, F.M., de Leoni, M., van der Aalst, W.M.: Declarative process mining in healthcare. Expert Syst. Appl. **42**(23), 9236–9251 (2015)
5. OMG: BPMN 2.0, recommendation, OMG (2011)
6. van der Aalst, W., Pesic, M., Schonenberg, H.: Declarative workflows: balancing between flexibility and support. Comput. Sci. R&D **23**(2), 99–113 (2009)
7. Schönig, S., Cabanillas, C., Jablonski, S., Mendling, J.: A framework for efficiently mining the organisational perspective of business processes. Decis. Support Syst. **89**, 87–97 (2016)
8. Rinderle-Ma, S., van der Aalst, W.M.: Life-cycle support for staff assignment rules in process-aware information systems, Technical report, TU/e (2007)
9. Baumgrass, A.: Deriving current state RBAC models from event logs. In: International Conference on Availability, Reliability and Security, pp. 667–672 (2011)

10. Burattin, A., Sperduti, A., Veluscek, M.: Business models enhancement through discovery of roles. In: IEEE CIDM, pp. 103–110 (2013)
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: Pastor, O., Falcão e Cunha, J. (eds.) CAiSE 2005. LNCS, vol. 3520, pp. 216–232. Springer, Heidelberg (2005). https://doi.org/10.1007/11431855_16
12. Cabanillas, C., Knuplesch, D., Resinas, M., Reichert, M., Mendling, J., Ruiz-Cortés, A.: RALph: a graphical notation for resource assignments in business processes. In: Zdravkovic, J., Kirikova, M., Johannesson, P. (eds.) CAiSE 2015. LNCS, vol. 9097, pp. 53–68. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-19069-3_4
13. Schönig, S., Rogge-Solti, A., Cabanillas, C., Jablonski, S., Mendling, J.: Efficient and customisable declarative process mining with SQL. In: Nurcan, S., Soffer, P., Bajec, M., Eder, J. (eds.) CAiSE 2016. LNCS, vol. 9694, pp. 290–305. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-39696-5_18
14. Verbeek, H.M.W., Buijs, J.C.A.M., van Dongen, B.F., van der Aalst, W.M.P.: XES, XESame, and ProM 6. In: Soffer, P., Proper, E. (eds.) CAiSE Forum 2010. LNBIP, vol. 72, pp. 60–75. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-17722-4_5
15. Cabanillas, C., Resinas, M., del Río-Ortega, A., Ruiz-Cortés, A.: Specification and automated design-time analysis of the business process human resource perspective. Inf. Syst. **52**, 55–82 (2015)
16. Bose, R.P.J.C., Maggi, F.M., van der Aalst, W.M.P.: Enhancing declare maps based on event correlations. In: Daniel, F., Wang, J., Weber, B. (eds.) BPM 2013. LNCS, vol. 8094, pp. 97–112. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-40176-3_9
17. Song, M., van der Aalst, W.M.: Towards comprehensive support for organizational mining. Decis. Support Syst. **46**(1), 300–317 (2008)
18. Pika, A., Leyer, M., Wynn, M.T., Fidge, C.J., ter Hofstede, A.H.M., van der Aalst, W.M.P.: Mining resource profiles from event logs. ACM Trans. Manag. Inf. Syst. **8**(1), 1:1–1:30 (2017)
19. Nakatumba, J., van der Aalst, W.M.P.: Analyzing resource behavior using process mining. In: Rinderle-Ma, S., Sadiq, S., Leymann, F. (eds.) BPM 2009. LNBIP, vol. 43, pp. 69–80. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-12186-9_8
20. Hompes, B.F.A., Maaradji, A., La Rosa, M., Dumas, M., Buijs, J.C.A.M., van der Aalst, W.M.P.: Discovering causal factors explaining business process performance variation. In: Dubois, E., Pohl, K. (eds.) CAiSE 2017. LNCS, vol. 10253, pp. 177–192. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-59536-8_12
21. Wynn, M.T., Poppe, E., Xu, J., ter Hofstede, A.H.M., Brown, R., Pini, A., van der Aalst, W.M.P.: ProcessProfiler3D: a visualisation framework for log-based process performance comparison. Decis. Support Syst. **100**, 93–108 (2017)
22. Jin, T., Wang, J., Wen, L.: Organizational modeling from event logs. In: International Conference on Grid and Cooperative Computing (GCC), pp. 670–675 (2007)
23. Zhao, W., Zhao, X.: Process mining from the organizational perspective. Adv. Intell. Syst. Comput. **277**, 701–708 (2014)
24. American National Standards Institute, Inc.: Role-Based Access Control. ANSI INCITS 359–2004, February 2004. http://csrc.nist.gov/rbac

25. Zeising, M., Schönig, S., Jablonski, S.: Towards a common platform for the support of routine and agile business processes. In: IEEE International Conference on Collaborative Computing: Networking, Applications and Worksharing, pp. 94–103 (2014)
26. Kupferman, O., Vardi, M.Y.: Vacuity detection in temporal model checking. Int. J. Soft. Tools Technol. Transfer **4**(2), 224–233 (2003)
27. Montali, M., Pesic, M., van der Aalst, W.M.P., Chesani, F., Mello, P., Storari, S.: Declarative specification and verification of service choreographies. TWEB **4**(1), 3:1–3:62 (2010)
28. Burattin, A., Maggi, F.M., Sperduti, A.: Conformance checking based on multi-perspective declarative process models. Expert Syst. Appl. **65**, 194–211 (2016)
29. van Dongen, B.F., Shabani, S.: Relational XES: data management for process mining. In: CAiSE Forum 2015, pp. 169–176 (2015)
30. Schönig, S.: SQL Queries for Declarative Process Mining on Event Logs of Relational Databases, CoRR, vol. abs/1512.00196 (2015)
31. Eiglsperger, M., Siebenhaller, M., Kaufmann, M.: An efficient implementation of Sugiyama's algorithm for layered graph drawing. In: Pach, J. (ed.) GD 2004. LNCS, vol. 3383, pp. 155–166. Springer, Heidelberg (2005). https://doi.org/10.1007/978-3-540-31843-9_17
32. Cabanillas, C., del Río-Ortega, A., Resinas, M., Cortés, A.R.: CRISTAL: collection of resource-centric supporting tools and languages. In: BPM (Demos), pp. 51–56. CEUR-WS.org (2012)