Bus Inf Syst Eng 61(6):649–663 (2019)
https://doi.org/10.1007/s12599-018-0546-0

RESEARCH PAPER

Guided Interaction Exploration and Performance Analysis in Artifact-Centric Process Models

Maikel L. van Eck · Natalia Sidorova · Wil M. P. van der Aalst

Received: 31 October 2017 / Accepted: 17 April 2018 / Published online: 7 June 2018

Abstract Artifact-centric process models aim to describe complex processes as a collection of interacting artifacts. Recent development in process mining allow for the discovery of such models. However, the focus is often on the representation of the individual artifacts rather than their interactions. Based on event data, composite state machines representing artifact-centric processes can be discovered automatically. Moreover, the study provides ways of visualising and quantifying interactions among different artifacts. For example, strongly correlated behaviours in different artifacts can be highlighted. Interesting correlations can be subsequently analysed to identify potential causes of process performance issues. The study provides a strategy to explore the interactions and performance differences in this context. The approach has been fully implemented as a ProM plug-in; the CSM Miner provides an interactive artifact-centric process discovery tool focussing on interactions. The approach has been evaluated using real life data, to show that the guided exploration of artifact interactions can successfully identify process performance issues.

Accepted after one revision by Prof. Dr. Zdravkovic.

ir. M. L. van Eck (✉) · Dr. N. Sidorova · Prof. Dr. ir. W. M. P. van der Aalst
Eindhoven University of Technology, Eindhoven, The Netherlands
e-mail: m.l.v.eck@tue.nl

Dr. N. Sidorova
e-mail: n.sidorova@tue.nl

Prof. Dr. ir. W. M. P. van der Aalst
e-mail: wvdaalst@pads.rwth-aachen.de

Prof. Dr. ir. W. M. P. van der Aalst
RWTH Aachen University, Aachen, Germany

Keywords Process discovery · Artifact-centric processes · Performance analysis · Interactive process exploration

1 Introduction

Process discovery is the automated creation of process models that explain the behaviour captured in event data (van der Aalst 2016). These process models can be studied e.g. to identify interesting process flows that differ from the process behaviour expected by a process expert or analyst. However, complex process behaviour can result in unstructured process models, which makes them difficult and time-consuming to analyse.

One of the sources of complexity of discovered process models is that many process discovery approaches produce models that provide a monolithic view on the real process (van der Aalst 2016; van Eck et al. 2016b). Such monolithic models explain the behaviour of a process in terms of the life-cycle of a single process instance. However, in reality a process instance may involve several interacting process objects or artifacts, each with their own life-cycle (van der Aalst et al. 2001; Popova et al. 2015; Cohn and Hull 2009). Examples include a procurement process with order and invoice objects, the usage process of a smart product with sensors measuring different physical aspects like movement or temperature, and the status of a single resource in terms of its status in the different processes it is involved in.

Recently, it has become possible to automatically discover models for process artifacts and their behavioural interactions (Lu et al. 2015; van Eck et al. 2016b; Popova et al. 2015). These techniques produce individual process models for each artifact, similar to traditional process discovery approaches. The addition of artifact interaction

enriches the individual models, connecting process elements from different artifact models. Such information highlights e.g. whether a specific state in one artifact coincides with the state of another artifact.

Artifact-centric techniques can provide more structured process models than traditional discovery approaches (van Eck et al. 2016b). By decomposing the process, smaller and simpler models are obtained with fewer states and state transitions per model. However, decomposing the behaviour of a process into interacting artifacts does not necessarily make the overall process easier to understand. To facilitate the understanding, we present an approach to support the *analysis of behavioural interactions between process artifacts*. The goal is to find the most interesting or relevant interactions so that an analyst can inspect these first. The next step is understanding how these interactions affect process performance. This helps process analysts faced with complex processes involving artifacts interacting in a bigger system.

There are different ways to interpret the interaction of artifacts (Lu et al. 2015; van Eck et al. 2016b; Popova and Dumas 2013). We are interested in finding implications that given the occurrence of a state or activity related to one artifact-lifecycle provide information on the possible behaviour of other artifacts. Process data generally does not explicitly contain these interactions or causal relations between artifact behaviour, so instead, we use information on *correlations between artifact behaviour to obtain such insights*.

The analysis guidance involves the use of measures of interestingness to quantify artifact interactions. Such measures have been developed in the field of association rule learning to quantify the relevance of relations between sets of items (Tan et al. 2004; Liu et al. 2000). In van Eck et al. (2017) we discussed how these measures can be defined in the context of process artifact interaction. Based on these measures a ranking of artifact interactions can be presented to process analysts when inspecting process discovery results.

In this paper we build upon the work of van Eck et al. (2017) in the following way: (1) we define how artifact interaction relations can be partitioned and then compared, (2) we discuss a strategy to analyse artifact interactions, and (3) we show how the entire approach is used to identify possible root causes of process performance issues. We have extended our artifact-centric process discovery tool, the CSM Miner van Eck et al. (2016a) in the ProM process mining framework, to support the explanation and analysis of interactions.

To evaluate the use of analysis guidance in practice we have used the developed tool with real life process data. We explored the top results suggested by the measures of interestingness and then looked how variations in artifact interaction resulted in performance differences in order to identify the underlying causes for these differences. This evaluation shows that the analysis guidance provides insights into the overall process behaviour by highlighting interesting artifact interactions.

The remainder of this paper is structured as follows. First, in Sect. 2 we discuss related work on artifact-centric process mining and measures of interestingness. In Sect. 3 we introduce a way to model processes representing artifact systems and define artifact interactions. Then in Sect. 4 we explain how interactions can be partitioned and compared to highlight performance differences. We present the CSM Miner, which implements the analysis approach described in this paper, in Sect. 5. In Sect. 6 we present a strategy to analyse artifact interactions and their effects on overall process performance. We evaluate the approach using real life process data in Sect. 7. Finally, in Sect. 8 we present future work and conclusions.

2 Related Work

A plethora of algorithms and tools for automated process discovery emerged over the last decade (van der Aalst 2016). These produce models in various process model notations. Several approaches have also been developed to take an object-oriented or artifact-centric view of process mining (van der Aalst et al. 2001; Popova et al. 2015). However, the number of techniques that can automatically discover the interactions between artifact models is limited (Lu et al. 2015; van Eck et al. 2016b).

There are different types of behavioural interaction between artifacts that can be mined from process execution data. Like in monolithic process discovery, it is possible to establish causal dependencies between events that occur in different artifacts (Lu et al. 2015). It is also possible to link a stage in one artifact lifecycle to stages in related artifact lifecycles by discovering synchronisation conditions (Popova and Dumas 2013). Similarly, one can identify artifact interaction defined as the co-occurrence of states and transitions from different artifacts as part of the states and transitions of the entire process (van Eck et al. 2016b).

The goal of the analysis of process artifacts and their interaction is to help the user understand complex behaviour by providing additional structure to the process through decomposition. There are several other existing approaches in process mining to deal with model complexity. Most process discovery tools have filtering options or sliders to adjust which activities and dependencies between activities are shown, often based on frequency information (van der Aalst 2016). For some types of processes it is also possible to discover hierarchical process models that allow the analysis of a process at different

levels of detail (Bose et al. 2011). Trace clustering is a technique to decompose the process data of flexible processes with many different process instance variants that share little overlap in behaviour (Weerdt et al. 2013). The clustered process instances are used to mine a more limited model with fewer and stronger dependencies between activities. However, all these approaches simplify the real behaviour shown by the data and hide information instead of using the complete information to guide the analyst.

Understanding the relations between artifacts and their effect on the overall process behaviour is a challenge (Lu et al. 2015). For complex processes this requires the analysis of large numbers of possible artifact interactions, many of which are not interesting. This problem is related to the problem in association rule learning that association rule mining algorithms produce large numbers of rules that are not equally relevant (Bazaldua et al. 2014; Liu et al. 2000; Tan et al. 2004). A solution in association rule learning for this problem involves the quantification of the interestingness of the association rules using specific measures of interestingness.

In process mining it is common to look at different variants of a process to see how instance characteristics influence the overall process behaviour and related aspects such as performance (Rosa et al. 2017; Bolt et al. 2017). Techniques like trace clustering and process cubes can be used to partition a dataset in order to compare different process variants (Weerdt et al. 2013; Vogelgesang et al. 2016; van der Aalst 2013). Generally, with this type of technique each partition results in a separate process model, possibly annotated with performance information, which can then be compared. Other approaches use the control-flow context and process-specific information to train decision or regression trees that can predict performance at given points in a process (Bolt et al. 2017). However, to the best of our knowledge there are no approaches that can discover how process variants affect artifact interactions and their effects on the performance of artifact-centric processes.

3 Modelling of Artifact Systems

In this work we use the notion of state machines to model processes representing artifact systems and the life-cycles of artifacts as presented in van Eck et al. (2017). Note that unlike (Cohn and Hull 2009) we do not consider an information model describing associated data.

Regarding notation, we write σ_k for the k-th element of a sequence $\sigma \in S^*$ of elements from some set S, and $|\sigma|$ denotes the length of σ. We write $s \in \sigma$ if $s = \sigma_k$ for some k and $\sigma\langle s, \ldots, s'\rangle$ for the concatenation of σ with sequence $\langle s, \ldots, s'\rangle$. Additionally, for $s \in S_1 \times \cdots \times S_n$ we write $s(i)$ for the value of the i-th component of s ($i \in \{1, \ldots, n\}$).

3.1 Composite State Machines

A process consisting of a number of interacting artifacts is called an artifact system, and we model its behaviour as a *Composite State Machine* (CSM). The state of a CSM is defined as the composition of the states of its artifacts, i.e. it is a vector of states. The set of all possible states of a CSM is a subset of the cartesian product of the sets of states of its artifacts, as not all combinations of artifact states are necessarily possible. Each transition in a CSM represents a change in the state of at least one artifact; we do not allow self loops. Formally:

Definition 1 A *Composite State Machine* $\mathcal{M} = (S, T, b, f)$ is a model of a process with n artifacts where $S \subseteq (S_1 \times \ldots \times S_n)$ is a set of states, with $S_1, \ldots, S_n$ the sets of artifact states, $b = (b_1, \ldots, b_n)$ is the initial source state, $f = (f_1, \ldots, f_n)$ is the final sink state, $T \subseteq (S \cup \{b\}) \times (S \cup \{f\})$ is the set of transitions, and $\forall (s, s') \in T : s \neq s'$. We define $\overline{S} = S \cup \{b, f\}$ and $\overline{S}_i = S_i \cup \{b_i, f_i\}$ for $i \in \{1, \ldots, n\}$.

The explicit initial and final states have no incoming and outgoing transitions, respectively. They are not true states: they only mark the points in time where a process instance begins and finishes. As a special case, we call a CSM with only one artifact an *Artifact Model*, which represents the behaviour of the artifact in isolation.

We can project a CSM onto a specific subset of its artifacts to focus only on their behaviour. A *CSM Projection* is obtained by reducing the cartesian product of each state to the given subset of artifacts, merging the identical states, and omitting unnecessary transitions and self loops. As transitions represent state changes, two states of a projection are only connected by a transition if there is a transition in the CSM whose source and target are reduced to these different states.

Definition 2 Given a CSM $\mathcal{M}$ with n artifacts and an ordered subset of indices $\Pi = \{i_1, \ldots, i_m\} \subseteq \{1, \ldots, n\}$, with $i_1 < i_2 < \ldots < i_m$, we define the state projection function $\pi_\Pi : (\overline{S}_1 \times \ldots \times \overline{S}_n) \rightarrow (\overline{S}_{i_1} \times \ldots \times \overline{S}_{i_m})$ as follows: $\forall s \in \overline{S}, i_j \in \Pi : (\pi_\Pi(s))(j) = s(i_j)$. A *CSM Projection* of $\mathcal{M}$ on Π, $\mathcal{M}^\Pi = (S^\Pi, T^\Pi, b^\Pi, f^\Pi)$, is defined as:

$$
\begin{aligned}
S^\Pi &= \{\pi_\Pi(s) | s \in S\},\\
T^\Pi &= \{(\pi_\Pi(s), \pi_\Pi(s')) | (s, s') \in T \wedge \pi_\Pi(s) \neq \pi_\Pi(s')\},\\
b^\Pi &= \pi_\Pi(b),\\
f^\Pi &= \pi_\Pi(f).
\end{aligned}
$$

The *Artifact Model* $\mathcal{A}_i$ is defined as the projection $\mathcal{M}^{\{i\}}$ of $\mathcal{M}$ on $\{i\}$.

Note that the projection of a CSM is itself again a CSM, modelling only the behaviour of the artifacts projected on.

In Fig. 1 we present a simple healthcare process, which we use as a running example. This process (model $\mathcal{M}$) has two distinct perspectives or artifacts: the status of the patient being treated (model $\mathcal{A}_1$), and the status of lab tests of the patient (model $\mathcal{A}_2$). The artificial initial and finial states are marked without border.

The healthcare process starts when the patient is registered, after which a lab test is planned to diagnose the patient. If the patient misses their appointment or if the results are inconclusive, then a new test is planned, but if the test results are ready then the treatment can proceed. During the treatment additional tests may be required, until the patient is healthy again and the process ends. Note that the composite process is smaller than the cartesian product of the artifacts ($4 \times 5 = 20$ states) because not all state combinations can be observed due to interdependencies. For example, once the patient is healthy no extra lab tests are needed. Such dependencies between artifacts can be interesting to analyse.

3.2 Process Execution Data

The CSM models as introduced above provide only limited insights into the dependencies and interaction between the artifacts whose behaviour makes up the process of the artifact system. There are no expected sojourn times for the different states or frequencies for transitions. For the process in Fig. 1 an analyst could be interested e.g. in the average time spent *Waiting on result* (*C*) while the patient is *In treatment* (*Y*) or the difference in probability of transitioning to *New test needed* (*E*) before and after the patient is *Diagnosed* (*X*). To enrich the model with such information, we require a collection of process execution data.

In this work we assume the availability of both a CSM of the process of interest and a matching collection of process instance data consisting of execution sequences of the process. Each *State Entry* in an *Execution Sequence*, or trace, specifies the new state of the artifact system at a certain point in time. A collection of execution sequences together form a *Log*. Given a log, a CSM can be discovered that matches the execution sequences in the log using the approach presented in van Eck et al. (2016a).

Definition 3 Let $\mathcal{M}$ be a CSM and $\mathbb{T}$ a time domain. We call $e \in (\overline{S} \times \mathbb{T})$ a *State Entry*. Function $\mathsf{state}(e)$ returns the state, $\mathsf{time}(e)$ returns the time, and $\mathsf{state}_i(e) = \pi_{\{i\}}(\mathsf{state}(e))$ returns the state projection of the state entry e.

$\sigma \in (\overline{S} \times \mathbb{T})^*$ is an *Execution Sequence* of $\mathcal{M}$ iff:

- $\mathsf{state}(\sigma_1) = b$,
- $\mathsf{state}(\sigma_{|\sigma|}) = f$,
- $(\mathsf{state}(\sigma_k), \mathsf{state}(\sigma_{k+1})) \in T$ for $k \in \{1, \ldots, |\sigma| - 1\}$,
- $\mathsf{time}(\sigma_1) = \mathsf{time}(\sigma_2)$, and
- $\mathsf{time}(\sigma_k) < \mathsf{time}(\sigma_{k+1})$ for $k \in \{2, \ldots, |\sigma| - 1\}$.

The set $\mathsf{Traces}_{\mathcal{M}}$ is the set of all possible execution sequences of $\mathcal{M}$. A *Log* $\mathcal{L}_{\mathcal{M}} : \mathsf{Traces}_{\mathcal{M}} \to \mathbb{N}$ is a multiset of execution sequences.

An example of an execution sequence for the CSMs from Fig. 1 is provided in Fig. 2. Note that no time is spent in the artificial initial state *b*, representing the beginning of the known execution, but it is included in execution sequences to enable the calculation of the frequency of the different possible ways to start a process. Artificial final

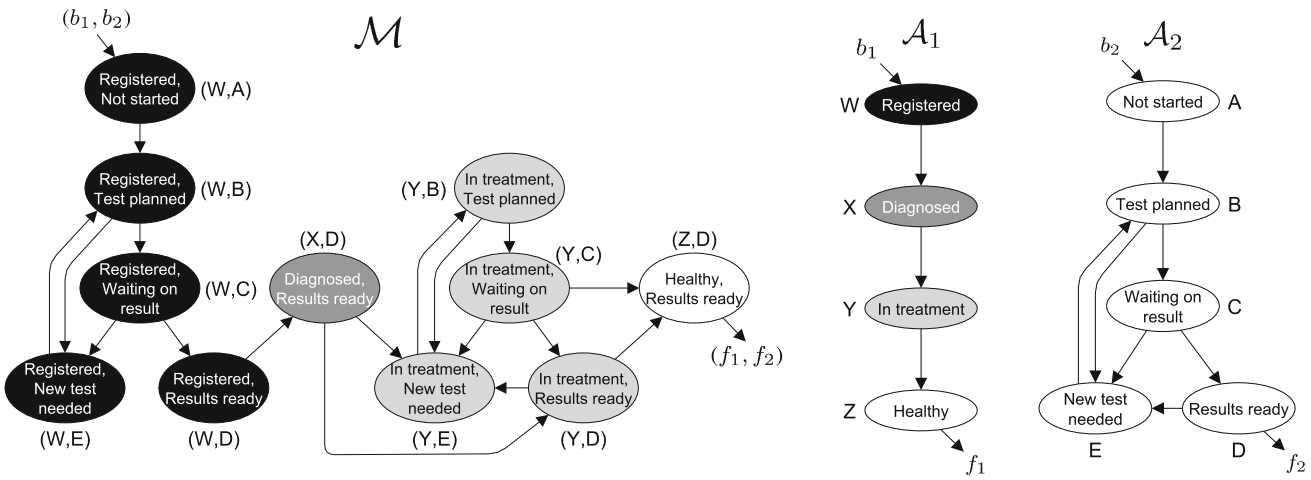

Fig. 1 A model $\mathcal{M}$ of a simple healthcare process and its two artifact models $\mathcal{A}_1$ and $\mathcal{A}_2$. Every state in the process is a combination of a state from each artifact

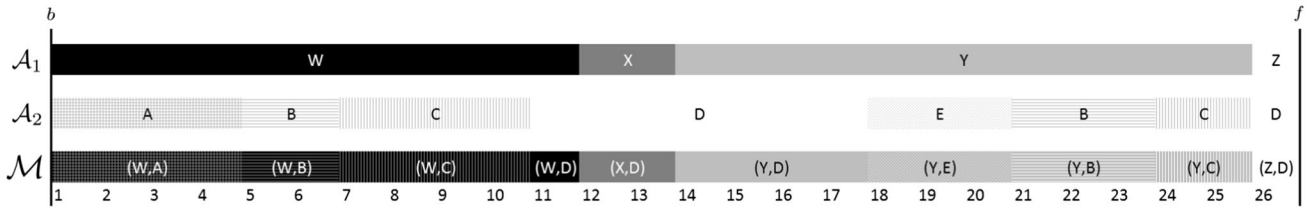

Fig. 2 An execution sequence for the running example process from Fig. 1

Table 1 The state entries of the execution sequence σ of $\mathcal{M}$ from Fig. 2, the sequence projected on the first artifact $\sigma' = \pi_{\{1\}}(\sigma)$, and the sequence projected on the second artifact $\sigma'' = \pi_{\{2\}}(\sigma)$

k	σ_k	$\delta(\sigma_k)$	l	σ'_l	$\delta(\sigma'_l)$	m	σ''_m	$\delta(\sigma''_m)$
1	((b_1, b_2), 1-1-'17)	0	1	(b_1, 1-1-'17)	0	1	(b_2, 1-1-'17)	0
2	((W,A), 1-1-'17)	4	2	(W, 1-1-'17)	11	2	(A, 1-1-'17)	4
3	((W,B), 5-1-'17)	2	3	(X, 12-1-'17)	2	3	(B, 5-1-'17)	2
4	((W,C), 7-1-'17)	4	4	(Y, 14-1-'17)	12	4	(C, 7-1-'17)	4
5	((W,D), 11-1-'17)	1	5	(Z, 26-1-'17)	1	5	(D, 11-1-'17)	7
6	((X,D), 12-1-'17)	2	6	(f_1, 27-1-'17)	0	6	(E, 18-1-'17)	3
7	((Y,D), 14-1-'17)	4				7	(B, 21-1-'17)	3
8	((Y,E), 18-1-'17)	3				8	(C, 24-1-'17)	2
9	((Y,B), 21-1-'17)	3				9	(D, 26-1-'17)	1
10	((Y,C), 24-1-'17)	2				10	(f_2, 27-1-'17)	0
11	((Z,D), 26-1-'17)	1						
12	((f_1,f_2), 27-1-'17)	0						

state f represents the point in time after which the process instance finished and the state is unknown.

We can use the time information in an execution sequence to calculate the time spent in a given state. By aggregating the durations of state entries over a log the models can be enriched with sojourn time statistics for each state. Similar to state sojourn times, we can also count the number of transitions occurring in a log. These numbers can be used to annotate the transitions in the process models with frequency statistics.

Definition 4 Let σ_k be a state entry of an execution sequence $\sigma \in \mathsf{Traces}_{\mathcal{M}}$ of CSM $\mathcal{M}$. The state entry's duration is given by:

$$\delta(\sigma_k) = \begin{cases} \mathsf{time}(\sigma_{k+1}) - \mathsf{time}(\sigma_k), & \text{if } 1 \leq k < |\sigma| \\ 0, & \text{if } k = |\sigma| \end{cases}$$

The total sojourn time of a state $s \in S$ for a log $\mathcal{L}_{\mathcal{M}}$ is:

$$\mathsf{soj}(s, \mathcal{L}_{\mathcal{M}}) = \sum_{\sigma \in \mathcal{L}_{\mathcal{M}}} \sum_{k | \mathsf{state}(\sigma_k) = s} \delta(\sigma_k) * \mathcal{L}_{\mathcal{M}}(\sigma)$$

The frequency of a transition $(s, s') \in T$ for a log $\mathcal{L}_{\mathcal{M}}$ is:

$$\mathsf{freq}_T((s, s'), \mathcal{L}_{\mathcal{M}}) = \sum_{\sigma \in \mathcal{L}_{\mathcal{M}}} |\{k | \mathsf{state}(\sigma_k) = s \wedge \mathsf{state}(\sigma_{k+1}) = s'\}| * \mathcal{L}_{\mathcal{M}}(\sigma)$$

An execution sequence of a CSM can also be projected onto a subset of its artifacts such that it is an execution sequence of the matching projected CSM. The projection abstracts from state entries where the state of the specified artifacts does not change from the previous state entry, as these entries no longer represent transitions in the projected process model. With such projections we can calculate sojourn and frequency statistics to enrich projected CSMs as before.

Definition 5 Let $\mathcal{M}$ be a CSM, Π a set of artifact indices, and π_Π a state projection function. We lift the application of projection function π_Π to sequences $\sigma \in \mathsf{Traces}_{\mathcal{M}}$ so that $\pi_\Pi(\sigma) \in \mathsf{Traces}_{\mathcal{M}^\Pi}$. We define $\pi_\Pi(\sigma)$ recursively:

If $\sigma = \langle\rangle$ then $\pi_\Pi(\sigma) = \langle\rangle$, and if $\sigma = \langle e \rangle$, with $e \in (\overline{S} \times \mathbb{T})$, then $\pi_\Pi(\sigma) = \langle(\pi_\Pi(\mathsf{state}(e)), \mathsf{time}(e))\rangle$. For an execution sequence $\sigma\langle e_1, e_2\rangle$,

$$\pi_\Pi(\sigma\langle e_1, e_2\rangle) = \begin{cases} \pi_\Pi(\sigma\langle e_1\rangle), & \text{if } \pi_\Pi(\mathsf{state}(e_1)) = \pi_\Pi(\mathsf{state}(e_2)) \\ \pi_\Pi(\sigma\langle e_1\rangle)\pi_\Pi(\langle e_2\rangle), & \text{otherwise} \end{cases}$$

A *Log Projection* $\mathcal{L}^\Pi_{\mathcal{M}} : \mathsf{Traces}_{\mathcal{M}^\Pi} \to \mathbb{N}$ of a log $\mathcal{L}_{\mathcal{M}}$ is a multiset of execution sequences such that: $\forall \varsigma \in \mathsf{Traces}_{\mathcal{M}^\Pi} : \mathcal{L}^\Pi_{\mathcal{M}}(\varsigma) = \sum_{\sigma \in \mathcal{L}_{\mathcal{M}} : \varsigma = \pi_\Pi(\sigma)} \mathcal{L}_{\mathcal{M}}(\sigma)$.

Table 1 shows an execution sequence σ of the running example process and its projections $\pi_{\Pi}(\sigma)$ for $\Pi = \{1\}$ and $\Pi = \{2\}$, together with their corresponding durations.

The information in a collection of execution sequences can be used to enrich a CSM and its projections with state sojourn statistics and transition frequencies as described above. Figure 3 shows the running example process of Fig. 1 annotated with frequency and average sojourn time information. Process execution data can also be used for the identification of relations between artifact model elements and the calculation of measures of interestingness for such relations.

3.3 Artifact Interaction

Given a CSM $\mathcal{M}$ with multiple artifacts and a log $\mathcal{L}_{\mathcal{M}}$, we want to find interesting artifact interactions that are a part of the artifact system behaviour. Interestingness in the context of pattern analysis generally means unexpected to the user, i.e. new knowledge or contradicting expectations, and potentially actionable, e.g. leading to process improvement (Liu et al. 2000). For example, if the state of an artifact cannot be advanced until a certain state in a different artifact has been reached then this may represent a bottleneck in the overall process. Similarly, the probability of making specific choices at a decision point in one artifact may be affected by the state of another artifact.

The executions in a log do not explicitly describe causal dependencies between the behaviour of different artifacts, but we can infer *correlations* between sets of artifact states or transitions. Based on this, we distinguish three types of artifact interaction: *state co-occurrence*, *transition co-occurrence* and *forward-looking co-occurrence*.

We focus here only on the interaction between pairs of artifacts, but the interaction definitions can be generalised to involve sets of artifacts. We formulate each interaction as an *implication* $(X \Rightarrow Y)$ between two statements regarding the states or execution behaviour of the artifacts. In van Eck et al. (2017) we already presented the formulas to calculate these interactions in detail.

State co-occurrence $(s_i \Rightarrow_S s_j)$ is defined as the conditional probability that artifact model $\mathcal{A}_j$ is in state s_j given that artifact model $\mathcal{A}_i$ is in state s_i. From the execution sequences in a log we can determine the strength of this interaction in the observed data. It is calculated as the amount of time the system state contains both states divided by the total time spent in s_i.

Transition co-occurrence $((s_i, s_i') \Rightarrow_T (s_j, s_j'))$ is defined as the conditional probability that, given that $\mathcal{A}_i$ is in a transition from s_i to s_i', $\mathcal{A}_j$ has a state s_j before and a state s_j' after the transition. If $s_j = s_j'$ this co-occurrence specifies the state of $\mathcal{A}_j$ during the given transition in $\mathcal{A}_i$, but if they differ then it specifies a transition in $\mathcal{A}_j$ that co-occurs with the transition in $\mathcal{A}_i$. The strength of this interaction is calculated as the number of times we observe transitions for which both the condition and the consequence hold divided by the total number of observed transitions for which the condition holds.

Forward-looking co-occurrence $(s_i \wedge s_j \Rightarrow_{\mathcal{F}} (s_j, s_j'))$ is defined as the conditional probability that the next transition executed in $\mathcal{A}_j$ goes to state s_j', given that $\mathcal{A}_j$ is in state s_j and that $\mathcal{A}_i$ is in state s_i during and after the next transition in $\mathcal{A}_j$. The strength of this interaction is calculated as the number of times we observe a transition from s_j to s_j' while $\mathcal{A}_i$ has the specified state s_i divided by the total number of outgoing transitions from s_j while $\mathcal{A}_i$ is in s_i.

It is possible to calculate the artifact interactions defined above for all pairs of states and transitions of all pairs of artifacts. However, it is clear that this results in a very large number of interactions for a process analyst to inspect. One solution to this problem is to rank and filter the list of

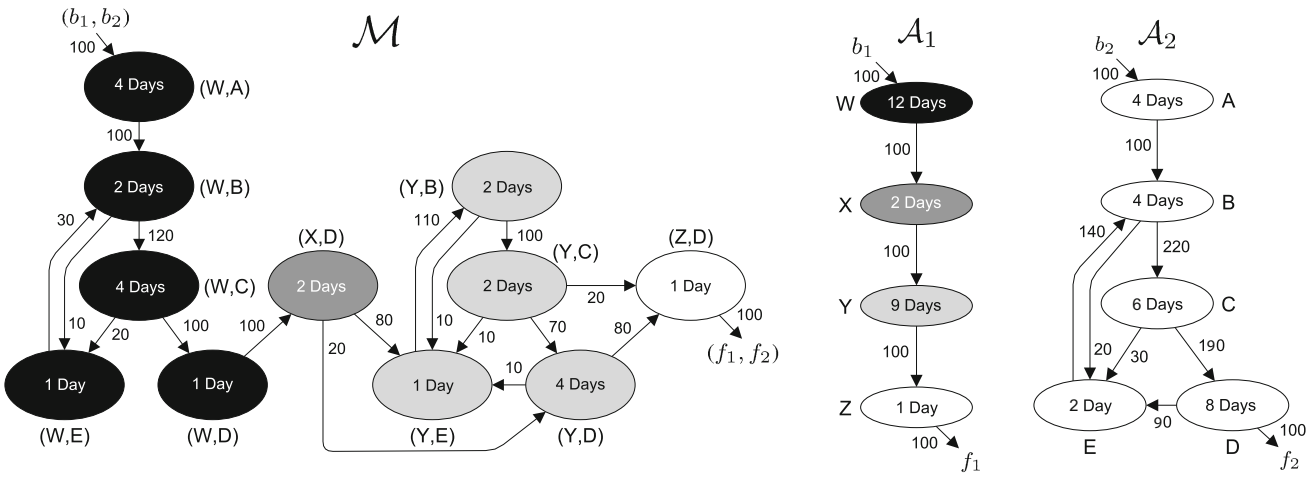

Fig. 3 The models of the healthcare process from Fig. 1 annotated with transition frequencies and average state sojourn times per trace

interactions to obtain the most interesting artifact relations and to present those to the analyst first.

In order to rank and filter artifact interactions based on their interestingness it is necessary to be able to quantify "interestingness". As we discussed in Sect. 2, work has been performed in the field of association rule learning to develop measures of interestingness to help with the analysis of large sets of association rules (Tan et al. 2004; Liu et al. 2000). In van Eck et al. (2017) we select a number of such measures and we discuss their meaning and applicability in the context of artifact interactions that represent process behaviour.

4 Focussed Performance Analysis

Ranking and filtering artifact interactions based on measures of interestingness results in a shortlist of interactions that can be studied in detail to get insights in the overall process performance. The interactions discussed above show how the behaviour of one artifact is influenced given a condition on the behaviour of another artifact. Applications of this allow an analyst to see e.g. how a state with a high sojourn time is correlated with states of other artifacts and how the occurrence of a state affects the progression at decision points in other artifacts. A next step can then be a root cause analysis of the overall process performance in terms of such interactions affecting specific parts of the overall process.

The interactions defined above are formulated as implications, i.e. having a condition and a consequence. For a given condition there can be multiple possible consequences, e.g. in Fig. 1 if the process is in state (X, D) there exist two *forward-looking co-occurrence* interactions related to the transitions to states (Y, D) and (Y, E). In this example, whether or not a new test is needed would likely depend on the type of treatment given to the patient. Therefore, if we know the treatment type of each case then we can investigate the strength of these interactions for each treatment type separately, in order to confirm the above hypothesis. So, given additional information it is possible to further restrict the condition of an artifact interaction in order to analyse variants of the process and the possible causes of differences in overall process performance.

In the following, we assume that traces and state entries can have additional attributes associated to them. Examples of such attributes are treatment type or patient characteristics, on a trace level, and the specific type of test planned, on a state entry level. We do not provide formal definitions of these extensions here, but they can be interpreted as mappings from specific traces or state entries to attribute values.

Given a set of attributes and associated values for the traces in a log, we can divide the traces into subsets that can be used to compare artifact interaction in different variants of the same process. So, to investigate how treatment type affects the number of new tests needed in our running example, we partition a log such that each partition forms a sub-log contains all the traces belonging to a single treatment type. For each sub-log we then calculate the strength of the *forward-looking co-occurrence* interactions, e.g. between states (X, D) and (Y, E). If there are three treatment types then this would result in three separate artifact interaction strength measures, which we can compare to see what the estimated expected probability of a new test is for each treatment type. In this way we can perform a focussed performance analysis to test a specific hypothesis.

As described in Sect. 3.3 and discussed in detail in van Eck et al. (2017), the strength of artifact interactions is calculated in terms of state sojourn times or transition frequencies, as defined in Definition 4. Therefore, to calculate the interactions for a sub-log we restrict the calculation of state sojourn times to traces of the specific process variant of interest. This is defined as follows, with the calculation of transition frequencies being analogous:

Definition 6 Let $\mathcal{M}$ be a CSM. The total conditional sojourn time of a state $s \in S$ for a log $\mathcal{L}_{\mathcal{M}}$ and a trace mask $\chi \subseteq \mathsf{Traces}_{\mathcal{M}}$ is:

$$\mathsf{soj}(s, \mathcal{L}_{\mathcal{M}}, \chi) = \sum_{\sigma \in \mathcal{L}_{\mathcal{M}} \wedge \sigma \in \chi} \sum_{k | \mathsf{state}(\sigma_k) = s} \delta(\sigma_k) * \mathcal{L}_{\mathcal{M}}(\sigma)$$

By providing a trace mask, traces are either included or excluded in the calculation of interaction measures depending on trace characteristics, i.e. the trace mask defines which traces belong to a specific sub-log we are interested in. We can then test whether these characteristics are responsible for differences in overall process behaviour. A trace mask can be based on both provided and derived attributes. For the running example, a mask based on a provided trace attribute is e.g. the set of traces specific to one treatment type. A mask based on a derived attribute is e.g. 50% of all traces with the longest total duration of the treatment. Both trace masks highlight differences in the probability of a new test being needed during treatment and may offer an explanation for the root cause of the differences.

It is also possible to compare differences in artifact interaction based on characteristics of the state entries directly. This can be done by specifying a state entry mask restricting the calculation of state sojourn times to a set of entries as follows, with transition frequency calculations being analogous:

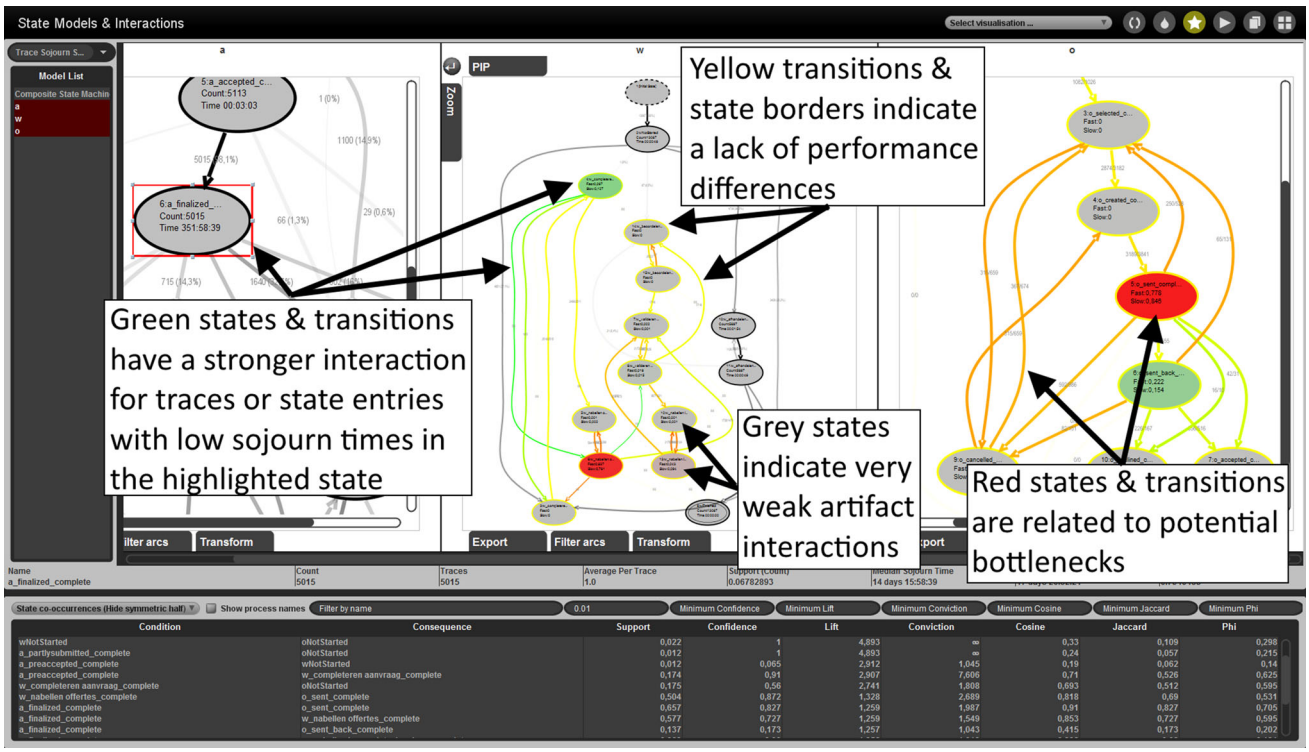

Fig. 4 The analysis guidance is shown below the process models. Users can sort and filter on the different measures of interestingness, and then click on an artifact interaction to highlight it. The colours of the highlights indicate differences in performance

Definition 7 Let $\mathcal{M}$ be a CSM. The total conditional sojourn time of a state $s \in S$ for a log $\mathcal{L}_{\mathcal{M}}$ and a state entry mask $\varphi \subseteq (\overline{S} \times \mathbb{T})$ is:

$$\mathsf{soj}(s, \mathcal{L}_{\mathcal{M}}, \varphi) = \sum_{\sigma \in \mathcal{L}_{\mathcal{M}}} \sum_{k | \mathsf{state}(\sigma_k) = s \wedge \sigma_k \in \varphi} \delta(\sigma_k) * \mathcal{L}_{\mathcal{M}}(\sigma)$$

Restricting the calculation of artifact interaction measures by specifying a state entry mask is very similar to the use of a trace mask. The difference is that two state entries from the same trace may have different characteristics and therefore may belong to different state entry masks. In the running example, the probability that a new test is needed will depend on the latest test results, which can be different for subsequent tests performed on the same patient. By creating a state entry mask for each test result type, it is possible to analyse and compare how these types affect the artifact interactions. A state entry mask can also be based on derived attributes such as the duration of the specific state entry.

5 Analysis Guidance Implementation

In this section we discuss the implementation of the analysis guidance and focussed performance analysis in the CSM Miner van Eck et al. (2016a), a plug-in[1] in the process mining framework ProM.

The CSM Miner discovers a model of the artifact system and of each artifact in the input log, annotates them with sojourn times and frequencies, and presents them in an interactive visualisation. The input log can be a standard XES event log (Verbeek et al. 2011), as long as an attribute is present per event that denotes to what artifact it belongs. The interaction allows the user to click on a state or transition and this will highlight all other states and transitions that co-occur with the selected element. The colour of the highlighting is dependent on the strength of the artifact interaction.

The analysis guidance for the exploration of artifact interactions is provided below the interactive model visualisation, as shown in Fig. 4. It provides a list of artifact interactions and for each interaction the measures of interestingness discussed in van Eck et al. (2017) are calculated. The user can sort the interactions by the measure values and can set minimum values for each measure to filter the list. By clicking on interactions in the table, the corresponding states or transitions are highlighted in the models.

[1] Contained in the *CSMMiner* package of the ProM 6 nightly build, available at http://www.promtools.org/.

The user can also change the artifact interaction highlighting functionality to enable both trace and state entry-based focussed performance analysis. In the *Trace Sojourn Split* mode, the highlighting shows how the artifact interaction differs for two trace masks consisting of the half of traces with the lowest and the half with the highest sojourn time in the selected state. Similarly, in the *State Entry Sojourn Split* mode the highlighting shows the differences for two state entry masks consisting of the half of the state entry occurrences with the lowest and the half with the highest sojourn time in the selected state.

In both modes, the highlighting uses a color scheme from green through yellow to red to denote the magnitude of the differences in artifact interaction. A yellow state border indicates no significant difference between the two groups. A green border indicates that the traces or state entries with a lower sojourn time have a stronger interaction between the highlighted state and the selected state than the traces or state entries with a higher sojourn time, while a red border indicates the reverse. Similar to the border color, the color of the transitions indicates whether the transition co-occurs more frequently in the low or high sojourn time masks. The state fill color indicates the absolute strength of the interaction for the mask with the stronger relative artifact interaction strength, scaling from grey to either green or red, or yellow if there is no difference between the masks.

6 Artifact Interaction Analysis Strategy

The CSM Miner can be used for analysis of artifact interactions and their effect on overall process performance. To help process analysts use the tool effectively we suggest the following approach, especially for large and complex processes.

6.1 Preprocessing and Mining

The first step is the preparation of a state log to be used as input for the CSM Miner. The CSM Miner does not apply any filtering during the mining of the artifact models, so if the input data is expected to contain noise then it can be useful to apply filtering or outlier detection. Removing noise or infrequent behaviour from the input can result in more structured models, but it may also hide interesting deviations from the main process behaviour. Therefore, the correct preprocessing depends on the goal of the process analysis. The *State Log Creator* plugin in ProM can be used to create a log that complies with the input assumptions of the CSM Miner.

Given a state log, the CSM Miner creates the artifact models and computes the artifact interactions. The analyst can then inspect the individual artifact models to identify whether their general structure makes sense. A set of transformation operations is available in the visualisation to modify the models by removing or merging states. Removing states where little time is spent simplifies the models without having a large effect on process behaviour, while the merge operation can be used to hide sub-processes or isolated parts of the process in which the analyst is not interested at the moment. These operations cause a recomputation of the models and all interactions. The final result is a set of artifact models for which the analyst wants to explore interactions.

6.2 Exploring Artifact Interaction

We identify two strategies to explore artifact interactions. The first is using the measures of interestingness presented in van Eck et al. (2017) and the second is to inspect the interactions of each state of a single artifact in a specific order.

In the CSM Miner a list of all artifact interactions is presented below the models, which can be sorted and filtered. The first step is to limit the number of artifact interactions by setting thresholds for some measures. *Support* and *Confidence* are most suitable for this, as their values are intuitive to understand. By setting a minimum support of e.g. 0.01 the interaction has to occur during at least 1% of the time on average and a minimum confidence of e.g. 0.5 means that the condition implies the consequence at least 50% of the time. The exact thresholds to use differ per process and also depend on the goal of the analysis, e.g. finding strong artifact interactions or identifying infrequent patterns.

The second step is sorting the remaining interactions on a specific measure and analysing the top results. For each artifact interaction in e.g. the top 10 results, we propose that the analyst answers the following questions:

- Is the artifact interaction valid?
- Is the insight relevant?
- What is the root cause?

To answer the first question the analyst needs to validate the correctness of the interaction. It may happen that an interaction contradicts known constraints on the process behaviour. This could occur due to incorrect logging or mistakes in the preprocessing of the input data, in which case the analyst has to correct these errors. Invalid interactions related to (artificial) start and end states or states in which a negligible amount of time was spent may also occur due to timestamp granularity issues.

The second question can only be answered by the analyst using domain knowledge. The CSM Miner helps to guide the analyst to interactions that are interesting from a

statistical point of view, which can differ from what a process expert may consider as interesting (Bazaldua et al. 2014). For example, when sorting on *Conviction* the top results will be the interactions for which the consequence always occurs if the condition holds. These strong causal relations between the behaviour of different artifacts are likely well known to a process expert and hence probably not interesting to explore. However, if an interaction very rarely occurs then it may also not be relevant for an analysis of the overall process behaviour.

Answering the third question also requires domain knowledge, but the focussed performance analysis described above can help an analyst to answer this question. We discuss this aspect in more detail in the next subsection.

As an alternative to the use of the measure of interest, an analyst can use the traditional approach of inspecting the states of a certain artifact in a specific order. By selecting an execution sequence, e.g. of the most frequent trace through a single artifact model, the analyst can decide on an order in which to inspect the states and their artifact interactions. For each interaction inspected in this manner it is still useful to use the questions above for guidance. This type of exploration is more time consuming than using the measures of interest, so the analyst may want to skip states in which relatively low amounts of time are spent.

6.3 Root Cause Analysis

For a given state there are usually multiple co-occurrence relations with states and transitions of the other artifacts. This is because of independent behaviour of artifacts or the presence of decision points from where multiple paths are possible. By comparing different variants of the process, an analyst can try to identify the root causes that influence which states and transitions co-occur.

As a proof of concept, the CSM Miner currently supports the comparison of trace and state entry masks based on sojourn time in the selected state. This can be used to determine if there are significant differences between the behaviour of instances which are slowly processed versus those that are quickly processed. Such differences can point to root causes of bottlenecks in the overall performance of the process. For example, delays in a given state may be highly related to the co-occurrence with a choice made in a different artifact. Another example is that quickly processing instances can result in a larger chance for the occurrence of an undesirable outcome in another artifact, e.g. because of mistakes made due to time pressure. The analyst can report on such observations and start thinking of potential ways to improve the process performance.

For a given artifact state co-occurrence relation, the analysis works as follows. By selecting the condition state in the CSM Miner, its corresponding consequences are highlighted. The fill colour of a state indicates the overall strength of the interaction, so grey states can be ignored as they are not very likely to be important causes of performance differences compared to red or green states. If the border of a state is highlighted in yellow then this co-occurrence relation is not an explanation for delays in the condition state. If the border is orange or red then it is correlated stronger with the slow traces or state entries and hence it may be a possible cause of a bottleneck or performance issue. The opposite holds if the border is between yellow and green, indicating a positive correlation to lower sojourn times. After inspecting a given co-occurrence, an analyst will have to determine if there is a logical explanation for a causal relation between the artifact interaction and performance differences. Following this, the analyst should inspect the other co-occurrences with the same condition state to see the related performance differences.

An analyst can also look at the transitions to identify causes of performance differences. Transitions are coloured based on their relative frequency when co-occurring with slow or quickly processed traces and state entries. This can be used to investigate the links between state sojourn times and undesirable transitions or to identify choices that lead to a quicker processing times. As an example, additional lab tests occur more frequently if the treatment of a patient is longer than average. However, this insight is probably well-known to a process analyst, showing that domain knowledge is again required to determine the relevance of the insights.

7 Evaluation

In this evaluation we aim to show that exploring a process using the measures of interest results in relevant artifact interactions and that the proposed focussed performance analysis can help a process analyst to find possible explanations for performance issues. Therefore, we discuss the results we obtained by applying the CSM Miner on real life process data.

7.1 Process Description

We used event data taken from the BPI Challenge 2012 (van Dongen 2012). This dataset concerns process instances of a personal loan and overdraft application process at a Dutch financial institute. The events and activities in the log are related to three interrelated sub-processes, which can be considered as interacting process artifacts. The first artifact concerns the state of the application (*A*-states), the second relates to the work-items performed by the financial institute (*W*-states), and the third concerns the state of a potential offer that the institute can make to the applicant

(*O*-states). This process has been analysed in several other papers (Bautista et al. 2012; Adriansyah and Buijs 2012).

The overall process behavior is as follows. The process starts with the submission of the application. An unlogged check determines whether the application is pre-accepted or declined immediately. The application is accepted once all necessary information has been provided to complete the application. After the acceptance, the institute sends a concrete offer for the terms of the loan or overdraft to the applicant. When the response is returned, the application is validated and then accepted or declined. At any point in the process the applicant can decide to cancel their application and exit the process. In cases where the applicant does not respond in a timely manner, or if the application does not meet the criteria of the financial institute, then the application can be declined by the institute. In exceptional cases the financial institute checks the applications for fraud.

7.2 Results

For the analysis of the described process our primary goal was to identify where in the process potential bottlenecks occur and what may be their root cause. To achieve this, we explored the artifact interaction results first using the measures of interestingness and then by focussing on the manual tasks in the process. We applied focussed performance analysis to identify differences between the traces and state entries with quick processing times versus those that were delayed.

To start the analysis of the artifact interactions, we applied the CSM Miner on the unfiltered data and did not remove or merge any states in the resulting artifact models. To filter the artifact interactions we applied a minimum support of 0.01 to filter out the very infrequent patterns and sorted on *Lift* to find the patterns that are unlikely under assumption of statistical independence. The lift measure expresses the ratio between the probability of co-occurrence and the expected co-occurrence under statistical independence (van Eck et al. 2016a). Note that lift is a symmetric measure, so a co-occurrence between states *A* and *B* has the same lift score independent of which state is the condition and which the consequence. We looked at the top 20 results, shown in Table 2, for further analysis.

For each artifact interaction, we identified whether it is valid and relevant, and what its potential cause could be when looking at performance differences. All artifact interactions in the top 20 are valid, so below we only discuss their relevance and the subsequent analysis to identify performance differences and their possible root causes.

The interaction between *Validating Application* and *Offer Sent Back* is a strong correlation with high confidence, but the sojourn time in *Validating Application* has no significant effect on the interaction strength. This is a good example of an artifact interaction that is valid, but not interesting for an analyst familiar with the process. An application can only be validated if the customer has responded to the offer of the financial institute. The interaction's reverse is a weak correlation where sojourn time is again not a relevant factor for process differences. However, there are other interactions with *Offer Sent Back* that do show such differences.

One such interaction is the reverse of the second co-occurrence in the top list of Table 2, between *Offer Sent Back* and *Calling Incomplete Files*. As shown in Fig. 5a, the *Trace Sojourn Split* mode was used to split the traces into two equal sized sets: the traces that spend most time in *Offer Sent Back* in one set and the remainder in the other set. Set of traces with most time in *Offer Sent Back* are on average spending 41.8% of this time calling or waiting for customers to complete their application. The other traces spend on average 11.6% of their time in *Calling Incomplete Files* at this point. This interaction is relevant and it shows that a possible cause of delays in the decision to accept or decline an application is the significant differences in the time spent calling the customer to provide additional information to resolve incomplete files.

Another relevant artifact interaction is between *Offer Not Started* and *Application Preaccepted*, shown in Fig. 5b. This

Table 2 The top 20 artifact interaction results, sorted on *lift*

Condition	Consequence	Support	Confidence	Lift
W Validating Application	Offer Sent Back	0.013	0.917	6.68
W Calling Incomplete Files	Offer Sent Back	0.049	0.775	5.647
Application Preaccepted	Offer Not Started	0.191	1	4.893
W Not Started	Offer Not Started	0.022	1	4.893
Application Partly Submitted	Offer Not Started	0.012	1	4.893
Application Preaccepted	W Not Started	0.012	0.065	2.912
Application Preaccepted	W Complete Application	0.174	0.91	2.907
W Complete Application	Offer Not Started	0.175	0.56	2.741
W Call After Offer	Offer Sent	0.504	0.872	1.328
Application Finalised	Offer Sent	0.657	0.827	1.259

Lift is a symmetric measure, so the lift score for the reverse of each condition and consequence pair is identical to the 10 interactions shown

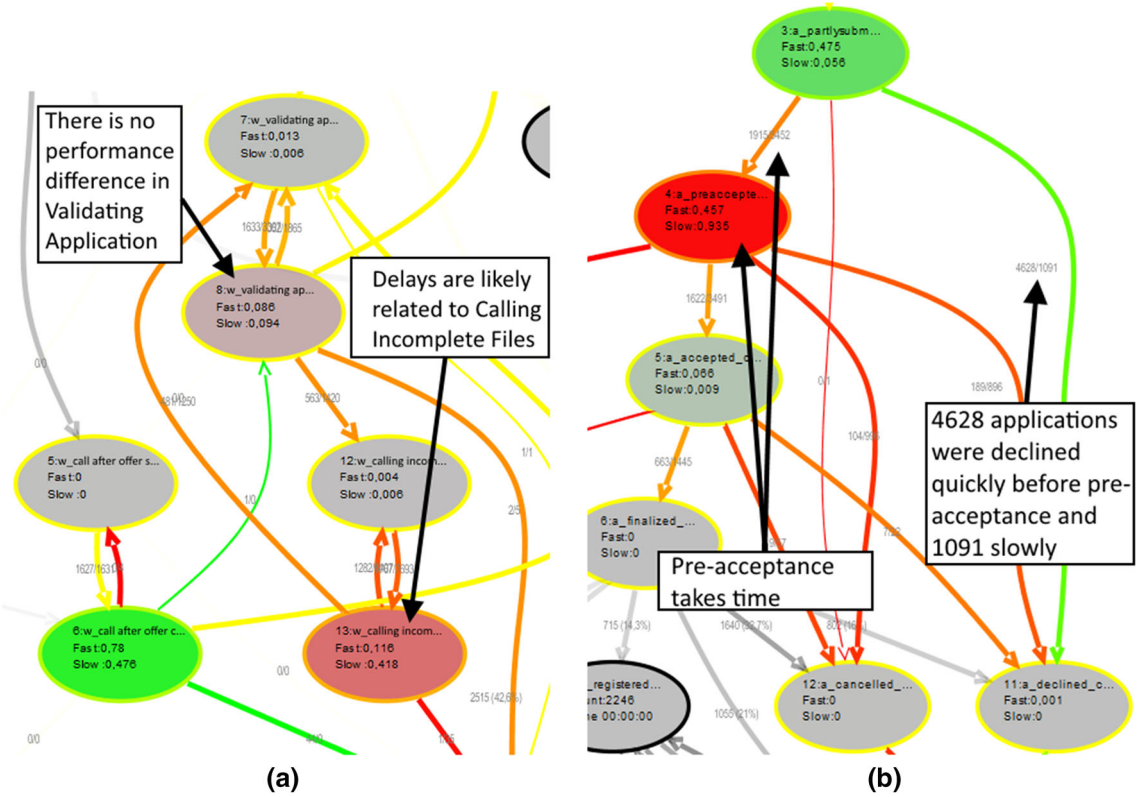

Fig. 5 **a** The co-occurrences in the *Workflow* artifact with *Offer Sent Back*. **b** The co-occurrences in the *Application* artifact with *Offer Not Started* (color figure online)

co-occurrence is stronger if there is a bigger delay before an offer is made to the customer. The cause of this is actually in the state directly before the preaccept, an interaction also in the top list, in which many instances are quickly declined. This is also visible from the transition, showing that 4628 applications got declined there for the half of traces spending least time before an offer was sent, compared to 1091 for the half with the largest sojourn time. Although perhaps not very surprising, it confirms that the automated checks of the financial institute generally remove unsuitable applications quickly and thereby save valuable time to be spent on applications that benefit the organisation.

This initial part of the process also has strong co-occurrence relations with a high lift related to *W Not Started*, as the manual work on the application has not yet started. Once the workflow has started with the *Complete Application* step then we again see strong differences in the process flow for quick and slow traces, as shown in Fig. 6a. This again shows that the majority of the actual working time is spent on cases for which an offer is made rather than on unprofitable cases that are declined.

The interaction analysis also reveals where improvements can still be made, later in the process. The co-occurrence between *Application Finalised* and *Offer Sent*, shown in Fig. 6b, indicates that a major cause of delay in the last stage of the application process is caused by customers responding very slowly to the offers sent out to them. This is not very surprising, however the visualisation also highlights a possible deeper root cause for the slow response of the customers: from the transition frequencies we can see that for the slow applications many offers are cancelled and a new offer is subsequently created. Apparently customers have room to negotiate on the terms of their loan, which leads to delays in the process. As a result of this analysis, a possible follow-up would be an analysis of the benefits of using this strategy or whether it may actually be cheaper to offer better initial terms to customers and thereby reduce the amount of work and long-running applications.

As an alternative to the use of the measures of interestingness, one can also explore the models using domain knowledge and focussing on a common sequence of occurrences. Given that we are interested in finding performance problems, we focussed on the manual parts of the workflow artifact and looked for relevant insights that were not covered above. Regarding the handling of leads, there was interestingly no significant relation between the time spent on this activity and the rate of declined applications, as shown in Fig. 6c. This suggests that there are objective criteria for declining a loan application. The checking of

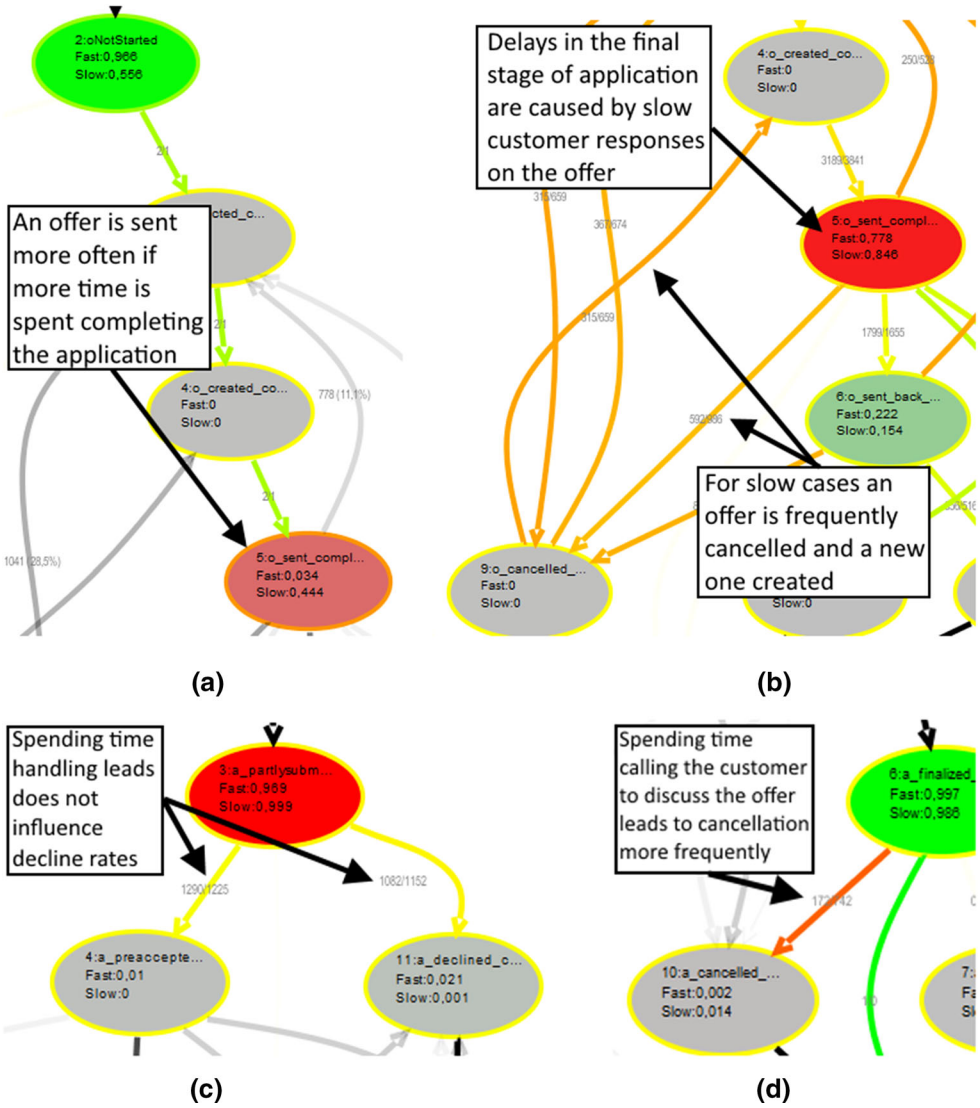

Fig. 6 **a** The co-occurrences in the *Offer* artifact with *Complete Application*. **b** The co-occurrences in *Offer* with *Application Finalised*. **c** The co-occurrences in *Application* with *Handle Leads*. **d** The co-occurrences in *Application* with *Call After Offer*. For **a** and **b** a *Trace Sojourn Split* is made and for **c** and **d** a *State Entry Sojourn Split* to show performance differences

the completion of the application did not reveal interesting insights. However, if the employees spend a lot of time on the phone with a client to discuss the offer then it is more likely that the entire application is cancelled, as shown in Fig. 6d. Further analysis is needed to determine whether this simply means that employees spend a significant amount of time on the phone to try to negotiate with customers and whether this effort is actually worthwhile. The same was true for the time spent calling after incomplete files. Finally, the time spent on the final validation of the application has no significant effect on the acceptance rate.

The above analysis has shown that by just looking at the top artifact interactions as ranked on one of the measures of interestingness, it is already possible to obtain surprising and valuable insights into the performance of this process. Although the user is generally presented with several closely related co-occurrence relations that are not guaranteed to be relevant or interesting for performance analysis, it is much less time consuming than exhaustively exploring the entire set of artifact models. We also found a relevant direction for further analysis by looking at the states in which significant amounts of time is spent on manual work in an average application. This shows that using a guided approach to explore the artifact interactions in a complex process allows an analyst to quickly get insights into the overall process performance.

8 Conclusion and Future Work

In this paper we have presented an approach to objectively quantify the interestingness and performance effects of interactions between artifacts in artifact-centric processes. This approach is based on measures of interestingness that have been defined in the context of process models. It highlights useful or surprising artifact interactions and thereby enables process analysts to deal with large or complex models.

We have also discussed how artifact interactions can be partitioned and compared. Together with a strategy to guided the user in analysing these interactions, the partitioned interactions can be used to identify potential root causes of performance issues in artifact-centric processes.

The approach has been implemented using an interactive process discovery tool, the CSM Miner. We have evaluated the approach and shown that it provides relevant and valuable insights on real life process execution data. However, a limitation of our evaluation is that the insights have only been compared to the results from other authors that analysed the same open dataset, but the results have not been discussed with domain experts. An evaluation with the involvement of domain experts is a direction of future work.

We aim to extend this work in several ways. Instead of only looking at pairs of artifacts, we can also generalise artifact interaction to sets of artifacts. One variant of this is related to having multiple concurrent instances of the same artifact type, e.g. multiple invoices related to one order. Another variant is that conditions over multiple different artifacts can form a more complex interaction, similar to item sets and association rules. There is also room to improve the transformation of execution sequences into observations of artifact interaction. For example, correlations based on time intervals could be used to handle noise or non-fitting executions in the process data. Additionally, we have only explored co-occurrence relations, while long-term dependencies between states are also important in many processes.

Acknowledgements This research was performed in the context of the IMPULS collaboration project of Eindhoven University of Technology and Philips: “Mine your own body”.

References

Adriansyah A, Buijs JCAM (2012) Mining process performance from event logs. In: Business process management workshops – BPM 2012 international workshops, Tallinn, Estonia, 3 Sept 2012, revised papers, pp 217–218

Bautista AD, Wangikar L, Akbar SMK (2012) Process mining-driven optimization of a consumer loan approvals process – the BPIC 2012 challenge case study. In: Business process management workshops – BPM 2012 international workshops, Tallinn, Estonia, 3 Sept, revised papers, pp 219–220

Bazaldua DL, Baker RS, Pedro MOS (2014) Comparing expert and metric-based assessments of association rule interestingness. In: Proceedings of the 7th international conference on educational data mining (EDM) 2014, London, UK, 4–7 July, pp. 44–51

Bolt AJ, van der Aalst WMP, de Leoni M (2017) Finding process variants in event logs (short paper). In: On the move to meaningful internet systems, OTM 2017 conferences – confederated international conferences: CoopIS, C&TC, and ODBASE 2017, Rhodes, Greece, 23–27 Oct, proceedings, part I, pp 45–52

Bose RPJC, Verbeek EHMW, van der Aalst WMP (2011) Discovering hierarchical process models using ProM. In: IS olympics: information systems in a diverse world – CAiSE forum 2011, London, UK, 20–24 June 2011, selected extended papers, pp 33–48

Cohn D, Hull R (2009) Business artifacts: a data-centric approach to modeling business operations and processes. IEEE Data Eng Bull 32(3):3–9

De Weerdt J, vanden Broucke SKLM, Vanthienen J, Baesens B (2013) Active trace clustering for improved process discovery. IEEE Trans Knowl Data Eng 25(12):2708–2720

Liu B, Hsu W, Chen S, Ma Y (2000) Analyzing the subjective interestingness of association rules. IEEE Intell Syst 15(5):47–55

Lu X, Nagelkerke M, van de Wiel D, Fahland D (2015) Discovering interacting artifacts from ERP systems. IEEE Trans Serv Comput 8(6):861–873

Popova V, Dumas M (2013) Discovering unbounded synchronization conditions in artifact-centric process models. In: Business process management workshops – BPM 2013 international workshops, Beijing, China, 26 Aug 2013, revised papers, pp 28–40

Popova V, Fahland D, Dumas M (2015) Artifact lifecycle discovery. Int J Coop Inf Syst 24(1):1550001

Rosa ML, van der Aalst WMP, Dumas M, Milani F (2017) Business process variability modeling: a survey. ACM Comput Surv 50(1):2:1–2:45

Tan P-N, Kumar V, Srivastava J (2004) Selecting the right objective measure for association analysis. Inf Syst 29(4):293–313

van der Aalst WMP (2013) Process cubes: slicing, dicing, rolling up and drilling down event data for process mining. In: Asia Pacific business process management – first Asia Pacific conference, AP-BPM 2013, Beijing, China, 29–30 Aug 2013, selected papers, pp 1–22

van der Aalst WMP (2016) Process mining – data science in action, 2nd edn. Springer, Berlin

van der Aalst WMP, Barthelmess P, Ellis CA, Wainer J (2001) Proclets: a framework for lightweight interacting workflow processes. Int J Coop Inf Syst 10(4):443–481

van Dongen BF (2012) BPI Challenge 2012. Eindhoven University of Technology. Dataset. https://doi.org/10.4121/uuid:3926db30-f712-4394-aebc-75976070e91f

van Eck ML, Sidorova N, van der Aalst WMP (2016a) Composite state machine miner: discovering and exploring multi-

perspective processes. In: Proceedings of the BPM demo track 2016 co-located with the 14th international conference on business process management (BPM 2016), Rio de Janeiro, Brazil, 21 Sept, pp 73–77

van Eck ML, Sidorova N, van der Aalst WMP (2016b) Discovering and exploring state-based models for multi-perspective processes. In: Business process management – 14th international conference, BPM 2016, Rio de Janeiro, Brazil, 18-22 Sept, proceedings, pp 142–157

van Eck ML, Sidorova N, van der Aalst WMP (2017) Guided interaction exploration in artifact-centric process models. In: 19th IEEE conference on business informatics, CBI 2017, Thessaloniki, Greece, 24–27 July, vol 1, conference papers, pp 109–118

Verbeek HMW, Buijs JCAM, van Dongen BF, van der Aalst WMP (2011) XES, XESame, and ProM 6. In: Information systems evolution, pp 60–75. Springer, Berlin

Vogelgesang T, Rinderle-Ma S, Appelrath H-J (2016) A framework for interactive multidimensional process mining. In: Business process management workshops – BPM 2016 international workshops, Rio de Janeiro, Brazil, 19 Sept, Revised Papers, pp 23–35