Mining Process Performance from Event Logs

The BPI Challenge 2012 Case Study

A. Adriansyah and J.C.A.M Buijs

Department of Mathematics and Computer Science
Eindhoven University of Technology
P.O. Box 513, 5600 MB Eindhoven, The Netherlands
{a.adriansyah,j.c.a.m.buijs}@tue.nl

Abstract. Having reliable performance information is often crucial in many business process improvement efforts. In systems where process executions are not strictly enforced by a predefined process model, obtaining such information is difficult. In this paper, we analyzed an event log of a real-life process, taken from a Dutch financial institute, using process mining techniques. In particular, we used the *alignment* technique to gain insights into the control flow and performance of the process execution. We showed that alignments between event logs and discovered process models from process discovery algorithms reveal frequent occurring deviations. Insights into these deviations can be exploited to repair the original process models to better reflect reality. Furthermore, we show that the projection of alignments onto process model provides reliable performance information. All analysis in this paper is performed using existing and dedicated plug-ins within the open-source process mining toolkit ProM.

1 Introduction

The BPM-lifecycle Figure 1 is the leading development model describing the different phases of managing a business process. One of the phases is the diagnosis phase, which focusses on keeping a business process working optimally over time. Before a business process can be improved however, an analysis of the as-is process is required. In the diagnosis phase the recorded history of process executions is analyzed in detail in order to find improvement opportunities. These opportunities can then be implemented in the business process after which the changes can be monitored again for possible further improvements.

Process mining is a special type of data mining, specifically focussed on analyzing historical data of process executions in the form of event logs. Process mining techniques are able to provide insights into the current execution of the business process based on observed facts as recorded in the event log. Process mining techniques exist to *discover* a process model, check the *conformance* of a process model and *enhance* a process model with performance information or animations. Discovering a process model from the observed behavior provides a process model based on the actual observed behavior, without the need to perform interviews. Process model conformance uses the recorded behavior to verify how well the process model conforms with the observed behavior, or vice versa. It also indicates where the actual execution differs from the process model. Enhancement of a process model uses the recorded behavior to project information, such as performance or decision information, on the process model. In this paper we will apply techniques from all three areas of process mining on the challenge event log [10].

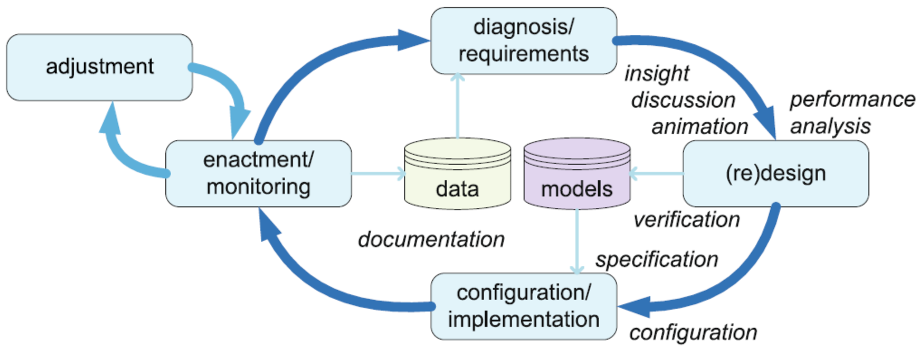


Fig. 1: The BPM life-cycle showing the different uses of process models (from [8]).

In this paper, we campaign an iterative approach based on existing process discovery, verification, and enhancement techniques (e.g. all types) to gain insights into the process execution of a financial institution. As shown in Figure 2, we advocate for an approach where after each phase you might go back one or two phases in order to improve the results obtained so far. In particular, we exploit the *alignment techniques* from [9] to manually improve the automatically discovered process model to better reflect reality. Once we obtained a qualitatively good process model we use the projection of alignments onto the process model to reveal process bottlenecks.

Let us consider the process model, given in Petri net notation [7], as shown in Figure 3. This model allows only for the execution of the traces `ABCD` and `ACBD`. Furthermore, consider the trace `ACCD`. If we replay the trace on the process model then not all movements of events in the trace are allowed by the process model. The goal is to find an optimal alignment between the process model and the trace, that matches as many movements of the process model with movements of the trace. If we try to replay the trace `ACCD`, as shown at the bottom of Figure 3, we obtain the (optimal) alignment as shown in the middle. In the initial state of the process activity A is able to be fired. So, if we try to align the first event of the trace, event A, with the process model both can execute A. This is shown in the alignment by placing an A in the top, as action for the process model, and at the bottom of the alignment, as action for the trace. We mark

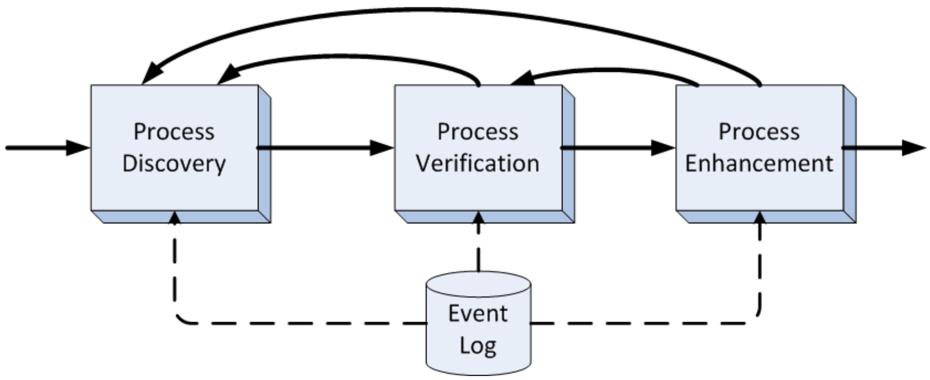


Fig. 2: The three phases of Process Mining and our suggested iterative approach.

this as a green, or synchronous, action since the process model and trace are 'in sync'. The next event in the trace is event C, which can also be executed by the process model, hence C is also marked as synchronous. The process model now has tokens in places p1 and p4 and is only able to execute activity B. The trace however contains event C. We therefore need to fire activity B in the process model without the trace beeing able to follow. We mark this with purple, or as a mode on the process model only. Next, we need to process event C from the trace. We color this with yellow or as a move on trace only.[1] The process model now has a token in places p3 and p4 and is able to follow event D from the traces, therefore a synchronous move. In the end we obtain an alignment with 3 synchronous moves (A, C and D), one model move (B) and one log move (C). The fitness of a trace on a given process model is calculated by dividing the total costs by the total costs of aligning an empty trace. In this example, assuming all costs are set to one, the fitness is $1 - \frac{4}{8} = 0.5$.

In this paper we use the open-source process mining toolkit ProM [11], version 6.1. We use both existing as dedicated plug-ins for the analysis based on alignments.

This paper is structured as follows. In section 2 we globally analyze the event log. We provide some default statistics but also discuss performance related observations without the use of a process model. section 3 discusses the discovered process models, and how we obtained them, for the different views on the event log. Then, in section 4 the process models are enhanced with performance information, showing where bottlenecks in the processes are. section 5 concludes the paper and summarizes our main findings.

It should be noted however that this paper is intended as an initial process mining analysis. Since we have very limited background knowledge about the

[1] Please note that we could also have chosen to first perform the C of the trace and then the B of the process model. In this case those alignments are equal.

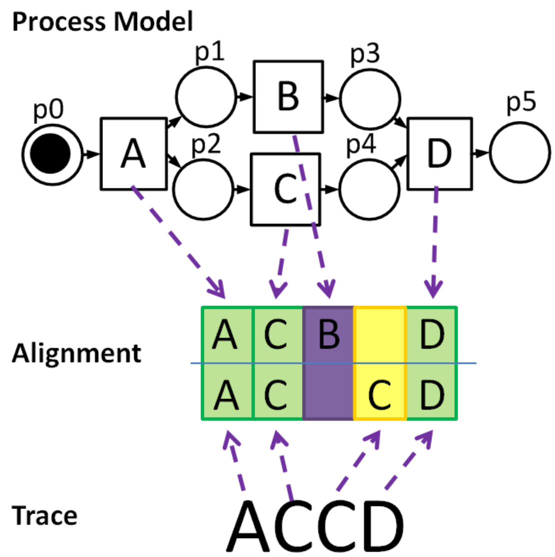


Fig. 3: Example of verification of a process model (top) and trace (bottom) using alignments (middle).

actual process, and no option to verify our findings with the process owner, we will only make observations.

2 Global Analysis

The event log provided for this challenge [10] is taken from a Dutch financial institution and describes applications for personal loans or overdraft. According to the description provided on the challenge website [1], the event log contains events from three intertwined subprocesses, which can be distinguished by the first letter of each event name (A, O and W). The A subprocess is concerned with handling the applications themselves. The O subprocess handles offers send to customers for certain applications. The W process describes how work items, belonging to the application, are processed.

A global overview of the event log characteristics is shown in the ProM dashboard in Figure 4. The event log contains 13,087 traces and 262,200 events, recorded from October 1, 2011 to March 14, 2012. The dashboard also shows that, although on average 20 activities are recorded per case, the distribution of the number of recorded events per case varies greatly. There are quite some cases with only a few recorded events (only 3) while there are also other cases for which a lot of events are recorded, up to 175. There are 36 distinct event classes spread across 3 event types. As provided in the description, for subprocesses A and O only the event type 'complete' is present, indicating that a task is

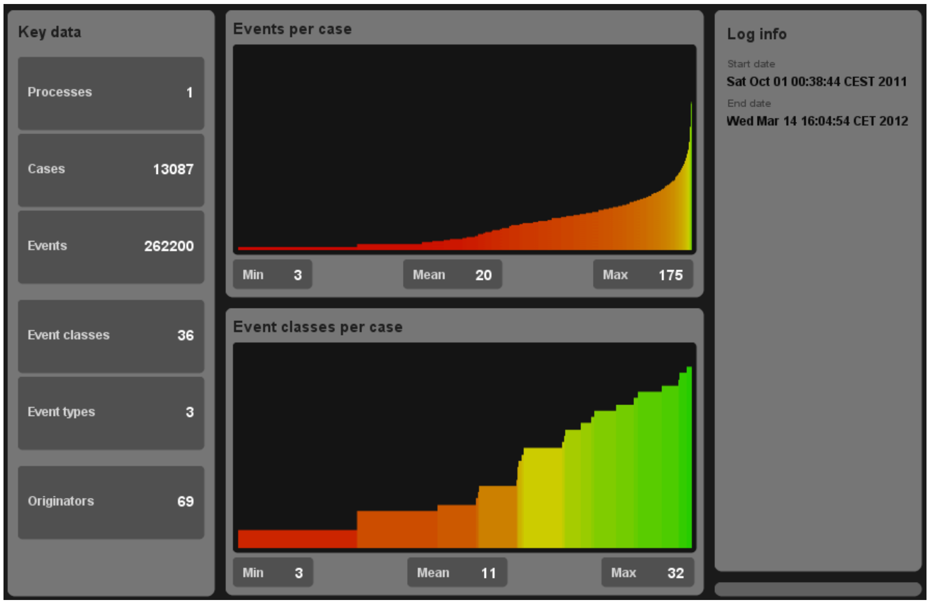


Fig. 4: ProM Dashboard overview of the original event log.

completed. For the W subprocess however work items can be created in the queue ('schedule' event type), obtained by the resource ('start') and released by the resource ('complete'). Looking at the distribution of event classes over the cases we see a similar pattern as for the events, where only a few event classes are executed for some cases while for others a lot of different event classes have been recorded. The event log spans a period from October 1, 2011 until March 14, 2012 in which 69 different originators were observed.

Using the ProM log summary (see Table 12, 13 and 14 in the appendix) it is clear that *every* case starts with the activity 'A_SUBMITTED' which is always executed by user '112'. This resource is the only one executing all occurrences of this activity and the activity 'A_PARTLYSUBMITTED'.

The last case is started on February 29, 2012 which means that 13,087 cases were received within a 152 day period. This means than on average 86 new applications are received per day, including the weekend and holidays.

Resource investigation using ProM 5.2's *Basic Performance Analysis* plug-in shows that some activities were only performed by specific resources. For example, both activities 'A_SUBMITTED' and 'A_PARTLYSUBMITTED' were always performed by resource '112'. Some resources are specialists as they only performed specific activities in the process. For example, resources '10125' and '10821' only performed activity 'W_Valideren aanvraag', and resource '11254' only performed activity 'W_Completeren aanvraag'. Resource '11304' is possibly an auditor, as he only performed activity 'W_Beoordelen fraude'. Resource

'10188' performed 'A_DECLINED' in most cases, but it also performed a small number of 'W_Beoordelen fraude'.

2.1 Performance information based on Event Log only

We visualized the elapsed time between first occurrences of activities in a *Performance matrix*. In this matrix the color of the cells indicates the average elapsed time between the first occurrences of that pair of activities, as compared to other pairs. Green implies a relatively low value and red implies a high value, where yellow implies a value in between. Cells that have a blue dot in it have a duration below a specified threshold. Part of the performance matrix showing the average elapsed time between activities for this event log is shown in Figure 21 in the appendix. From this matrix Table 1 is derived, showing those activity pairs with very short elapsed time in between them. For instance the activities 'A_SUBMITTED' and 'A_PARTLYSUBMITTED', which are always executed by user '112', are executed within half a second of each other. We know that 'A_SUBMITTED' always starts the process and the process description explicitly mentions that the first activities are automated. Therefore user '112' is likely to be an system or application user. From the performance matrix it also became clear that 'A_ACTIVATED', 'A_APPROVED', 'A_REGISTERED' and 'O_ACCEPTED' are all executed within a millisecond after each other. Other activities that are recorded within less than 1 millisecond after each other are: 'A_DECLINED' and 'O_DECLINED', 'A_CANCELLED' and 'O_CANCELLED', 'O_CANCELLED' and 'O_SELECTED' and 'A_FINALIZED' and 'O_SELECTED'. Other interesting patterns are activities 'O_CREATED' and 'O_SEND' beeing executed 59.3 milliseconds after each other.

The performance matrix also shows that some scheduling of activities in process W is synchronized with the completion of activities in process O. In 5,015 cases, completion of both activities 'O_CREATED' and 'O_SENT' occurred soon after/before scheduling of activity 'W_Nabellen Offertes' (with average elapsed time less than 0.5 seconds). This indicates that these two activities trigger 'W_Nabellen Offertes'. Other activities in process W that are triggered in a similar way is the scheduling of 'W_Valideren aanvraag' in 3,254 traces (triggered by activity 'O_SENT_BACK') and scheduling of 'W_Completeren aanvraag' in 7,365 traces (triggered by activity 'A_PREACCEPTED').

3 Process Discovery

One of the main powers of process mining is the discovery of process models from recorded behavior [8]. However, discovering a process model that describes the behavior well is not an easy task. Especially for more complicated behavior most algorithms fail in creating a process model that is able to replay the observed behavior, does not allow for unseen behavior and is simple enough to be understood. However, a good process model is needed for further analysis of the event log, such as performance analysis (see section 4).

Table 1: Elapsed time and number of occurrences of selected activity pairs.

First Activity	**Followed by**	**Elapsed time**	**#Cases**
A_SUBMITTED	A_PARTLYSUBMITTED	581.67 ms	13,087
A_PREACCEPTED	W_Completeren aanvraag (schedule)	521.63 ms	7,365
O_CREATED	O_SENT	59.3 ms	5,015
O_SENT	W_Nabellen offertes (schedule)	205.66 ms	5,014
A_FINALIZED	O_SELECTED	0 ms	4,289
O_SENT_BACK	W_Valideren aanvraag (schedule)	319.54 ms	3,254
A_APPROVED	A_REGISTERED	0.69 ms	2,246
A_REGISTERED	A_ACTIVATED	0.09 ms	2,246
O_ACCEPTED	A_REGISTERED	0.97 ms	2,243
A_ACTIVATED	A_REGISTERED	0 ms	2,050
A_APPROVED	O_ACCEPTED	0 ms	1,698
A_REGISTERED	A_APPROVED	0 ms	1,662
A_ACTIVATED	A_APPROVED	0 ms	1,466
A_CANCELLED	O_CANCELLED	0.07 ms	1,222
A_REGISTERED	O_ACCEPTED	0 ms	1,126
O_CANCELLED	O_SELECTED	0.07 ms	1,020
A_ACTIVATED	O_ACCEPTED	0 ms	938
A_DECLINED	O_DECLINED	0.03 ms	800
O_DECLINED	A_DECLINED	0 ms	779

This section discusses the results of some popular process discovery algorithms and how we improved the discovered process models. In Section 3.1 we first describe our approach to obtain good quality process models. Then in Section 3.2 we describe the final process models obtained for each of the subprocesses. In Section 3.3 we describe the process models for the different trace types that are present in the event log.

3.1 Process Discovery Approach

Discovering a process model from the observed behavior recorded in the event log is not a trivial task. Several process discovery algorithms exist but for more complex processes, such as the one under investigation, they do not provide process models of enough quality. However, for many types of analysis, such as the ones we present later on, a good quality process model is mandatory. However, the algorithms do provide a good starting point for manual improvement. We execute the following steps to get to a good quality process model:

1. **Log Filtering.** First the event log needs to be filtered, both for the specific view taken but also to include only complete cases. To improve the results of the process discovery algorithms, artificial start and end events are added to the trace, if required. Based on the Dotted Chart, as shown in Figure 5 it appears that cases starting in 2011 are likely to be finished while cases starting in 2012 might not be. Therefore we run the process discovery algorithms

on event logs where each trace start and ends with an artificial activity and only contains cases starting in 2011.

2. **Initial Process Model Discovery.** Based on earlier experiences in [6] we selected three well-known process discovery algorithms: the *α-algorithm* [2], the *heuristics miner* [12], and the *ILP-Miner* [13, 14]. Each of these algorithms produce results that are in, or can be translated to, the Petri net [8] modelling notation. This allows us to use the process model for further detailed analysis of the event log.

3. **Evaluate Quality of Discovered Models.** Each of the automatically discovered models is evaluated on three of the four process quality dimensions described in [6]: fitness, precision, and simplicity. The *(Replay) Fitness* [3, 9] is a measure of how well the process model is able to replay, or follow, the behavior observed in the event log. The other dimension, *precision* [4, 9] indicates how much additional behavior the process model allows that is not seen in the event log. Although fitness is the most important quality dimension, precision makes sure that the process model is still specific enough for the event log under investigation. The quality dimension of simplicity prefer simple models to complex ones. This dimension is not measured but taken into account while manually improving the process model. Generalization is, for our specific purpose, not of importance. Please note that the quality is measured on the event log with all traces, so also those starting in 2012, and while ignoring the artificial start and end activities.

4. **Improve Process Model Manually Using Alignment with Event Log.** To calculate the replay fitness, the events of the traces in the event log are related to the activities in the process model. This is called an alignment between the event log on the one hand and the process model on the other hand. This alignment indicates where the process model is not able to follow the traces in the event log. This allows us to repair the process model using two phases:

   (a) **Fix common deviations.** In several iterations severe deviations between the process model and event log are fixed by adding extra tasks to the process model for accommodating the recorded behavior. After each repair the alignment is again calculated and new severe deviations are fixed until the process model is of enough quality.

   (b) **Clean-up model.** Once the deviations are repaired, the process model is cleaned-up by removing unnecessary and unused transitions which simplifies the process model and thus makes it easier to read and interpret.

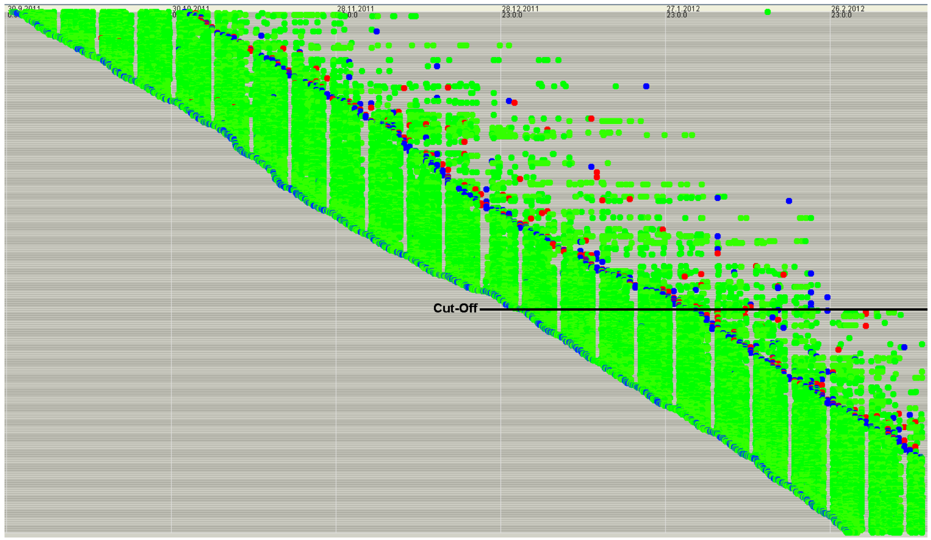


Fig. 5: Dotted Chart of the original event log where activities in process A are colored blue, O are colored red and W are colored green. The line indicates the cut-off point between those cases starting in 2011 and those starting in 2012.

3.2 Process Models per Subprocess

As indicated in the description of the event log on [1], the process consists of three sub processes: A, O and W. The statistics of the original event log and the event logs with only events related to each subprocess is shown in Table 2. In this section the process models for each of the sub processes are shown and a discussion on how the models were obtained is provided.

Table 2: Statistics for the different event logs (original, subprocess A, subprocess O and subprocess W).

	Original	**Subprocess A**	**Subprocess O**	**Subprocess W**
#Traces	13,087	13,087	5,015	9,658
#Events	262,200	60,849	31,244	170,095
#EventClasses	36	10	7	18
#Resources	69	61	60	60
First event	October 1, 2011	October 1, 2011	October 1, 2011	October 1, 2011
Last event	March 14 ,2012	March 14 ,2012	March 14 ,2012	March 14 ,2012

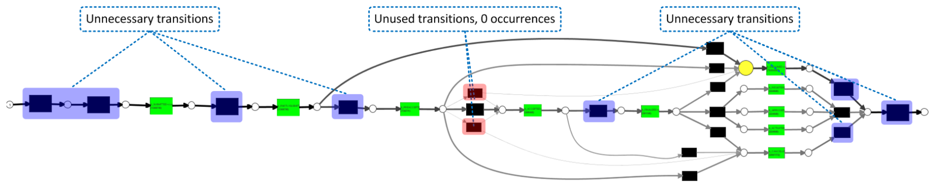
Subprocess A: Loan Applications Applying the approach described above results in process models for subprocess A with characteristics as shown in the top 2 rows of Table 3. The α-algorithm results in a model where the final marking is not reachable resulting in a fitness of 0 and hence the process model is not usable. The ILP-miner results in a process model where most transitions are not restricted in firing, hence precision is bad. The result of the heuristics miner is rather good and is shown in Figure 6a. Checking conformance revealed only minor deviations from the process model, as is indicated by a fitness of 0.99. The only thing necessary to improve this model is to clean it up by removing 8 unnecessary transitions and 2 unused transitions. After this clean-up the process model as shown in Figure 6b is the result. Projection of alignments between all traces in the event log of subprocess A onto the result process model yields a visualization that shows location of deviations as shown in Figure 6b. Deviations of this process model only occur in 4 locations: 'A_DECLINED' is skipped 327 times, 'A_CANCELLED' is skipped 72 times, 'A_ACCEPTED' is executed before 'A_DECLINED' 29 times and there is one occasion where 'A_PREACCEPTED' is skipped. These deviations are however minor and do not warrant further modification of the process model.

The resulting process model shows a nice sequence of the activities 'A_SUBMITTED', 'A_PARTLYSUBMITTED', 'A_PREACCEPTED', 'A_ACCEPTED' and 'A_FINALIZED'. After the activities 'A_PARTLYSUBMITTED', 'A_PREACCEPTED' and 'A_FINALIZED' there is the option to end the process by executing 'A_DECLINED'. After the activities 'A_PREACCEPTED', 'A_ACCEPTED' and 'A_FINALIZED' there is the option to end the process by executing 'A_CANCELLED'. If this is not the case then the application is approved by executing the activities 'A_APPROVED', 'A_REGISTERED' and 'A_ACTIVATED' in parallel after which the process ends.

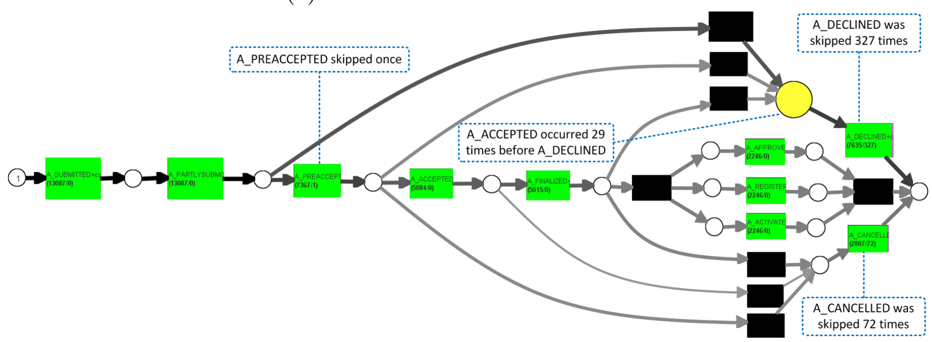
Table 3: Quality metrics for different process models for all subprocesses.
[1] Calculation of both fitness and precision are skipped as the calculation takes more than 1 hour and the model is already too complex to be improved.

Process	**Metric**	**α-algorithm**	**Heuristics Miner**	**ILP Miner**	**Final Model**
A	**Fitness**	0.00	0.99	1.00	0.99
	Precision	0.87	0.93	0.53	1.00
O	**Fitness**	0.00	0.80	1.00	0.99
	Precision	1.00	0.79	0.48	0.99
W	**Fitness**	-[1]	0.77	1.00	0.99
	Precision	-[1]	0.81	0.11	0.77

Subprocess O: Loan Offers The process model resulting from the Heuristics miner provided the best process model to start with for subprocess O, as shown in Table 3. Figure 7a shows the alignments between traces of the event log and the model, projected onto the model. Using this visualization we can identify some clear deviations. For instance, the activity 'O_CANCELLED' was



(a) Result of the Heuristics Miner



(b) Result after removing unnecessary transitions.

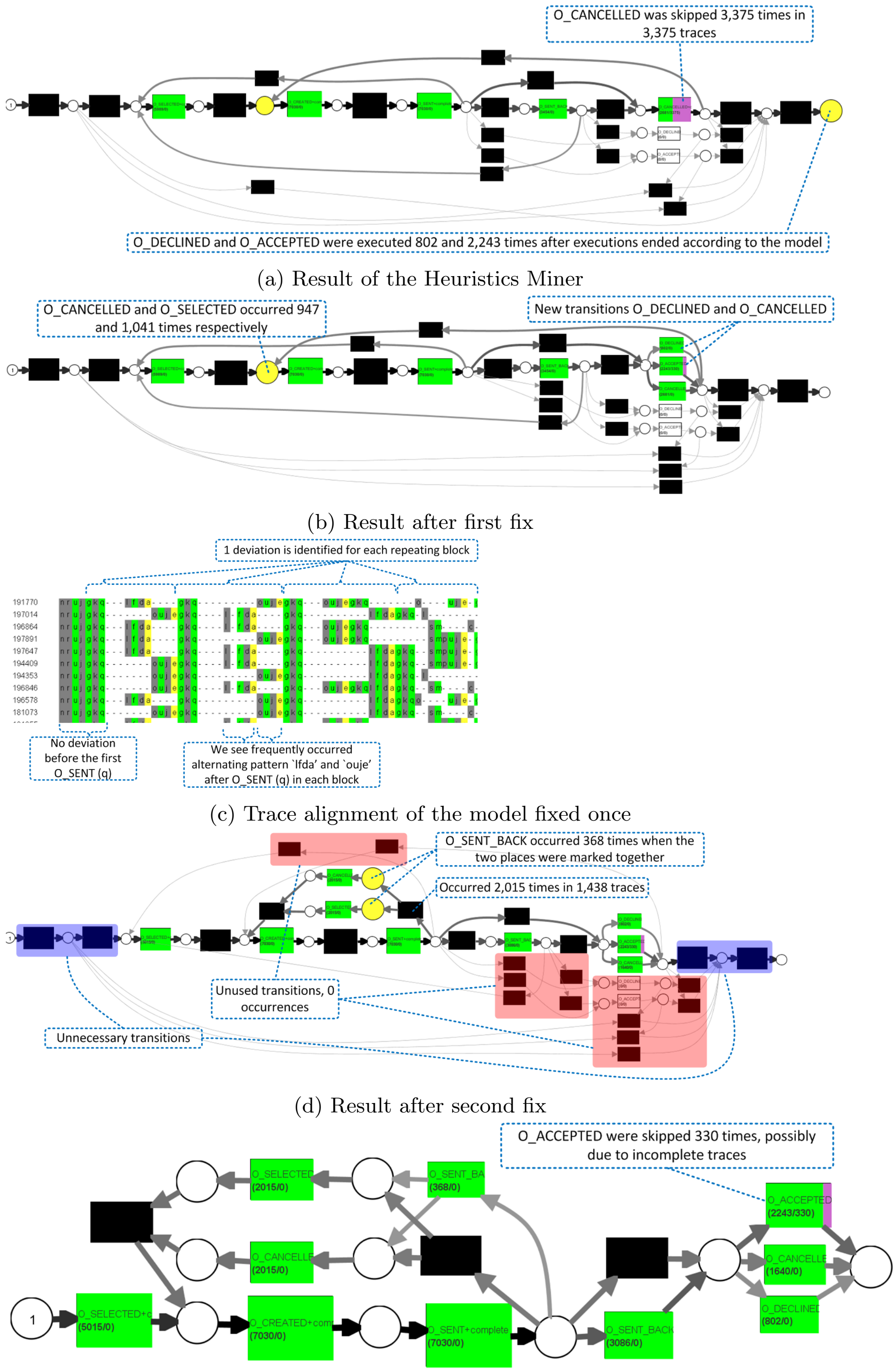
Fig. 6: Process models discovered and fixed for process A.

skipped 3,375 times in total. Furthermore, at the end of the process model we see frequent executions of 'O_DECLINED' and 'O_ACCEPTED'. Fixing these deviations is not difficult and the resulting process model is shown in Figure 7b. By creating a choice between the activities 'O_CANCELLED', 'O_DECLINED' and 'O_ACCEPTED' we improved the both the replay fitness and precision of the original model.

However, after this initial modification there are quite a few deviations in the place before activity 'O_CREATED'. Using the trace abstractions [5] visualization of all alignments between traces in the log and the model, as shown in Figure 7c, several patterns can be discovered. In this case, it appeared that in the loop back after 'O_SENT' to 'O_CREATED' activities 'O_CANCELLED' and 'O_SELECTED' are executed in parallel. Applying this fix leads to the process model as shown in Figure 7d.

In this process model there is still a deviation namely the execution of the activity 'O_SENT_BACK' in some cases. Furthermore, we can also remove unnecessary transitions, marked blue, and unused transitions, marked red, to clean up the model. this results in the process model as shown in Figure 7e, which is our final version. The only deviation that is still present is activity 'O_ACCEPTED' being skipped 330. This can be explained by the fact that some cases might still be running.

The resulting model of Figure 7e allow different behavior than the one allowed by the model found by the Heuristics miner. By improving the process model manually we were able to increase replay fitness from 0.80 to 0.99 and precision



(a) Result of the Heuristics Miner

(b) Result after first fix

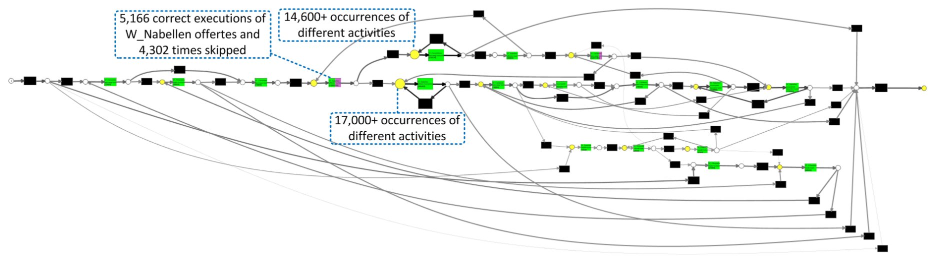
(c) Trace alignment of the model fixed once

(d) Result after second fix

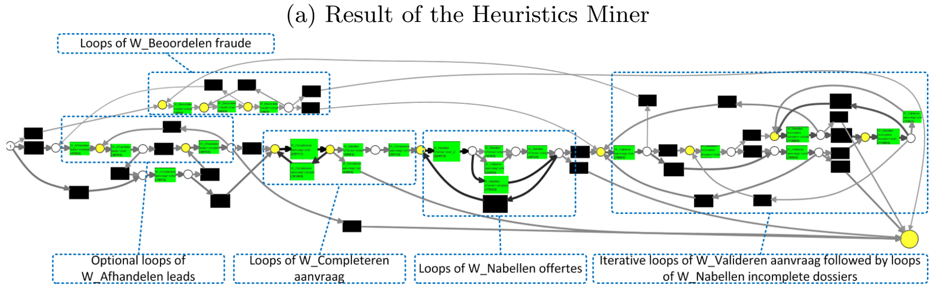
(e) Result after second fix

Fig. 7: Intermediate and final process models for process O

from 0.75 to 0.92. We now have model that is of high enough quality to be used for performance measurements, as will be discussed in Section 4.2.



(a) Result of the Heuristics Miner



(b) Result after final fix

Fig. 8: Discovered and final process models for process W

Process W: Work Items The process model of subprocess W is more complicated than that of subprocesses A and O. None of the models are readable, i.e. they all have very low simplicity. The α-algorithm results in a process model that has unconnected nodes. The ILP miner discovered a process model that does have a perfect fitness (by design) but the number of incoming and outgoing arcs in its transitions make it unreadable. The only algorithm that provides a reasonable model is the heuristics miner, which is shown in Figure 8a. However, this model still has problems, some of which are indicated in Figure 8a. It took 8 iterations to improve this model. These fixes contained adding loops of start and complete of several activities, adding loop back and skip opportunities, and several iterations of removing unused and infrequent transitions. This resulted in the process model as shown in Figure 8b. This process model is far better than the original, as shown by the figures in the last column of Table 3, although it is not really precise due to the many (necessary) loops.

The process model mainly contains of several loops of the different event types of a certain activity. Many of the loops allow to first execute the 'schedule' type of the activity followed by loops of the sequence of 'start' and then 'complete'. The main part of the process allows the workitems 'W_Afhandelen leads', 'W_Completeren aanvraag', 'W_Nabellen offertes' to be handled in sequence, with the loops of the different event types. After that alternative executions of the loops for activities 'W_Valideren aanvraag' and 'W_Nabellen incomplete dossiers' follows. There is also the option to execute 'W_Beoordelen fraude' but this also happens on some other places in the process.

3.3 Process Models for different Trace Types

Although three different subprocesses can be identified, each subprocess is not executed for all cases. The relation between traces that contain activities from subprocesses A, O, and W is shown in the Venn diagram in Figure 9. Activities that belong to process A occurred in all 13,067 traces. In 9,658 of these traces, the trace also contains activities that belong to process W. In all of the 5,015 traces where O occurred, activities that belong to process A and W also occurred. This indicates a clear hierarchical relation between the three subprocesses. Therefore we split the original event log in three sublogs: one that contains traces that only contains events from subprocess A, a second sublog that contains traces with only activities from subprocesses A and W (and not O), and a third sublog with traces that contain activities from all three subprocesses. More details about the size of these three sublogs is shown in Table 4. In this section the discovered process models for each of these sublogs are discussed.

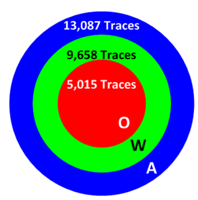


Fig. 9: Venn diagram showing the relation between traces that contain activities from process A, process O, and process W.

Table 4: Statistics for the different trace types.

	Original	**Only A**	**Only A and W**	**A, W and O**
#Traces	13,087	3,429	4,643	5,015
#Events	262,200	10,287	54,161	197,752
#EventClasses	36	3	13	36
#Resources	69	1	63	68
First event	October 1, 2011	October 1, 2011	October 1, 2011	October 1, 2011
Last event	March 14 ,2012	February 29, 2012	March 14 ,2012	March 14 ,2012

Table 5: Quality metrics for different process models for all trace types.
[1] Calculation of both fitness and precision does not terminate because of unconnected nodes.

Process	Metric	α-algorithm	Heuristics Miner	ILP Miner	Final Model
Only A	**Fitness**	1.00	1.00	1.00	1.00
	Precision	1.00	1.00	1.00	1.00
Only A and W	**Fitness**	0.18	0.90	0.99	1.00
	Precision	0.50	0.87	0.27	0.87
A, W and O	**Fitness**	-[1]	0.81	-[1]	0.99
	Precision	-[1]	0.62	-[1]	0.61

Cluster 1: Traces with activities from process A and not W or O In traces where activities of process A occurred but no activities of process W and O occurred, the process is a sequence of activities 'A_SUBMITTED', 'A_PARTLYSUBMITTED', and 'A_DECLINED' (see Figure 10). We confirmed that this is the case for all traces by aligning all traces in the original log that contains activities of subprocess A, but not subprocess W or O, to the sequential model. The alignments of the traces, projected onto the model (see Figure 10), shows that this is indeed true for all traces. All traces that follow this process are declined and, as discussed in section 2, handled automatically.

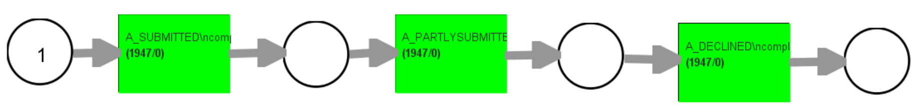
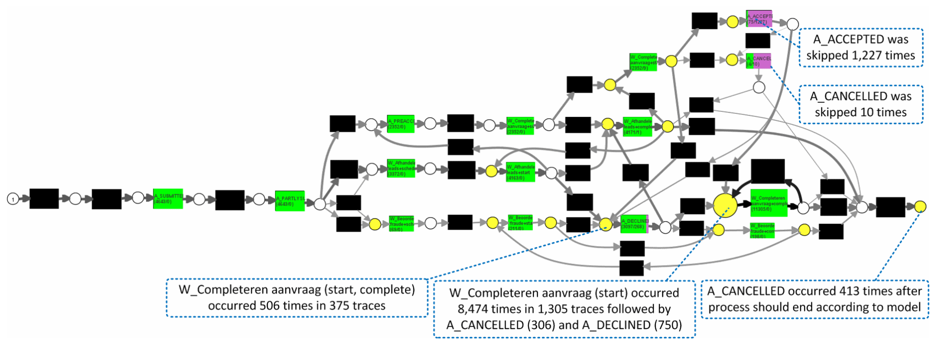


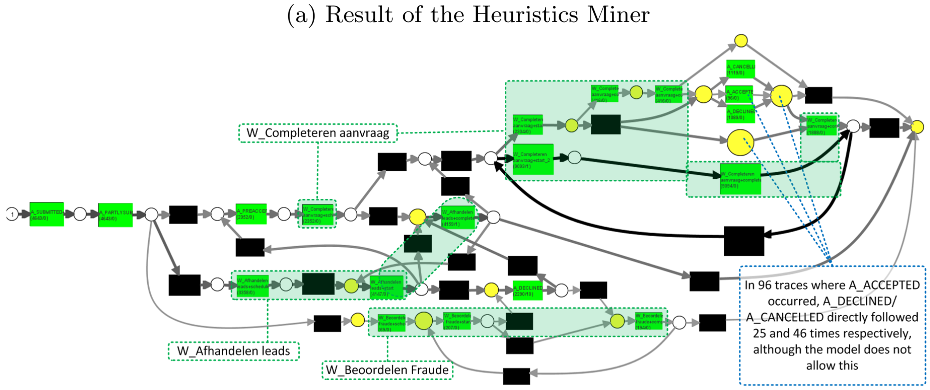
Fig. 10: Discovered Process Model for traces where only activities from process A occurred

Cluster 2: Traces with activities from processes A and W and not O The process behind the execution of traces with activities from subprocesses A and W but not O is more complicated. According to the fitness and precision metrics on the process models discovered by the process discovery algorithms no fitting and precise model is found. Again, the process model found by the heuristics miner, see Figure 11a, seems to be the best basis for manual improvement. After 7 improvement rounds the process model as shown in Figure 11b was obtained. Improvements included addition of duplicate instances of certain activities and the option to repeat or skip certain activities. And, of course, the removal of unnecessary and unused transitions.

The final process model is the most complex one found so far and allows for accepting (observed 96 times), cancelling (observed 1,119 times) and declining (observed 3,379 times in total, in two locations in the model) an ap-



(a) Result of the Heuristics Miner



(b) Result after final fix

Fig. 11: Discovered and final process models for traces with activities from A and W and not O.

plication. However, this decision is not always made as sometimes the process can already finish after the execution of 'W_Afhandelen leads', which was observed 2, 235 times. In general, the process starts with 'A_SUBMITTED' and 'A_PARTLYSUBMITTED' after which 'W_Afhandelen leads' or 'W_Beoordelen fraude' can be started. If 'W_Beoordelen fraude' is actually executed, which is the 'complete' type, the process has the option to stop, which happened 56 times, out of 194 times in total. Furthermore, all three types of 'W_Beoordelen fraude' occur regularly in other states of the process where the model actually not allows it.

A second part in the process, as is shown in the top right of Figure 11b, completes the application by executing 'W_Completeren aanvraag' and then either cancelling ('A_CANCELLED') or declining ('A_DECLINED') the application. Although the activity 'A_ACCEPTED' is recorded 96 times, after each occurrence the activity 'A_CANCELLED' or 'A_DECLINED' appears effectively annulling the acceptance of the application. Therefore we can conclude that traces where activities from subprocesses A and W are observed, but not from O, are either cancelled or declined and never accepted and finalized.

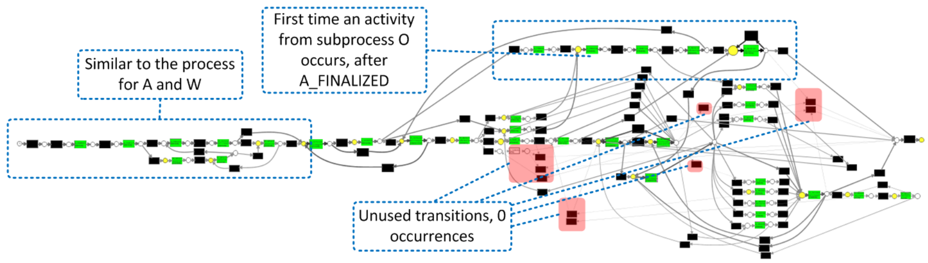
Cluster 3: Traces with activities from processes A, W, and O Although expected to be rather easy, considering that the process model for the O subprocess was rather simple, adding the O subprocess to the process model of A and W was the most difficult of all. The process models found by the α-algorithm and ILP miner made no sense. The process model found by the Heuristics miner, as shown in Figure 12a has a very low fitness to be directly used. After several iterations of fixing the model based on alignment projected onto the original model, we finally obtained (Figure 12b) a qualitatively good enough process model requiring 8 rounds of fixing. Again, fixing meant adding loops, possibilities to skip certain parts and adding more instances of certain activities. However, in the end we managed to obtain a model with a fitness of 0.99 and a precision of 0.61.

In the obtained process model one can recognize parts of the different subprocesses and how they interact. For instance the initial part of this process is similar to the process model discovered for the traces with only activities from A and W. An interesting discovery is that there seem to be loops of 'W_Nabellen offertes' with different characteristics. In the part of the process marked with 1 in the picture for instance, it is followed by 'O_SENT BACK' or a reselection ('O_CANCELLED' and 'O_SELECTED' are executed in parallel). In the part indicated with 2, the loop of 'W_Nabellen offertes' is followed by 'O_SENT BACK' and a (re)scheduling of 'W_Valideren aanvraag'. The part indicated with 3 contains only loops of 'W_Nabellen offertes'. In the part indicated with 4, 'W_Nabellen offertes' is followed by the activities 'O_CANCELLED' and 'A_CANCELLED', executed in parallel. Furthermore, in the center of the process we see a loop of the 'W_Nabellen incomplete dossiers' work item. We can also see that decisions (accept, decline or cancel) are only made after the execution of the work item 'W_Valideren aanvraag'.

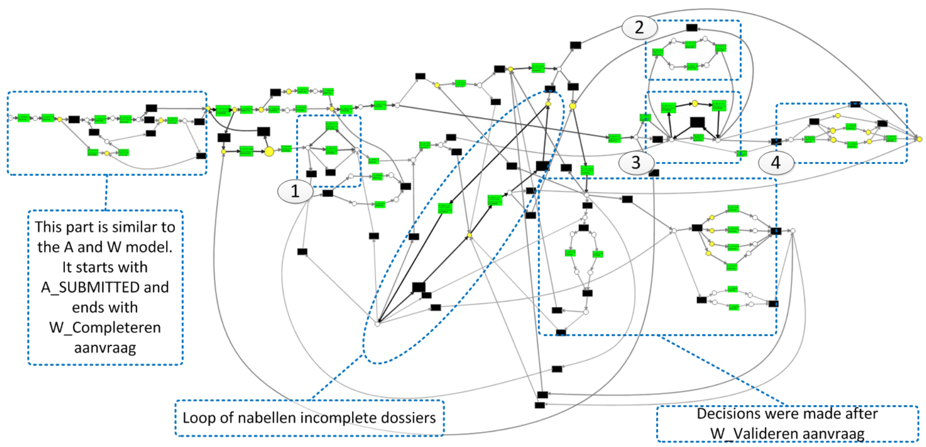
In this section, we showed that visualization of alignments, i.e. projection of alignments onto process model and trace alignment of all alignments, shows locations of deviations that are useful to repair process model to better reflect the reality in event logs. In section 4, we use the discovered process models to obtain insights into the performance of process executions, including batched activities, bottleneck analysis, and synchronization.

4 Performance Analysis

The goal of performance analysis is to find possible bottlenecks in process executions. To avoid misleading insights, performance is measured without considering activity executions that deviate from the discovered process models. We use alignments [9] between execution traces in the event log and the process models to identify non-deviating activity executions as much as possible and use them to measure performance of process executions. The obtained information is projected back onto the models to provide a way of visual analysis of performance.



(a) Result of the Heuristics Miner



(b) Result after final fix

Fig. 12: Discovered and final process models for traces with activities from A and W and O.

In this section we show performance-related findings on the different subprocesses as discussed in the previous section. This analysis is based mainly on the visual projection of performance information constructed using alignments.

Performance analysis focusses on the time aspect of process executions. By replaying the executed traces on the process model, timing information of the different steps in the process become available. This helps indicating bottlenecks in the process. We color the waiting times accordingly such that (relative) bottlenecks in the process are colored red. Parts of the process with moderate waiting times are colored yellow and those with relatively short waiting times are colored green.

Since only timing information of traces that can actually be replayed on the process model can be taken into account, the fitness of the process model w.r.t. the event log needs to be high. Therefore we spend much effort, as described in the previous section, to obtain process models with a fitness of at least 0.99. If the fitness would be low than the further we get 'into' the process (e.g. the further from the initial place), the lower the number of traces that can actually be replayed by the process model. This would mean that the number of actual measurements we use for our reported statistics reduces quickly and the results become untrustworthy and therefore unusable very quickly. Because we have good fitting models only few traces are 'lost' hence all performance information is reliable.

In this section we describe, for each of the three subprocesses and each of the three trace types, performance related measurements. It should be noted however that in general we can only measure the time *between* executions of activities (and not the execution time itself). Therefore, we can only indicate how long traces, on average, has waited in certain states of the process until the next activity was executed. We provide the average waiting time, together with a 99% confidence interval. This interval indicates our level of certainty of the average. If we report an average of 10 ± 1 days for instance, we mean that the true average is expected to be between 9 and 11 days, with a 99% confidence. In general we only report the average durations and not the minimum and maximum observed durations since these might be single rare occurrences which do not provide much information on the overall execution of the process.

4.1 Process A: Loan Applications

The process model of subprocess A with performance information projected onto it is shown in Figure 13. It can be immediately seen that there are two bottlenecks. The most severe one, with an average time of 17.99 ± 0.27 days after the previous activity recording, is the cancelation of loan applications. If an application is accepted, registered and activated it happens 16.00 ± 0.50 days after beeing finalized. Loan applications that are declined wait on average 1.92 ± 0.17 days before the decision is recorded. These figures indicate that a decision to decline a loan application is made much quicker than the decision to cancel or accept.

The beginning of the process, with the execution of 5 activities in a sequence, is executed quickly with waiting times in the order of minutes or hours. The first two activities are executed automatically after each other as is indicated by the average weighting time in between of 581.67 ± 27.92 milliseconds. The average waiting time of 2.08 ± 0.20 hours before an application is preaccepted does not clearly indicate whether that action is performed automatic or manual. The waiting time after preaccepted and before accepted of 18.74 ± 1.61 hours indicates it might be a manual activity but probably with a high priority. The time after accepting and before finalizing the application of 9.42 ± 4.82 minutes indicates that this is a manual step, suggested by the high confidence interval.

Table 6 shows that on average a case takes 8.08 ± 0.27 days to go through the complete application subprocess. However, if we split the cases on their outcome, e.g. approved, declined or cancelled, we can see some clear differences. A case that is declined takes on average 2.05 ± 0.18 days while a case that will be cancelled takes 18.6 ± 0.71 days. This is only slightly longer than a case that will be approved, which takes 16.7 ± 0.51 days on average. This indicates that a case can be declined quickly, within roughly 2 days, but the decision between

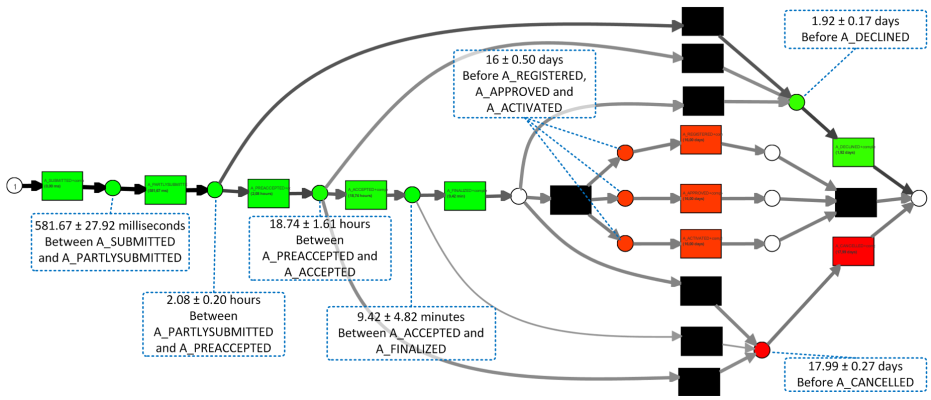


Fig. 13: Performance projection on the loan application process.

Table 6: Case types and their durations for loan applications. **merge avg. and 99p. cols (as in text)**

Cases	Freq	Min	Max	Average	99% conf. interval
All	13,087	1.86 seconds	3.05 months	8.08 days	0.27 days
Declined	7,635	1.86 seconds	2.56 months	2.05 days	0.18 days
Cancelled	2,807	1.08 minutes	3.05 months	18.60 days	0.71 days
Approved	2,246	11.68 minutes	2.86 months	16.66 days	0.51 days

either approving or cancelling the application takes roughly 2.5 weeks. It should be noted that for 399 cases in the event log neither of the activities cancelled, declined or accepted have been recorded. These cases can be assumed to be still running.

4.2 Process O: Loan Offers

The process for handling loan offers shows a lot of similarities with the process for handling loan applications, also when considering the performance information. Just as in the A process, cancelling an offer takes the longest, as is shown in Figure 14. It takes on average 18.70 ± 0.69 days after the previous execution of an activity for a loan offer to be cancelled. In the case of an acceptance this is 4.05 ± 0.25 days and if the offer is declined it is decided 4.02 ± 0.38 days after the last activity. If the offer is sent back, it happens 9.59 ± 0.24 days after it is sent. There is no clear difference in terms of waiting time between the two types of 'O_SENT_BACK'. If a new offer is send however, it is done 4.99 ± 0.44 days after the offer has been sent or after the offer has been sent back. The activities to select, create and sent an offer are executed within seconds after each other. As was indicated in section 2, it is likely that these activities are automated.

On average handling a loan offer from beginning to end takes 17.2 ± 0.42 days, as is shown in Table 7. If we split cases again on their outcome we see different

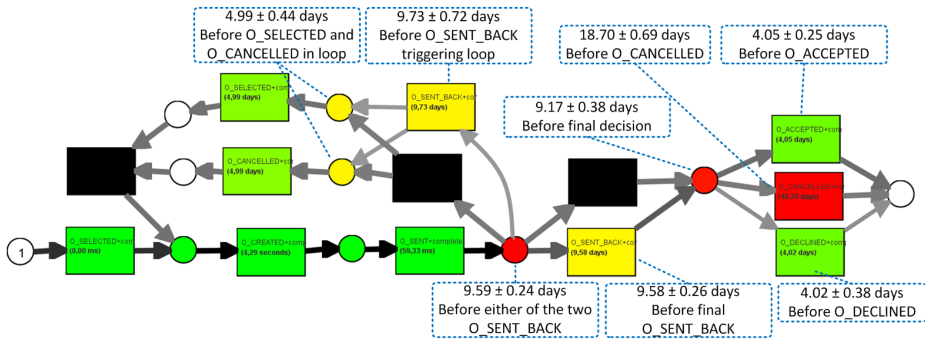


Fig. 14: Performance projection on the loan offer process.

statistics as for the A process. Where in the A process a claim was declined in 2 days on average, if an offer has been sent it takes 15.5 ± 0.81 days on average. If an offer is cancelled a typical case takes 21.52 ± 0.85 days to complete the process, which is roughly 3 days slower than for the A process. An accepted loan offer takes with 16.00 ± 0.50 days less time than cancelled offers, and roughly the same time as approved cases in the A process.

Since the process for handling loan offers has a loop back in the process, in order to send new offers, this will influence case throughput times. If no loop back is required, e.g. if there is only one offer sent, a case takes 15.23 ± 0.41 days on average. Sending at least two offers will increase the average throughput time to 22.05 ± 0.96 days.

Table 7: Case types and their durations for loan offers.

Cases	Freq	Min	Max	Average	99% conf interval
All	5,015	667.00 milliseconds	2.98 months	17.18 days	0.42 days
Declined	802	3.27 minutes	2.10 months	15.47 days	0.81 days
Cancelled	1,640	3.25 minutes	2.98 months	21.52 days	0.85 days
Approved	2,243	3.15 minutes	2.86 months	16.00 days	0.50 days
No Loop Back	3,577	667.00 milliseconds	2.80 months	15.23 days	0.41 days
Loop Back	1,438	37.69 seconds	2.98 months	22.05 days	0.96 days

The process for handling loan offers (O) is one of the processes for which improving the originally discovered model gives significant differences in performance measurements. Figure 15 shows that without first repairing the discovered model, both activities 'O_DECLINED' and 'O_ACCEPTED' (shown as white-colored transitions) never occurred as their occurrences are considered as deviations. Hence, no performance measurement can be obtained from both activities. However, in the repaired version of the process this information is available.

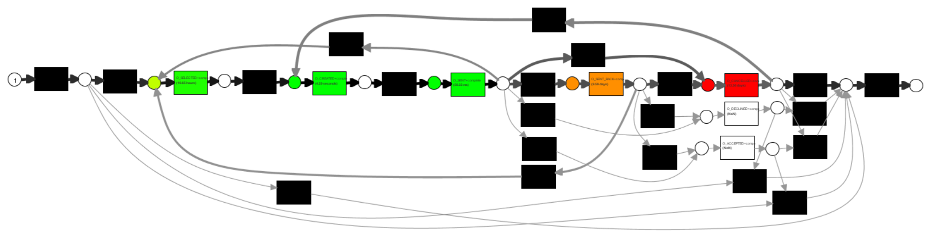


Fig. 15: Incorrect performance projection on the loan offer process, using the model discovered from Heuristic miner.

4.3 Process W: Work Items

The process with performance information projected onto it for the process of handling work items is shown in Figure 16. Cases spend most time waiting before execution of 'W_Nabellen offertes+start', 3.05±0.06 days. This is executed straight after 'Completeren aanvraag'. On average cases also wait 1.30 ± 0.05 days before the activity 'W_Valideren aanvraag+start' is recorded, which occurs mainly after 'nabellen offertes' or 'nabellen incomplete dossiers' is executed. At other places in the process cases seem to be waiting around 14-18 hours between activities from subprocess W.

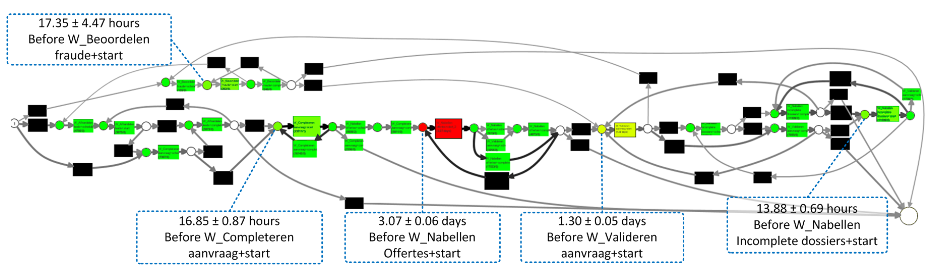


Fig. 16: Performance projection on the work items process.

Table 8: Case durations for handling work items.

Cases	Freq	Min	Max	Average	99% conf interval
All	9,658	30.81 seconds	2.86 months	10.76 days	0.31 days

When considering the average case throughput time, as shown in Table 8, it can be seen that cases spend on average 10.76 ± 0.31 between the first and last recording of activities from process W. This is less than most cases spend in subprocess A and O, which could indicate that the W subprocess is only executed in a specific part of the process execution.

4.4 Performance of Traces with only Activities from Subprocess A

Although the process model for traces containing only activities from subprocess A is very simple, applying performance analysis on this model is still interesting. The performance projection on the process model is shown in Figure 17. This projection shows for instance that the execution of 'A_PARTLYSUBMITTED' after 'A_SUBMITTED' is generally performed within half a second (524.99 ± 46.19 milliseconds). This again indicates that this is an automated activity. The execution of the activity 'A_DECLINED' after 'A_PARTLYSUBMITTED' is also likely to be automated which is indicated by the average waiting time of 38.00 ± 0.43 seconds. However, it seems that more calculations are necessary for this decision to be made since the waiting time is significantly larger than the 0.5 seconds of the other activity. Therefore we think that for 3,429 of the 13,087 loan applications an automated decision has been made to decline the application. This is also supported by the fact that all three activities are executed by user '112'.

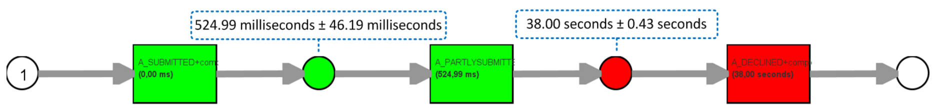


Fig. 17: Performance projection on the process for traces with only activities from subprocess A.

Table 9: Case durations for traces with only activities from subprocess A.

Cases	Freq	Min	Max	Average	99% conf interval
All	3,429	1.86 seconds	2.00 minutes	38.52 seconds	0.44 seconds

The average total execution time for cases that only contain activities from the A subprocess is not surprisingly the summation of the two queues in the process: 38.52 ± 0.44 seconds. In Table 9 also the minimal and maximal case durations are shown. Especially the maximal duration of 2.00 minutes indicates that this is an automated process.

4.5 Performance of Traces with only Activities from both Subprocess A and W

When we take the process model as discovered for traces with only activities from subprocesses A and W we obtain the performance projection as shown in Figure 18. In this model we only observe one real bottleneck, the delay between the first and second execution of 'W_Completeren aanvraag+complete'. However, with only 416 occurrences, out of the total 4,643 traces in this cluster, this bottleneck only occurs for a small fraction of the traces. On average the waiting time between the two executions is 11.10 ± 1.22 days. The hidden bottleneck is however in the decision point where a choice needs to be made which 'W_Completeren aanvraag' to continue with, which is visited by 48% of the traces. In other parts of the process we observe waiting times of several hours but also in the order of minutes, seconds or milliseconds. This strengthens our suspicion that many activities are automated and/or executed in batches.

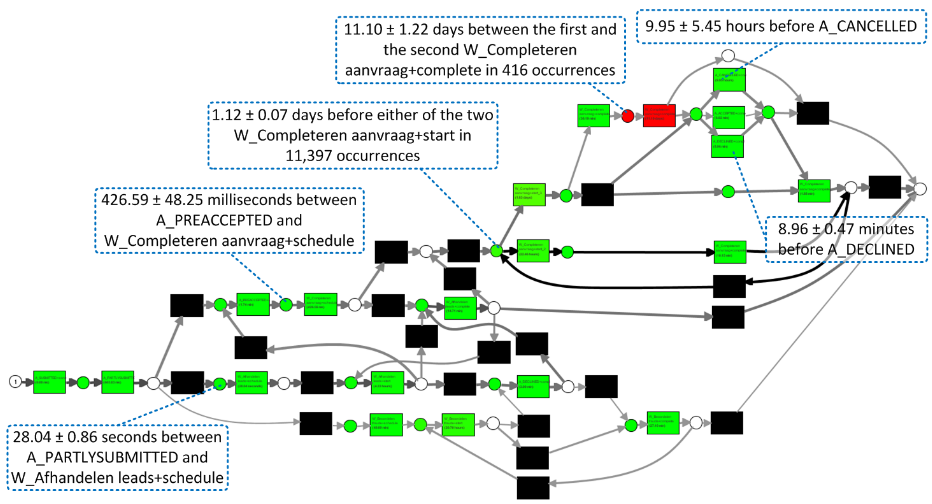


Fig. 18: Performance projection on the process for traces with only activities from subprocesses A and W.

Table 10: Case durations for traces with only activities from both subprocess A and W.

Cases	Freq	Min	Max	Average
All	4,643	1.11 minutes	1.09 months	4.06 ± 0.35 days

4.6 Performance of Traces with only Activities from all subprocesses A, W, and O

When enhancing the process model for traces with activities from all three subprocess we observe more bottlenecks. An overview of the whole process is shown in Figure 19a while in Figure 19b the bottleneck area is shown enlarged. In this area we observe two transitions marked red on one marked orange, indicating relatively long waiting times. All three activities are instances of 'W_Nabellen offertes+start', in different parts of the process. The average waiting times are 4.22 ± 0.14 (top) and 4.13 ± 0.31 (bottom right) and 3.11 ± 0.08 (orange) days. However, once this activity is started, it seems to finish within less than 50 seconds. Therefore this could be executed in batch or only started when the call has already been made. Here too we observe possibly synchronized activities. We again observe identical waiting times for 'A_CANCELLED' and 'O_CANCELLED' but also for 'W_Valideren aanvraag (schedule)' and 'O_SENT_BACK'. In general, it seems that the time spent before 'W_Nabellen offertes+start' is the main bottleneck in this process.

Table 11: Case durations for traces with only activities from both subprocess A and W.

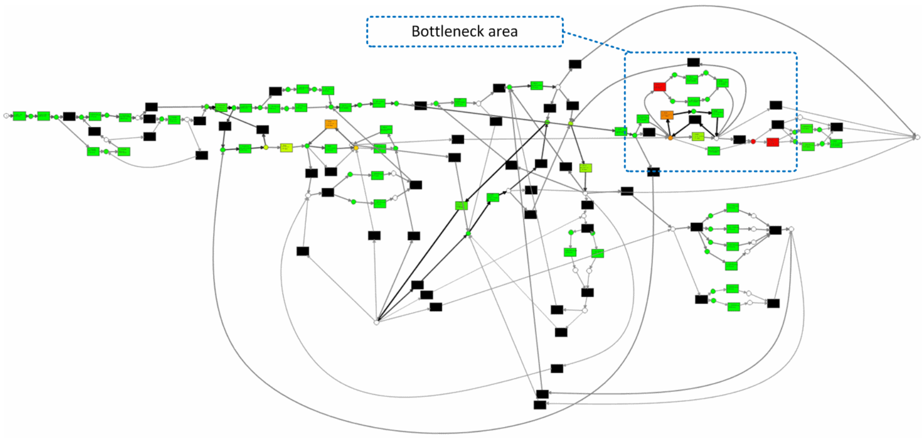
Cases	**Freq**	**Min**	**Max**	**Average**
All	5,015	8.47 minutes	3.05 months	3.05 ± 0.01 months

5 Conclusion

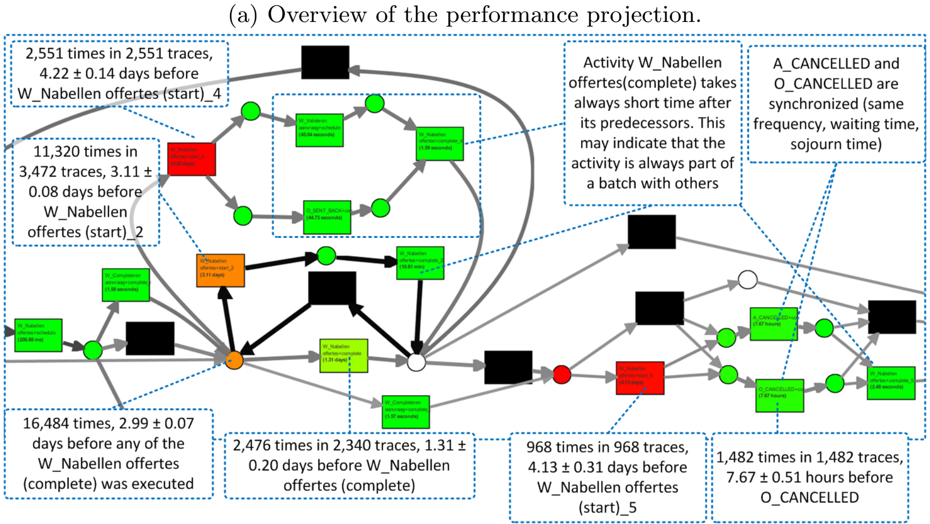
In this paper we used *alignments* between a process model and an event log to manually improve the process models obtained by algorithms and to project performance information on them.

We were able to discover a process model for each of the three subprocesses but also for the three different trace types. For each of these 6 sublogs we were able to improve the discovered process models to reach both a high fitness (of 0.99 in all cases) and a high precision. Obtaining a high quality process model is important when information, such as performance measurements, is projected onto the process models. We showed that by spending effort in obtaining high quality process models, reliable performance information becomes available which provides important insights. We could for instance pinpoint several bottlenecks and automated activities. Especially the combined process of A, O and W gives a good insight into how the mainly automatic processes A and O intertwine with the manual handling of work items from subprocess W. Furthermore, we were able to distinguish between different types of activity 'W_Nabellen offertes' in the W process.

Besides concrete process mining results we also demonstrated that the process models as discovered by process discovery algorithms can be improved significantly. By using the alignments between the process model and event log major deviations can be fixed manually. By repeating this process a high quality process model can be obtained. This is especially the case for behavior that is



(a) Overview of the performance projection.



(b) Part of the performance projection in more detail.

Fig. 19: Performance projection on the process for traces with activities from subprocesses A, W and O.

not extremely structured but shows clear patterns. Furthermore, we showed that a good alignment is a requirement in order to obtain trustworthy performance information about the process.

References

1. 8th International Workshop on Business Process Intelligence 2012. `http://www.win.tue.nl/bpi2012/doku.php?id=challenge`. Accessed: 29/07/2012.
2. W.M.P. van der Aalst, A.J.M.M. Weijters, and L. Maruster. Workflow Mining: Discovering Process Models from Event Logs. *IEEE Transactions on Knowledge and Data Engineering*, 16(9):1128–1142, 2004.
3. A. Adriansyah, B. van Dongen, and W.M.P. van der Aalst. Conformance Checking using Cost-Based Fitness Analysis. In *IEEE International Enterprise Computing Conference (EDOC 2011)*, pages 55–64. IEEE Computer Society, 2011.
4. A. Adriansyah, J. Munoz-Gama, J. Carmona, B.F. van Dongen, and W.M.P. van der Aalst. Alignment Based Precision Checking. In *Proceedings of the 8th International Workshop on Business Process Intelligence (BPI 2012)*, 2012. (to appear).
5. J.C.R.P. Bose and W.M.P. van der Aalst. Process diagnostics using trace alignment: Opportunities, issues, and challenges. *Information Systems*, 37(2):117–141, April 2012.
6. Buijs, J.C.A.M., Dongen, B.F. van, and Aalst, W.M.P. van der. On the Role of Fitness, Precision, Generalization and Simplicity in Process Discovery (to appear). In *20th International Conference on Cooperative Information Systems (CoopIS 2012)*, lncs, 2012.
7. T. Murata. Petri nets: Properties, analysis and applications. *Proceedings of the IEEE*, 77(4):541–580, August 1989.
8. W.M.P. van der Aalst. *Process Mining: Discovery, Conformance and Enhancement of Business Processes*. Springer Verlag, 2011. ISBN:978-3-642-19344-6.
9. W.M.P. van der Aalst, A. Adriansyah, and B.F. van Dongen. Replaying history on process models for conformance checking and performance analysis. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 2(2):182–192, 2012.
10. B.F. van Dongen. Event Log for the BPI Challenge 2012. `http://dx.doi.org/10.4121/uuid:3926db30-f712-4394-aebc-75976070e91f`, 2012.
11. H. M. W. Verbeek, J. C. A. M. Buijs, B. F. van Dongen, and W. M. P. van der Aalst. Xes, xesame, and prom 6. In *Information System Evolution*, volume 72, pages 60–75. Springer, 2011.
12. A.J.M.M. Weijters and W.M.P. van der Aalst. Rediscovering Workflow Models from Event-Based Data using Little Thumb. *Integrated Computer-Aided Engineering*, 10(2):151–162, 2003.
13. J.M.E.M. van der Werf, B.F. van Dongen, C.A.J. Hurkens, and A. Serebrenik. Process Discovery using Integer Linear Programming. *Fundamenta Informaticae*, 94:387–412, 2010.
14. T. van der Wiel. Process mining using integer linear programming. Master's thesis, Eindhoven University of Technology, 2010.

A Original Event Log Statistics

In this Appendix basic log statistics from the original event log are shown.

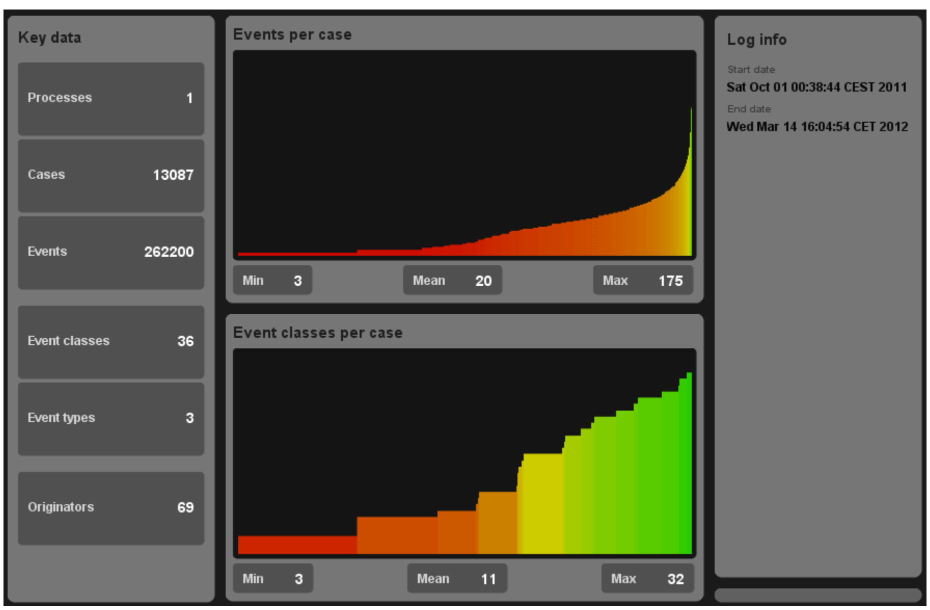


Fig. 20: ProM Dashboard overview of the original event log.

Table 12: Event Classes and their occurrences (event name + lifecycle state).

Event Class	Occurrences (absolute)	Occurrences (relative)
W_Completeren aanvraag+COMPLETE	23967	9,141%
W_Completeren aanvraag+START	23512	8,967%
W_Nabellen offertes+COMPLETE	22976	8,763%
W_Nabellen offertes+START	22406	8,545%
A_SUBMITTED+COMPLETE	13087	4,991%
A_PARTLYSUBMITTED+COMPLETE	13087	4,991%
W_Nabellen incomplete dossiers+COMPLETE	11407	4,350%
W_Nabellen incomplete dossiers+START	11400	4,348%
W_Valideren aanvraag+COMPLETE	7895	3,011%
W_Valideren aanvraag+START	7891	3,010%
A_DECLINED+COMPLETE	7635	2,912%
W_Completeren aanvraag+SCHEDULE	7371	2,811%
A_PREACCEPTED+COMPLETE	7367	2,810%
O_SELECTED+COMPLETE	7030	2,681%
O_CREATED+COMPLETE	7030	2,681%
O_SENT+COMPLETE	7030	2,681%
W_Nabellen offertes+SCHEDULE	6634	2,530%
W_Afhandelen leads+COMPLETE	5898	2,249%
W_Afhandelen leads+START	5897	2,249%
A_ACCEPTED+COMPLETE	5113	1,950%
W_Valideren aanvraag+SCHEDULE	5023	1,916%
A_FINALIZED+COMPLETE	5015	1,913%
W_Afhandelen leads+SCHEDULE	4771	1,820%
O_CANCELLED+COMPLETE	3655	1,394%
O_SENT_BACK+COMPLETE	3454	1,317%
A_CANCELLED+COMPLETE	2807	1,071%
W_Nabellen incomplete dossiers+SCHEDULE	2383	0,909%
A_APPROVED+COMPLETE	2246	0,857%
A_REGISTERED+COMPLETE	2246	0,857%
A_ACTIVATED+COMPLETE	2246	0,857%
O_ACCEPTED+COMPLETE	2243	0,855%
O_DECLINED+COMPLETE	802	0,306%
W_Beoordelen fraude+START	270	0,103%
W_Beoordelen fraude+COMPLETE	270	0,103%
W_Beoordelen fraude+SCHEDULE	124	0,047%
W_Wijzigen contractgegevens+SCHEDULE	12	0,005%

Table 13: Start Event Classes

Class	Occurrences (absolute)	Occurrences (relative)
A_SUBMITTED+COMPLETE	13087	100%

Table 14: End Event Classes

Class	Occurrences (absolute)	Occurrences (relative)
A_DECLINED+COMPLETE	3429	26,20%
W_Valideren aanvraag+COMPLETE	2745	20,98%
W_Afhandelen leads+COMPLETE	2234	17,07%
W_Completeren aanvraag+COMPLETE	1939	14,82%
W_Nabellen offertes+COMPLETE	1289	9,85%
A_CANCELLED+COMPLETE	655	5,01%
W_Nabellen incomplete dossiers+COMPLETE	452	3,45%
O_CANCELLED+COMPLETE	279	2,13%
W_Beoordelen fraude+COMPLETE	57	0,44%
W_Wijzigen contractgegevens+SCHEDULE	4	0,03%
W_Valideren aanvraag+START	2	0,02%
W_Nabellen offertes+START	1	0,01%
A_REGISTERED+COMPLETE	1	0,01%

B Performance Matrix

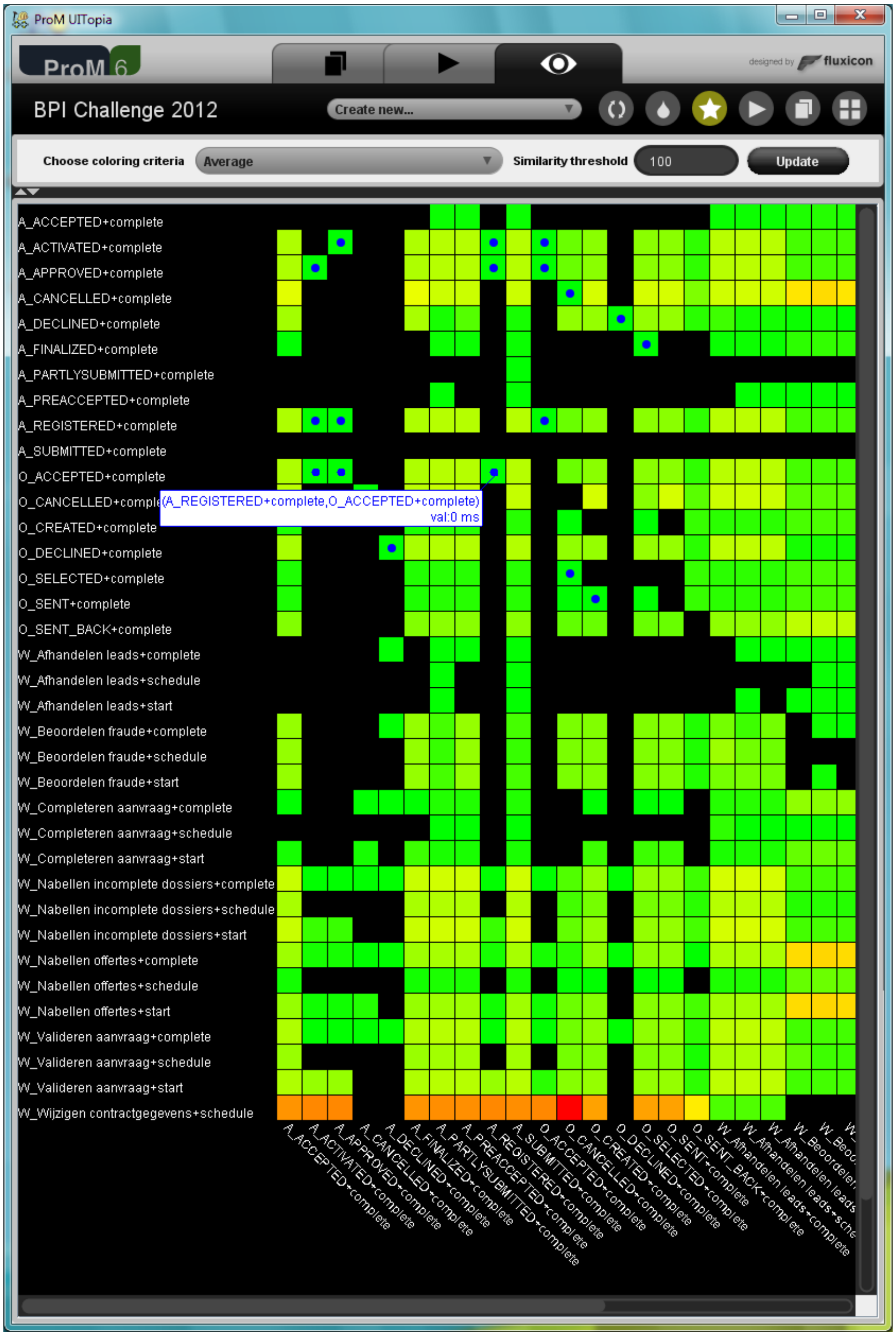


Fig. 21: Part of the Log Performance Matrix showing the elapsed time between pairs of activities.

C Event Log statistics for Subprocess A

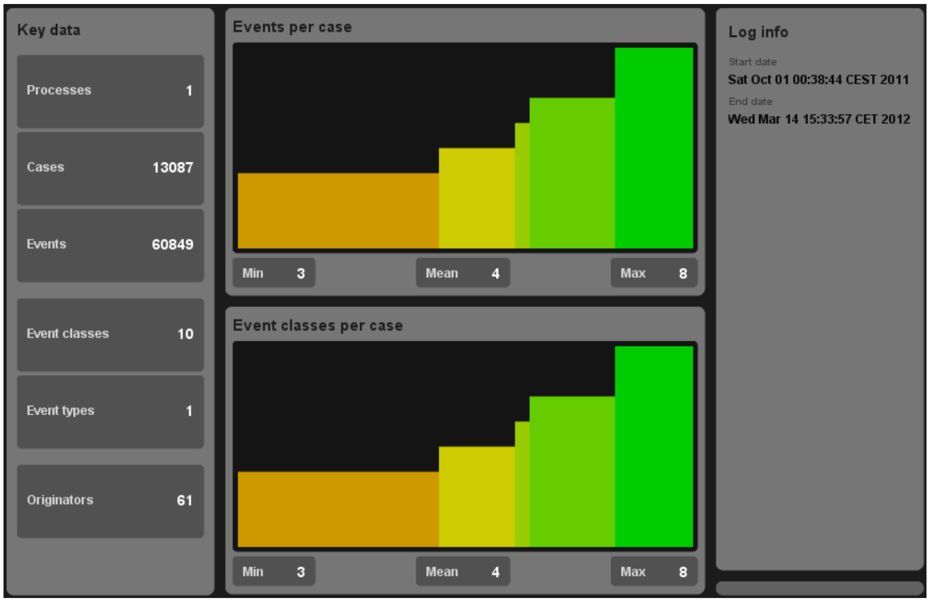


Fig. 22: ProM Dashboard for subprocess A.

Table 15: Event Classes and their occurrences for the A subprocess.

Class	Occurrences (absolute)	Occurrences (relative)
A_PARTLYSUBMITTED+complete	13,087	21.51%
A_SUBMITTED+complete	13,087	21.51%
A_DECLINED+complete	7,635	12.55%
A_PREACCEPTED+complete	7,367	12.11%
A_ACCEPTED+complete	5,113	8.40%
A_FINALIZED+complete	5,015	8.24%
A_CANCELLED+complete	2,807	4.61%
A_REGISTERED+complete	2,246	3.69%
A_APPROVED+complete	2,246	3.69%
A_ACTIVATED+complete	2,246	3.69%

Table 16: Event classes for subprocess A that occur as the first event of traces.

Class	Occurrences (absolute)	Occurrences (relative)
A_SUBMITTED+complete	13,087	100.00%

Table 17: Event classes for subprocess A that occur as the last event of traces.

Class	Occurrences (absolute)	Occurrences (relative)
A_DECLINED+complete	7,635	58.34%
A_CANCELLED+complete	2,807	2.14%
A_ACTIVATED+complete	1,122	8.57%
A_REGISTERED+complete	787	6.01%
A_APPROVED+complete	337	2.58%
A_FINALIZED+complete	327	2.50%
A_PREACCEPTED+complete	69	0.53%
A_ACCEPTED+complete	3	0.02%

D Event Log statistics for Subprocess O

Fig. 23: ProM Dashboard for subprocess O.

Table 18: Event Classes and their occurrences for the O subprocess.

Class	Occurrences (absolute)	Occurrences (relative)
O_SENT+complete	7,030	22.50%
O_SELECTED+complete	7,030	22.50%
O_CREATED+complete	7,030	22.50%
O_CANCELLED+complete	3,655	11.70%
O_SENT_BACK+complete	3,454	11.06%
O_ACCEPTED+complete	2,243	7.18%
O_DECLINED+complete	802	2.57%

Table 19: Event classes for subprocess O that occur as the first event of traces.

Class	Occurrences (absolute)	Occurrences (relative)
O_SELECTED+complete	5,015	100.00%

Table 20: Event classes for subprocess O that occur as the last event of traces.

Class	Occurrences (absolute)	Occurrences (relative)
O_ACCEPTED+complete	2,243	44.73%
O_CANCELLED+complete	1,640	32.70%
O_DECLINED+complete	802	15.99%
O_SENT+complete	241	4.81%
O_SENT_BACK+complete	89	1.78%

E Event Log statistics for Subprocess W

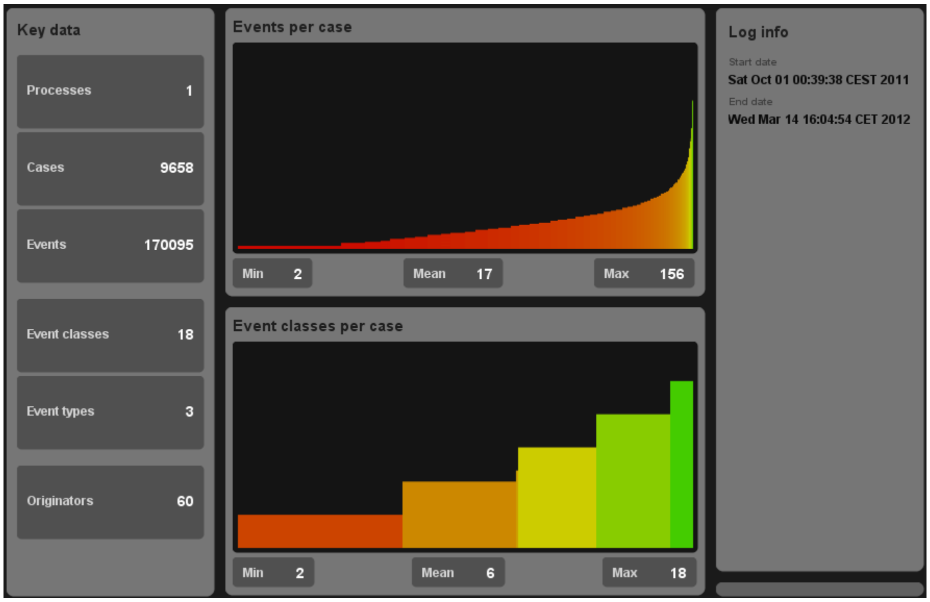


Fig. 24: ProM Dashboard for subprocess W.

F Event Log statistics for Traces with only Activities from Subprocess A

G Event Log statistics for Traces with only Activities from Subprocesses A and W

H Event Log statistics for Traces with Activities from Subprocesses A, W and O

Table 21: Event Classes and their occurrences for the W subprocess.

Class	Occurrences (absolute)	Occurrences (relative)
W_Completeren aanvraag+complete	23,967	14.09%
W_Completeren aanvraag+start	23,512	13.82%
W_Nabellen offertes+complete	22,976	13.51%
W_Nabellen offertes+start	22,406	13.17%
W_Nabellen incomplete dossiers+complete	11,407	6.71%
W_Nabellen incomplete dossiers+start	11,400	6.70%
W_Valideren aanvraag+complete	7,895	4.64%
W_Valideren aanvraag+start	7,891	4.64%
W_Completeren aanvraag+schedule	7,371	4.33%
W_Nabellen offertes+schedule	6,634	3.90%
W_Afhandelen leads+complete	5,898	3.47%
W_Afhandelen leads+start	5,897	3.47%
W_Valideren aanvraag+schedule	5,023	2.95%
W_Afhandelen leads+schedule	4,771	2.81%
W_Nabellen incomplete dossiers+schedule	2,383	1.40%
W_Beoordelen fraude+complete	270	0.16%
W_Beoordelen fraude+start	270	0.16%
W_Beoordelen fraude+schedule	124	0.16%

Table 22: Event classes for subprocess W that occur as the first event of traces.

Class	Occurrences (absolute)	Occurrences (relative)
W_Completeren aanvraag+schedule	4,852	50.24%
W_Afhandelen leads+schedule	4,739	49.07%
W_Beoordelen fraude+schedule	67	0.69%

Table 23: Event classes for subprocess W that occur as the last event of traces.

Class	Occurrences (absolute)	Occurrences (relative)
W_Valideren aanvraag+complete	2,750	28.47%
W_Completeren aanvraag+complete	2,355	24.38%
W_Afhandelen leads+complete	2,235	23.14%
W_Nabellen offertes+complete	1,801	18.65%
W_Nabellen incomplete dossiers+complete	457	4.73%
W_Beoordelen fraude+complete	57	0.59%
W_Valideren aanvraag+start	2	0.02%
W_Nabellen offertes+start	1	0.01%

Table 24: Event Classes and their occurrences for the log with only traces with activities from the A subprocess.

Class	Occurrences (absolute)	Occurrences (relative)
A_PARTLYSUBMITTED+complete	3,429	33.33%
A_SUBMITTED+complete	3,429	33.33%
A_DECLINED+complete	3,429	33.33%

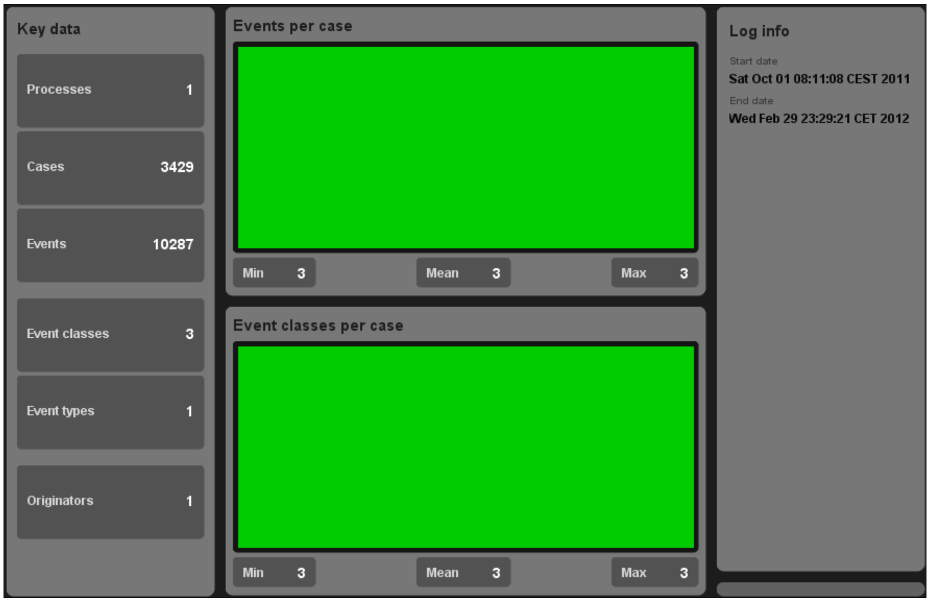


Fig. 25: ProM Dashboard for traces with only activities from subprocess A.

Table 25: Event classes for traces with only activities from subprocess A that occur as the first event of traces.

Class	Occurrences (absolute)	Occurrences (relative)
A_SUBMITTED+complete	3,429	100.00%

Table 26: Event classes for traces with only activities from subprocess A that occur as the last event of traces.

Class	Occurrences (absolute)	Occurrences (relative)
A_DECLINED+complete	3,429	100.00%

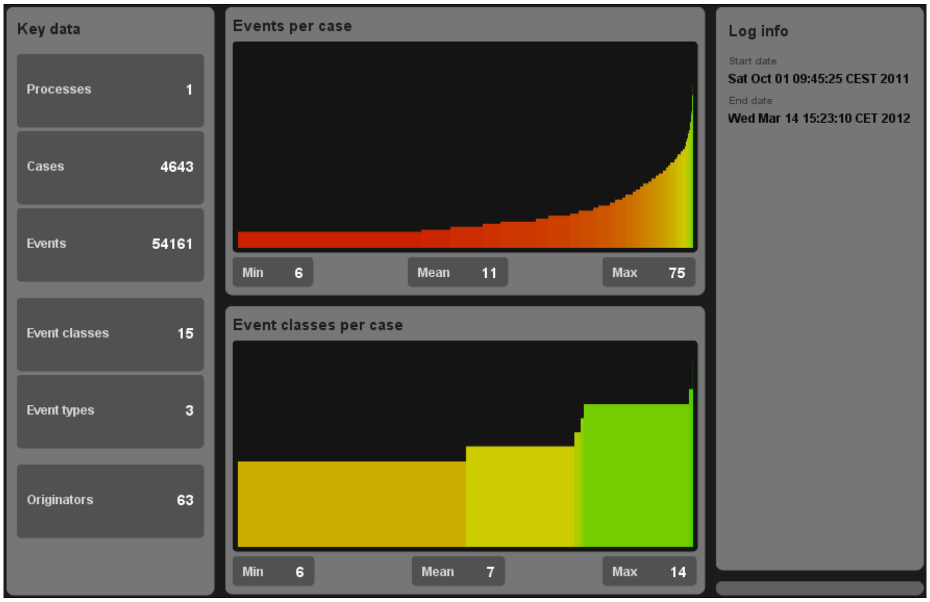


Fig. 26: ProM Dashboard for traces with activities from subprocesses A and W.

Table 27: Event Classes and their occurrences for the log with only traces with activities from the A and W subprocesses.

Class	Occurrences (absolute)	Occurrences (relative)
W_Completeren aanvraag+complete	11,848	21.88%
W_Completeren aanvraag+start	11,398	21.05%
A_SUBMITTED+complete	4,643	8.57%
A_PARTLYSUBMITTED+complete	4,643	8.57%
W_Afhandelen leads+complete	4,177	7.71%
W_Afhandelen leads+start	4,176	7.71%
A_DECLINED+complete	3,404	6.29%
W_Afhandelen leads+schedule	3,390	6.26%
W_Completeren aanvraag+schedule	2,352	4.34%
A_PREACCEPTED+complete	2,352	4.34%
A_CANCELLED+complete	1,167	2.16%
W_Beoordelen fraude+complete	214	0.40%
W_Beoordelen fraude+start	214	0.40%
A_ACCEPTED+complete	98	0.18%
W_Beoordelen fraude+schedule	85	0.16%

Table 28: Event classes for traces with only activities from subprocesses A and W that occur as the first event of traces.

Class	Occurrences (absolute)	Occurrences (relative)
A_SUBMITTED+complete	4,643	100.00%

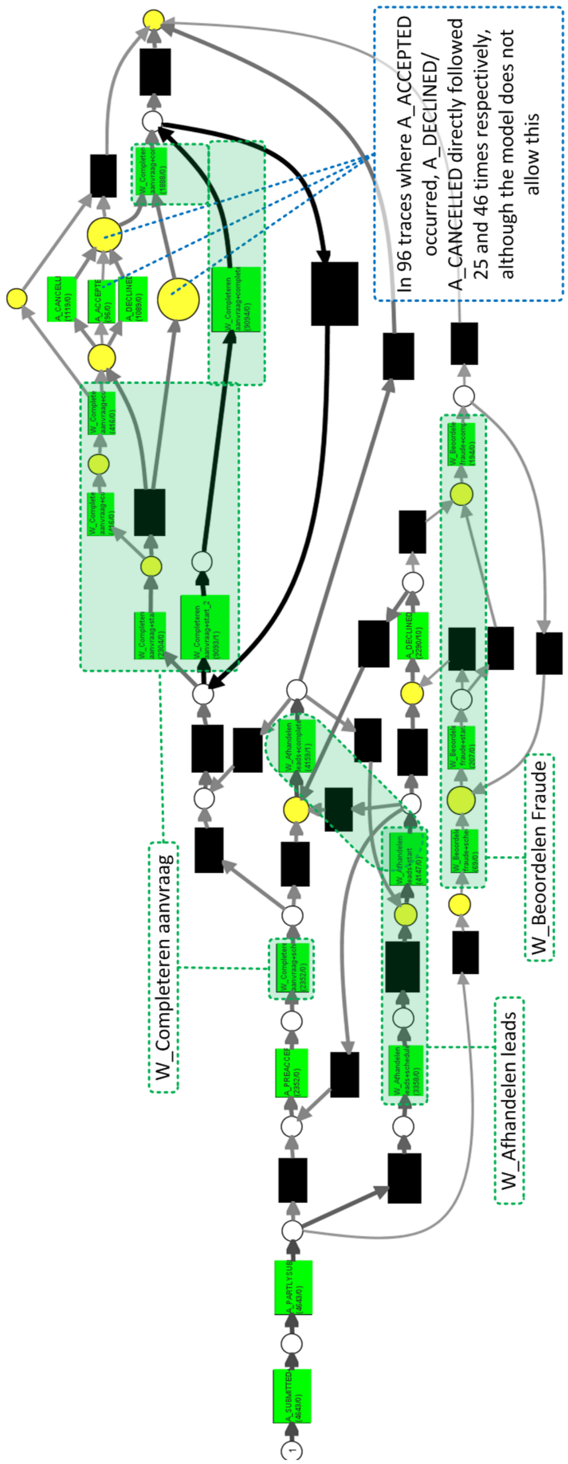


Fig. 27: Result after final fix

Table 29: Event classes for traces with only activities from subprocesses A and W that occur as the last event of traces.

Class	Occurrences (absolute)	Occurrences (relative)
W_Afhandelen leads+complete	2,234	48.12%
W_Completeren aanvraag+complete	1,935	41.68%
A_CANCELLED+complete	417	8.98%
W_Beoordelen fraude+complete	57	1.23%

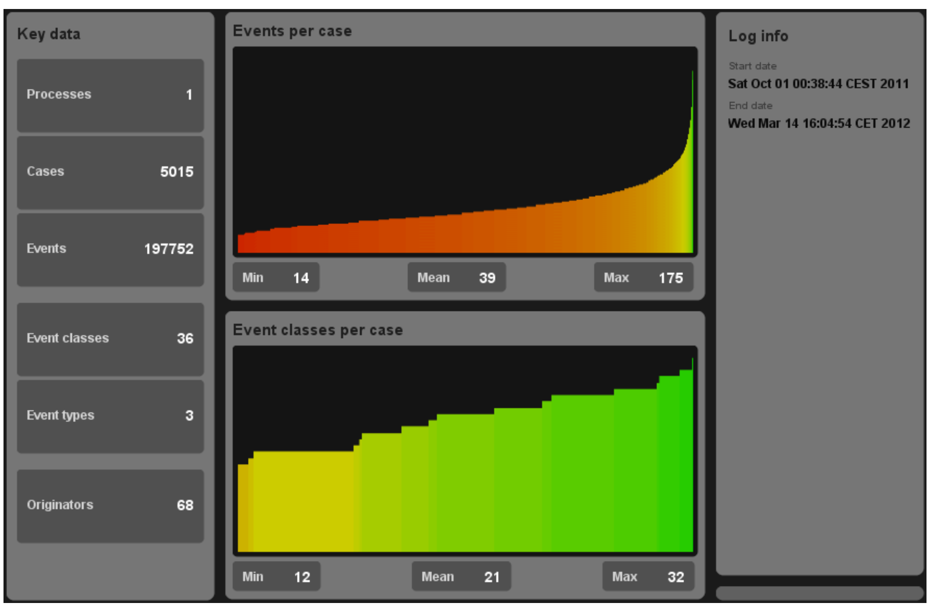


Fig. 28: ProM Dashboard for traces with activities from all subprocesses.

Table 30: Event Classes and their occurrences for the log with only traces with activities from all three subprocesses.

Class	Occurrences (absolute)	Occurrences (relative)
W_Nabellen offertes+complete	22,976	11.62%
W_Nabellen offertes+start	22,406	11.33%
W_Completeren aanvraag+complete	12,119	6.13%
W_Completeren aanvraag+start	12,114	6.13%
W_Nabellen incomplete dossiers+complete	11,407	5.77%
W_Nabellen incomplete dossiers+start	11,400	5.77%
W_Valideren aanvraag+complete	7,895	3.99%
W_Valideren aanvraag+start	7,891	3.99%
O_CREATED+complete	7,030	3.56%
O_SENT+complete	7,030	3.56%
O_SELECTED+complete	7,030	3.56%
W_Nabellen offertes+schedule	6,634	3.36%
W_Valideren aanvraag+schedule	5,023	2.54%
W_Completeren aanvraag+schedule	5,019	2.54%
A_FINALIZED+complete	5,015	2.54%
A_PREACCEPTED+complete	5,015	2.54%
A_SUBMITTED+complete	5,015	2.54%
A_PARTLYSUBMITTED+complete	5,015	2.54%
A_ACCEPTED+complete	5,015	2.54%
O_CANCELLED+complete	3,655	1.85%
O_SENT_BACK+complete	3,454	1.75%
W_Nabellen incomplete dossiers+schedule	2,383	1.21%
A_REGISTERED+complete	2,246	1.14%
A_APPROVED+complete	2,246	1.14%
A_ACTIVATED+complete	2,246	1.14%
O_ACCEPTED+complete	2,243	1.13%
W_Afhandelen leads+complete	1,721	0.87%
W_Afhandelen leads+start	1,721	0.87%
A_CANCELLED+complete	1,640	0.83%
W_Afhandelen leads+schedule	1,381	0.70%
O_DECLINED+complete	802	0.41%
A_DECLINED+complete	802	0.41%
W_Beoordelen fraude+complete	56	0.03%
W_Beoordelen fraude+start	56	0.03%
W_Beoordelen fraude+schedule	39	0.02%
W_Wijzigen contractgegevens+schedule	12	0.01%

Table 31: Event classes for traces with only activities from all three subprocesses that occur as the first event of traces.

Class	Occurrences (absolute)	Occurrences (relative)
A_SUBMITTED+complete	5,015	100.00%

Table 32: Event classes for traces with only activities from all three subprocesses that occur as the last event of traces.

Class	Occurrences (absolute)	Occurrences (relative)
W_Valideren aanvraag+complete	2,745	54.74%
W_Nabellen offertes+complete	1,289	25.70%
W_Nabellen incomplete dossiers+complete	452	9.01%
O_CANCELLED+complete	279	5.56%
A_CANCELLED+complete	238	4.75%
W_Completeren aanvraag+complete	4	0.08%
W_Wijzigen contractgegevens+schedule	4	0.08%
W_Valideren aanvraag+start	2	0.04%
A_REGISTERED+complete	1	0.02%
W_Nabellen offertes+start	1	0.02%