Estimating the Processing Time of Process Instances in Semi-Structured Processes - A Case Study

Andreas Wombacher
Center for Telematics and Information Technology,
University of Twente,
Enschede, The Netherlands
Email: a.wombacher@utwente.nl

Maria Iacob
Center for Telematics and Information Technology,
University of Twente,
Enschede, The Netherlands
Email: m.e.iacob@utwente.nl

***Abstract*—Performance analysis of Web applications are rather difficult since people can perform parts of an activity outside the application or get interrupted while performing an activity in the system. The lack of a performance model makes it hard to plan resources or get a better understanding of the way available resources are used. In this paper an approach for determining a performance model for semi-structured processes is applied to a case study.**

I. Introduction

Semi-structured processes are business workflows, where the execution of the workflow is not completely controlled by a workflow engine, i.e., an implementation of a formal workflow model. Examples can be found in scenarios where several people potentially from different organizations cooperate, e.g., in creating a yearly progress report or writing a scientific paper. Other examples are workflows where people interact with clients and/or paper documents which are used to insert, approve, or validate information in a potentially Web based information system. Such a Web-based information system can be an application server, or an orchestration of services, e.g., using BPEL.

In these scenarios it is important for an organisation's management to understand well the process, the characteristics of activities, and the performance of individual employees. Lacking such knowledge makes it hard to predict the load of resources, and to make a balanced resource planning. For example, it is difficult to predict the ability of the business to handle higher workload due, for example, to a promotional activity or to vacations.

Independent of the workflow's implementation, the underlying information system may keep track of the completion time of an activity but cannot record the start time of an activity. Such an information system cannot detect for instance when a conversation with a client starts or whether an employee interrupts his work on one activity to perform a more urgent activity. Thus, it is not possible to build a classical performance model and use existing process analysis techniques like those described in [1]. These assumptions hold for most web based applications and therefore the approach discussed in this paper has a high applicability. Further, due to the nature of the problem at hand there are, to the best of our knowledge no other approaches addressing this problem. The naive approach of estimating the start time of an activity as the completion time of the preceding activity is only applicable under strong (and unlikely) assumptions, such as, people working on the system under investigation without interruptions or breaks.

Therefore, in this paper we aim to use the available log information to perform data analysis and data cleansing in order to get an estimate of the overall time spend by a user on the process. We propose a structured approach to investigate and cleanse the observed event data. The result is an estimated starting time for each event. In case the estimated starting time is not trustworthy we report it as 'unknown'. Based on this estimated stating time and the observed behavior of the user we are able to provide a good estimate of the processing time of a process. We apply the proposed method to a case study at the University of Twente. The results are validated empirically by comparison with reported process durations.

The paper is organized as follows. We first present the general cleansing and start-time estimation approach (Sect II), followed by the description of our use case (Section III). Section III presents the application of our approach and the results of the analysis we performed on the use case data. An evaluative discussion of the proposed approach from the perspective of the case-study results is included in Section V. We conclude the paper with related work (Section VI) and conclusions (Section VII).

II. Approach

As we said before, independent of the workflow's implementation, the underlying information system may keep

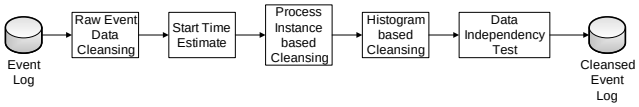

Figure 1. Cleansing rules overview

track of the completion time of an activity but cannot record the start time of an activity [2]. Such an information system cannot detect for instance when a conversation with a client starts. Thus, it is not possible to build a classical performance model and use existing process analysis techniques like those described in [1] before enriching the data with the activities' start times. Therefore, in [2] we proposed an approach for cleansing the data and estimating the activity start times based on the steps depicted in Fig 1. A first cleansing step is performed on the raw event data eliminating data which is unusable, because of abnormal operation of the system due to, for instance, infrastructure problems (e.g., network problems). Next, the cleansed data is used to infer an initial estimate of the start time for each activity. The initial estimates may be overwritten in later cleansing steps. The following cleansing step investigates special situations per process instance (also called case) like, for example, system tests and dead-lock or live-lock instances. The last cleansing step is the histogram based cleansing (per activity) that removes outliers, i.e., exceptionally high durations of an activity. The final step investigates dependencies of activity durations cross process instances and categorical data (e.g., the weekday or the experience of a user), thus checking whether the independence assumption used in a performance model is actually supported by available data. The final result is a cleansed event log, which can be used for the mining of a control flow and for performance analysis.

In [2] we have reached the conclusion that the inferred performance estimates for the case study were significantly low compared to the performance measures expected by domain experts. Therefore, in this paper we re-visit the start time estimation approach and the histogram based cleansing. In particular, we present additional insights into start time estimates including a mathematical description of the problem. Furthermore, a new histogram based cleansing technique is proposed. The findings will be evaluated on a new use case introduced in the following section.

III. Introduction Use Case

The use case concerns a digital library based on the eprints system[1] used at the University of Twente. A detailed description of the underlying strategy and goals of the system is accessible at [3]. The configuration used is illustrated by means of a typical workflow depicted in Fig 2. The figure shows a state transition diagram representing a single case, and specifies the process of inserting a publication as an eprints item into the system.

Each node represents an activity of the case, i.e., filling in some information in a web-form in the web-application. An activity is annotated by its name and the user id of the user performing this activity. The transitions depicted as edges represent the sequence of activity executions in the case. Each edge is annotated with an order number representing the order of executing this activity in the sequence of this case. Further, each edge contains, between brackets, the time difference between the source activity start time and the target activity start time in seconds. That is, the time difference is the time between entering the source and entering the target web page, thus the time potentially spent by the user on the source web page. The case starts at the activity with no incoming transition.

The first part of the case in Fig 2 is executed by user id 745 and the second part by user 1025. The user initiating a case is called the owner of the case. Thus user id 745 is the owner of the case. This user first selects the *type* of publication he or she is about to add to the system. Next, several meta data are collected. In particular, information about the authors, the title and the relations of the authors in the organization are acquired in activity *meta.core*. Then information about the publication are collected (*meta.pubinfo* activity) like the page numbers or ISSN number. The remaining meta data (*meta.status* activity) is about the abstract of the paper and key words. Next the *files* activity shows the already uploaded files and enables to upload files. Then the process requires to upload the publication which is split up in two activities: first in *docmeta* activity the document type and the access policy are specified and afterwards in the *fileview* activity the file is actually uploaded. After the upload the uploaded files are shown again in the *files* activity, before the user is asked to *verify* the provided data and to submit the data to the editor (*done* activity).

The editor - in this case the user id 1025 - follows the same process to check and potentially change values. The only difference is the *return* activity is replacing the *done* activity since the editor is completing the task and not depositing it again. The editor has a *quickverify* activity not contained in this case allowing the editor to get an overview of the data related to a case.

Usually a case is initiated by a user, who is owning the case, i.e., which is one of the authors of the publication added to the digital library. It is possible that other people (editors) get involved in the process to verify or posteriori modify a case.

Note that the workflow model may contain cycles since the web application allows to go backwards and forwards in the process but does not support to jump to a particular screen. As a consequence, cycles happen actually quite often. Such a small cycle is also shown in Fig 2 with *files*, *docmeta* and *fileview* activities.

IV. Analyzing Use Case

In this section the steps of our approach (see Sect II) are discussed one by one, while being executed for the selected case study, and the results of each step are presented. The basis for this analysis is a log file generated by the eprints logging system containing the timestamp of the request of

[1] http://www.eprints.org/

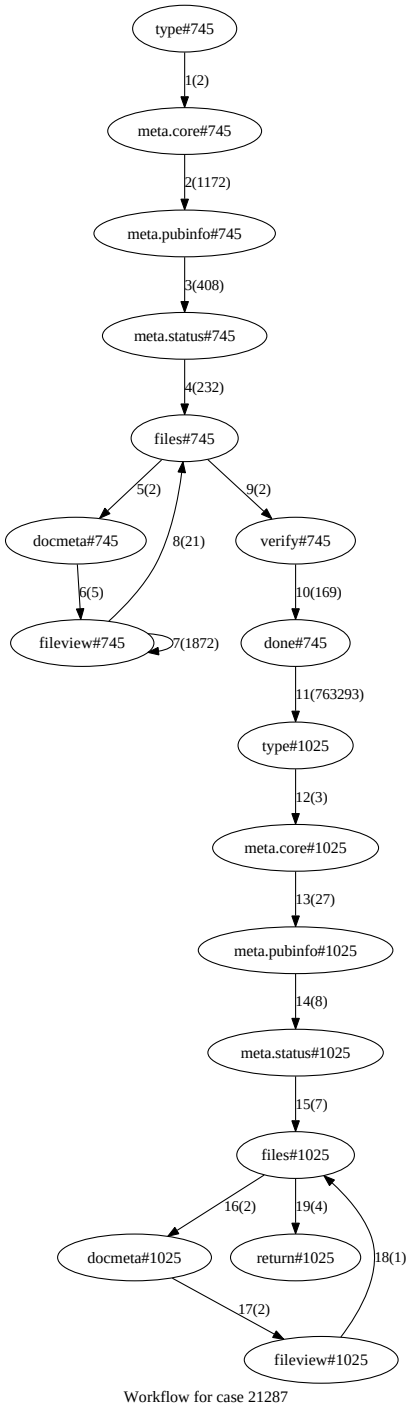

Figure 2. CPN workflow

a web site, the user making the request and the page being requested. The available log file contains 1003 cases.

A. Raw Event Data Cleansing

The initial step of the cleansing aims to ensure the reliability of the data, thus, establishes whether the data

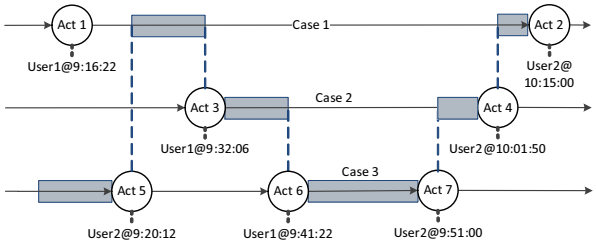

Figure 3. Start Time Estimation

at hand reflects normal operation of the system or an exceptional mode of operation. An example of an exceptional mode of operation are network problems in a distributed infrastructure. In the use case at hand there are no known infrastructure problems, therefore no data has to be omitted.

B. Start Time Estimate

The aim of the start time estimation step is to determine automatically a possible start time of an activity given the completion time of this activity and the completion time of other activities. The start time estimation approach is elaborated in detail in [4]. Based on the start time and the completion time of an activity, the duration of an activity can be inferred. However, as we said, the system only logs the completion times of activities. The estimation of the start time of activities is based on the observation that an activity in a workflow can only start after its preceding activity has been completed. Further, we assume that a user can perform only one activity at the time. The proposed approach is illustrated in Fig 3. The start time of activity 2 (Act 2) is initially estimated by the completion time of the preceding activity of the same process, which is activity 1. From this initial estimate the chunks are removed, which fall into this processing time interval and which have been associated to executing other activities by the same user. A chunk is represented by a gray box in the figure. In this case, the processing of activity 2 is performed in two chunks: the first one is the non assigned time of user 2 between activity 5 and 3 and the second one is the non assigned time between activity 4 and 2. The more processes are interwoven the higher the number of chunks.

The log information extended by the estimated start times and the associated chunks are the input for the following step.

C. Process Instance based Cleansing

The step investigates the event log per process instance, also called case, and marks complete cases as unsuitable for performance model mining. Examples of omitted cases are special test cases performed on the system, as well as special deadlock and livelock errors in cases.

A first data analysis reveals that the number of activities performed in a case correlates with the case ID (see Fig 4).

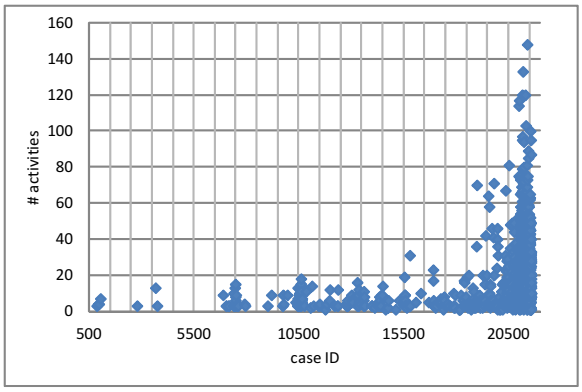

Figure 4. Histogram of the number of activities per case

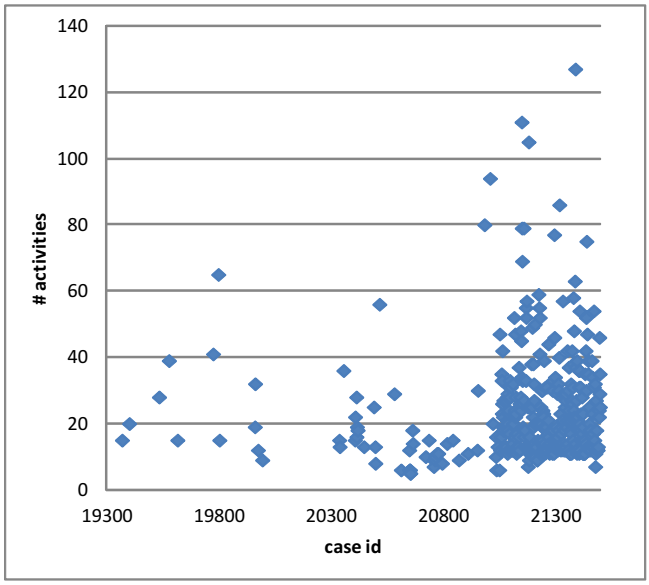

Figure 5. Relation between number of activities performed by the case owner and case ID

Further, the number of users involved in a case increases with increasing case ID. This is unexpected since the workflow has not changed in the time period the log file has been created. When discussing these observations with the owner of the data, the conclusion was that some of the cases are incompletely recorded. In particular, the logging option has been switched on during the operation of the system, thus the log file contains also cases which have been started before the logging had been initiated. Identifying and removing these incomplete cases turned out to be difficult, since the eprints system only documents when a case is completed, that is, the publication has been initially approved by an editor. We use this approval date as a criteria to eliminate cases which have been approved before the logging has been switched on. This reduces the number of relevant cases in the log file to 442 cases.

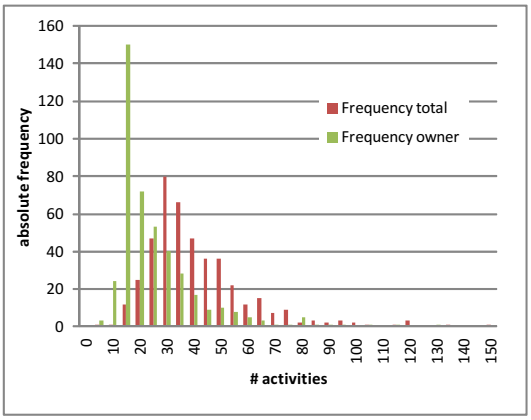

Figure 6. Histogram of the number of activities per case

The number of activities per case performed by the case owner is less dependent on the case ID now, as depicted in Fig 5. Looking at it from the perspective of the number of activities performed in a case, either by the case owner, or by another user, results in Fig 6. The conclusion is that most cases have around 15 activities executed by the case owner and around 30 activities executed by different users, i.e., the case owner and one or more editors. The maximum number of activities in a single case (performed by all involved users in that case) is 148 and there are seven cases with more than 100 activities.

Looking into the data it turns out that, in most cases, more than one user is involved in a case, thus at least one editor is looking at a case, besides the owner. Fig 8 depicts the number of cases where a certain number of users is involved. At most there are five users involved in a case. Looking the other way around, thus, determining the number of cases a particular user is involved in (see Fig 6) indicates that almost 90% of the users are owning less than six cases. As a consequence, data independence tests (see Sect IV-E) based on owners of cases are infeasible due to the low number of cases. Furthermore, since the main focus is on the time spend by the person reporting a publication, i.e., the case owner, in the following we are only looking at activities performed by the owner of a case.

D. Histogram based Cleansing

Based on the remaining cases in the event log, the next step is to investigate the histograms of activity durations with the same label over all cases. The aim is to find the most dominant duration which should reflect the mean duration of an activity. Please note that the duration of an activity is the difference between the observed completion time and the estimated start time minus the duration of the chunks observed in this time interval (see Sect IV-B). In [4] we investigated mainly two different approaches: slope based clustering and an approach based on kernel density

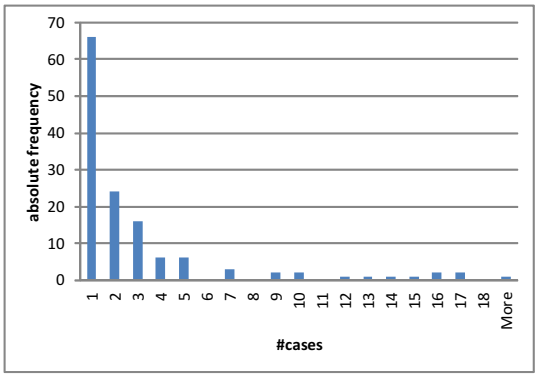

Figure 7. Histogram of number of cases owned by a particular user

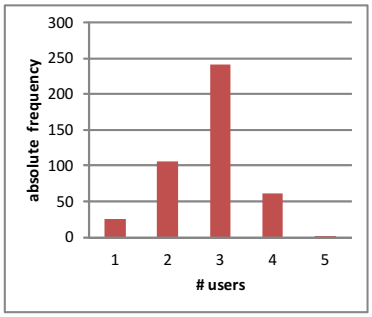

Figure 8. Histogram of the number of users involved in a case

estimation.

Kernel Density Estimation. The basic idea is to build a histogram of all observed durations of an activity and to use the bin in the histogram with the highest frequency as the mean of the main distribution. Since such an approach is dependent on the definitions of the bins in the histogram, an approach for estimating the probability density function is used, which is called kernel density estimation (kde) [5], [6]. The inferred kernel can be then used to create the density value for the complete domain. The mean of the activity durations is the maximum in the probability density function.

Slope based Clustering. The basic idea is that the mean activity duration has the strongest representation in the observed durations. Therefore, the aim is to find the largest subset of observed durations where the slope or gradient between two subsequent durations does not exceed a specific value. Furthermore, the subset must contain at least 5% of all observed durations. This can be formalised as follows: the observed durations can be described as an ordered set $X = \{x_0, \ldots, x_N\}$. The mean of the main distribution corresponds to the mean of the largest subset $X' \subseteq X$ with $X' := \{x'_0, \ldots, x'_n \in X | \forall j \in \{1, \ldots, n\}, x'_j - x'_{j-1} < \varepsilon\}$ of observed durations with the lowest threshold $\varepsilon \in \{0.1, \ldots, \frac{n}{2}\}$, where the size of the subset X' must be significant, thus it must contain more than 5% of the total

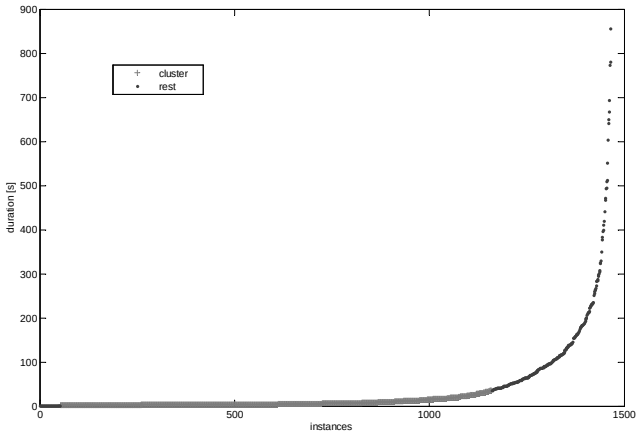

Figure 9. Clustering result example for activity *verify*

size of the data set $n > 0.05 * N$. An example of a cluster is depicetd in Fig 9 for the activity *verify*.

Applying both approaches on the data results in mean duration estimates of the different activities as given in Table I. The table provides the activity name as well as the duration estimates in seconds for the slope based clustering and the kernel density estimate (kde) based approach. Further, the number of instances contained in the log file and the number of instances contained in the cluster derived from the slope based approach are given. The ratio of the latter is an indication on how many values are actually influencing the estimate of the slope based cluster.

activity name	slope			kde
	estimate [s]	# instances	# cluster	estimate [s]
docmeta	19	552	483	10
done	6	894	772	2
files	3	51	7	9
fileview	13	1007	870	4
meta.core	5	538	402	2
meta.event	15	1544	1156	3
meta.pubinfo	14	649	398	2
meta.status	9	535	277	5
quickverify	10	277	137	7
return	20	1147	854	2
type	22	1432	977	2
verify	7	1465	1107	2

Table I
ESTIMATES BASED ON MAX DURATION PER ACTIVITY AND CASE OF THE INITIATOR OF THE PROCESS ONLY

The estimates are low for the slope based approach and are even lower for the kde based approach. The mean and standard deviation for cases are usually much higher than the estimates provided in Tab I. This has two reasons:

- First, there are activities which are not executed directly in sequence. For example the user starts a case but

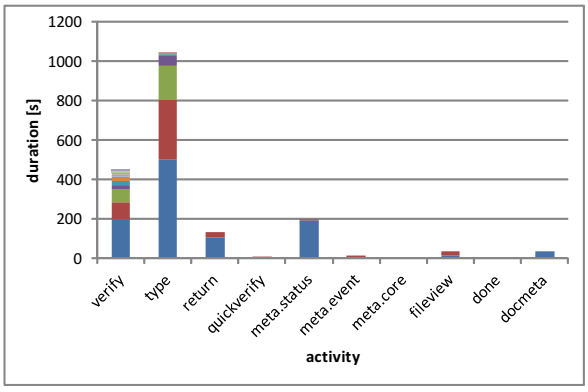

Figure 10. Activity Durations per Activity for Case 21150

he is missing some piece of information, such as, the document object identifier (DOI) and therefore stops the case and continues it a week later. As a consequence the duration of the first activity when the user picks up the case again is longer than a week, which significantly influences the mean and standard deviation.

- Second, it turns out that the duration of a few activity instances in a case dominate the duration of the case. In Fig 10 the durations for each activity are depicted, where instances of the same activity are denoted on the same horizontal position. The five instances with durations longer than 200 seconds dominate the duration of the overall case. The estimates derived above are addressing mean durations of instances, thus instances with a duration less than 100 seconds.

Based on these observations a filter on the maximum duration is introduced which cuts-off instances with a duration of more than 15 minutes, thus 900 seconds. The threshold of 15 minutes have been estimated by the owner of the data.

A further consequence of the above observations is that building a performance model for the duration of cases purely based on the duration estimates of activities will not be feasible. This is because the duration of a few activities dominates the duration of the complete case. As a consequence, the best estimate we can provide is based on the observed activity durations as available in this step fro activity durations below the threshold. The activities with durations above the threshold are replaced by duration estimates for the particular instance.

A separate investigation of activity instances with longer durations is not possible with the current log file since the number of instances is between 4 and 80 per activity. Thus, the number of instances for a statistical analysis is too small.

E. Data Independence Test

The aim of the data independence test is to see whether the assumption that the duration of instances of an activity are actually independent of a specific user or work day or any other additional property available. Unfortunately, the only information available in the log file is the user information. Furthermore, the number of cases per user is too small for such a statistical analysis (this is due to the fact that in average one person will produce a rather small number of publication in one year time). As stated in Sect IV-C almost 90% of the users have performed less than six cases.

V. Evaluation

The evaluation is based on 15 cases where users observed their own behavior and reported the time they spent on adding the information into the eprints system. The characteristics of the cases are given in Table II. This contains the case ID, the duration in minutes provided by the user, as well as the total number of activities performed by the user, i.e., the owner of the case.

Calculating the mean durations per activity for the actually observed executed activities, results in the estimated durations (measured in minutes) given in Table II. The estimates are under the observed execution time of a case significantly. The reason for that is that a few activity instances dominate the duration of the overall case as stated in Sect IV-D.

Therefore the approach is to use the observed durations and replace overly large durations of an activity by the activity estimate. As a threshold for overly large durations, based on discussions with the data owner, we decided to set the duration threshold to 15 minutes for a single activity. Applying this processing to the data results in three duration columns in Table II: the observed duration is the sum of durations of activity instances below the threshold; the ignored activity column is the sum of estimates per activity being above the threshold; the total duration column contains the sum of the previous two columns. The table also shows how many activities have a duration below the threshold as well as the total number of activities. The number of activities below and above the threshold are also depicted in Fig 11. Here the number of instances rejected compared to the number of accepted instances per activity are shown. As it can be seen, for many activities very little instances are excluded, except for *meta.status* activity, where about 38% of the instances are omitted. As expected the duration of the ignored activities is negligible compared to the duration of the activity instances below the threshold.

A visualization of the comparison between the evaluation duration reported by the owner of the process and the estimated duration is depicted in Fig 12. The worst results are reported for cases of shorter durations, in particular for cases 21287 and 21244. Here the error is actually 300% and 240%. However, there are 10 cases which have an error of less than 15%. The remaing three cases are below 60% error.

These results are rather good, in terms of accuracy, compared to the alternatives: a summation of the durations of all activity instances is much further off, since they are based on the same numbers but without the threshold.

case	duration [min]					# activities	
	user	estimated	observed	ign. act.	total	within cut-off	total
21150	22	8.1	32.4	0.5	32.9	31	34
21287	7	2.6	20.1	0.6	20.7	9	11
21244	8	6.3	18.9	0.6	19.5	21	24
21283	9	4.7	8.4	0.1	8.5	17	18
20666	10	3.8	10.0	0.0	10.0	14	14
21131	10	2.9	16.7	0.0	16.7	12	12
21248	10	5.7	9.0	1.0	10.0	18	22
21492	15	5.6	17.2	0.0	17.2	22	22
21448	15	8.7	18.2	0.7	18.8	32	35
21533	16	3.3	16.9	0.2	17.1	14	15
21496	17	5.1	15.2	0.2	15.4	21	22
21495	25	5.7	25.2	0.0	25.2	24	24
21447	30	9.1	28.6	0.7	29.3	36	39
21545	30	4.6	31.8	0.0	31.8	24	24
21544	35	5.9	34.8	0.2	34.9	23	24

Table II
SUMMARY OF THE EVALUATION RESULTS

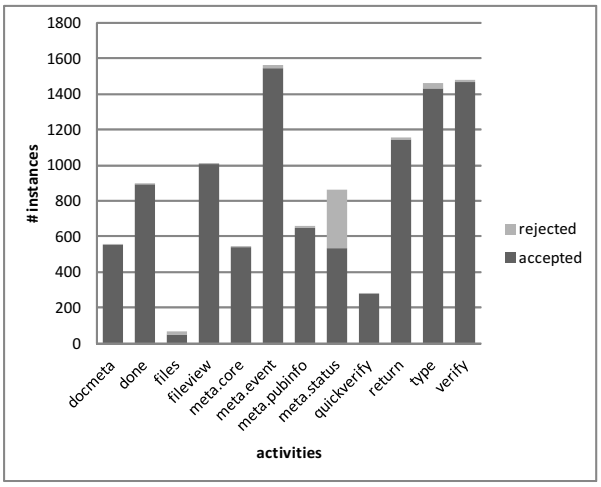

Figure 11. Duration cut-off per Activity Instance

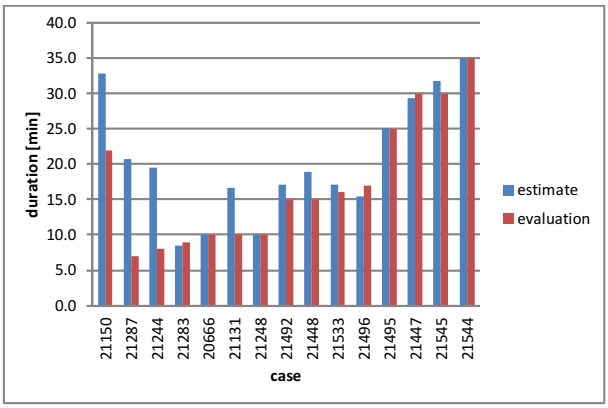

Figure 12. Comparison of the Evaluation Results

Thus, it adds at least 15 minutes for every activity instance above the threshold for each case. Further, this evaluation relies on the self reported observations of our users. We do not have any insight on how reliable (i.e., how precise) these measurements are. We performed a plausibility check, that is, we checked whether the sum of observed activity durations is at least as big as the duration provided by the owner of the case. We are convinced that these results are a good basis to estimate for example the time spend by personal to insert and maintain the digital library. This was also confirmed by the owner of the data.

VI. Related Work

Several approaches on performance model mining are relevant as related work. Some are related to ProM [7] and are based on event logs provided in the Mining Extensible Markup Language (MXML) [8]. Rozinat et al. [8] present an approach to mine simulation models from MXML event logs. The idea is to generate a process model, represented as a Colored Petri Net (CPN). Depending on the event log's richness, the resulting CPN may cover not only the control-flow perspective, but also the resource and performance perspective. However, all approaches around the ProM tool assume the event log contains both start and end times of an activity, which is not the case in our scenario.

There is also some literature making less assumptions on the available event logs. For example, in [9] the authors try to derive the relation between events and process instance assuming there is no explicit data available to make the link. In [10] the authors address noisy event logs and ways of dealing with it. However, the focus there is not on performance models.

Classical performance models, such as, Queuing Networks [11] or stochastic Petri Nets [12] assume that the complete system is modeled. The models can then be used either to perform an equilibrium analysis or a transient

analysis. In our situation the event log does not capture the complete system but only a part of it. To be able to apply classical performance models we have to make strong assumptions on the non-represented (parts of) the system(s).

It should be also noted that not all event logs are focusing on control flow performance mining. For example, in [13] the authors base their work on change logs, documenting ad-hoc changes performed on process instances. These change logs are then used to mine reference models.

VII. Conclusion

In this paper we elaborate a systematic approach to prepare event log data from semi-structured processes for the derivation of a performance model using a case study. In particular, the main goal is to estimate the start time of an activity in the process. This is necessary, since in a semi-structured process, activities are not always performed solely in one computer system and therefore the start time of an activity cannot be acquired automatically. The start time estimates are checked for outliers based on various errors and the independence of situational characteristics is checked. Future work will focus on testing the robustness of our approach using other data sets generated from logging processes from other sectors.

References

[1] A. Rozinat, R. S. Mans, M. Song, and W. Aalst, "Discovering simulation models," *Information Systems*, vol. 34, no. 3, pp. 305–327, 2009.

[2] A. Wombacher, M. Iacob, and M. Haitsma, "Towards a performance estimate in semi-structured processes," in *Service-Oriented Computing and Applications (SOCA), 2011 IEEE International Conference on*, dec. 2011, pp. 1 –5.

[3] P. H. Hartel, "On the cost and benefits of building a high-quality institutional repository," http://eprints.eemcs.utwente.nl/15019/, Centre for Telematics and Information Technology University of Twente, Enschede, Technical Report TR-CTIT-09-07, January 2009.

[4] A. Wombacher and M. Iacob, "Start time estimation in semi-structured processes," in *submitted to BPM 2012*, 2012.

[5] E. Parzen, "On the estimation of a probability density function and mode," *Annals of Mathematical Statistics*, vol. 33, pp. 1065–1076, 1962.

[6] M. Rosenblatt, "Remarks on some nonparametric estimates of a density function," *The Annals of Mathematical Statistics*, vol. 1956, pp. 832–837, 1956.

[7] B. Dongen, A. Medeiros, H. M. W. Verbeek, A. J. M. M. Weijters, and W. Aalst, "The proM framework: A new era in process mining tool support," in *Application and Theory of Petri Nets 2005*. Springer, 2005, pp. 444–454.

[8] A. Rozinat, R. S. Mans, M. Song, and W. M. P. van der Aalst, "Discovering colored petri nets from event logs," *STTT*, vol. 10, no. 1, pp. 57–74, 2008.

[9] H. Motahari-Nezhad, R. Saint-Paul, F. Casati, and B. Benatallah, "Event correlation for process discovery from web service interaction logs," *The VLDB Journal*, vol. 20, pp. 417–444, 2011.

[10] K. Musaraj, T. Yoshida, F. Daniel, M.-S. Hacid, F. Casati, and B. Benatallah, "Message correlation and web service protocol mining from inaccurate logs," in *IEEE International Conference on Web Services*, 2010, pp. 259–266.

[11] P. King, *Computer and Communication Systems Performance Modelling*. Prentice Hall, 1990.

[12] M. A. Marsan, "Stochastic petri nets: an elementary introduction," in *Advances in Petri Nets*, pp. 1–29.

[13] C. Li, M. Reichert, and A. Wombacher, "Discovering reference models by mining process variants using a heuristic approach," in *Business Process Management*, ser. LNCS. Springer Berlin / Heidelberg, 2009, vol. 5701, pp. 344–362.