Towards a better assessment of event logs quality

Mohammed Oussama Kherbouche, Nassim Laga, Pierre-Aymeric Masse
Orange Gardens
44 Avenue de la République
92320 Châtillon, France
Email : {mohammedoussama.kherbouche, nassim.laga, pierreaymeric.masse}@orange.com

***Abstract*—It is widely observed that the poor event logs quality poses a significant challenge to the process mining project both in terms of choice of process mining algorithms and in terms of the quality of the discovered process model. Therefore, it is important to control the quality of event logs prior to conducting a process mining analysis. In this paper, we propose a qualitative model which aims to assess the quality of event logs before applying process mining algorithms. Our ultimate goal is to give process mining practitioners an overview of the quality of event logs which can help to indicate whether the event log quality is good enough to proceed to process mining and in this case, to suggest both the needed preprocessing steps and the process mining algorithm that is most tailored under such a circumstance. The qualitative model has been evaluated using both artificial and real-life case studies.**

Keywords—event logs; process mining; process mining algorithms; qualitative model.

I. INTRODUCTION

The process mining is a research field which provides a bridge between data mining and business process analysis [1], [2]. It aims at discovering, monitoring and improving operational processes by extracting knowledge from event logs collected manually or generated by a variety of software applications [3]. However, a general consensus among process mining practitioners is that poor quality event logs often lead to wrong discovered process models, since the event logs quality has an influence not only on the process mining results but also on the choice of process mining algorithm (e.g. applying alpha algorithm [4] to real-life event logs is unrealistic, or applying heuristic miner [5] to less structured event log can provides spaghetti-like models). A decision tree is proposed in [6] to help the practitioners to decide which mining algorithm is appropriateness for each circumstance.

The importance of the event logs quality has been highlighted in several research studies. Indeed, according to [3], it is necessary to review the quality of event logs before proceeding in the process mining project, this suggestion was taken up by the process mining manifesto [2] which defines five event log maturity levels ranging from poor quality denoted by one star (★) to excellent quality denoted by five stars (★★★★★) in order, to judge in how far one can rely on the validity of event log data. However, no means was provided to assess the event logs quality in a concrete way. On similar lines, the authors, in [7], [8] have identified four classes of data quality issues arising in analyzing event logs in process mining, namely, (i), missing data (i.e. different information can be missing in event log although they are mandatory), (ii) incorrect data (i.e. some logged data within event log would be incorrect), (iii) imprecise data (i.e. some logged information within event log would be too coarse and imprecise), and (vi) irrelevant data (i.e. the logged information may be relevant for a particular context of analysis, but not for another). The authors have also strongly encouraged the process mining researchers to focus on techniques that address these quality issues.

In addition to the importance of the event log quality emphasized by these authors, there exist some log-based process metrics proposed in the literature, which serves to achieve different goals. For instance, in the process mining book [3], the author provides two metrics to measure the diversity of a data set. In fact, entropy is used to measure the diversity, whereas the Gini index of diversity measures the "impurity" of a data set.

Yang *et al.* [9], [10], [11] present a set of metrics and compare several estimators for evaluating the local and global completeness of event logs which is needed for the applicability of several process mining techniques [12], [13]. Finally, in [14] an approach to identify false traces in process event logs is proposed. The approach allows estimating the latent generation probabilities of observed traces by means of minimizing a distance function between the occurrence frequencies of traces and the occurrence probabilities of the traces respectively.

However, the review of related work led us to the conclusion that despite these efforts, there are, to date, still rather few initiatives to assess the event log quality before the process discovery steps in an objective manner. In this paper, we propose a qualitative model which allows evaluating event logs quality before applying process mining algorithms, in order to help the practitioners to cope with needed preprocessing steps and choose an appropriate process mining algorithm according to the event log quality. The proposed qualitative model shall be composed of a set of quality dimensions, quality attributes and measurements.

This paper is structured as follows: Section II provides some concepts needed to describe our approach. Then the proposed approach for assessing a quality of event logs is outlined in a step-wise manner in Section III. Section IV outlines some important metrics, used to evaluate different quality attributes, whereas Section V presents the realization of the proposed approach in the ProM plug-in. Section VI demonstrates the usefulness of our approach using a real-life

978-1-5090-4240-1/16/$31.00 ©2016 IEEE

case study and Section VII concludes the paper and outlines potential future work.

II. NOTATIONS

In this section, we give a formal description of the concepts used to describe our approach for assessing the quality of event logs.

A. Basic Definitions

The event log represents the starting point for any process mining project. It consists of a set of traces representing the records executions of a process model. In fact, a trace is seen as a sequence of ordered events by timestamps and which reflect the result of the completion of a process instance.

An event may also contains additional information or attributes like timestamp, cost, event type, resource, data elements, etc. The first attribute is required to analyze the performance related aspects of the process, whereas resource information such as the performer or originator which is involved in performing the activity is useful when analyzing the organizational perspective.

For a more formal definition of event logs used in process mining, the reader is referred to [4], [5].

B. Definition 1 (event log)

Let T be a finite set of tasks. T^+ is a set of all non-empty finite sequences of tasks from T. An event log over T is a multiset of traces defined as: $W = (E, C, A, \partial, \wp, \Gamma, \succ) \subseteq T^+$, where:

- *E* represents a finite set of events,
- *C* represents a finite set of cases,
- *A* represents a finite set of additional information which can be partitioned into disjoint sets of timestamps *S*, cost *O*, resources *R*, event types *I*, and data elements *D*,
- $\partial: E \rightarrow C$ represents a surjective function relating each event to a case,
- $\wp: A \rightarrow E$ represents a surjective function relating each additional information to an event,
- $\Gamma: E \rightarrow T$ represents a function relating each event to task,
- $\succ \subseteq E \times E$ represents a total ordering on the events in *E*. The ordered set of events belonging to a case is called "*trace*".

C. Definition 2 (trace)

Let T be a finite set of tasks, T^+ is a set of all non-empty finite sequences of tasks from T, and $W = (E, C, A, \partial, \wp, \Gamma, \succ) \subseteq T^+$. For all ordered events *E* relating to case $c \in C$, we can define a trace as $\sigma_c = \{e|$ for all $e \in E, \partial(e) = c\}$, the events in σ_c are totally ordered. Each trace corresponds to an execution of a process, and the same trace may appear multiple times in an event log. Henceforth, we denote a trace σ with $\sigma = \sigma(1), \sigma(2), \sigma(3) \ldots \sigma(n) \in T^+$. $|\sigma|$ represents the length of the trace *σ*. *σ (i)* is the i^{th} event in the trace and $\sigma(i, j) = \langle e_i \ldots e_j \rangle$ is the subsequence of σ that starts at position i and ends at position j.

D. Definition 3 (trace classes)

Let T be a finite set of tasks, T^+ is a set of all nonempty finite sequences of tasks from T, and $W = (E, C, A, \partial, \wp, \Gamma, \succ) \subseteq T^+$. Let $\sigma_1 \backslash CLP(\sigma_1, \sigma_2)$, $\sigma_2 \backslash CLP(\sigma_1, \sigma_2) \in T^+$ be two traces in event log. Let $CLP(\sigma_1, \sigma_2) = (LS_1 \cap LS_2)$ be a finite set of common loop patterns of both σ_1 and σ_2, where, $LS_i = \{LRES_1, LRES_2 \ldots LRES_n\} \subseteq \sigma$ represents a finite set of loop sequences in σ (see. section IV.A.6). We say that σ_1 and σ_2 are equivalent ($\sigma_1 \equiv \sigma_2$), iff: $\sigma_1(i) = \sigma_2(j) \Longrightarrow \Gamma(\sigma_1(i)) = \Gamma(\sigma_2(j)) \wedge |\sigma_1| = |\sigma_2|$ where $1 \leq i \leq n\ 1 \leq j \leq m$.

The trace σ_1 and σ_2 are equivalent, this means that each event of σ_1 and σ_2 refers to the same task and their lengths are equal.

Our definition of trace classes is similar to that in [9], except that, the loops which may occur in the traces are not taken into consideration in the comparison, since they dramatically increase the number of trace classes, due to presence of recurring behavior without yielding any additional information as mentioned by [9].

III. EVENT LOG QUALITY MODEL

As mentioned above, the availability of high-quality event logs is essential for a success of process mining. Indeed, the event data should be accurate, available and complete, otherwise we run the risk of drawing the wrong conclusions. Unfortunately the real-life event logs tend to be less structured, fine-granular, heterogeneous, noisy, imprecise, and/or voluminous.

The proposed event log quality model comes very handy. It aims to evaluate the quality of event log addressed before proceeding with process mining. It shall be composed of a set of dimensions, where each dimension represents a facet of the quality of event log, a set of quality attributes and measurements classified on four hierarchical levels as shown in Fig. 1.

Indeed, the evaluation of the root level, also denoted as Event log Quality (EveLoQ), depends on the evaluation of quality dimensions lying on the subsequent level. These include Complexity, Accuracy, Consistency, and Completeness. Each quality dimension of second level is devoted to a set of quality attributes for their precise evaluation. Finally, at the lower level, are the metrics which aims to capture a specific aspect of a quality attributes.

In the next paragraphs, different quality dimensions and corresponding attributes used in our event log quality model will be explained.

A. Complexity

In our proposed model, we define complexity as the degree to the simplicity with which the event log can be mined, and therefore the understandability of the model that would be discovered from the event log. In fact, there is a high correlation between event log complexity and the comprehensibility of discovered process model. This is partly due to highly complex real-life event logs exhibiting a large variety in process behavior. For this purpose, it would be convenient to measure the event log complexity prior to discovery steps because the calculation of process models is expensive when the resulting model is spaghetti-like.

The quality dimension of complexity can be specified with two attributes, structural complexity and behavioral complexity.

1) Structural Complexity: Structure is the syntactic means by which behavior can be specified in reference process model e.g. sequence, choice, parallelism, split, join, loop, etc. This can be reflected by the likely existence of loops, duplicate tasks, hidden tasks, a considerable amount of events and traces in a given event log. The structural complexity allows evaluating the presence of these elements in the event log.

2) Behavioral Complexity: This attribute measures the complexity of behavior present in the event log which refers to the number and intricacy of events in each trace and a variety between all traces within the event log.

B. Accuracy

The accuracy dimension allows assessing the degree to which the event log data reflects the underlying reality. i.e. if it satisfies the requirements of its intended use by describing what has happened in the real-life in accurate manner, since any inaccuracy in the event data could jeopardize the analysis results and hinders the extraction of meaningful insights.

The event log accuracy can be faced from different perspectives, precision, and trustworthiness which represent the attributes that can be part of this dimension.

1) Precision: The precision attribute gauges the level of preciseness, or the state of exactness of event data present in the event log. The event data precision concerns events, traces, timestamps, and/or resources.

2) Trustworthiness: According to [2], the events should be trustworthy, i.e., the recorded events have actually happened and that the attributes of events are correct. To evaluate the trustworthiness of event data, we need to answer questions like "Where did the event data come from? How trustworthy is the originator of event data (handle automatically or manually)? How many sources the event data come from? Are the original event data source trustworthy?".

C. Consistency

The consistency dimension attempts to estimate some characteristics such as: noise, number of elusive traces, etc, in order to ensure the definition, the understandability and

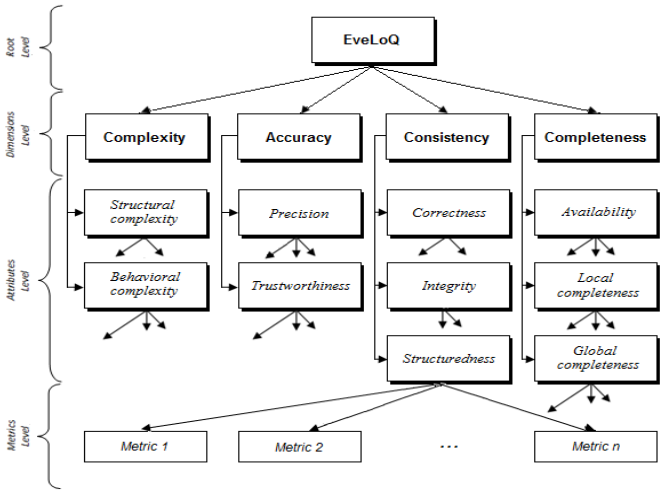

Fig. 1. The event log quality model.

the integrity of event data in event log, even if they come from heterogeneous sources.

To estimate the consistency of an event log, we propose some attributes that can be part of this dimension, namely, correctness, integrity, and structuredness.

1) Correctness: The correctness attribute evaluates the degree to which the different information containing in the event log can be judged correct. (e.g. no outlier data in the event log).

2) Integrity: The integrity attribute measures the validity of event data which can be compromised by human errors when data is entered, the heterogeneity of the data event sources or a hardware malfunctions. This attribute needs to ensure that each event refers to a case and each case consists of a sequence of well-ordered events.

3) Structuredness: Structuredness means the event data need to be available in a structured form i.e no free text that first needs to be processed into structured data.

D. Completeness

The completeness is undoubtedly reckoned as one of the most important evaluation dimension of event log quality on the same level as the complexity dimension. It refers to the extent to which required information is not missing and is of sufficient breadth for process mining effort (i.e. is all the requisite information available? Are event data missing or incomplete?). The completeness of an event log relies on the objective of the exact process mining task and the type of information which is needed. For example, in some cases, missing data is irrelevant, but when the information that is missing is mandatory to specific types of process mining (e.g. performers or originators for organizational miner [15]), completeness becomes an issue.

To estimate the completeness of an event log, we propose three attributes that can be part of this dimension, namely, availability, local completeness, global completeness.

1) Availability: The availability attribute refers to the degree of the availability of required event data in the event log. Indeed, missing events, missing cases, missing task names, missing timestamps, or missing resources, entails a result which do not match with the real execution of the cases, even worse, an inability to apply certain specific types of analysis such as organizational miner [15] and the social network miner [16].

For this purpose, it would be convenient to measure the event log completeness to determine which types of process mining could be envisaged and estimate the amount of available information in the given event log.

2) Local completeness: This attribute measures the local completeness of the given event log, i.e. whether all direct succession relations between tasks are recorded in the event log. This requirement is essential to applied the alpha algorithm [4], [17].

3) Global completeness: This attribute measures the degree of the occurrence of all possible behaviour of the process model in event log [9]. The global completeness of event logs is required for the application of some process mining algorithms as mentioned in [4], [17].

IV. METRICS FOR EVENT LOG QUALITY ASSESSMENT

When evaluating the quality of event logs, appropriate metrics need to be at hand although the measurement of some quality attributes as "*trustworthiness* (i.e. answer all the questions raised in section III.B.2)" are more difficult than others as "*correctness*". Due to the page number limitation, we outline some metrics proposed in our framework, used to evaluate different quality attributes.

A. Structural complexity measurements

The measurement of the structural complexity attribute can be done by using some straightforward metrics such as: event log length (L), number of process (NP), number of cases (NC), number of events (NE), number of resources (NR), number of trace classes (NTC), number of event classes (NEC) and average of loops per trace (ALT).

1) Event log length (L): The event log length L relates to the number of traces within an event log W [9]. It is defined as follows:

$$L = \sum_{\sigma \in W} \sigma \tag{1}$$

*2) Number of events (NE):*The number of events within an event log W represents the number of elements in E.

$$NE = \sum_{e \in E} e \tag{2}$$

3) Number of resources (NR): The number of resources in the event log W refers to the number of elements in $R \subset A$:

$$NR = \sum_{r \in R} r \tag{3}$$

4) Number of trace classes (NTC): NTC expresses the number of trace equivalences or trace classes as follows:

$$NTC = \sum_{i=1}^{L} \sum_{j=i+1}^{L} (\sigma_i \equiv \sigma_j) \mid \sigma_i, \sigma_j \in W \tag{4}$$

5) Number of event classes (NEC): The number of distinct events or event classes within an event log W is defined as:

$$NEC = n(E) \tag{5}$$

6) Average of loops per trace (ALT): The loops are manifested as the repeated occurrence of one or more events in one or several traces. To determine the set of the loop sequences of each trace, we retrieve what we have called the "*Longest Repeated Event Subsequence (LRES)*" of a trace that occurs at least twice, and which following the same principle to find the "*Longest Repeated Substring*" [19].

We can efficiently find *LRES* in σ in linear time (O (n) where n = |σ|) by building a suffix trees based on the Ukkonen's algorithm [20]. We define a loop sequence as:

Let $LS \subseteq \sigma$ be a finite set of loop sequences in σ where: $LS = \{LRES_1, LRES_2 \ldots LRES_n\} / \forall LRES_k \in LS$, $LRES_k = \sigma(i, j)$, $1 \leq k \leq n$ with $i \leq j$.

The formula (6) defines the average of loops per trace in a given event log W, as follows:

$$ALT = \frac{1}{L} \sum_{\sigma \in W} |LS| \tag{6}$$

B. Behavioral complexity measurements

We can evaluate the behavioral complexity by measuring some metrics such as: average trace length (ATL), average trace size (ATS), density (Dn), traces heterogeneity rate (THR), traces similarity rate (TSR), and complexity factor (CF).

1) Average trace length (ATL): The average trace length is defined as:

$$ATL = \frac{1}{L} \sum_{i=1}^{L} |\sigma| \tag{7}$$

where L be the event log length, as in (1) and |σ| represents the number of events per trace.

This metric is simple but could bias our judgment on the event log complexity, if it is not associated with other metrics. Indeed, a long trace does not mean that the event log is complex, since this may due to recurring behavior caused by loops.

2) Average trace size (ATS): The average trace size represents the average number of event classes per trace. It is defined as follows:

$$ATS = \frac{1}{L}\sum_{i=1}^{L} | n(e_i) | \quad (8)$$

where L be the event log length and $| n(e_i) |$ represents the number of event classes per trace.

3) Density (Dn): The density expresses the average of the number of event classes over traces, as in (8) to the average of the number of events over traces, as in (7) :

$$Dn = \frac{ATS}{ATL} \quad (9)$$

The density is used to indicate whether the average trace length arises from unique behavior or from recurring behavior. Indeed, when the metric value is close to 1, this means that the traces mostly contain a unique behavior, otherwise, when the metric value is close to 0, this means that the traces contain a recurring behavior caused by loops.

4) Traces heterogeneity rate (THR): The traces heterogeneity rate relates to the ratio of the number of trace classes, as in (4) to the number of traces within the event log L, as in (1). The traces heterogeneity rate is defined as follows:

$$THR = \frac{ln(NTC)}{ln(L)} \quad (10)$$

The metric is close to 1, this means that the event log contains a high diversity of behaviors, whereas, when the metric value is close to 0, this means that the event log contains a low diversity of behaviors.

5) Traces similarity rate (TSR): The traces similarity rate allows measuring the discrepancy that might occur between different traces within the event log by taking into account the different trace lengths. The metric can be defined as follows:

$$TSR = \frac{1}{NTC * (NTC-1)} \times \sum_{i=1}^{NTC}\sum_{j=i+1}^{NTC} \frac{max(|\sigma_i|,|\sigma_j|) - LD(\sigma_i,\sigma_j)}{max(|\sigma_i|,|\sigma_j|)} \quad (11)$$

where $LD(\sigma_i, \sigma_j)$ called also Levenshtein Distance allows to measure a distance between two traces σ_i and σ_j [21] and *NTC* represents the number of trace classes, as in(4).

The choice of the Levenshtein Distance which is a specific case of the generic edit distance is justified by the fact that the metric is able to compare two strings with different length.

The metric value can be interpreted as the mean extent of modifications necessary to transform any trace class $\sigma_i \in W$ to a trace class $\sigma_j \in W$ within the event log, which is in fact, a quite useful metric to express event log complexity.

6) Complexity factor (CF): The complexity factor metric investigated the degree of the understandability of the future model mined from the given event log W. Indeed, more the complexity factor value is high; more the discovered model will be complex. The metric can be computed as:

$$CF = (\ln(NTC)^{((1-TSR)+Dn)} \times ATS) \quad (12)$$

where 1 - TSR represents the traces dissimilarity rate, *NTC* is the number of trace classes, as in (4), *Dn* is a density, as in (9), and *ATS* is the average trace size, as in (8).

The *CF* is a numerical score that corresponds to a description of the complexity of the discovered model which has been established and refined through several experimentations using both artificial and real-life case event logs. These descriptions are given in Tab. 1 below.

TABLE I. THE COMPLEXITY FACTOR DESCRIPTIONS

Complexity Factor Numeric value	Description
0 – 30	Simple
30 – 53	Somewhat Complex
53 – 75	Complex
75 – 95	Very Complex
95 or higher	Highly Complex

C. Precision measurements

The precision can be evaluated with measurements such as: erroneous timestamps rate (ETR).

1) Erroneous timestamps rate (ETR): The erroneous timestamps rate evaluates too coarse or imprecise timestamps within the event log. This metric is defined as follows:

$$ETR = \frac{\sum_{i=1}^{NE} \neg match(t(e_i), P)}{NE} \; / \; t \in S, e \in E \quad (13)$$

where *P = "YYYY-MM-DD'T'HH:MM:SS[.fraction]"* is a pattern which represents a required timestamp format for each event, recall that *NE* is the number of elements in *E* and $t(e_i) \in S \mid S \subset A$ represents the timestamp relating to each event $e \in E$.

D. Availability measurements

Availability can be measured by detecting the number of missing timestamps, and missing resources. The following metrics can be used to achieve that goal: timestamps availability rate (TAR), and resources availability rate (RAR).

1) Timestamps availability rate (TAR): The timestamps availability rate relates to the ratio of the events without timestamps to the total number of events.

$$TAR = 1 - \frac{\sum_{i=1}^{NE} \wp'(t(e_i))}{NE} \quad (14)$$

where *NE* is the number of elements in *E*, and $\wp' \subseteq \wp: A \nrightarrow E \mid S \subset A, \exists t(e_i) \in S, \; \wp(t(e_i)) = \perp$ is a partial function

which is regarded as undefined on those events which do not have timestamps.

2) Resources availability rate (RAR): The resources availability rate relates to the ratio of the events without resources to its total number of events.

$$RAR = 1 - \frac{\sum_{i=1}^{NE} \wp'(r(e_i))}{NE} \quad (15)$$

where *NE* is the number of elements in *E*, and $\wp' \subseteq \wp: \mathrm{A} \nrightarrow \mathrm{E} \mid \mathrm{R} \subset \mathrm{A}, \exists \mathrm{r}(\mathrm{e_i}) \in \mathrm{R}, \wp(\mathrm{r}(\mathrm{e_i})) = \perp$ is a partial function which is regarded as undefined on those events which do not have resources.

To measure the local and global completeness, we have used the set of estimators proposed by Yang *et al.* [9], [10] and [11].

V. IMPLEMENTATION

The concepts discussed in this paper have been implemented as the event logs quality assessment plug-in in the ProM[1] tool. Indeed, the ProM is an extensible plug-in that provides a wide range of plug-ins for many different mining algorithms, as well as analysis, preprocessing operations, conversion, and export modules.

The event logs quality assessment plug-in implements the proposed qualitative model which consists of several quality dimensions, quality attributes and more than one hundred metrics. Through the interface presented in Fig. 2, the plug-in displays for each of the quality dimensions, the associated quality attributes and metrics according to our classification model in the form of a tree.

This plug-in starts with an event log and computes the different metric values. Then, the interpreter module compares each metric value with the corresponding threshold value indicating for what specific value the measures quality begins to decline. This is done in two stages. Firstly, a weighted sum is used to achieve the final score of each attribute quality. Indeed, the interpreter computes for each quality attribute A_k the sum δ_k of the associated metric values computed previously as follows:

$$\delta_k = \frac{1}{W} \sum_{i=1}^{n} \omega_i \, v_i \quad (16)$$

where $W = \sum_{i=1}^{n} \omega_i$ represents the sum of all assigned weights to the metrics, ω_i is weight of each metric (e.g. the weight of the CF metric is more important than the weight of the OHR metric in the assessment of the behavioral complexity attribute), and v_i is the metrics values.

After computing the weighted sum, the interpreter computes for the attribute quality A_k the sum δ'_k of the metrics thresholds values using the same method. Finally, the comparison between δ_k and δ'_k provides an assessment (high, medium or low) of the quality attribute A_k.

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

In the same way, and according to the quality attributes associated with the metrics, the interpreter gives an evaluation of the quality dimensions and therefore, the global quality of the event log based on a "bottom-up" evaluation.

This comparison is being conducted by using Drools[2], which is a business logic platform providing an integrated unified platform for rules, workflow, and event processing. We make use of it to write quality expert rules as well as rules to automate the quality assessment report generation.

Indeed, the plug-in describes in a report the information relating to the duration, the behavior, the accuracy, the consistency, and the completeness of event log. It allows also indicating whether the quality of event log we have as input is good enough to proceed in the process mining and in this case, the required preprocessing steps if necessary and suggests an appropriate process mining algorithm according to the quality assessment results (see. Tab. 2).

TABLE II. AN EXAMPLE OF A PREPROCESSING SUGGESTION RULE

```
rule "unlabelledEvents"
dialect "mvel"
when
 eval ((NumberUtil.parseDouble(MetricsRepository

.getInstance().getByMetricName("ENA").getValue()))>0
then
 processing.put("ue","Remove the unlabeled event(s)
by using plug-in name :\""+
PreProcessing.INDUCTIVE_VISUAL_MINER.getName() +
"\".");
end
```

The plug-in supports visualization of the significant variation of the different metrics through a set of charts and allows saving the generated report for further analysis.

VI. CASE STUDY

To validate the approach discussed in this paper, we have performed several artificial and real-life case studies. This section explains one of them in detail. The case study uses a process log from the "*BPI Challenge 2012*". It contains events related to the application process for a personal loan within a Dutch financial institute[3].

The application of the heuristic miner algorithm on the "*BPI Challenge 2012*" event log without any prior overview of the quality gives us an incomprehensible spaghetti-like process model. This result is the logical consequence of carrying out a process mining project without any overview of the quality of event log, since this would amount to exploring an unknown country without a roadmap.

Having analyzed the "*BPI Challenge 2012*" with our plug-in (see Fig. 2), it was felt that the event log contained 13 087 cases, 848 trace classes, 262 200 events distributed over 36 tasks and 69 distinct resources.

The traces length vary between 3 and 20 event classes, the average of events per case was to ATL = 20.035

[2] http://www.drools.org/

[3] https://data.4tu.nl/repository/uuid:3926db30-f712-4394-aebc-75976070e91f

Fig. 2. The Event Logs Quality Assessment Plug-in.

and the average of the different events per case was to ATS = 8.1765.

According to the average trace length and the event density, the plug-in concluded that the cases contained a recurring behavior caused by loops.

The total duration of the "*BPI Challenge 2012*" event log is 5 months 2 weeks 26 minutes 10 seconds 135 milliseconds. The average duration of each trace in the event log is 1 week 1 day 14 hours 58 minutes 20 seconds 95 milliseconds and the average duration of each event is 11 hours 31 minutes 13 seconds 323 milliseconds.

The number of loop patterns in the event log was to 40571 with an average of 3.10 per case. The average of traces frequency in the event log was to 15.43 and the average of events frequency was to 10 925. The plug-in concluded that the event log complexity seemed to be complex. Indeed, the traces are little similar and heterogeneous, which partly explains the initial results.

As regards the accuracy information, the plug-in shown that the "*BPI Challenge 2012*" event log shown that there exist 0,1091% of incorrect event timestamps values and there exist 3 end events which appear also as suspicious seeing their frequency.

The plug-in cautioned us about the presence of 0.0059% of noise, 38.26% of traces that are not ordered by timestamps, and no trace with the duplicate events (i.e. the events with the same attributes).

Overall, all of event names and event timestamps are available, but resource information availability rate is 93.13%. Indeed, there are 18 009 events of 3528 traces that have missing resource information, i.e., 6:87% of events and 26:98% of the traces have partially missing resource information. There are 1 042 traces where association between start and complete events of tasks are. The global completeness of the event log was high (i.e. tune of 88.71%).

Based on this analysis, the plug-in concluded that global quality of "*BPI Challenge 2012*" event log was medium, which will allow us to carry out our process mining project successfully. Indeed, the event log is complex (tune of 46.66%). The accuracy of the events and the traces was medium (tune of 60%). The consistency was medium (tune of 63.33%) and the completeness of the event log was high (tune of 76.66%). It suggested us also to applying several preprocessing steps, namely, filter the events with incorrect event timestamps values, filter the traces with an abnormal behavior, etc.

It's also advocated us to use fuzzy miner algorithm [22], which was appeared as the most tailored under this circumstance

During the various experiments conducted with our plug-in as raised previously, we have observed a limitation concerning the computation time. That the computation of the trace classes and the traces similarity rate takes a quadratic time O(n^2). In other words, the computation time increase with the scalability of the event log size. This requires the use of alternate techniques explained in the conclusion.

VII. CONCLUSION

The event logs quality is an important concern for any process mining project. However, despite its importance, there is currently very little work in assessing event logs quality before applying process mining algorithms. In this paper we have presented a qualitative model involving a set of quality dimensions refined in several quality attributes and measurements. We have also described some important metrics, used to evaluate different quality attributes.

Our approach has been fully implemented in the context of the ProM plug-in and evaluated using both real and synthetic event logs. It is noteworthy that our proposed event logs quality model is not exhaustive and can be enriched by empirical research.

As a future work, we intend to address current limitations of our proposal, namely use the sampling approach to speed-up the process of assessment, which is actually computationally expensive in the case of voluminous event logs. We wish also to gain more experience in how the metric values scale by conducting more comprehensive case studies in order to make the measurement process more reliable.

ACKNOWLEDGMENT

The authors are grateful to Orange (a French telecommunication operator) for funding the research in process mining. They also wish to thank all those who participated in the development of the process mining tool ProM, which enabled to integrate the proposed plug-in.

REFERENCES

[1] W.M.P van der Aalst, "Process mining: Overview and opportunities," *ACM Transactions on Management Information Systems* 3(2), pp. 1–17, 2012.

[2] W.M.P van der Aalst and *al.*, "Process mining manifesto," *Daniel, F., Barkaoui, K., Dustdar, S. (eds.) BPM Workshops 2011*, Part I. LNBIP, vol. 99, pp. 169–194. Springer, Heidelberg, 2012.

[3] W. M. P. van der Aalst, "Process Mining: Discovery, Conformance and Enhancement of Business Processes," *Springer-Verlag*, 2011.

[4] W. M. P. van der Aalst, A.J.M.M. Weijters, L. Maruster, "Workflow Mining: Discovering Process Models from Event Logs," *IEEE Trans. on Knowledge and Data Engineering* vol. 16, no. 9, pp. 1128-1142, 2004, doi:10.1109/TKDE.2004.47.

[5] A.J.M.M. Weijters, J.T.S. Ribeiro, "Flexible Heuristics Miner (FHM)," *BETA Working Paper Series, WP 334. Eindhoven University of Technology*, Eindhoven, 2010.

[6] G.T. Lakshmanan, R. Khalaf, "Leveraging process mining techniques to analyze semi-structured processes," *IT Professional*, vol.15, no. 5, pp. 22-30, 2013, doi: 10.1109/MITP.2012.88.

[7] R.S. Mans, W. M. P. van der Aalst, R. J. B.Vanwersch, "Process Mining in Healthcare: Evaluating and Exploiting Operational Healthcare Processes," *SpringerBriefs in Business Process Management*, 2015.

[8] R. Bose, R. Mans, W. M. P. van der Aalst, "Wanna improve process mining results?," *Proc. IEEE Symp. on Computational Intelligence and Data Mining (CIDM)*, pp. 127–134, Apr. 2013, doi: 10.1109/CIDM.2013.6597227.

[9] H. Yang, A. H. M. ter Hofstede, B. F. van Dongen, M. T. Wynn, J. Wang, "On global completeness of event logs," BPM Center Report BPM-10-09, 2010.

[10] H. Yang, L. Wen, J. Wang, "An Approach to Evaluate the Local Completeness of an Event Log," *Proc. IEEE 12th international conference on Data mining (ICDM)*, pp. 1164–1169, 2012, doi: 10.1109/ICDM.2012.66.

[11] H. Yang, B. Van Dongen, A. ter Hofstede, M. Wynn, J. Wang, "Estimating Completeness of Event Logs," Technical report, BPM Center, 2012.

[12] B. F. van Dongen, A. K. Alves de Medeiros, L. Wen, "Process Mining: Overview and Outlook of Petri Net Discovery Algorithms," *T. Petri Nets and Other Models of Concurrency*, vol. 2, pp. 225–242, 2009.

[13] L. Wen, W.M.P. van der Aalst, J. Wang, J. Sun , "Mining Process Models with Non-free-choice Constructs," *The Journal of Data Mining and Knowledge Discovery,* vol. 15, no. 2, pp. 145-180, 2007.

[14] H. Yang, L. Wen, J. Wang, "An Approach to Identifying False Tracesin Process Event Logs," *Proc. 17th Pacific-Asia Conference, PAKDD 2013*, pp. 533-545, 2013.

[15] M.S. Song, W. M. P. van der Aalst, "Towards Comprehensive Support for Organizational Mining," *Decision Support Systems* , vol. 46, no. 1, pp. 300-317, 2008.

[16] W. M. P. van der Aalst, H.A Reijers, M. Song, "Discovering social networks from event logs," *Comput Support Cooper Work (CSCW)* vol. 14, no. 6, pp. 549–593, 2005.

[17] B. F. van Dongen, A. K. Alves de Medeiros, L. Wen, "Process Mining: Overview and Outlook of Petri Net Discovery Algorithms," *T. Petri Nets and Other Models of Concurrency*, vol. 2, pp. 225–242, 2009.

[18] L. Wen, W.M.P. van der Aalst, J. Wang, J. Sun , "Mining Process Models with Non-free-choice Constructs," *The Journal of Data Mining and Knowledge Discovery,* vol. 15, no. 2, pp. 145-180, 2007.

[19] D. Gusfield, "Algorithms on Strings, Trees, and Sequences," *Computer Science and Computational Biology*, vol. 19, no. 8, pp. 521-585, 1997.

[20] E. Ukkonen, "On-Line Construction of Suffix Trees," *Algorithmica*. vol. 14, no. 3, pp. 249-260, 1995.

[21] R.P.J.C. Bose, W.M.P. van der Aalst, "Context Aware Trace Clustering: Towards Improving Process Mining Results," *SIAM International Conference on Data Mining*, pp. 401–412, 2009.

[22] C.W. Günther, W.M.P. van der Aalst, "Fuzzy Mining: Adaptive Process Simplification Based on Multi-perspective Metrics," *International Conference on Business Process Management (BPM 2007)*, Springer-Verlag, Berlin, pp. 328-343, 2007.