Enhancing Event Log Quality: Detecting and Quantifying Timestamp Imperfections

D.A. Fischer[1], K. Goel[2], R. Andrews[2], C.G.J. van Dun[1], M.T. Wynn[2], and M. Röglinger[1]

[1] FIM Research Center, University of Bayreuth, Project Group Business & Information Systems Engineering of the Fraunhofer FIT, Bayreuth, Germany
{dominik.fischer,christopher.vandun,maximilian.roeglinger}@fim-rc.de
[2] Queensland University of Technology, Brisbane, Australia
{k.goel,r.andrews,m.wynn}@qut.edu.au

Abstract. Timestamp information recorded in event logs plays a crucial role in uncovering meaningful insights into business process performance and behaviour via Process Mining techniques. Inaccurate or incomplete timestamps may cause activities in a business process to be ordered incorrectly, leading to unrepresentative process models and incorrect process performance analysis results. Thus, the quality of timestamps in an event log should be evaluated thoroughly before the log is used as input for any Process Mining activity. To the best of our knowledge, research on the (automated) quality assessment of event logs remains scarce. Our work presents an automated approach for detecting and quantifying timestamp-related issues (timestamp imperfections) in an event log. We define 15 metrics related to timestamp quality across two axes: four levels of abstraction (event, activity, trace, log) and four quality dimensions (accuracy, completeness, consistency, uniqueness). We adopted the design science research paradigm and drew from knowledge related to data quality as well as event log quality. The approach has been implemented as a prototype within the open-source Process Mining framework ProM and evaluated using three real-life event logs and involving experts from practice. This approach paves the way for a systematic and interactive enhancement of timestamp imperfections during the data pre-processing phase of Process Mining projects.

Keywords: Process Mining · Event log · Data quality · Timestamps · Data quality assessment

1 Introduction

Process Mining, a sub-discipline of data science and business process management (BPM), is a powerful technique that allows deriving insights into business process performance and behaviour using historical records from event logs [1]. Previous research shows that Process Mining is widely applied in practice [7,10]. Thereby, using an event log as the fundamental data source for Process Mining, organizations gain insights into process performance, conformance of processes to

existing process models, and process improvement opportunities [26]. Reliable Process Mining results are, however, contingent on starting with high-quality event log(s) [1,4,13]. In practice, event logs are often far from the desired quality [7,26]. Therefore, an event log should not be naively used for Process Mining without ensuring that it is of adequate quality [1]. Data scientists spend up to eighty percent of their work with identifying, assessing, and remedying data quality issues [32]. Therefore, in the BPM community the interest in exploring the roots of data quality problems and the related assurance of accurate data is rising [2,32]. Following the words of Edward Deming ("you can't manage what you can't measure"), it is essential to have a set of metrics for detecting and quantifying event log quality [32].

However, to the best of our knowledge, research that focuses on the (automated) quality assessment of event logs remains scarce. We intend to bridge this gap in research specifically for timestamp-related data quality issues since timestamps are the principal means for ordering events and the foundation for many use cases [9,12,13]. Precise timestamps are essential to reproduce the correct ordering of activities and, thus, to obtain accurate process models (*discovery*), to measure the alignment between the process model and the actual process flow (*conformance*), and to determine effectiveness and efficiency in the execution of activities (*performance*) [9,12]. In contrast, inaccurate and coarse timestamps often lead to convoluted process models that may result in erroneous analyses [9]. This paper, therefore, focuses on the following research question: *How can we detect and quantify timestamp-related data quality issues in event logs?*

To address the research question, we adopt the design science research (DSR) paradigm [11] and develop an automated approach for detecting and quantifying timestamp imperfections in an event log. We evaluate the quality of timestamps in an event log across two axes: four levels of abstraction (event, activity, trace, log) and four quality dimensions (accuracy, completeness, consistency, uniqueness). We define 15 timestamp quality-related metrics and demonstrate how they can be computed. Following the DSR process by Peffers et al. [21] and the evaluation framework by Sonnenberg and vom Brocke [24], the remainder of this paper is structured as follows: In Section 2, we derive essential design objectives for a timestamp quality quantification approach from mature knowledge on data and event log quality. Section 3 introduces the required preliminaries and explains the approach employed. Section 4 demonstrates the prototype implemented in the Process Mining framework ProM. In Section 5, we describe the evaluation of our approach by using three real-life event logs and involving experts from research and practice. The paper concludes with Section 6.

2 Background

In this section, we present a brief overview of the background on data quality and event log quality research in Process Mining and propose two design objectives that underpin our approach.

It is well-recognised that the quality of input data is of high importance to Process Mining techniques [1,13]. Previous research, therefore, established data quality dimensions for evaluating data quality as they verify the "fitness for use" of data by data consumers [32]. A variety of studies on data quality dimensions exist that are renowned and widely adopted (see [18,22,29,30]). However, they often do not provide metrics for measuring data quality in databases or event logs. Hence, we conducted a scan of recent literature on data and event log quality while focusing on extracting metrics for quantifying event log quality.

We analysed 48 papers synthesized from a larger set of results (n=412) by abstract screening and subsequently by text screening [31]. We created a concept matrix in which we attributed data quality dimensions to each article and whether the studies provide metrics for quality quantification [31]. We identified six studies that provide metrics for data quality quantification [3,5,15,17,23,25]. Furthermore, we clustered 118 different data quality dimensions that were named by the literature based on syntactic and semantic similarities which finally led us to 25 different data quality dimensions.

Since we aim to optimize the automation of our technique, we excluded dimensions that are not quantifiable without user input according to the literature (e.g., objectivity, usability, valued-added). This reduced our set to eight different data quality dimensions. From these dimensions, we excluded dimensions, that do not align with our intention to quantify timestamp quality (e.g., conciseness, understandability) and decided on the four data quality dimensions: *accuracy*, *completeness*, *consistency*, and *uniqueness*. We specify the following design objective (DO):

> **DO 1.** An approach for detecting and quantifying timestamp imperfections should consider multiple data quality dimensions, such as *accuracy*, *completeness*, *consistency*, and *uniqueness*.

Regarding event log quality, which refers to the data quality of event logs, the IEEE Task Force on Process Mining provides maturity levels for the suitability of different data sources for Process Mining [13]. For instance, they categorize semantically annotated logs of BPM systems as the pinnacle of high-quality event logs ($\star\star\star\star\star$) while they rank paper-based medical records to the lowest level ($\star$). They consider logs that at least fulfil the conditions of 3-stars ($\star\star\star$) as suitable for Process Mining techniques. However, most real-life logs do not comply with these conditions as they tend to be incomplete, noisy, and imprecise [7,10].

To understand which quality issues affect event logs, Suriadi et al. [26] proposed eleven event log imperfection patterns often found in real-life logs. Three of these patterns highlight timestamp-related issues, namely *form-based event capture*, *inadvertent time travel*, and *unanchored event* [26]. The *form-based event capture* pattern describes the existence of sets of events that mostly occur at the same time. Electronic-based forms creating multiple events with equal timestamps commonly cause the presence of this pattern. The *Inadvertent time travel* pattern outlines the existence of inaccurate timestamp values that lead to incorrect ordering of events. The *unanchored event* pattern characterizes timestamps that are recorded in a different format than expected from the Process Mining

tool. The confusion between the month-day and day-month format is a frequent manifestation of the named pattern.

Beyond the mentioned timestamp-related issues, we identified other factors that may impact Process Mining analysis. *Mixed granularity of traces* may cause incorrect event ordering [9]. For instance, in a hospital log, 'admission' may be recorded at second-level granularity, e.g., '03 Feb 2019 10:23:12', while within the same trace 'examination by the doctor' may be recorded at hour-level, e.g., '03 Feb 2019 10:00:00'. In the discovered process model, this will lead to incorrect orderings: the 'admission' activity follows the 'examination by the doctor' activity. In the majority of cases, the 'admission' activity may happen before the 'examination by the doctor', but, as the example has shown, in some instances, mixed granularity may cause *incorrect and infrequent event ordering* [9].

There exists an approach which tests for an *inconsistent format* to detect the *unanchored event* pattern. This aims, for instance, to identify date specifications in the format common in the United States ('MM-DD-YYYY'). Studies also discover timestamp issues through *stacked or parallel events* [8,19]. For instance, let the doctor submit a form at the end of each examination in which he declares having 'measured blood pressure', 'measured temperature', and 'intercepted airways'. Submitting a form may lead to three events in the log containing the same timestamp, i.e. the time the form was submitted (see *form-based event capture* in [26]) rather than the time these three events happened. We also identified research that addresses the issue of *overlapping events*, which aims to detect resource overlap between events. For instance, according to the log, a nurse may begin a new patient transfer before completing the previous patient transfer [6]. As a result, for our approach, we specified the following objective:

> **DO 2.** An approach for detecting and quantifying timestamp imperfections should consider existing imperfection detection approaches, such as *inconsistent format* (satisfies the *unanchored event* pattern), *mixed granularity of traces*, *infrequent event ordering* (satisfies the *inadvertent time travel* pattern), *overlapping events*, and *stacked events* (satisfies the *form-based event capture* pattern).

3 Approach

3.1 Preliminaries

Before we present our approach, we introduce required definitions and preliminaries. An event log is the necessary input to gain insights into a recorded process via Process Mining techniques. Central to Process Mining are the notions of *case* and *trace* in such event logs. A case is the set of events carried out in a single process instance with a *trace* being the execution sequence of the events in a case. Consequently, to conduct Process Mining analysis, an event log needs to contain, at a minimum, enough information such that each event record can be attributed to a case, and can be placed in sequence within the case. For ordering events, timing information (e.g., date and time) of when an event occurred is

frequently used, although some discovery algorithms rely on implicit ordering of events in an event log instead of explicit timestamps. Optionally, an event record may contain attributes such as a resource and costs [7,9,13].

Definition 1 (event, attribute, event log). *Let $\mathcal{E}$ be the set of all possible event identifiers and $AN = \{a_1, a_2, .., a_n\}$ be the set of all possible attribute labels. For each attribute $a_i \in AN$ $(1 \leq i \leq n)$, $\mathcal{D}_{a_i}$ is the set of all possible values for the attribute a_i. For any event $e \in \mathcal{E}$ and an attribute name $a \in AN$, we denote $\#_a(e) \in \mathcal{D}_a$ as the value of attribute named a for event e.*

For any event $e \in \mathcal{E}$ we define the following standard attributes: $\#_{id}(e) \in \mathcal{D}_{id}$ is the event identifier of e; $\#_{case}(e) \in \mathcal{D}_{case}$ is the case identifier of e; $\#_{act}(e) \in \mathcal{D}_{act}$ is the activity label of e; $\#_{time}(e) \in \mathcal{D}_{time}$ is the timestamp of e; $\#_{ts}(e) \in \mathcal{D}_{ts}$ is the lifecycle transition of e. We also define $\#_{res}(e) \in \mathcal{D}_{res}$ as the resource who triggered the occurrence of e as an optional attribute. An event log $\mathcal{L} \subseteq \mathcal{E}$ is a set of events.

Definition 2 (case, trace). *Let $\mathcal{C}$ be the set of all possible case identifiers. For any $c \in \mathcal{C}$ and an attribute name $a \in AN$, we denote $\#_a(c) \in \mathcal{D}_a$ as the value of the attribute named a for case c. We denote $\mathcal{E}^*$ as the set of all finite sequences of events over $\mathcal{E}$ where a finite sequence of length n over $\mathcal{E}$ is a mapping $\sigma \in \{1, \cdots, n\} \rightarrow \mathcal{E}$ and is represented as $\sigma = \langle e_1, e_2, \cdots, e_n \rangle$ where $e_i = \sigma(i)$ for $1 \leq i \leq n$. We define the special attribute $\#_{trace}(c) \in \mathcal{E}^*$ as representing the trace of case c, which consists of all events associated with c. We denote $\hat{c} = \#_{trace}(c)$ as shorthand for referring to the trace of a case and further note that the ordering in a trace should respect timestamps, i.e. for any $c \in \mathcal{L}, i, j$ such that $1 \leq i \leq j \leq |\hat{c}| : \#_{time}(\hat{c}(i)) \leq \#_{time}(\hat{c}(j))$.*

Next, we define quality metrics and position each metric along two axes: four quality dimensions (accuracy, completeness, consistency, uniqueness) and four levels of abstraction (event, activity, trace, log).

Definition 3 (quality dimensions, quality metrics, abstraction levels). *Let $DQ = \{dq_1, dq_2, \cdots, dq_n\}$ be the set of quality dimensions labels, $M = \{m_1, m_2, \cdots, m_n)$ be the set of quality metrics labels and $V = \{v_1, v_2, \cdots, v_n)$ be the set of quality attributes labels. Let $LL \subseteq AN^*$ be the set of all possible log abstraction levels. For any $ll \in LL$ we denote $\mathcal{E}_{ll}$ as the set of all event identifiers such that for any event $e \in \mathcal{E}_{ll}$ only attributes $a \in ll$ are accessible. We define some special log abstraction levels as: $ll_{event} = \{eventid,\ timestamp\}$; $ll_{activity} = \{eventid, activity\ label, transition, timestamp\}$; $ll_{trace} = \{eventid, traceid, transition, timestamp\}$; $ll_{log} = AN$.*

For any metric $m \in M$ and quality attribute $v \in V$ we denote $\#_v(m) \in \mathcal{D}_v$ as the value of quality attribute v for metric m where $\mathcal{D}_v$ is the set of all possible values for the quality attribute v. We define the following attributes for each metric $m \in M$: $\#_{score}(m) \in [0, 1]$; $\#_{weight}(m) \in \mathbb{R}^+$; $\#_{ll}(m) \in \mathcal{D}_{ll}$ and $ll \in LL$; $\#_{dq}(m) \in \mathcal{D}_{ll}$ and $dq \in DQ$. In a similar way we denote: $\#_{score}(dq) \in [0, 1]$ and $dq \in DQ$; $\#_{weight}(dq) \in \mathbb{R}^+$ and $d \in DQ$; $\#_{score}(ll) \in [0, 1]$ and $ll \in LL$.

	TIMESTAMP QUALITY			
	QD_1: Accuracy	QD_2: Completeness	QD_3: Consistency	QD_4: Uniqueness
Log Level		M_5: Missing Trace[c]	M_9: Mixed Granularity of the Log[c] M_{10}: Format[a]	M_{13}: Duplicates within Log[c]
Trace Level	M_1: Infrequent Event Ordering[a] M_2: Overlapping Events per Resource[a]	M_6: Missing Activity[c]	M_{11}: Mixed Granularity of Traces[a]	M_{14}: Duplicates within Trace[b]
Activity Level		M_7: Missing Event[c]	M_{12}: Mixed Granularity of Activities[c]	M_{15}: Duplicates within Activity[c]
Event Level	M_3: Future Entry[c] M_4: Precision[c]	M_8: Missing Timestamp[c]		

Table 1. Timestamp quality assessment framework

3.2 Detection and quantification of timestamp imperfections

Table 1 depicts our proposed framework to detect and quantify timestamp imperfections in an event log. We measure the quality of timestamps at various log abstraction levels (event, activity, trace, log) [1], using the four data quality dimensions *accuracy*, *completeness*, *consistency*, and *uniqueness* (DO 1). In total, we defined a set of 15 novel quality metrics, consisting of metrics based on existing detection approaches[a] (DO 2), modifications of existing detection approaches[b] and ten new detection approaches that we designed ourselves[c] based on insights from literature. However, we found no metrics that apply to accuracy at log or activity level. Nonetheless, the framework should be seen as an extensible foundation in event log quality quantification and, thus, further metrics or dimensions can be integrated.

We now describe all four quality dimensions and one exemplary metric for each dimension and show how we detect timestamp-related quality issues and quantify each metric to receive a score between 0 and 1. Due to lack of space, we provide details on all 15 metrics here: `http://bit.ly/33hz4SM`.

QD_1: Accuracy describes the deviation of the recorded value from the real value represented by the data [29]. Following the stated definition, we allocated every metric that investigates imprecise timestamps to accuracy. At the event level, we inspect the *precision* (M_4) of timestamps. We examine to what granularity the timestamp of an individual event is recorded in the log. For instance, we consider the quality of a timestamp that contains information down to millisecond as optimal while hour-level granularity is not. We also developed the metric *future entry* at the event level, which indicates whether there are future-dated timestamps in an event log. At the trace level, we quantify accuracy through the

metric *infrequent event ordering* as this phenomenon often occurs as a result of inaccurate timestamps [9]. Infrequent event ordering also covers the *inadvertent time travel* pattern since this pattern typically manifests itself in the existence of some traces in which event ordering deviates significantly [26]. We also detect *overlapping events* per resource provided that the start and end times of an activity that is executed by a resource are recorded in an event log. This can indicate imprecise recordings of start or end timestamps of activities if we assume that a resource does not multitask since the metric identifies activities that are started by a specific resource before they finished their prior activity [6].

M_4: Precision (Quality dimension: accuracy, abstraction level: event) aims to detect events containing coarse timestamps. Particularly events that are mostly recorded manually show coarse granularity as it is difficult for the user to provide information, for instance, about milliseconds granularity.

Detection. For each event $e \in \mathcal{L}$, we determine (i) the granularity $g(e)$ by investigating up to which time unit $tu \in \mathcal{TU} = \{year, month, ..., millisecond\}$ an event is recorded, and (ii) the number $\#_{tu < g(e)}(e)$ of time units tu that are more granular than $g(e)$.

Quantification. By default, the scores of all metrics should indicate a similar influence on Process Mining techniques. Hence, we assign a power value to each metric. Using this and $\#_{tu < g(e)}(e)$ of each event e, we calculate the score of *precision* with the following equation:

$$\#_{\text{score}}(precision) = (1 - \frac{\sum_{e \in \mathcal{L}} \#_{tu < g(e)}(e)}{6 * |\mathcal{L}|})^2 \quad (1)$$

QD_2: Completeness manifests as the recording of all values for a specific variable [29]. We quantify completeness at the event level through the metric *missing timestamp* of events. For this purpose, we examine whether a timestamp is recorded for each event. Furthermore, if an event contains a timestamp before the year 1971, this timestamp is also considered as missing, since many systems convert time-related null values into the year 1970 (so-called Unix time). At the activity level, we check for *missing events* (M_7). Since an activity requires at least an event with a start transition and an event with a complete transition, we consider an activity that either has no start event or end event as an indicator of a missing event. At the trace level, we aim to detect *missing activities*. To identify this issue, we first scan the event log for infrequent predecessor-follower relations. If a trace contains an infrequent relation, the trace is investigated to see whether it contains an activity that frequently follows the predecessor under investigation. Unless we are unable to find a frequent follower, we assume that an activity is missing in this trace. Lastly, we detect possible *missing traces* at the log level by examining differences between timestamps of the first events of two consecutive traces. We consider a gap between two traces that is significantly larger than the expected mean value as an indicator of a missing trace in between.

M_7: Missing event (completeness, activity) describes cases in which an activity is either missing a start or an end event. Potential reasons are failures or

omissions in recording the start or end event, or the expected event has been mapped to the wrong trace.

Detection. In each trace, we count all events e with $\#_{act}(e) = a$ and $\#_{ts}(e) = start$. We then count all events e in the trace with $\#_{act}(e) = a$ and $\#_{ts}(e) = complete$. If the two counts are not the same, the trace contains missing events for activity a.

Quantification. To calculate a score, we use the ratio of affected activities to total activities and the ratio of affected traces to total traces in the log. Let $\mathcal{L}_{act} = \{\#_{act}(e) | e \in \mathcal{L}\}$ be the set of all activity labels and $\mathcal{L}_{case} = \{\#_{case}(e) | e \in \mathcal{L}\}$ be the set of all case identifiers in log $\mathcal{L}$. We define $affected : \mathcal{L} \times \mathcal{L}_{case} \times \mathcal{L}_{act} \rightarrow \mathbb{N}$ such that $affected(\mathcal{L}, c, a)$ returns the difference between the number of events $e \in \mathcal{L}$ with $\#_{case}(e) = c$, $\#_{act}(e) = a$ and $\#_{ts}(e) = start$ and those with $\#_{ts}(e) = complete$. $\#affected\ activities = \sum_{c \in \mathcal{L}_{case}, a \in \mathcal{L}_{act}} affected(\mathcal{L}, c, a)$. Let $T = \{\#_{case}(e) | e \in \mathcal{L}, affected(\mathcal{L}, \#_{case}(e), \#_{act}(e)) > 0\}$ be the set of traces which contain at least one affected activity and $\#affected\ traces = |T|$. Then Thus, we calculate the score for using the following equation:

$$\#_{\text{score}}(missing\ events) = (1 - \frac{\#affected\ activities}{2 * |\mathcal{L}_{act}|} - \frac{\#affected\ traces}{2 * |\mathcal{L}_{case}|})^4 \quad (2)$$

QD$_3$: Consistency means the equal representation of data values in all events [29]. As multiple units are necessary to evaluate equality, it is not possible to measure consistency for a single event at the event level. At the activity level, we group all events according to their activity label. After that, we examine whether all timestamps of a specific activity show the same granularity because mixed granularity potentially leads to the wrong order of activities [9]. We also inspect individual traces for *mixed granularity of traces* at the trace level. For this metric, we group the events according to their trace IDs. Furthermore, at the log level, we compare the granularity of all events applying the metric *mixed granularity of the log* (M_9) regardless of trace and activity labels. We also assess whether all event timestamps have been recorded in the same day-month *format*. As addressed by the imperfection pattern *unanchored event* [26], a log may contain timestamps that have been stored in a day-month format and timestamps that have been stored in a month-day format when logs from multiple systems are combined in a single event log.

M$_9$: Mixed granularity of the log (consistency, log) measures the extent of mixed granularity of the events in the log. We check whether certain events are recorded in coarser or finer granularity. This may occur, among others, if parts of the log are recorded automatically through electronic systems while other parts are recorded manually by the user.

Detection. $g(e)$ of each event $e \in \mathcal{E}$ by investigating up to which time unit $tu \in \mathcal{TU} = \{year, month, ..., millisecond\}$ an event is recorded. Then, we count how often each time unit tu was found as the most precise granularity and determine the dominating time unit $max(\mathcal{TU})$. Thereupon, we calculate the

observed granularity distribution and the expected granularity distribution. We calculate the observed distribution as follows:

$$d_{\text{obs}}(tu) = \frac{tu}{|\mathcal{TU}|} \tag{3}$$

We use $max(\mathcal{TU})$ for the calculation of the expected distribution. For instance, let $max(\mathcal{TU}) = second$, we assume that $d_{\text{exp}}(second) = 59/60$. The remaining 1/60 is distributed accordingly over the coarser time units. From there, we calculate the deviation of the two distributions:

$$dev = \sum_{tu \in \mathcal{TU}} \frac{|d_{\text{exp}}(tu) - d_{\text{obs}}(tu)|}{2} \tag{4}$$

Quantification. To determine a score for the metric *mixed granularity of the log* we use the following equation:

$$\#_{\text{score}}(mixed\, gran.\, (log)) = (1 - dev)^2 \tag{5}$$

QD_4: Uniqueness is defined as the existence of unwanted duplicates within or across systems for a particular data set [29]. At the activity level, we measure uniqueness using the metric *duplicates within activity*. Therefore, we detect all activities that contain more than one event with the same timestamp. Frequently detected issues include identically timestamped start and end events for the same activity. We identify *duplicates within trace* (M_{14}) at trace level. This metric detects events of a trace that show the same timestamp. As the imperfection pattern *form-based event capture* shows, several events of a trace are often recorded using the same timestamp through e-forms, even though they happened in a sequence [26]. Finally, applying the metric *duplicates within log*, we reveal events that do not belong to the same trace but use the same timestamp. This issue may also be caused by electronic form-based event storage, for instance, if the user can record multiple trace start events through an e-form.

M_{14}: Duplicates within trace (uniqueness, trace) aims to detect events with an exact same timestamp in the same trace. As the imperfection pattern *form-based event capture* shows, recording of events after the fact (e.g., via e-forms) often causes multiple events stored with equal timestamps, although these events did not happen concurrently in real life [26].

Detection. As duplicates within a trace we describe a set of n events $\mathcal{E}_{\text{dt}} \subseteq \mathcal{E}$ where for every event $e_{\text{i}} \in \mathcal{E}_{\text{dt}}$: $\#_{\text{trace}}(e_{\text{i}})$ is equal, $\#_{\text{time}}(e_{\text{i}})$ is equal, and $\#_{\text{act}}(e_{\text{i}})$ is unequal.

Quantification. To calculate a score, we use the ratio of detected events to total events, and the ratio of affected traces to traces. Let $\mathcal{L}_{time} = \{\#_{time}(e) | e \in \mathcal{L}\}$ be the set of all timestamps, and $\mathcal{L}_{case} = \{\#_{case}(e) | e \in \mathcal{L}\}$ be the set of all case identifiers in log $\mathcal{L}$. We define $duplicates : \mathcal{L} \times \mathcal{L}_{case} \times \mathcal{L}_{times} \rightarrow \mathbb{N}$ such that $duplicates(\mathcal{L}, c, t)$ returns the number of events $e \in \mathcal{L}$ with $\#_{case}(e) = c$ and $\#_{time}(e) = t$ having distinct values of $\#_{act}(e)$.

Let $\#detected\ events = \sum_{c \in \mathcal{L}_{case}, a \in \mathcal{L}_{ts}} duplicates(\mathcal{L}, c, t)$.

Let $T = \{\#_{case}(e) | e \in \mathcal{L}, duplicates(\mathcal{L}, \#_{case}(e), \#_{time}(e)) > 1\}$ be the set of traces which contain at least one timestamp with multiple events and $\#affected\ traces = |T|$. Thus, we calculate the score for the metric *duplicates within trace* with the following equation:

$$\#_{\text{score}}(duplicates\ (trace)) = (1 - \frac{\#detected\ events}{2 * |\mathcal{L}|} - \frac{\#affected\ traces}{2 * |\mathcal{L}_{case}|}) \quad (6)$$

4 Implementation

4.1 Software prototype

Figure 1 shows the main window of the prototype that displays the results of metrics and calculated scores in Panel A after importing an event log in XES format. The top row and the left column indicate the aggregated scores for each quality dimension and event log level. The scores are also visualized with colours: red, yellow, green for low (under 0.25), medium (0.25 - 0.75), and high (above 0.75), respectively. These thresholds are configurable in the prototype.

By using the "details" button for a metric, the prototype provides the user with a list of detected issues for a particular metric (Panel B). For instance, using the metric *duplicates within trace*, the list shows the case id, event label, timestamp, and lifecycle transition for all events detected by the corresponding

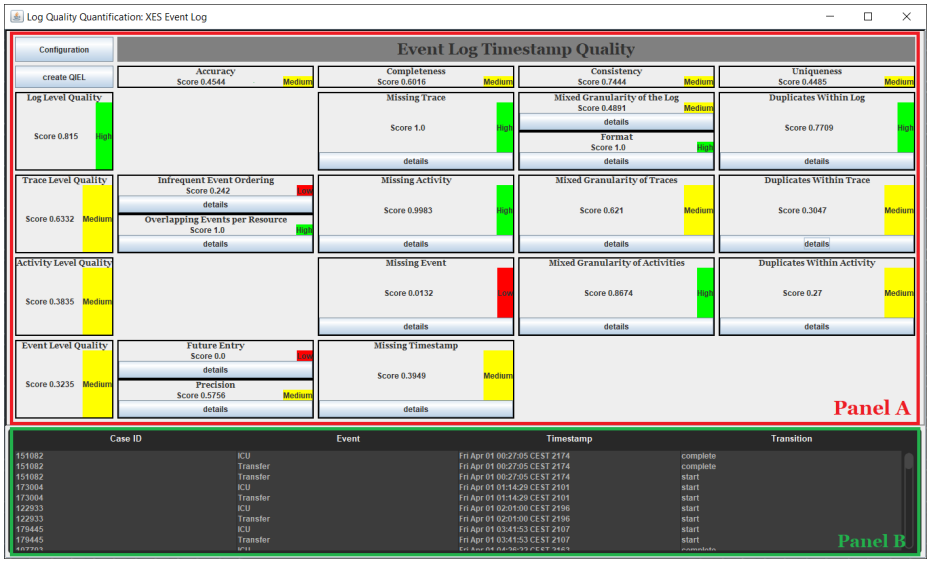

Fig. 1. Main window: the quality quantification of the MIMIC-III log (panel A) and the error list (panel B) of the metric *duplicates within trace*

metric as duplicate events. Thus, the user is able to discover which events are affected. This list can also be sorted by all columns.

A separate "User Configuration" window (not shown here) can be accessed through the button "Configuration". It allows the user to suppress metrics or dimensions which are not of concern and to adjust the weight of metrics. The default weight per metric is one. Any positive real number can be assigned. The accumulated dimension and event log scores are then recalculated for the new configuration. The option of suppressing irrelevant measures or dimensions and adjusting the weight of metrics turns our approach into a domain-agnostic solution, as it allows the user to customize the timestamp quality quantification for the specific use case and to address the issue of potential dependencies between metrics. For instance, the results of the metric *precision* may have an impact on the results of the consistency and uniqueness metrics. Thus, the user can hide the metric *overlapping events per resource* if multitasking is possible in the examined use case or lower the weight of *precision* if millisecond granularity is not necessary in the case under consideration. Using the "create QIEL" (QIEL = "quality-informed event log") button, the configuration information set by the user as well as the assessment results are stored in the metadata of the imported XES log file. Thus, the quality information can be used in later Process Mining phases.

4.2 Sources

The implementation of the automated approach for detecting and quantifying timestamp imperfections is available in the ProM nightly build which can be downloaded here: `http://bit.ly/38KVKvJ` [28]. The source code is available in the package "LogQualityQuantification": `http://bit.ly/39OAgj0`. In addition, we provide detailed instructions for the use of the prototype in the appendix (`http://bit.ly/33hz4SM`) and describe the implemented features.

Among the three event logs used for the evaluation, log A and log B are logs from Australian partners and, therefore, cannot be made available publicly in accordance with relevant Australian legislation. Log C, however, is openly available and can be accessed after completing the CITI "Data or Specimens Only Research" course here: `http://bit.ly/3aNRnkW` [14].

5 Evaluation

5.1 Evaluation strategy

Aligning with DSR purposes, our overall goal is to evaluate the usefulness and applicability of the proposed approach and give an indication towards its real-world fidelity [24]. The evaluation, therefore, involves analyzing three diverse real-life event logs and comparing the outcomes created by the application of our approach to the manual assessment of log experts. We define log experts as persons with Process Mining experience who are capable of making informed

data quality statements about a particular event log. In showing that our approach returns similar quality assessments as log experts in three varying circumstances and in a fraction of the time needed for manual analysis, we provide evidence that the approach adequately supports users in detecting and quantifying timestamp-related data quality issues in event logs and, therefore, addresses our research question. The three used logs are:

- **Log A**: represents the activities related to processing of 2090 annual progress reports for PhD students at an Australian University
- **Log B**: represents the waypoints (dispatched, on scene, at patient, ...) in over 40,000 ambulance attendances to, and transport to hospital of, patients injured in road traffic crashes in Queensland, Australia
- **Log C**: represents an openly available data set comprising desensitized health data associated with 40,000 critical care patients. It includes demographics, vital signs, laboratory tests, medications, and more [14]

We decided on these logs, as many of the addressed issues are present (i.e. insufficient precision and duplicates in log A, ordering and granularity issues in log B, and future timestamps and completeness issues in log C).

We defined an iterative evaluation process (Figure 2). One evaluation round contains the following steps: First, the timestamp quality of log A was assessed using the prototype and its results compared to a quality report manually created by a corresponding log expert A. If the quality report created by expert A matched the outcomes of the prototype closely enough (see definition below), we proceeded with expert B and log B and, subsequently, with expert C and log C. When, at one point in this process, the outcomes of the prototype differed from the results reported by the expert, we reconfigured and improved the prototype based on the insights from the respective evaluation round and subsequently started a new round, again starting with log A. The process terminates when the prototype results match the experts' quality reports in all three cases.

The manual reports are based on ordinal quality levels (low $\widehat{=}$ 1, medium $\widehat{=}$ 2, and high $\widehat{=}$ 3), the prototype output has been scaled down accordingly. To determine whether the assessment of an event log by the prototype matches the report created by the respective expert, we calculated the agreement ratio and reliability value α using Krippendorff's alpha [16]. Following the recommendations of Krippendorff, we consider the outcomes as reliable if $\alpha \geq 0.667$ [16].

We decided first to run the evaluation within the author team to preconfigure the prototype. Thus, three co-authors (who have worked with the respective logs

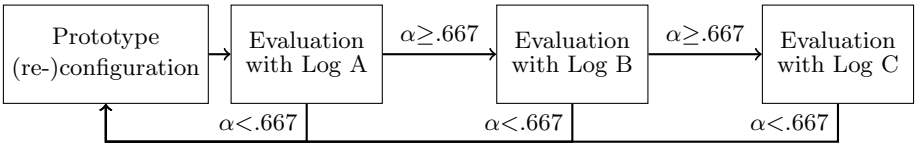

Fig. 2. Evaluation process (performed both with internal and with external experts)

before) represented the log experts for log A, B, and C until the approach passed each evaluation step. After the internal evaluation was completed, we also went through the evaluation process with three external log experts from academia and practice. The detailed results are presented in the following.

5.2 Findings

Overall, one evaluation within the author team and one evaluation with external experts were completed and one prototype reconfiguration was performed. Below, we describe these evaluation rounds and the implemented changes.

For the **first internal evaluation round** (see Table 2, 1.A), the assigned quality levels of expert A regarding log A and the results of the prototype did not match closely enough. A number of changes were made in the prototype as a result. One main difference is the score for the metric *form-based event capture*. Expert A is certain that no events of log A were recorded through electronic forms and, therefore, the score should not be less than 3. After further analysis, we noticed that this metric detects sets of events which have the same timestamps in the same trace but for different activities. Such a pattern does not only occur due to the *form-based event capture* pattern but can also be caused by other

	1.A			2.A			2.B			2.C			3.A			3.B			3.C		
	P_s	P_{ql}	A	P_s	P_{ql}	A	P_s	P_{ql}	B	P_s	P_{ql}	C	P_s	P_{ql}	A	P_s	P_{ql}	B	P_s	P_{ql}	C
M_1	.616	2	3	.616	2	3	.674	2	1	.242	1	2	.616	2	3	.674	2	2	.242	1	2
M_2	1	3	3	1	3	3	1	3	2	1	3	3	1	3	3	1	3	3	1	3	3
M_3	1	3	3	1	3	3	1	3	3	0	1	1	1	3	3	1	3	3	0	1	1
M_4	.683	2	2	.683	2	2	.586	2	2	.576	2	2	.586	2	3	.586	2	2	.576	2	2
M_5	.977	3	-	.977	3	-	.983	3	3	1	3	3	.977	3	3	.983	3	3	1	3	-
M_6	.936	3	3	.936	3	3	.947	3	3	.998	3	3	.936	3	3	.947	3	3	.998	3	2
M_7	.951	3	2	.741	2	2	0	1	1	.013	1	1	.741	2	2	0	1	1	.013	1	1
M_8	1	3	3	1	3	3	1	3	3	.395	2	2	1	3	3	1	3	3	.395	2	1
M_9	.956	3	3	.956	3	3	.393	2	2	.489	2	2	.956	3	3	.393	2	2	.489	2	2
M_{10}	1	3	3	1	3	3	1	3	3	1	3	3	1	3	3	1	3	2	1	3	3
M_{11}	.918	3	3	.918	3	3	.533	2	2	.621	2	2	.918	3	3	.533	2	2	.621	2	3
M_{12}	.956	3	3	.956	3	3	.972	3	3	.867	3	3	.956	3	3	.972	3	3	.867	3	3
M_{13}	.695	2	3	.869	3	3	.601	2	2	.771	3	2	.869	3	3	.601	2	2	.771	3	3
M_{14}	.488	2	3	.397	2	2	.820	3	2	.305	2	2	.397	2	2	.820	3	3	.305	2	2
M_{15}				.509	2	2	.999	3	3	.270	2	3	.509	2	2	.999	3	3	.270	2	3
M_{16}[1]	.522	2	3																		
AG	**64.29%**			**92.86%**			**80.00%**			**80.00%**			**86.67%**			**93.33%**			**64.29%**		
α	**0.082**			**0.842**			**0.777**			**0.801**			**0.670**			**0.890**			**0.692**		

P_s = indicated scores of the prototype; P_{ql} = indicated quality levels of the prototype; A, B, C = quality levels assigned by experts; AG = agreement; α = reliability

Table 2. Evaluation of the applied metrics

[1]The initial version of the prototype (for the first evaluation) contains the metric *M_{16}: form-based event capture* instead of the metric *M_{15}: duplicates within activity*.

reasons. Thus, we removed the metric *form-based event capture* and proposed a new metric *duplicates within activity* that detects sets of events with the same timestamp, the same event name and in the same trace. Originally, the metric *duplicates within log* detected sets of events with the same timestamp and same event name. However, using this detection method, we identified issues that are already covered by *duplicates within activity* or *duplicates within trace*. Thus, we modified the metric *duplicates within log* so that it detects sets of events with the same timestamp but in different traces. For the metric *missing event*, we adjusted the score quantification. Initially, we considered only the ratio of affected events to total events. For event log A, just 1.26% of total events were affected and, thus, the prototype assigns a high score to the metric *missing event*. However, expert A observed that 12.85% of the total traces were affected by *missing events* and, therefore, expected the score to be medium. Hence, we concluded that, in terms of score quantification, it is necessary to consider the ratio of affected traces to total traces.

In the **second internal evaluation round** (see Table 2, 2.A-2.C), the outcomes of the reconfigured prototype met the expectations of each expert for at least 12 of 15 metrics. As we obtained sufficient reliability, we assumed the internal evaluation to be successful.

We continued with an **evaluation round involving external experts** from academia and practice who were not engaged in the design of the approach. A data expert from an Australian university acted as the expert for log A, a data scientist working in the healthcare domain as the expert for log B, and a research associate from Germany with Process Mining focus as the expert for log C. Each of these external experts has gathered expertise with the corresponding log during previous research or industry projects and is therefore capable of providing information on all relevant quality characteristics of these logs. In all steps of the evaluation with external experts (see Table 2, 3.A-3.C), we obtained sufficient reliability and, therefore, assumed the external evaluation to be successful.

Our evaluation allowed us to improve our approach iteratively. Morever, by using real-life logs and automatically delivering outcomes matching the quality levels manually assigned by log experts from academia and practice in a fraction of the time needed for manual analysis, we demonstrate the approach's applicability in practice. Real-world fidelity is indicated by successfully using the prototype with three logs with different characteristics. We received feedback from the experts regarding the usefulness of both our approach and the prototype: The experts confirmed that timestamps are the cornerstones of event logs and should be analyzed in detail. Therefore, the approach as a way of detecting and quantifying timestamp quality issues in event logs was deemed useful to practitioners and researchers alike. This applies to both the proposed framework (Table 1) as well as the underlying set of metrics. At the same time, the experts frequently proposed two extensions to further increase the approach's usefulness and completeness: (i) including domain-specific metrics to increase the approach's usefulness in highly specific fields and (ii) including more intelligent metrics to increase the explanatory power and guidance of the approach.

The implementation as part of the ProM framework allows for interoperability with other tools and was therefore welcomed by all experts. In summary, we are confident that the approach and its implementation adequately support users in detecting and quantifying timestamp quality issues in event logs and address our research question.

6 Conclusion

Following DSR principles, we designed and implemented an approach for detecting and quantifying timestamp imperfections in event logs based on 15 novel data quality metrics structured along four data quality dimensions and log levels each. The applied metrics and dimensions were subsequently evaluated using real-life event logs and involving experts from academia and practice. Our framework focuses on timestamp quality issues and provides a first step in quantifying event log quality. The approach can detect common timestamp-related issues and measure the quality of timestamp information in event logs. Furthermore, our approach is domain-agnostic (e.g. by suppressing irrelevant metrics or adjusting the weight of metrics). Thus, we support process stakeholders in determining the suitability of an event log for Process Mining analysis. We also assist data scientists in automatically identifying and assessing data quality issues in event logs. Finally, our approach paves the way for future research on detecting and quantifying quality issues of further event log components (e.g., activity labels).

These insights come with limitations: First, we intend to minimize the risk that existing issues remain undetected (false negatives). This sensitivity, however, can cause the approach to identify issues that may be false alarms (false positives) such as uniqueness issues caused by batched events [20]. However, we mitigate potential over-detection by allowing users to review detected issues and plan to implement a white-list functionality as part of future work. Second, the evaluation was only performed on three different logs and under the assumption that expert assessments are correct. Thus, we are running the risk of the prototype being configured improperly if experts are biased or the logs only capture very special situations. We consider this risk to be low, given that the logs cover a broad range of characteristics and the assessments of the internal and external experts for each log were very similar. Although we followed established principles and systematically identified potential timestamp-related quality issues from the literature and our evaluation showed promising results for the approach and the prototype, a more thorough evaluation involving further logs and experts should be conducted in the future (see [24]).Thereby, evaluations (with case studies or controlled experiments) need to focus on generality and completeness of the metrics and further examine the real-world fidelity of the approach's results.

We also identified areas where the approach can be extended. First, we want to provide a more interactive detection approach whereby fine-grained user configuration can be taken into account (e.g., for minimum and maximum values or rules for certain metrics to detect violations). Second, the framework should

be seen as a foundation for future extensions. Therefore, we want to encourage researchers to introduce further metrics to underpin the quality dimensions. Third, our approach can also be extended with other log attributes such as event labels. Moreover, research so far lacks a systematic approach on how to define quality measures. Our work, therefore, constitutes a starting point to design a set of axioms for the definition of measures [27]. Finally, a natural extension to this work is to provide the user with an opportunity to repair the detected timestamp issues in a similar manner to those presented in [8,9]. Our vision is to provide an integrated approach to detecting, quantifying, repairing and tracking (timestamp) quality issues in event logs.

Acknowledgements:

We would like to thank Queensland's Motor Accident Insurance Commission and the Queensland University of Technology for allowing us access to their datasets.

References

1. van der Aalst, W.M.P.: Process mining: data science in action, vol. 2. Springer, Berlin, Heidelberg (2016)
2. van der Aalst, W.M.P., Bichler, M., Heinzl, A.: Responsible Data Science. Business and Information Systems Engineering **59**(5), 311–313 (2017)
3. Alkhattabi, M., Neagu, D., Cullen, A.: Assessing information quality of e-learning systems. Computers in Human Behavior **27**(2), 862–873 (2011)
4. Andrews, R., van Dun, C.G.J., Wynn, M.T., Kratsch, W., Röglinger, M.K.E., ter Hofstede, A.H.M.: Quality-informed semi-automated event log generation for process mining. Decision Support Systems **132**(3) (2020)
5. Askham, N., Cook, D., Doyle, M., Fereday, H., Gibson, M., Landbeck, U., et al.: The six primary dimensions for data quality assessment (2013)
6. Awad, A., Zaki, N.M., Di Francescomarino, C.: Analyzing and repairing overlapping work items. Information and Software Technology **80**, 110–123 (2016)
7. Bose, R.P.J.C., Mans, R.S., van der Aalst, W.M.P.: Wanna improve process mining results? In: CIDM 2013. pp. 127–134. IEEE (2013)
8. Conforti, R., la Rosa, M., ter Hofstede, A.H.M.: Timestamp repair for business process event logs. Tech. rep., University of Melbourne (2018)
9. Dixit, P.M., Suriadi, S., Andrews, R., Wynn, M.T., ter Hofstede, A.H.M., Buijs, J.C.A.M., et al.: Detection and interactive repair of event ordering imperfection in process logs. In: CAiSE 2018. pp. 274–290. Springer, Cham (2018)
10. Emamjome, F., Andrews, R., ter Hofstede, A.H.M.: A Case Study Lens on Process Mining in Practice. In: OTM 2019. pp. 127–145. Springer, Cham (2019)
11. Gregor, S., Hevner, A.R.: Positioning and presenting design science research for maximum impact. MIS quarterly pp. 337–355 (2013)
12. Gschwandtner, T., Gärtner, J., Aigner, W., Miksch, S.: A taxonomy of dirty time-oriented data. In: ARES 2012. pp. 58–72. Springer, Berlin, Heidelberg (2012)
13. IEEE Task Force on Process Mining: Process mining manifesto. In: BPM 2011. pp. 169–194. Springer, Berlin, Heidelberg (2011)
14. Johnson, A.E.W., Pollard, T.J., Lu, S., Lehman, L.w.H., Feng, M., Ghassemi, M., et al.: MIMIC-III, a freely accessible database. Scientific data **3**, 160035 (2016)

15. Kherbouche, M.O., Laga, N., Masse, P.A.: Towards a better assessment of event logs quality. In: IEEE SSCI 2016. pp. 1–8. IEEE (2016)
16. Krippendorff, K.: Reliability in Content Analysis. Human communication research **30**(3), 411–433 (2004)
17. Lee, Y.W., Pipino, L.L., Funk, J.D., Wang, R.Y.: Journey to data quality. The MIT Press (2009)
18. Lee, Y.W., Strong, D.M., Kahn, B.K., Wang, R.Y.: AIMQ: a methodology for information quality assessment. Information and Management **40**(2), 133–146 (2002)
19. Lu, X., Fahland, D., Andrews, R., Suriadi, S., Wynn, M.T., ter Hofstede, A.H.M., et al.: Semi-supervised log pattern detection and exploration. In: OTM 2017. pp. 154–174. Springer, Cham (2017)
20. Martin, N., Swennen, M., Depaire, B., Jans, M., Caris, A., Vanhoof, K.: Retrieving batch organisation of work insights from event logs. Decision Support Systems **100**, 119–128 (2017)
21. Peffers, K., Tuunanen, T., Rothenberger, M.A., Chatterjee, S.: A design science research methodology for information systems research. Journal of management information systems **24**(3), 45–77 (2007)
22. Pipino, L.L., Lee, Y.W., Wang, R.Y.: Data quality assessment. Communications of the ACM **45**(4), 211–218 (2002)
23. Sattler, K.U.: Data quality dimensions. In: Liu, L., Özsu, T.M. (eds.) Encyclopedia of Database Systems, pp. 612–615. Springer (2009)
24. Sonnenberg, C., Vom Brocke, J.: Evaluations in the science of the artificial. In: DESRIST 2012. pp. 381–397. Springer, Berlin, Heidelberg (2012)
25. Stvilia, B., Gasser, L., Twidale, M.B., Smith, L.C.: A framework for information quality assessment. Journal of the American Society for Information Science and Technology (2007)
26. Suriadi, S., Andrews, R., ter Hofstede, A.H.M., Wynn, M.T.: Event log imperfection patterns for process mining. Information Systems **64**, 132–150 (2017)
27. Tax, N., Lu, X., Sidorova, N., Fahland, D., van der Aalst, W.M.P.: The imprecisions of precision measures in process mining. Information Processing Letters **135**, 1–8 (2018)
28. Verbeek, H.M.W., Buijs, J.C.A.M., van Dongen, B.F., van der Aalst, W.M.P.: XES, XESame, and ProM 6. In: CAiSE 2010. pp. 60–75. Berlin, Heidelberg (2010)
29. Wand, Y., Wang, R.Y.: Anchoring data quality dimensions in ontological foundations. Communications of the ACM **39**(11), 86–95 (1996)
30. Wang, R.Y., Strong, D.M.: Beyond accuracy: what data quality means to data consumers. Journal of Management Information Systems **12**(4), 5–33 (1996)
31. Webster, J., Watson, R.T.: Analyzing the past to prepare for the future: writing a literature review. MIS Quartely **26**(2), 13–23 (2002)
32. Wynn, M.T., Sadiq, S.: Responsible Process Mining - A Data Quality Perspective. In: BPM 2019. pp. 10–15. Springer, Cham (2019)