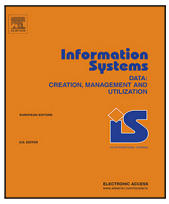
Information Systems 141 (2026) 102766

Contents lists available at ScienceDirect

Information Systems

journal homepage: www.elsevier.com/locate/is



Object-centric event-data imperfection patterns

Sareh Sadeghianasl [a,*], Moe Thandar Wynn [a], Robert Andrews [a], Wil M.P. van der Aalst [b], Jonghyeon Ko [a]

[a] Queensland University of Technology, Brisbane, Queensland, Australia
[b] RWTH Aachen University, Aachen, Germany

ARTICLE INFO

Keywords:
Process mining
Object-centric event log
Data quality
Patterns

ABSTRACT

The field of process mining offers a range of techniques for evidence-based improvement of business processes. The quality of the process data used, as stored in so-called event logs, is paramount to the reliability and usefulness of the process mining outcomes. Due to the increased uptake, at scale and for more complex types of applications, the field of process mining has evolved and event logs now need to be object-centric rather than event-centric. To understand and manage the quality problems that can occur in object-centric event logs a systematic approach is required that is different from past investigations into event-centric logs. To this end, we adopt a pattern-based approach, a tried and tested method to characterise problems that are otherwise hard to capture. A new collection of patterns is presented for object-centric logs, where each pattern captures the nature of the problem, how its manifestation can be detected, and how the problem can be remedied. The pattern collection is validated through a multi-prong approach, i.e., evidence-based, literature-based, empirical, and user-based (with the process mining community). The results show that these patterns are perceived as important to identify and that they do occur in practical settings.

1. Introduction

Since the emergence of the field of process mining [1] over twenty years ago, several attempts have been made to create a standard for event logs, including MXML, XES, XOC, and Parquet [2]. The adoption of a standard log format simplifies data preparation, ensures consistent data exchange and analysis across different systems and tools, and leads to more reliable insights. From these, the aforementioned XES format (approved by IEEE [3]) surfaced as the most prominent. However, the increased uptake of process mining in practice in the last decade has highlighted limitations of this standard. Reality turns out to be more complicated than the assumptions that underlie the XES standard. For example, the XES log standard assumes that events belong to only a single case (where the notion of *case* specifies the viewpoint of a process execution). This is often not realistic as events may be part of multiple cases (viewpoints on process execution) as in, for example, the Order to Cash (O2C) process managed by ERP systems. Consequently, the process mining community is moving towards a new event log standard, referred to as *Object-Centric Event Data (OCED)*. In this standard, there is no single case notion; instead, multiple *objects* correlate (e.g., customers, orders, items, and invoices in the O2C process) and one event can involve more than one object. For instance, placing an order involves a customer as well as the order, adding an item to an order involves both the order and an item, and payment involves an invoice and a customer.

OCED enables Object-Centric Process Mining (OCPM). OCPM helps to address the following three limitations of traditional process mining: (1) data extraction is time-consuming and needs to be repeated when new questions involving different case perspectives emerge, (2) interactions between objects are not captured and objects are analysed in isolation, and (3) a 3D reality needs to be squeezed into 2D event logs and models. In traditional process mining, one needs to pick a case identifier for each event [4,5]. This means selecting a particular viewpoint that does not show all relevant objects and may also be distorted due to the convergence (one event may be related to multiple cases) and divergence (multiple instances of the same activity within a case) problems [6]. For every new viewpoint, data needs to be extracted and transformed. Using OCED, this is not needed, and OCPM techniques can create process models without the usual distortions and show the interactions between objects and events. Fortunately, most of the existing process mining techniques can be lifted to multiple object types [4,5,7,8]. To date, various process mining

* Corresponding author.
E-mail addresses: s.sadeghianasl@qut.edu.au (S. Sadeghianasl), m.wynn@qut.edu.au (M.T. Wynn), r.andrews@qut.edu.au (R. Andrews), wvdaalst@pads.rwth-aachen.de (W.M.P. van der Aalst), jonghyeon.ko@qut.edu.au (J. Ko).

https://doi.org/10.1016/j.is.2026.102766
Received 8 June 2023; Received in revised form 5 April 2026; Accepted 29 May 2026
Available online 3 June 2026

techniques have been proposed to deal with OCED. Some examples include object-centric process discovery techniques [9–11], object-centric conformance checking techniques [12,13], and object-centric process prediction techniques [14–16]. Case studies [17,18] show that OCPM delivers valuable improvement opportunities for problems that live at the intersection points of processes and organisational units.

Event logs may suffer from a range of data quality problems, which may have a detrimental effect on process mining analysis. Suriadi et al. [19] devised a collection of so-called event log imperfection patterns to systematically deal with data quality problems commonly observed in XES format event logs. The new object-centric approach gives rise to a new collection of associated data quality problems which are fundamentally different from the event log imperfection patterns introduced for "classical" event logs, e.g., specified in the XES format. To get a grasp on these problems, we also choose a pattern-based approach in this paper. The collection of event log imperfection patterns introduced in [19] has had a significant impact on the advancement of the field of data quality management in process mining, attracting hundreds of citations so far [20–23]. These include real-life case studies that show the prevalence of event log imperfection patterns [24–26] as well as detection and repair techniques for these patterns [27–32]. Furthermore, the event log imperfection patterns proposed by Suriadi et al. [19] have been used as a basis PraeclarusPDQ,[1] an open-source framework for data quality management in process mining. This framework is available to the community, and the academic paper is currently under review. A patterns-based approach allows us to characterise object-centric data quality problems in sufficient detail and, at the same time, it helps us avoid an overly formalised approach that may narrow their interpretation and may compromise readability. Furthermore, pattern collections provide a template that allows them to be extended – completeness is something that cannot be achieved, as that would require a framework that formalises that notion – as well as terminology that makes it easier to communicate about the underlying problems and their resolution.

The proposed collection of patterns constitutes an extensible repository of knowledge about data quality issues specific to object-centric event data to identify well-suited detection techniques and remedies for specific root causes. Although some of the patterns may resemble classical relational database quality issues, their manifestation in object-centric event data and their impact on process mining results are semantically distinct. OCED logs introduce multi-object relationships, object–event mappings, and object CRUD operations that are not addressed in traditional relational or event-centric quality frameworks. For example, patterns such as missing object–object relations or ambiguous object–event links are unique to the object-centric paradigm and have direct implications for process mining outcomes—such as distorted process models or broken cardinality constraints and therefore require new detection and remediation strategies.

As an example, consider an emergency services event log compiled from datasets recorded by disparate, non-interoperable information systems, e.g., ambulance transport, emergency department, ICU admissions, and pathology. Reconstructing complete patient histories is challenging when some of these datasets do not share a common patient identifier. Fig. 1 shows an example of such an object-centric event log extracted from four datasets: (1) ambulance transport dataset, where each transport has a unique identifier tr_i, (2) emergency department dataset, where each presentation is identified by an encounter ID en_i, (3) ICU admissions dataset, where each stay has a unique identifier icu_i, and (4) pathology dataset with lt_i identifying individual lab tests. As shown in Fig. 1, sometimes the transport object ID is not recorded when the ambulance arrives at the hospital, and the crew hands the patient over to the emergency department staff (events e_5 and e_6). This happens because these events are missing in the ambulance transport dataset and only exist in the emergency department dataset. Another problem is that after the patient is transferred to the ICU, lab tests are ordered and conducted (e_9–e_{12}), but they are linked to the emergency department encounter rather than the ICU stay. In other words, the link between events e_9–e_{12} and the ICU stay object is missing.

The aforementioned missing links between events and objects (if they occur frequently) can lead to discovering incorrect flows in Object-Centric Directly Follows Graphs (OC-DFG) as shown in Fig. 2. We can see that the 'Initiate transport' activity is incorrectly considered as the final activity of the transport object, whereas in reality, there are two more: 'Arrive at hospital' and 'Handover patient'. Without these two activities, the transport object's OC-DFG is a disconnected fragment of the rest of the graph, making it challenging to reconstruct complete patient histories to, e.g., answer questions like "What is the average time it takes from a patient making an emergency call to the time they are discharged from the hospital?". A similar issue can be observed where the 'Transfer to ICU' activity is directly followed by 'Consult specialists', missing the lab test activities in the ICU objects' flow and oversimplifying its process behaviour. This leads to inaccurate measurement of ICU performance KPIs. For instance, there seems to be a long, unexplained waiting time between transfer to the ICU and the first specialist consultation. Furthermore, it is challenging to measure the frequency of lab testing per ICU stay.

Developed following the design science research (DSR) methodology, the pattern collection is designed to fulfil predefined objectives. These objectives ensure that the pattern collection captures imperfections that (1) exist and are pervasive in real-life object-centric event data, (2) their recognition is important because of their negative impact on process mining outcomes, (3) they are conceptually grounded in alignment with existing OCED [33] and OCEL 2.0 [34] standards, and (4) concrete techniques can be developed for their detection and/or repair. We abstract from a specific object-centric representation and base the patterns on a more conceptual footing, one that is object-centric but only incorporates the key features specified in terms of a core technology-independent metamodel in alignment with the two metamodels, OCED [33] and OCEL 2.0 [34], that are currently available in the process mining community. Sometimes, data quality problems, as codified in our patterns, are not picked up during event data extraction from raw data sources. Naturally, their prevention would be preferable to their subsequent detection and repair, but we do present strategies in case they are not treated before their appearance in the extracted event log.

The remainder of this paper is organised as follows. Section 2 reviews the relevant papers to data quality of object-centric event data. In Section 4, we propose a core metamodel for object-centric event data. Section 5 proposes the collection of object-centric event data imperfection patterns. Finally, Sections 6 and 7 present the evaluation of the patterns and suggest future work.

2. Background and related work

Data quality and its impact on reliable data-driven decision-making have been a topic of study for many decades [35–37]. This section reviews the existing works in this area and is structured as follows. First, in Section 2.1, we provide a general background on data quality, its definition and dimensions. Next, Section 2.2, provides an overview of process mining, event log structure, event-data quality frameworks, and approaches to detect and repair data quality issues in event logs. Section 2.3 focuses on object-centric process mining, describes what it is, reviews some of the techniques for object-centric process mining, and the few studies that focus on data quality of object-centric event logs. Finally, Section 2.4 reviews existing work on data quality in object-centric databases, a closely related field to object-centric process mining.

[1] https://github.com/praeclaruspdq/PraeclarusPDQ

ID	Activity	Transport	Encounter	ICU Stay	Lab Test	Timestamp
e_1	Receive emergency call	$\{tr_1\}$				15/03/2026 06:30:00
e_2	Ambulance arrive on scene	$\{tr_1\}$				15/03/2026 07:10:00
e_3	Provide on-scene treatment	$\{tr_1\}$				15/03/2026 07:30:00
e_4	Initiate transport	$\{tr_1\}$				15/03/2026 08:15:00
e_5	Arrive at hospital		$\{en_1\}$			15/03/2026 08:50:00
e_6	Handover patient		$\{en_1\}$			15/03/2026 09:30:00
e_7	Triage		$\{en_1\}$			15/03/2026 10:10:00
e_8	Transfer to ICU		$\{en_1\}$	$\{icu_1\}$		15/03/2026 11:01:00
e_9	Order lab test		$\{en_1\}$		$\{lt_1\}$	15/03/2026 12:20:00
e_{10}	Collect and label sample		$\{en_1\}$		$\{lt_1\}$	15/03/2026 13:05:00
e_{11}	Send sample to pathology		$\{en_1\}$		$\{lt_1\}$	15/03/2026 13:25:00
e_{12}	Receive test results		$\{en_1\}$		$\{lt_1\}$	15/03/2026 18:11:00
e_{13}	Consult specialists		$\{en_1\}$	$\{icu_1\}$		15/03/2026 20:02:00
e_{14}	Administer treatment		$\{en_1\}$	$\{icu_1\}$		15/03/2026 22:45:00
e_{15}	Discharge from ICU		$\{en_1\}$	$\{icu_1\}$		17/03/2026 09:24:00
e_{16}	Admitted to ward		$\{en_1\}$			17/03/2026 10:25:00
e_{17}	Discharge		$\{en_1\}$			20/03/2026 12:30:00

Fig. 1. An example of Object-Centric Event Log with missing links between events and objects.

2.1. Data quality

Of late, data has been recognised as an asset, valuable to both business and research. Data quality is an easily understood concept, at least so far as "high" quality data is more desirable than "low" quality data. However, various attempts to objectively define the characteristics of high/low quality data [35,38,39] demonstrate how difficult it is to arrive at a universal understanding of data quality. The ISO/IEC 25012 standard [40] defines data quality as "the degree to which the characteristics of data satisfy stated and implied needs when used under specified condition", i.e., data quality is defined in terms of the data's *fitness-for-purpose*. Data quality has long been determined to be a multi-dimensional concept [36] with a subsequent survey of data quality literature [37] showing that among authors writing about data quality at the time, there was no consensus as to the dimensions relevant to assessing data quality, and even, to the definition of the dimensions themselves. However, the most frequently mentioned dimensions in [37] were *Completeness*: the extent to which data are of sufficient breadth, depth and scope for the task at hand; *Accuracy*: the extent to which data are correct, reliable and certified; *Timeliness*: the extent to which the age of the data is appropriate for the task at hand; *Consistency*: the extent to which data are presented in the same format and compatible with previous data; and *Accessibility*: the extent to which information is available, or easily and quickly retrievable.

2.2. Data quality in process mining

Process mining is a well-established field of research and practice that aims to understand and improve business processes through the application of various techniques such as process discovery, conformance checking and performance analysis [41]. The main input for process mining is an event log, which is a collection of *events*, each describing an *activity* occurred at a specific *timestamp*, related to one or more particular process instance(s), a.k.a *case(s)*. These necessary features add additional characteristics to describing and measuring data quality for process mining. The Process Mining Manifesto [41] was the first attempt at quantifying (using a 1–5 star rating) how ready event logs are for use in process mining analysis (fitness for purpose). Attempts at classifying types of event data quality issues according to event/case attributes were described in frameworks proposed by Mans et al. [42] and Bose et al. [43]. Other frameworks for event data quality have been proposed in Kerbouche et al. [44] and in Fox et al. [45] that describe the Care Pathways Data Quality Framework (CP-DQF) for process mining of electronic health care records. Furthermore, Goel et al. [25] propose a framework for common event log quality issues electronic medical records. These frameworks are useful in classifying/labelling an identified quality issue, but are limited in that (i) the issues (usually) affect individual attributes of individual log records, and (ii) do not provide guidance for detecting specific examples of quality issues in a log.

Suriadi et al. [19] begin a move towards a systematic and generalisable approach to uncovering event data quality issues by describing 11 frequently encountered log imperfection patterns and including indicative rules for detecting these imperfections in event logs. Further, some of the patterns proposed refer to quality issues that may manifest across multiple event log records. Fischer et al. [31] and Sadeghianasl et al. [29,30,46,47] present approaches and supporting tools for detecting data quality issues associated with, respectively, timestamps and activity labels. Kherbouche et al. [44] detect and quantify missing and too coarse timestamps (which may lead to incorrect order of events). Busch et al. [48] present an approach for semantic detection of anomalies in event logs using sequence-to-sequence models. van Zelst et al. [49] present an approach for repairing homonymous activity labels using contextual information of the process. The Odigos (Greek for 'guide') framework [50] is a means of identifying root-causes of identified event data quality issues through understanding the personal, social, and material (IT) context in which a process executes. Odigos thus provides a means of looking at more than just symptoms of data quality, e.g., missing or inaccurate event/case/log attributes, but is useful for deriving prevention strategies to stop data quality issues from being introduced into process/event data. Finally, Ghalibafan et al. [51] propose eRooMiner, a data-driven technique for detecting root causes of event log data quality issues to facilitate their prevention.

2.3. Data quality in object-centric process mining

Traditionally, each record in the event log refers to exactly one process instance (*case*), has an attribute representing the process step (*activity*), and a timestamp. In [6], the authors describe the notions of *convergence* (one event may be related to multiple cases), and *divergence* (multiple instances of the same activity within a case), leading to the conclusion that there is a gap between real-world event data and the event logs required by traditional process mining techniques. The notion of object-centric logs described in [6] – with formalisations of such logs described in [52,53] (XOC format) and in [2,54] (OCEL format) – was devised to address this shortcoming.

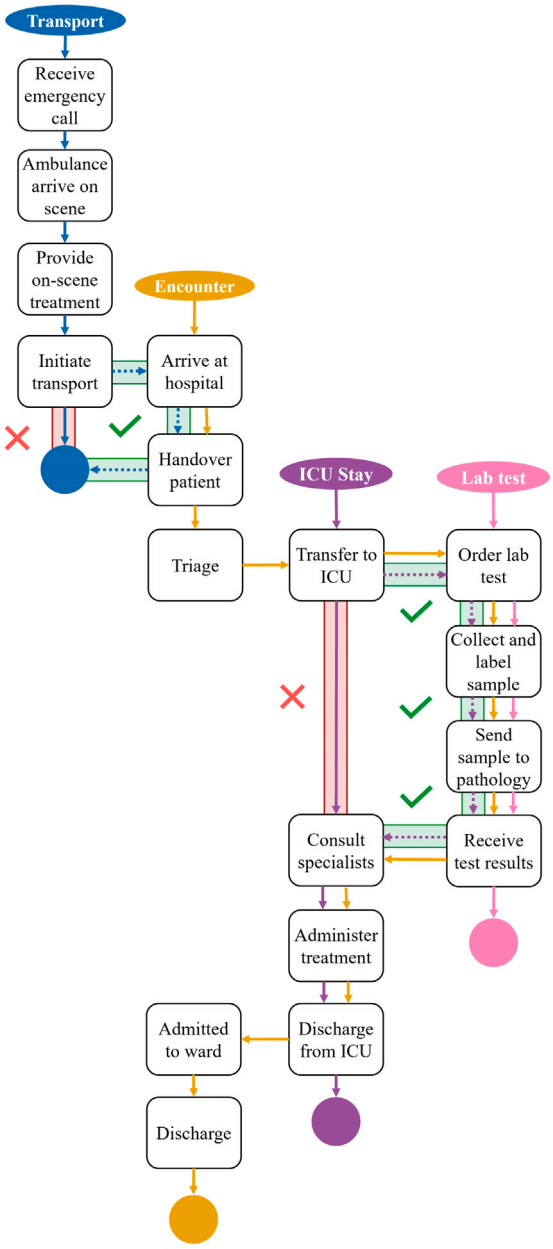


Fig. 2. An example of Object-Centric Directly Follows Graph (OC-DFG) with incorrect sequence flows, highlighted in red. The dotted line arrows, highlighted in green, show the correct sequence flows that should have been discovered instead. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

Here we mention some techniques for using object-centric data in process mining. Various process mining techniques (both academic and commercial) exist that seek to overcome limitations imposed by the single case notion. Berti and van der Aalst [55] describe a tool to construct so-called *Multiple Viewpoint* (MVP) models that encapsulate multiple case notions in a single diagram. Van der Aalst and Berti [56] describe a discovery technique that results in a process model in the form of an object-centric Petri net, while Fahland [57] introduces *Event Knowledge Graphs* that support modelling behaviour over multiple entities as a network of events. Adams et al. [58] describe the OCπ tool that makes explicit object-centric variants. Di Ciccio and Montali [59] describe extensions to the DECLARE [60] modelling language to capture object-centric processes. Such notions were first formalised in [61] which describes the combination of ideas from declarative, constraint-based languages like DECLARE and from data/object modelling approaches (like UML) to derive *Object-Centric Behavioural Constraint* (OCBC) models. The automatic discovery, from XOC format logs, of OCBC models which graphically describe control-flow and data/objects with cardinality constraints, is described in [52] and revised in [53]. Goossens et al. [62] introduce the Data-aware format of OCEL (DOCEL) to support dynamic object attributes with changing values. Other works such as [9–11] propose process discovery approaches from object-centric event logs. There are also works that focus on object-centric conformance checking [12,13], and object-centric process prediction [14–16]. Buss et al. [63,64] extract Object-Centric Event Logs (OCELs) from unstructured textual descriptions using natural language processing techniques and large language models. Celonis was a pioneer commercial vendor in supporting object-centric process mining with the release of Process Sphere in 2022 [5]. Furthermore, open-source tools such as the "OCELStandard" package in ProM,[2] the OC-PM tool,[3] Object-Centric Process Querying (OCPQ),[4] Object-Centric Event Log 2.0 Inspector (OCELOT),[5] and Object-Centric Process Insights (ocpi.ai) support OCED data very close to the conceptualisation chosen in this paper.

Data quality remains an important consideration for object-centric process mining. While much of the existing work devoted to event (log) data quality remains relevant, certain aspects become irrelevant, e.g., quality considerations specifically related to a case. Clearly, existing work relating to XES format logs does not include quality considerations that might apply to "objects" and "relationships".

With the growing interest in object-centric process mining, researchers have started to recognise the importance of data quality management in this area. For example, in a recent work [65], the authors have made an initial attempt to analyse data quality issues of object-centric event data. There are some overlaps between our OCED imperfection pattern collection and the quality issues analysed in [65], e.g., missing/incorrect object–object relations. However, this paper does not cover the concrete detection and repair approaches and a thorough evaluation of the presented data quality problems. In [53], the authors outline two log quality issues, "incompleteness" (too little data to cover the various objects and events, and "noise" (too much data with the log including events or objects unrelated to the target process). They describe techniques based on filtering infrequent events and objects for dealing with noisy logs, which (i) simplify complex discovered models, (ii) discover more precise behavioural constraints, and (iii) discover more precise cardinality constraints. In [66], the authors discuss the *garbage in–garbage out* challenge in process mining. They use an example scenario taken from a typical object-centric process and associated object-centric event data (quality) issues to argue for the augmentation of event data with so-called *commonsense knowledge* to improve the faithfulness and quality of process mining. The authors look to the field of Knowledge Representation & Reasoning (KR&R) as inspiration and look, as future work, to (i) develop a formalism capturing classes, relationships and temporal/dynamic operators and then (ii) understand how this formalism can be used to augment event data. In another work, Swevels et al. [67] present an approach for inferring missing object identifiers in object-centric event logs using event knowledge graphs.

2.4. Data quality in object-centric databases

Parallels may be drawn between object-centric event data and object-centric databases. In object-centric databases, data quality is

[2] https://promtools.org; last accessed 5 June 2026
[3] https://www.ocpm.info; last accessed 5 June 2026
[4] https://ocpq.aarkue.eu; last accessed 5 June 2026
[5] https://ocelot.pm; last accessed 5 June 2026

maintained through "integrity" constraints. In [68], the authors conceive database integrity in terms of a pattern language. The authors state that database information integrity has two essential attributes — validity and completeness. Validity guarantees that all false information is excluded from the database, and completeness guarantees that all true information is included in the database. The authors then define an integrity constraint as a "function that returns a Boolean, and implements a data assertion respected for all valid states of the database" [68, p. 51]. The authors then define a complete theory able to evaluate both validity and completeness. Finally, the authors define five (5) patterns that ensure correctness and safe concurrent access to the data, while maintaining the integrity of the database. These patterns are *Constrain* (to manage Entity, Referential, and Domain integrity), *Transaction*, *Locking Concurrency* and *Optimistic Concurrency* (to manage atomic units of work — create, read, update, delete), and *Resource Timer* (as a means of forcing rollback). The authors do not discuss any specific data quality issues affecting object-centric databases.

To sum up, this section aimed to provide a background on the topic of data quality in general and how it is managed in process mining, and more specifically, object-centric process mining. We also looked at data quality management in object-centric databases as a closely related area. Overall, our literature review shows that since its emergence, object-centric process mining has been popular and various process mining techniques have been proposed in this area [9–16,53,62]. There are several works that focus on data quality management in traditional process mining [19,29–31,44,46–49], but the new nature of object-centric process mining calls for new data quality management solutions, and this is an area that is still under-explored. This paper aims to fill this gap by proposing a collection of OCED imperfection patterns, each characterised by a concrete description, a manifestation, several real-life examples, their effect on process mining analysis, and detection and repair techniques.

3. Research method

The development of the object-centric event-data imperfection patterns followed a Design Science Research (DSR) methodology [69,70], aimed at producing an artefact to address a known problem in the domain of object-centric process mining. We followed a six-step process, as per Peffers et at. [70], consisting of (1) problem identification, (2) definition of design objectives, (3) design and development, (4) demonstration, (5) evaluation, and (6) communication.

(1) Problem identification. As highlighted in Section 2, existing work on event log data quality has primarily focused on case-centric event logs [19,42–44]. The increasing adoption of object-centric process mining calls for the recognition of new data quality issues that are not adequately addressed by existing frameworks. These issues — such as missing object identifiers, incorrect object–object relations, and incomplete object–event mappings — threaten the reliability of process mining outcomes. Despite initial attempts [65], a more systematic approach to characterising object-centric event data quality issues, their identification, and their remediation is required.

(2) Definition of design objectives. Based on the problems identified in the last step, the goal was to create a collection of imperfection patterns that satisfies four design objectives: (DO1) the pattern collection must characterise data quality issues that *exist and are pervasive* in real-life object-centric event data, rather than hypothetical problems that occur only in theory; (DO2) the *recognition of the patterns must be of importance*, meaning that if left untreated these imperfection patterns can distort process discovery, conformance checking, or performance analysis, thereby negatively impacting object-centric process mining results; (DO3) the pattern collection should have a *conceptual grounding*, to support methodological rigour, interoperability with existing standards, and extensibility for future research; and (DO4) *concrete detection and/or repair techniques can be developed*, demonstrating that the artefact is not merely descriptive but actionable. To improve process mining outcomes, imperfections must be identified and remedied; it is therefore essential to have tool support for (semi-)automated detection and repair of the patterns.

(3) Design and development. The pattern collection was guided by the design objectives and constructed through synthesising recurring data quality issues in real-life object-centric data sets (DO1), which have an impact on process mining analysis (DO2) into formalised patterns using the object-centric metamodel (Fig. 3) as a conceptual scaffold (DO3). Furthermore, each pattern is characterised by its nature, manifestation in data, impact on process mining, and potential detection and repair strategies (DO4).

(4) Demonstration The patterns are demonstrated through a detailed description with multiple examples of occurrence in real-life datasets presented in Section 5. This demonstration highlights how the patterns manifest across different domains and data sources, thereby illustrating their generalisability and practical relevance beyond a single case study.

(5) Evaluation The artefact (i.e., the OCED imperfection pattern collection) was evaluated through a multi-pronged strategy, consistent with the DSR method's emphasis on assessing how well the artefact addressed the design objectives. This included:

- *Evidence-based validation*, where we report on the evidence of the existence of the patterns in real-life data sets (DO1).
- *Literature-based validation*, where we report the literature that acknowledges the existence of the pattern in real-life data sets (DO1).
- *Empirical validation*, where we implement some automated detection techniques to show the feasibility of detecting the patterns in object-centric event logs (DO4).
- *User-based validation*, involving feedback from experts in the process mining community who reviewed the pattern collection and confirmed the pervasiveness and importance of the recognition of the patterns (DO1, DO2).

DO3 was addressed through designing the patterns in alignment with the core OCED metamodel presented in Fig. 3.

(6) Communication. The results — including the metamodel, the pattern collection, and the evaluation outcomes — are communicated through this paper, contributing a validated and extensible artefact to the field of object-centric process mining and supporting future research and practice in data quality management.

4. Object-centric event data core metamodel

In this section, we explore the core concepts of the object-centric paradigm, as this will allow us to structure the subsequent patterns and place them into context. These concepts will be shown and related to each other in the form of a conceptual data model. Our metamodel focuses on the essence of the object-centric paradigm in alignment with mainstream thinking on the topic [33,34].

Fig. 3 shows a core metamodel of object-centric event data. This model is not intended to be a complete metamodel for object-centric event data. It rather references the core elements of most object-centric event data metamodels [33,34]. Some of the elements of this core metamodel already exist in the XES standard, e.g., event, timestamp, and event attributes. However, the model is generalised to include some core elements specific to object-centric event data:

1. **Object**: An object is a *thing*, which can be physical, e.g., a patient, an invoice, a machine, or it can be abstract, e.g., a medical test. Objects should be uniquely identified by an identifier, though in practice, as we shall see, some objects may not have been identified correctly and thus not have a unique representation in the system or not even have a reflection in the system. Each object must have an **Object Type**. There can be multiple objects of the same type.

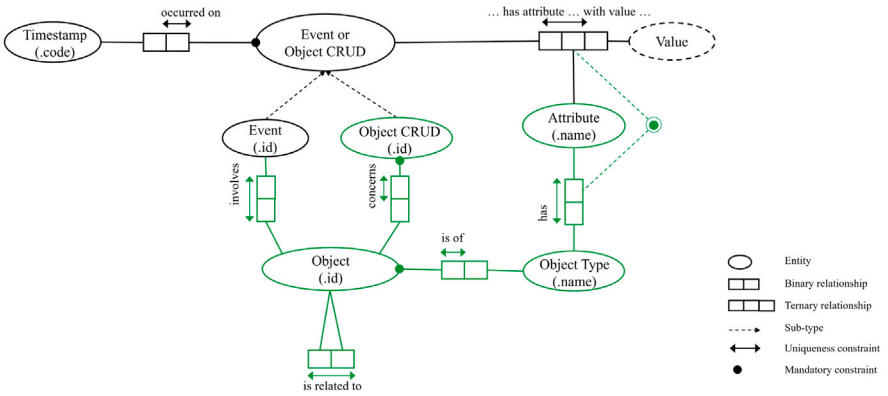


Fig. 3. Object-centric event data core metamodel in Object-Role Modelling notation (ORM [71] with the PSM extension [72]). Object-specific parts are highlighted in green. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

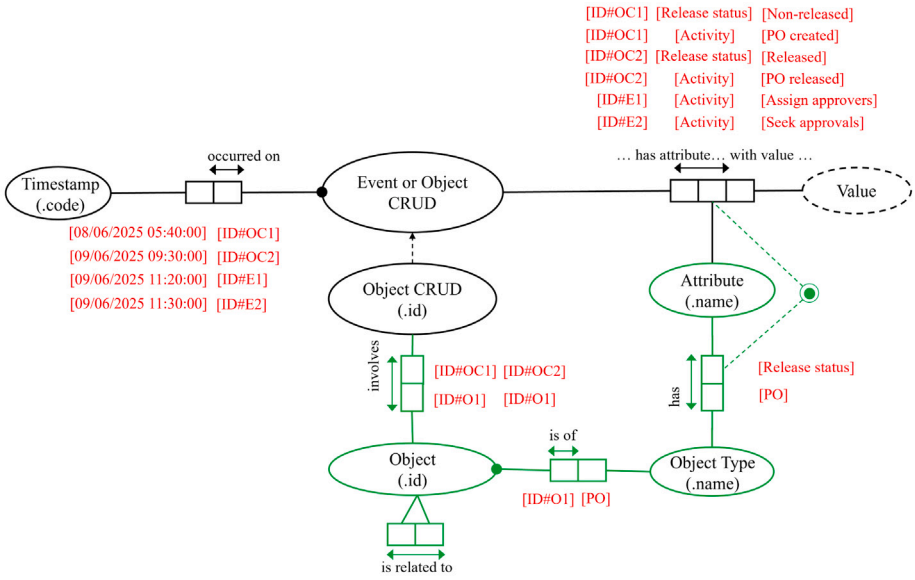


Fig. 4. Instantiation of the object-centric event data core metamodel for the Purchase Order (PO) creation Object CRUD.

2. **Object attributes**: An object (type) can have multiple attributes.
3. **Object CRUD**: An object CRUD (Create–Read–Update–Delete) shows a timestamped change to an object or its attributes. Events and object CRUDs are generalised [73] to **Event or Object CRUD** in the metamodel.
4. **Object–object relation**: An object can be related to any number of objects, for example, an object can be a subtype or a part of another object.
5. **Object–event relation**: An object can be related to any number of events and/or object CRUDs. An event can involve any number of objects, while an object CRUD can be related to only one object. Note that this model allows objects to exist without any linked events and vice versa.

Fig. 4 exemplifies the object-centric event data metamodel. This example describes a Purchase Order (PO) object with ID O1, the creation of which is recorded as an Object CRUD with ID OC1. There are two attributes involved in this creation, 'Activity' and 'Release status', the value of which is set to 'PO created' and 'Non-released', respectively. The next Object CRUD OC2 releases the purchase order O1. OC2 has two attributes involved, 'Activity' and 'Release status', with values 'PO released' and 'Released', respectively. Next, there are two events E1 and E2 happening in this process, where E1's activity is 'Assign approvers' and E2's activity is 'Seek approvals'.

5. Object-centric event-data imperfection patterns

Event data is not immune to quality issues, and this can threaten the reliability of process mining results. It is imperative to investigate and characterise event data quality problems to facilitate their prospective detection, mitigation, and repair strategies. Some common data quality issues have already been discussed in existing frameworks [19,43,74]. However, due to the maturity of the process mining field and the emergence of object-centric event data, new types of data quality issues have arisen.

In this section, we introduce eight new data quality issues, referred to as *object-centric imperfection patterns*, that can be observed in object-centric event data extracted from raw data sources. Our rationale for using a pattern-based approach is that the patterns presented derive from experience and allow one to encapsulate a number of important aspects, such as the description of the data quality issue involves, examples of the issue, the way the issue can affect process mining, and the detection and remediation of the issue. Patterns form an excellent starting point to get a grip on complex and fuzzy problems, to help build further understanding of these problems – by establishing concepts and terminology and through the creation of a community around the patterns – and to think about their solutions. Patterns "describe a problem which occurs over and over again in our environment" [75] and are thus a reflection and documentation of experience. This is

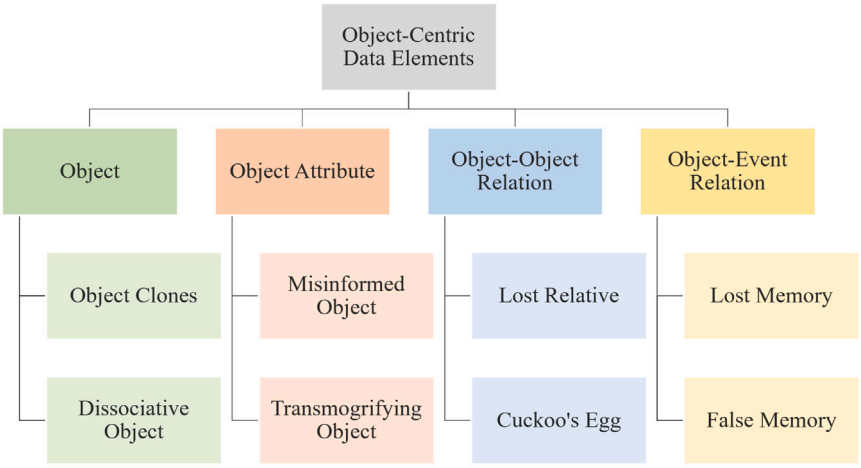


Fig. 5. Object-centric data elements and their associated imperfection patterns.

important as patterns that could theoretically occur, but do not really do so in practice, create a cognitive overload and distract from the key issues at hand. The documented imperfection patterns align with the core metamodel shown in Fig. 3, but the focus is only on the new elements that emerge from the use of object-centric event data (highlighted in green in the metamodel). In other words, there are other types of quality issues, e.g., those that affect events, that are not covered in the paper because they have already been studied and characterised in other data quality frameworks in process mining [19,43,74].

While we do not claim our set of patterns to be complete, and more patterns can be added to this set as new elements are introduced to the metamodel, we do believe that this pattern collection represents the common data quality issues that may appear in object-centric event logs, due to the systematic approach that was followed to collate the collection. Even with the addition of new patterns in the future, the proposed pattern collection is expected to remain relevant as it is based on the object-centric event data metamodel (Fig. 3) in alignment with the current OCED [33] and OCEL 2.0 [34] metamodels in the community.

Fig. 5 shows the set of eight object-centric event-data imperfection patterns. As depicted in Fig. 5, the patterns can be categorised based on the core object-centric data element involved, i.e., objects, object attributes, object–object relations, or object–event relations, in alignment with the metamodel of Fig. 3. While *object CRUD* is modelled as a separate entity in the metamodel, it is not treated as a separate category in the classification, as object manipulations fall in the scope of the Object Attribute category. The patterns are thus informed by the metamodel, but they are also based on practical experience and deductive reasoning. Our approach aims to capture patterns at a level of abstraction that may help their understanding by practitioners – not too fine-grained and not too high-level – and which may facilitate root-cause analysis.

In the remainder of this section, the eight object-centric imperfection patterns are described using the following components:

- *Definition*: outline of the pattern and some possible root-causes for its occurrence.
- *Real-life example*: example(s) of this pattern based on practical experience. An illustration is provided for the object-centric event data to visualise each pattern. The figures show multiple tables: including the events table, the objects table, and the object relations table; however, this is for demonstration purposes only and does not limit the implementation of object-centric event data to any representational format.
- *Effect*: the consequences of the presence of this pattern in the event log on process mining results.
- *Manifestation and detection*: suggestions to detect the existence of this pattern in the log.
- *Remedy*: some strategies to repair this pattern in the log.

5.1. Pattern: Object Clones

Definition. This pattern may happen where the same object, in reality, is associated with multiple records in the system, each with its own unique object identifier. Possible root causes for the occurrence of this pattern are a lack of effective record-matching mechanisms. This pattern relates to the problem of duplicate entities or linked records within the database community [76].

Real-life Example. This pattern was encountered in at least three data sets, two from the healthcare domain and one from the finance domain. In the Emergency Department (ED) of an Australian hospital, it was observed that sometimes multiple encounter IDs are assigned to a patient who visited the ED multiple times in a day. This could, e.g., occur as a result of a patient going for an outpatient appointment after an ED presentation and then having to return to the ED afterwards. However, the ED staff tends to create a new object/record with a different ID for the second presentation. Various recorded times for this presentation are copied from the first one. This is not a desirable practice as both encounters should be seen as a single presentation with a corresponding single object/record. Another possible scenario is that multiple patient IDs are assigned to a patient because, at the time of their second presentation, the first record could not be found in the system (e.g., because of a system error, a spelling error or an inconsistent date of birth format [77,78]), thus a new record is created. As shown in Fig. 6, both patient IDs p_1 and p_3 refer to the same patient 'John Smith'. Another example of this pattern manifested in a data set from a financial services firm, where multiple customer IDs were assigned to the same customer [79]. Also, this pattern was observed

Events

ID	Activity	Patient	Encounter	Timestamp
e_1	*Arrive* at hospital	$\{p_1\}$	$\{en_1\}$	3/5/2021 5:40:00
e_2	*Triage*	$\{p_1\}$	$\{en_1\}$	3/5/2021 5:44:00
e_3	*Dr seen*	$\{p_1\}$	$\{en_1\}$	3/5/2021 6:30:00
e_4	*Discharge*	$\{p_1\}$	$\{en_1\}$	3/5/2021 7:10:00
e_5	*Arrive* at hospital	$\{p_3\}$	$\{en_3\}$	10/5/2021 9:20:00
e_6	*Triage*	$\{p_3\}$	$\{en_3\}$	10/5/2021 9:21:00
e_7	*Dr seen*	$\{p_3\}$	$\{en_3\}$	10/5/2021 9:25:00
e_8	*Discharge*	$\{p_3\}$	$\{en_3\}$	13/5/2021 18:21:00

Object Relations

Object 1	Object 2
p_1	en_1
p_2	en_2
p_3	en_3
p_4	en_4

Objects

ID	Type	Name	DOB	Age	Nationality	Gender	Medicare Number
p_1	*Patient*	*John Smith*	01/01/1998	24	*Australian*	*Male*	6789
p_2	*Patient*	*Jill Brown*	*dd/mm/yyyy*	*xx*	*xxxxx*	*xxxx*	*xxxx*
p_3	*Patient*	*John Smith*	01/01/1998	24	*Australian*	*Male*	6789
p_4	*Patient*	*Joe Russel*	*dd/mm/yyyy*	*yy*	*xxxxx*	*xxxx*	*xxxx*

Objects p_1 and p_3 refer to the same patient

Fig. 6. An example of the Object Clones imperfection pattern.

in the data set of the St. Andrew's War Memorial Hospital's ED, where multiple records for the same patient were present [80].

Effect. When this pattern is present, the events related to a single object are scattered across multiple objects. This may result in objects with missing events, thus the discovery of incomplete process models. In the discovered OC-DFG, this manifests as missing direct follows relations and artificially short paths for individual objects. In conformance checking, cloned objects lead to false deviations, as alignments interpret the missing events as skipped process behaviour in reality. In performance analysis, waiting times and throughput times computed per object type can be underestimated because the time is distributed across multiple clones instead of a single object.

Manifestation and Detection. This pattern's signature is the existence of different object IDs for the same real-world object. A potential indicator of the presence of this pattern is a disproportionally high number of object IDs. For the detection of object clones, we need to look at their contextual information, either available in the log or coming from an external source. For instance, we can look at the proximity of object attributes [29,76], e.g., email addresses, date of birth, phone number, or home address of customers, to identify object clones. Entity matching techniques [76,81–83] can also be used to detect object clones. These methods use a variety of similarity measures, including string-based [84], contextual [82], or semantic [85]. Furthermore, clustering techniques [86] can be used to partition objects into groups of similar objects, which can potentially indicate object clones.

Remedy. Once detected, object clones can be repaired by updating their object IDs to the same ID. We can choose the ID of the object that is linked to more events and copy information from the other object or objects across. Moreover, approaches to finding cluster representatives [87] can guide the discovery of which objects to keep in the data.

5.2. Pattern: Dissociative Object

Definition. This pattern refers to different real-life objects to which the same object identifier is assigned in the system. This can happen in systems where a uniqueness constraint on object IDs is not enforced. Another possible root cause is the reuse of the object ID of an expired object for a new object. For instance, when a staff member leaves an organisation, their ID is assigned to a new staff member. This issue is known as *the day of the jackal* problem [88]. A third root cause of this pattern is heuristics/probability-based record linking of disparate data sources for which no common object identifier exists (such that the linking strategy is aimed at identifying instances of the same physical entity in the disparate data sets, and assigning a unique identifier to these instances).

Real-life Example. In the ED of an Australian hospital, there is a list of available Unknown Record Numbers (URNs) that are assigned to patients whose identity is unknown (e.g., because they are unconscious on arrival). As soon as a patient with a URN is identified, their record is linked to a new or an existing patient ID. In some cases, when a URN is assigned to a patient, staff forget to remove it from the list. This results in assigning the same URN to a future unknown patient and merging of their records. The problem may also occur where new patients present at the ED which are incorrectly linked to existing patients (e.g., because of name or DOB similarity). Fig. 7 shows an example of the first scenario, where the same ID urn_1 is assigned to two different unknown patients arriving at the ED unconscious. Please note that in this example, *object ID* is not the primary key in the object table, thus the same object ID can occur in multiple rows.

In an emergency services data set, compiled from logs recorded by disparate, non-interoperable information systems (ground-based ambulance, aeromedical transport, emergency department, hospital admissions, and death registry), no common person (patient) identifier exists. Reconstructing patient histories, i.e., assigning consistent, unique person identifiers across the disparate data sets involves heuristics/probability-based record linkage. Such heuristics include matching on identifiers that are known (such as the incident identifier assigned by the ambulance dispatcher, the ambulance unit identifier(s) attending the incident, timestamps, transport destinations, etc.). Where physically different people are involved in the same incident, and at approximately the same time, are treated by the same ambulance crew, and are transported to the same hospital, there is an increased probability that the heuristics will apply the same person identifier to these different people.

Effect. The presence of this pattern means that there are (unrelated) events linked to an object that should have been associated with several objects. In process discovery, this results in overly complex object-centric models with excessive concurrency, loops, or dependencies that do not exist in reality. In object-centric conformance checking, the merged object behaviour leads to over-fitting, where the discovered

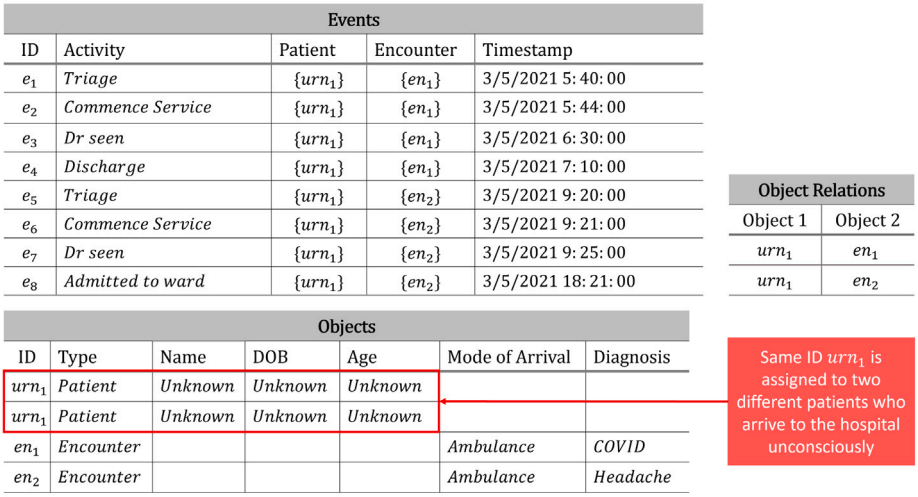


Fig. 7. An example of the Dissociative Object imperfection pattern.

process model allows too much behaviour, making deviations harder to detect. In performance analysis, cycle times and throughput times can be overestimated because the running times of multiple objects in reality are aggregated into a single instance in the data.

Manifestation and Detection. Dissociative objects manifest themselves as objects that are linked to incorrect events (potentially in different contexts). Making the column with object IDs a primary key (in case that was not already the case) allows one to find whether the same object ID has been used across different rows, a potential occurrence of dissociative objects. Another way of detecting dissociative objects can be through the use of attribute-based event clustering [86], notably using timestamps and/or activities. By clustering the events that relate to the same object, we can find out if there are two or more disjoint groups of events which could potentially be related to separate (new) objects.

Remedy. Each of the detected dissociative objects should be assigned a (new) unique ID. The next step is to unmerge the set of events linked to such objects into a number of disjoint sets (e.g., through density-based clustering methods [89] using event attribute-based similarity measures) and link these to the object ID of the correct new object.

5.3. Pattern: Misinformed Object

Definition. This pattern occurs when objects have contradictory attribute values. In some cases these may be combinations of values of attributes of the same object, while in other cases they may be combinations of attribute values of different, but related, objects. It can be understood that these combinations of values are contradictory without having to refer to historical attribute values, hence they are reflective of static constraints [90], e.g., integrity constraints, having been violated. Therefore, they can be caused by a lack of enforcement of constraints combined with incorrect data entry.

Real-life Example. This pattern is found in a third-party insurance data set, where the 'law firm' attribute has inconsistent values for the same claim object recorded in multiple tables. The Misinformed Object pattern is also observed in a data set from an Australian hospital. We have seen that a patient under the age of 20 was recorded as having a funding source of 'old age pension'. Fig. 8 shows this example for patient p_1.

Effect. The presence of this pattern indicates incorrect information about objects having been recorded. In object-centric process mining, object attributes can be used to define cohorts (e.g., patient groups, claim categories, customer types). As a result, Misinformed Objects cause incorrect cohort assignment, leading to misleading comparative analyses such as cohort-based performance analysis [91]. Performance indicators computed per cohort (e.g., average handling time per claim type) can become unreliable if objects are assigned to the wrong cohort. In data-aware process discovery, contradictory attributes can lead to the discovery of fake or unsatisfiable data constraints. In conformance checking with a data perspective [92,93], valid behaviour may be flagged as non-conformant (or vice versa) because the object's attribute values violate data conditions assumed by the model.

Manifestation and Detection. For the detection of this pattern, *knowledge conflict*, *contradiction detection* [94], or *textual entailment* [95] techniques may be employed. Textual entailment techniques aim to infer the veracity of a 'fact' from an already known 'fact'. These approaches usually combine rule-based feature specification with statistical classifiers to identify conflicting 'facts'. For instance, a set of domain-specific rules may be employed, e.g., IF '*Funding source*' = '*old age pension*' THEN '*age*' > *60*. The object or the linked objects that violate these rules can be detected as instances of this pattern through the use of rule-based data validation methods [96]. Tools such as Gritbot [97] have been shown to be able to, through statistical analysis of values in a data set, automatically identify examples of conflicting facts such as '*age = 73 and pregnant = true*'.

Remedy. This pattern can be repaired by replacing an inconsistent attribute value with a value estimated based on other object attribute values. The domain-specific rules can sometimes provide indications as to how corrections are best made (rule-based data cleaning [98]).

5.4. Pattern: Transmogrifying Object

Definition. This pattern occurs when there is an object CRUD operation that changes the value of an object attribute to an incorrect value. A change in value may be incorrect when it contradicts the historical value or values of the object and, in fact, can only be judged to be incorrect in relation to history and not based on a snapshot of values at a particular point in time. As such, this corresponds to a violation of a dynamic constraint [90] rather than a static constraint (as in the case of the Misinformed Object pattern). Therefore, one of the root causes of this pattern is the lack of enforcement of appropriate dynamic constraints combined with human error.

Real-life Example. In an Australian hospital data set, we have seen that the age of the same patient is recorded inconsistently across the two different systems being used. Therefore, when creating an event

Events

ID	Activity	Patient	Encounter	Staff	Timestamp
e_1	*Arrive at hospital*	$\{p_1\}$	$\{en_1\}$		3/5/2021 5:40:00
e_2	*Triage*	$\{p_1\}$	$\{en_1\}$	*Jack*	3/5/2021 5:44:00
e_3	*Commence service*	$\{p_1\}$	$\{en_1\}$	*Jack*	3/5/2021 6:30:00
e_4	*Dr seen*	$\{p_1\}$	$\{en_1\}$		3/5/2021 7:10:00
e_5	*Discharge*	$\{p_1\}$	$\{en_1\}$	*Jill*	10/5/2021 9:20:00

Inconsistency between the age and funding source of patient p_1

Objects

ID	Type	Name	Age	Funding Source	Mode of Arrival	Diagnosis
p_1	*Patient*	*John Smith*	20	*Old age pension*		
en_1	*Encounter*				*Ambulance*	*COVID*

Object Relations

Object 1	Object 2
p_1	en_1

Fig. 8. An example of the Misinformed Object imperfection pattern.

Events

ID	Activity	Patient	Timestamp
e_1	*Register*	$\{p_1\}$	5/7/2021 7:10:00
e_2	*Triage*	$\{p_1\}$	5/7/2021 9:01:00
e_3	*Dr Seen*	$\{p_1\}$	5/7/2021 11:21:00
e_4	*Discharge*	$\{p_1\}$	5/7/2021 13:30:00
e_5	*Follow Up Consult*	$\{p_1\}$	12/7/2021 10:00:00

Object CRUD

ID	Operation	Object	Age	Weight	Timestamp
oc_1	*Create*	p_1	24	75	5/7/2021 7:10:00
oc_2	*Update*	p_1	21	74.5	12/7/2021 11:00:00

Objects

ID	Type	Age	Weight
p_1	*Patient*	21	74.5
p_2	*Patient*	61	95

Unexpected change in the age attribute value

Acceptable change in the weight attribute value

Fig. 9. An example of the Transmogrifying Object imperfection pattern.

log from these systems, we will see a change in the value of the age attribute for a patient object, possibly going from older to younger – a physical impossibility. Fig. 9 shows an example of this pattern, where the age of patient p_1 is reduced from 24 to 21. Note that the weight attribute is also changed, but such a change may be acceptable in the context of the process, as weight can fluctuate over time.

Similarly, we observe this pattern in an emergency services data set where different information systems inconsistently record information about the same patient. There are numerous examples of a ground-based ambulance attending an incident recording a patient's gender as 'F' (female), and, when the patient is admitted to hospital, the gender is recorded as 'M' (male), or *vice versa*. As in the previous example, when creating an event log from these records, we will see a change in the patient's gender, leading to uncertainty as to the correct value of the gender attribute.

Effect. Transmogrifying Object pattern introduces invalid state changes in object attributes over time. In data-aware conformance checking [92,93], this can cause alignments to report deviations even when the control-flow is correct, because dynamic attribute constraints are violated. In cohort-based performance analysis [91], an invalid attribute change can lead to objects being assigned to the wrong cohorts, thereby distorting the cohorts' waiting and throughput times.

Manifestation and Detection. This pattern manifests itself as an object attribute with unexpected value changes over time (decreasing age, divorced while having never been married, etc.). Whether a change is actually possible can be determined using domain knowledge, e.g., in the form of an ontology, or through domain expert input. For instance, it may be acceptable that customers change their addresses, but they are not allowed to change their date of birth without proof or justification. Similarly, the value of the weight attribute can increase or decrease for a patient, but the age can only increase. Rule-based data validation techniques [99] can be used to detect unexpected value changes to object attributes.

Remedy. To repair manifestations of this pattern, in some cases, we can roll back attribute values to previously recorded ones. Knowledge-based or ontology-based data cleaning methods [100] can help to repair this pattern using domain-specific rules. Ideally, constraints are defined in the system, and there is an alert whenever an unexpected change is about to be made.

5.5. Pattern: Lost Relative

Definition. This pattern is encountered when relations between objects are missing. These relations are not recorded because of human error, and also due to the lack of constraints enforcing mandatory fields in the system.

Real-life Example. Lost relatives are observed in data sets from the healthcare, logistics, and education domains. In the healthcare domain, a scenario was observed where blood test and imaging results are not linked to the patients in the Emergency Department (ED) of an Australian hospital [80]. Furthermore, in the MIMIC-III dataset [101,102] (which records ICU patients), some prescriptions are not linked to any ICU stay, i.e., ICU stay ID is missing (consider Table 2 in [103]). Also, in an Australian hospital data set, some encounters are not linked to any team due to human error. Another example is when a laboratory specimen is not linked to any patient, because it is not labelled [104]. In a purchase order process data set, some invoices were not linked to any order [105]. Also, in the BPIC 2020 event log [106] which consists of events related to a reimbursement process at Eindhoven University of Technology in the Netherlands, we see that some international declarations and pre-paid travel costs are not linked to any travel permit applications. For example, as depicted in Fig. 10, we can see that the request with number r_2 for pre-paid travel costs is not linked to any permit IDs, while it should have been linked to p_2. The Lost Relative pattern is also observed in the emergency services data. Here we see instances where a patient is recorded as having been transported from the scene of an incident and arriving at a destination, but the actual destination is not recorded as part of the transport. For the patients identified in these transport instances, there are some that have hospital admissions or ED presentations temporally close to the transport date/time, but which are not definitively linked to the transport (by hospital name).

Another example of the Lost Relative pattern is also found in the third-party insurance data set. There are 233 Nominal Defendant (ND) claims which have both ISUNIDENTIFIED and ISUNREGISTERED flags set to 0 (indicating that the claim should appear as

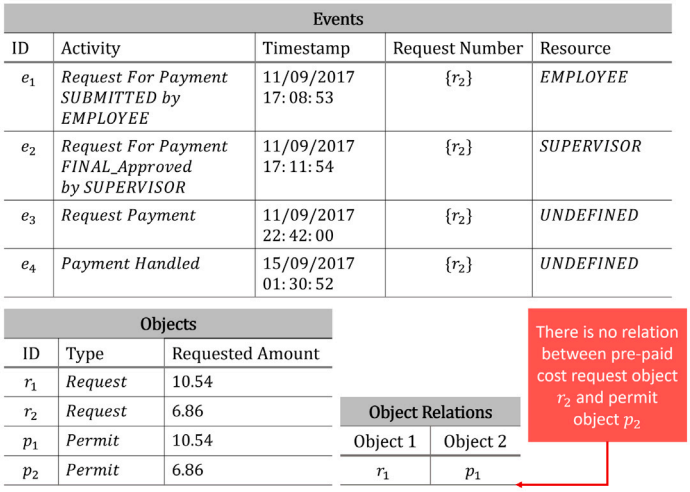


Fig. 10. An example of the Lost Relative imperfection pattern.

an object in the CTP Scheme data. That is, the ND claim should be related to a Scheme record by the Scheme record identifier (OBSERVATION_NUMBER). However, these 233 ND claims have a NULL value for OBSERVATION_NUMBER.

Effect. The presence of lost relatives in an event log can lead to missing some process steps or events related to an object. This happens because some events are indirectly linked to an object, e.g., event e is linked to object o_1, and object o_1 is linked to object o_2, thus event e is linked to object o_2. Therefore, the missing link between o_1 and o_2 can lead to not capturing event e related to object o_2. Missing process steps can influence process discovery and lead to an incomplete picture of the process, with missing paths and disconnected process fragments for affected object types. In conformance checking, alignments incorrectly report missing activities because events indirectly associated via another object are no longer visible in the model. Performance analysis is also affected. For example, the time from order creation to invoice payment becomes inaccurate because relevant events are not linked to the correct objects.

Manifestation and Detection. This pattern manifests itself as a set of objects with missing relations. In order to detect lost relatives, we need domain knowledge about the expected relations, e.g., order and order line item. In case of low frequencies of missing object relationships, outlier detection techniques such as density-based clustering [89] and data distribution prediction [107] can be used.

Remedy. Lost relatives can be repaired by adding missing object relations to the relevant objects. Imputation techniques [108], e.g., non-negative matrix factorisation, regression imputation, and stochastic imputation, can estimate a replacement for missing object relations based on existing data. We can approximate the missing object relationships based on existing relations between objects of the same type. If the missing relations happen due to a system logging mechanism, we can identify which objects should be linked together using clustering methods such as DBSCAN [109], taking an attribute-based function as a parameter. This function can estimate the distance between objects based on the proximity of the attributes of the events that are linked to them.

5.6. Pattern: Cuckoo's Egg

Definition. This pattern occurs when an object has been linked to one or more incorrect object(s). Some of the root causes for this pattern are human error and automatic approximate matching algorithms.

Real-life Example. This pattern was encountered in event logs from the healthcare domain. For instance, in an Australian hospital, when a patient arrives at the hospital and a staff member looks up their name in the system, it is possible that they choose another patient's record with a similar name by mistake and assign the encounter to that patient. Another example is when a laboratory report is linked to an incorrect patient due to, e.g., incorrect labelling of the samples by the nurses who take them [104,110]. As shown in Fig. 11, test t_1 is wrongly assigned to patient p_1 due to name similarity with patient p_2 (the intended patient).

Effect. The existence of incorrect links between objects may cause missing events in some cases and irrelevant events in others. For instance, we can see in Fig. 11 that test t_1 is incorrectly linked to patient p_1 instead of p_2. This implies that events e_1, e_2, and e_3 will be missing in the process instance related to p_2, and will be irrelevant in the process instance related to p_1. The presence of this pattern in an event log can lead to the discovery of incomplete and inaccurate object-centric process models, fragmenting the true process flow of some objects, while incorrectly expanding others. In conformance checking, valid behaviour appears as deviations. Performance analysis is also distorted because activity counts, durations, and frequencies are assigned to the wrong objects, leading to incorrect calculations of wait times and object throughput times.

Manifestation and Detection. We can use association mining techniques [111] to detect which object types are typically correlated in the event log. Object relations between objects of non-associated types can be instances of this pattern. Domain knowledge may also unveil some of the correlations between object types. For instance, we may see that a patient with a specific diagnosis is linked to tests not needed for that diagnosis.

Remedy. Occurrences of this pattern can be repaired by updating object relations. The right objects can be determined through association mining [111] or by matching object attributes through minimum weight graph matching algorithms [112] or ontology-based (i.e., semantic) matching techniques [113].

5.7. Pattern: Lost Memory

Definition. This pattern is encountered when there is a missing relation between an object and an event. This may be the result of human error or an incorrectly configured logging system.

Real-life Example. Evidence of the occurrence of this pattern was observed in a data set from the healthcare domain. In the MIMIC-III data set [101,102] (which records ICU patients), the hospital admission ID and the ICU stay ID are missing in some events, e.g., chart events, lab

Events

ID	Activity	Test	STAFF	Timestamp
e_1	*Test requested*	$\{t_1\}$	*John*	3/5/2021 5:40:00
e_2	*Test conducted*	$\{t_1\}$	*Jill*	3/5/2021 5:44:00
e_3	*Test results returned*	$\{t_1\}$	*Jack*	3/5/2021 6:30:00

Objects

ID	Type	Name	DOB	Age	Gender	Nationality
p_1	*Patient*	*Ann Smith*	03/04/1998	24	*Female*	*Australian*
p_2	*Patient*	*Annie Smith*	07/05/1991	30	*Female*	*Australian*
p_3	*Patient*	*xxxx*	*dd/mm/yyyy*	*xxx*	*xxxx*	*xxxx*
t_1	*Blood Test*					
t_2	*Blood Test*					

Test object t_1 should be related to Patient p_2 not p_1. This is recorded incorrectly due to name similarity of Patients p_1 and p_2.

Object Relations

Object 1	Object 2
t_1	p_1
t_3	p_3

Fig. 11. An example of the Cuckoo's Egg imperfection pattern.

Events

ID	Activity	Patient	Admission	ICU Stay	Item	Timestamp
e_1	*Item charted*	$\{p_1\}$	$\{ad_1\}$	$\{icu_1\}$	$\{i_1, i_2\}$	3/6/2021 6:30:00
e_2	*Item stored*	$\{p_1\}$	$\{ad_1\}$	$\{icu_1\}$	$\{i_1, i_2\}$	3/6/2021 7:10:00
e_3	*Item charted*	$\{p_2\}$	$\emptyset$	$\emptyset$	$\{i_3, i_4\}$	4/6/2021 8:35:00
e_4	*Item stored*	$\{p_2\}$	$\emptyset$	$\emptyset$	$\{i_3, i_4\}$	4/6/2021 9:01:00

Objects

ID	Type	Name	DOB	Age	Room number
p_1	*Patient*	*Jill Brown*	*dd/mm/yyyy*	*xx*	
p_2	*Patient*	*John Smith*	*dd/mm/yyyy*	*xx*	
ad_1	*Admission*				
icu_1	*ICU Stay*				*W213*

Missing links to Hospital Admission and ICU Stay objects in events e_3 and e_4.

Object Relations

Object 1	Object 2
p_1	ad_1
p_1	icu_2
p_1	i_1
p_1	i_2
ad_1	i_1
ad_1	i_2
icu_1	i_1
icu_1	i_2
p_2	i_3
p_2	i_4

Fig. 12. An example of the Lost Memory imperfection pattern.

events, and note events (consider Table 2 in [103]). As shown in Fig. 12, we can see that the hospital admission ID and the ICU stay ID are missing for events e_3 and e_4.

Effect. Lost Memory occurs when events are not linked to the objects they should be associated with. In object-centric process discovery, such events are excluded from object-specific views, resulting in gaps in object life cycles and incomplete process models. In object-centric conformance checking, the absence of event-object links can cause valid executions to appear incomplete, leading to false violations, such as missing mandatory steps. Performance analysis is affected because unlinked events are excluded from timing and frequency calculations, causing underestimation of activity durations, waiting times, or workload per object.

Manifestation and Detection. This pattern manifests itself as a set of events with missing object links. Sometimes, the lost memory pattern occurs when events that should logically have followed from earlier events are not recorded. For instance, an 'order test' event may be present for a patient in the log, but a corresponding 'test results returned' event cannot be found for that patient. If the missing relations are infrequent, outlier analysis techniques [114] can be used for their detection. Furthermore, discovering the correlation of events and object types, e.g., through association mining [111], can identify missing object–event relations.

Remedy. This pattern can be repaired by adding the missing object relations to the relevant events. The correct object relations can be determined by matching events to object attributes using techniques such as logistic regression [115] or probabilistic matching [116]. Furthermore, the missing relations can be estimated using stochastic imputation techniques [108], e.g., by considering object relations of events with the same name.

5.8. Pattern: False Memory

Definition. This pattern occurs when an event is related to one or more incorrect object(s).

Real-life Example. This pattern has been observed in the logistics domain. As shown in Fig. 13, events e_4 and e_5 are linked to the wrong customer c_2. This means that items i_1 and i_2 in order o_1 are wrongly delivered to customer c_2 instead of customer c_1.

Another example occurs in the emergency services data as a consequence of the Dissociative Object pattern, i.e., same object identifier (in this case patient identifier) being applied to two different persons. The False Memory pattern means that any events associated with an affected patient identifier cannot be definitively linked to the actual object (patient).

Effect. The existence of this pattern can lead to missing events in some process instances and irrelevant events in others. This happens because of the missing events that are incorrectly perceived to be part of other cases. In process discovery, this results in incorrect object flows where unrelated activities appear to occur for an object. In conformance checking, this creates false-positive deviations due to missing correct events. Performance analysis is compromised because execution times, waiting times, and frequency metrics are computed over object

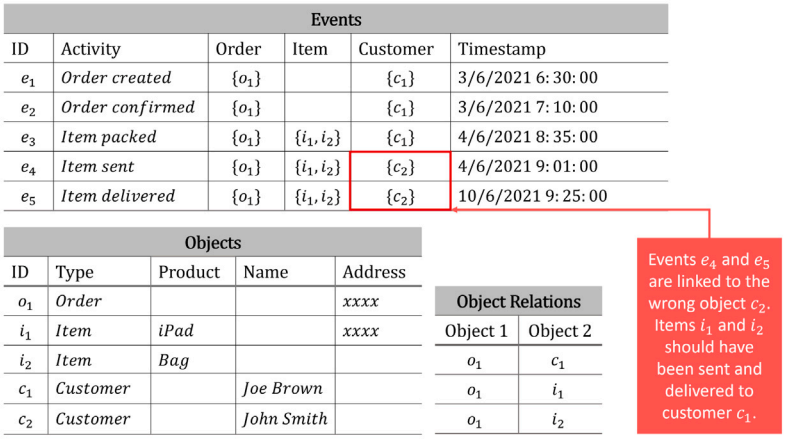


Fig. 13. An example of the False Memory imperfection pattern.

traces with incorrect events, leading to inaccurate conclusions about efficiency and bottlenecks.

Manifestation and Detection. For occurrences of this pattern, we may see some irrelevant events linked to some objects, because they should have been linked to other objects. For instance, we may see that a customer receives an item that they have not ordered at all. The correlated object types for each event type (i.e., activity) can be discovered through association mining [111], which can further indicate incorrect object–event relations.

Remedy. This pattern can be repaired by updating object links in the relevant events. The right objects to be linked to can be determined through matching objects and events. This matching can be semantic (ontology-based [113]) to, e.g., match a test type to a diagnosis, or it can be attribute-based (bipartite matching [117]), or it can be frequency-based (association mining [111]).

5.9. Comparison with other frameworks

In order to position our collection of OCED imperfection patterns and highlight their novelty, we compare them with data quality issues classified in three well-known data quality frameworks for event logs [1,19,74]. Table 1 shows the results of this comparison. We propose three levels of relations between our patterns and other data quality issues in the three aforementioned frameworks: close match (indicated by +), related (indicated by o), and no match (indicated by -). Only one of our patterns, i.e., Lost Memory, is related to an imperfection pattern of Suriadi et al. [19], i.e., the Elusive Case pattern, where events do not belong to a specific case. The Lost Memory pattern may have a manifestation of some object–event links being lost; so we have lost the links with the cases that these events belong to. As such, this pattern can be seen as a special case of the Elusive Case pattern where none of the events has a case ID.

Furthermore, we can see that three of our patterns closely match data quality issues described by Mans et al. [74], with two closely matching to data quality issues classified by Van der Aalst [1]. The Lost Memory and False Memory patterns correspond to the missing and incorrect relationships (between events and cases) as defined in [74] and missing and incorrect case data quality issues presented in [1]. The Misinformed Object pattern corresponds to the incorrect case attributes issues as defined in [74]. The patterns Object Clone and Dissociative Object are related to the incorrect relationship [74] (referred to as incorrect case by Van der Aalst [1]) data quality issue, where events are linked to incorrect cases. These two patterns are more detailed in describing the scenario that led to an incorrect link between events and cases. The Transmogrifying Object pattern is related to the incorrect case attribute data quality problem, but it describes the problem in more detail as it refers to the *changes* that were made to the values of objective attributes that made them incorrect.

It should be noted that the recurrence dimension defined in the framework of Van der Aalst [1] is not considered in the comparison presented in Table 1, as this seems to be a dimension orthogonal to our pattern (i.e., it could be combined with all our patterns).

All in all, it is apparent that many of the data quality problems for object-centric data (as presented in this paper) do not have an accurate reflection in an existing framework. The few that do either are not sufficiently elaborated to capture the object-centric notion, as in the case of the framework of Mans et al. [74] or that of Van der Aalst [1], or their approaches to detection, repair, and remedial action do not translate to an object-centric context, as in the case of the Elusive Case pattern. The proposed set of patterns for object-centric event data allows for the description of targeted detection and repair methods as well as a more customised approach to root-cause analysis (in fact, it could lead to an update of the root-cause analysis approach described by Andrews et al. [50], which was tailored to the event log imperfection patterns).

6. Evaluation

The proposed OCED imperfection patterns are evaluated against their design objectives. First, in Section 6.1 we provide evidence of the existence of these patterns in ten public or private real-life data sets (DO1). Then, Section 6.2 presents an initial attempt towards a dedicated tool support for detecting patterns (DO4) by implementing prototypical techniques and applying these to public event logs. Finally, we investigated the pervasiveness of the patterns (DO1) and the importance of recognising them in event logs (DO2) through a user evaluation within the process mining community (Section 6.3). DO3 was addressed through designing the patterns in alignment with the core OCED metamodel presented in Section 4.

6.1. Evidence of existence of the patterns in real-life datasets

Evidence of the existence of the OCED imperfection patterns in real-life data sets supports their relevance. Table 2 reports on the data sets (public and private, provided to us through our industry projects) where these patterns have been observed. The status in the cells uses the symbols + (pattern found) and – (not found), with a superscript to denote the type of validation: **v** for verified by an expert, and **d** for detected.

For public data sets, the pattern status is indicated by the superscript **d**. Specifically, the $+^{\mathbf{d}}$ symbol indicates that the pattern was detected by an automated technique, and the number in parentheses represents the frequency count of the detected instances. The $-^{\mathbf{d}}$ symbol indicates that

Table 1
Mapping of the OCED imperfection patterns to other frameworks [1,19,74]. The +, o, and − cells indicate a close match, a relation, and no match, respectively.

OCED Pattern	Description	Event Log Imperfection Patterns [19]	Event Log Quality Issues [74]	Data Quality Issue Classification [1]
Object Clones	Same object, multiple IDs	−	o (Incorrect relationship)	o (Incorrect Case)
Dissociative Object	Different objects, same ID	−	o (Incorrect relationship)	o (Incorrect Case)
Misinformed Object	Object(s) with inconsistent attribute values	−	+ (Incorrect case attribute)	−
Transmogrifying Object	Unexpected attribute values change	−	o (Incorrect case attribute)	−
Lost Relative	Missing object–object relation	−	−	−
Cuckoo's Egg	Incorrect object–object relation	−	−	−
Lost Memory	Missing object–event relation	o (Elusive case)	+ (Missing relationship)	+ (Missing Case)
False Memory	Incorrect object–event relation	−	+ (Incorrect relationship)	+ (Incorrect Case)

the pattern was not detected, which means it is either not present or its detection was limited by current automated approaches. Crucially, we were able to detect at least one instance of every OCED imperfection pattern in at least one public dataset.

For private data sets, the pattern status is indicated by the superscript **v**. The $+^{v}$ symbol means the patterns were verified based on the domain knowledge provided by industry experts or through verification against external reference documentation, as domain experts were available for confirmation. The $-^{v}$ symbol indicates that the pattern was confirmed by the domain expert to be not present. We reported our first-hand observations of these patterns within the industry datasets that our team had worked on. However, these datasets were not used to conduct the quantitative assessment, unlike those for the public datasets, due to ethics/access restrictions. To provide contextual information about these private datasets, we reported high-level statistics (i.e., the number of events, objects, and object types) as well as references to published studies that use these datasets.

The distinction between verified ($+^{v}$) and detected ($+^{d}$) is essential. Detected ($+^{d}$) means we applied a detection technique on an event log and found candidate instances. However, without domain knowledge, we cannot be certain if these truly represent an error. For example, to detect Object Clones, where the same real-world object is represented as two objects in the log, we typically need a domain expert to verify whether the detected candidates are indeed clones. In contrast, for private data sets, the $+^{v}$ symbol indicates that the pattern's existence was verified by experts, providing certainty. The two patterns related to missing data (Lost Relatives and Lost Memory) were often directly detectable even in public logs because their identification less critically requires domain-specific business knowledge. We were also able to detect Cuckoo's Eggs in the BPIC 2020 event log, which had a separate event table and an object table, finding mismatches between object links in the event and object tables. It is also important to note the inherent limitation of the field of object-centric process mining, which is that most public event logs are not object-centric due to the emerging nature of the topic [2]. The specific techniques used to detect the patterns marked as $+^{d}$ are detailed in Section 6.2.

The analysis of the ten real-life datasets (five public, five private) provides valuable insights into the pervasiveness of the OCED imperfection patterns. Across the ten logs, Lost Relative was the most frequently observed pattern, appearing in 5 out of 10 datasets, confirming a high prevalence of missing object links. The next most frequent patterns were Transmogrifying Object (4 out of 10), and Dissociative Object, Cuckoo's Egg, and Misinformed Object (each 3 out of 10). The other three patterns related to wrong or missing object assignment, i.e., Object Clones, Lost Memory, and False Memory, were observed less frequently (each 2 out of 10 datasets). Overall, all eight OCED imperfection patterns were found in real-life data: every pattern was detected ($+^{d}$) in at least one public log, and every pattern was verified ($+^{v}$) by domain experts in at least one private log.

Table 3 further supports the relevance of the OCED imperfection patterns by listing a number of research papers that report on the existence of one or more of the patterns in real-life data sets. These papers characterise data quality issues, the description of which matches the description of an OCED imperfection pattern, but they are not referred to by the related pattern name. For instance, in [78], the authors report on the occurrence of "duplicate patient records" in a healthcare data set. They state that "duplicate patient records occur when a single patient is associated with more than one record". We take this report as evidence of the existence of the Object Clones pattern in a real-life data set. Overall, the results confirm the occurrence of the patterns in real-life data sets.

6.2. Pattern detection techniques

This section aims to provide an initial attempt towards a dedicated tool support for detecting and/or repairing the proposed OCED imperfection patterns (DO4). We implemented simple detection techniques and applied them to two real-life public event logs. One of these logs already contained instances of two of the patterns. The other log was successively injected with instances of the remaining six patterns. The injected event logs provide the ground truth to measure the success of the detection technique. The datasets and implementation of the proposed approaches are publicly available.[6]

6.2.1. Event log acquisition and error injection

We consider the object-centric event logs made available by the Business Process Intelligence Challenge (BPIC) in 2017 and 2020. BPIC2017 [119] is a log of a loan application process from a Dutch financial institute, and BPIC2020 [106] contains events pertaining to two years of travel expense claims.

We injected each of the six imperfection patterns (Object Clones, Dissociative Object, Misinformed Object, Transmogrifying Object, Lost Memory, and False Memory) into BPIC2017 by perturbing 10% of traces for each pattern. The Lost Relative and Cuckoo's Egg patterns were the only patterns that were already present in BPIC2020, and therefore, no injection was required for these patterns. The injected and real imperfection patterns are described below:

Object Clones. We injected the Object Clones pattern into the BPIC2017 event log by replacing "Application" object IDs with incorrect values, e.g., from "Application_123" to "Application_123XYZ", in 10% of traces. This injection is applied to events occurring after a randomly chosen event within each of the selected traces.

[6] https://github.com/jonghyeonk/AnomalyDetection_OCIP

Table 2
The OCED imperfection patterns as evidenced in real-life data sets. The + cells indicate that the pattern was found, while the − cells indicate it was not detected. The superscript **v** denotes a result that was verified by an expert (used for private data), and **d** denotes a result that was only detected (used for public data). Note that for the BPIC 2019 dataset, the 30% most recent events (based on timestamp) were sampled. The frequency count for the detected instances ($+^{d}$) is also included in parentheses.

Data set	Domain	Object Clones	Dissociative Object	Misinformed Object	Transmogrifying Object	Lost Relative	Cuckoo's Egg	Lost Memory	False Memory
Public									
BPIC 2014 [118]	Finance	$-^{d}$	$-^{d}$	$-^{d}$	$-^{d}$	$-^{d}$	$-^{d}$	$+^{d}$(187)	$-^{d}$
BPIC 2017 [119]	Finance	$-^{d}$	$-^{d}$	$+^{d}$(3230)	$+^{d}$(1709)	$-^{d}$	$-^{d}$	$-^{d}$	$-^{d}$
BPIC 2019 [120] (30%)	Supply Chain	$-^{d}$	$+^{d}$ (2743)	$-^{d}$	$-^{d}$	$-^{d}$	$-^{d}$	$-^{d}$	$+^{d}$(2743)
BPIC 2020 [106]	Education	$-^{d}$	$-^{d}$	$-^{d}$	$-^{d}$	$+^{d}$ (126)	$+^{d}$(1354)	$-^{d}$	$-^{d}$
MIMIC-IV [121]	Healthcare	$+^{d}$ (6)	$-^{d}$	$-^{d}$	$-^{d}$	$-^{d}$	$-^{d}$	$-^{d}$	$-^{d}$
Private									
Australian hospital [25] 2,329,864 events, 211,227 objects, 4 object types	Healthcare	$+^{v}$	$+^{v}$	$+^{v}$	$+^{v}$	$+^{v}$	$+^{v}$	$-^{v}$	$-^{v}$
St. Andrew's War Memorial Hospital [80] 10,794 events, 1582 objects, 2 object types	Healthcare	$+^{v}$	$-^{v}$	$-^{v}$	$-^{v}$	$+^{v}$	$-^{v}$	$-^{v}$	$-^{v}$
Australian emergency medical services [122] 1,013,635 events, 286,084 objects, 3 object types	Healthcare	$-^{v}$	$+^{v}$	$-^{v}$	$+^{v}$	$+^{v}$	$+^{v}$	$-^{v}$	$+^{v}$
Australian compulsory third-party claims [123] 203,959 events, 8387 objects, 2 object types	Insurance	$-^{v}$	$-^{v}$	$+^{v}$	$-^{v}$	$+^{v}$	$-^{v}$	$-^{v}$	$-^{v}$
Pusan National University Hospital [124] 387,293 events, 64,448 objects, 3 object types	Healthcare	$-^{v}$	$-^{v}$	$-^{v}$	$+^{v}$	$-^{v}$	$-^{v}$	$-^{v}$	$-^{v}$

Table 3
The OCED imperfection patterns reported in the literature.

Paper	Domain	OCED Imperfection Pattern(s)
Suchý et al. [105]	Logistics	Lost Relative
Weis et al. [79]	Finance	Object Clones
Astion et al. [104]	Healthcare	Cuckoo's Egg, Lost Relative
Joffe et al. [78]	Healthcare	Object Clones
McClellan [77]	Healthcare	Object Clones
Bonini et al. [110]	Healthcare	Cuckoo's Egg
Kurniati et al. [103]	Healthcare	Lost Relative, Lost Memory
Yang et al. [125]	Multiple	Object Clones
Bauer et al. [126]	OO/RDB Programming	Dissociative Object

Dissociative Object. The Dissociative Object pattern is injected into the BPIC2017 data set by reassigning "Application" object IDs within 10% of traces to other existing "Application" objects sharing similar attribute values. Similarity is based on matching identical values across the "LoanGoal", "ApplicationType", "RequestedAmount", and "org:resource" attributes.

Misinformed Object. The Misinformed Object pattern is injected by introducing outlying values into the BPIC2017 data set by randomly altering "OfferedAmount" values within 10% of traces, based on their correlation with "CreditScore". This is achieved by replacing actual scores with one of the boundary values of the calculated 99.9% confidence intervals of the regression model's prediction, which can be regarded as outliers.

Transmogrifying Object. The Transmogrifying Object pattern is injected by modifying the "RequestedAmount" attribute in 10% of BPIC2017 traces. For constant values within a trace, the attribute value of a randomly selected event is multiplied by 10; otherwise, for continuous values, it is perturbed to a value at the 99.9% confidence interval boundary, assuming a normal distribution.

Lost Relative & Cuckoo's Egg. Due to the absence of ground truth error labels within the BPIC2020 data set, our analysis of Lost Relative and Cuckoo's Egg error patterns relied on the following assumptions: (1) the object relationships between request IDs and permit IDs within the BPIC2020-PrepaidTravelCost log are inherently correct; and (2) the external object table, sourced from BPIC2020-PermitLog, is potentially contaminated with Lost Relative and Cuckoo's Egg errors. Specifically, in BPIC2020-PrepaidTravelCost, request IDs are associated with the "RfpNumber" attribute, and permit IDs with the "Permit id" attribute. Conversely, in BPIC2020-PermitLog, request IDs are linked via the "travel permit number" attribute, and permit IDs via the "case:concept:name" attribute.

Lost Memory. The Lost Memory pattern, where object associations are removed, was injected by setting "Application" as the leading object type and "Offer" as the related object, with 10% of "Offer" IDs being replaced with missing values.

False Memory. The False Memory pattern simulates incorrect object associations by randomly reassigning relationships. In this injection, "Application" was the leading object, and 10% of "Offer" IDs were

Table 4
The detection performance of the six injected imperfection patterns.

OCED Pattern	Performance		
	Recall	Precision	Accuracy
Object Clones	0.811	0.130	0.682
Dissociative Object	0.898	0.565	0.926
Misinformed Object	0.999	0.890	0.977
Transmogrifying Object	0.884	1.000	0.972
Lost Memory	1.000	0.952	0.975
False Memory	0.987	0.627	0.943

replaced with existing "Offer" IDs from other traces, effectively creating false links. This mimics scenarios where events are mistakenly attributed to the wrong objects.

6.2.2. Detection approaches and results

To evaluate the detection success, we employed key performance metrics: Recall, Precision, and Accuracy. The detailed performance for each approach is summarised in Table 4. Within anomaly detection, Recall — representing the proportion of actual anomalous traces correctly identified — is of paramount importance. This is because a high Recall ensures that the majority of true anomalies are captured, and overlooking even a small number of these critical events can often lead to disproportionately significant consequences, such as unaddressed system failures, security breaches, or financial damage. Consequently, we prioritised achieving a high Recall, accepting that this might sometimes correspond to a moderately lower Precision, as our primary goal was to minimise missed anomalies. For each pattern, we selected and applied a tailored detection approach, as detailed below:

Object Clones. A context-based similarity approach can provide a simple but effective detection method for potential cloned objects as long as the right context variables are selected for similarity assessment. We selected six attributes: "LoanGoal", "ApplicationType", "RequestedAmount", "CreditScore", "MonthlyCost", and "NumberOfTerms" for checking the context similarity. We identified groups of object IDs exhibiting identical values for these selected attributes as potential instances of cloning errors. This similarity-based approach achieved a Recall score of 0.811 in identifying cloned objects. The low Precision score (0.130), however, can be attributed to the characteristics of the BPIC2017 data set, in which attribute values among traces show little variation. Consequently, the similarity-based method flagged a large number of normal traces that shared attribute values with anomalous instances, leading to a high false-positive rate and thus, poor Precision.

Dissociative Object. The reassignment of object IDs caused the merging of process traces, leading to significantly longer traces. The event lengths in the perturbed traces exhibited a distribution with a considerably higher mean and standard deviation [mean = 14.64, standard deviation (SD) = 3.69] compared to the original traces [mean = 6.86, SD = 1.84]. Based on this characteristic of trace perturbation, a control-flow anomaly score was calculated to identify anomalous sequences of activities in traces. Specifically, an information-theoretic measure for this control-flow anomaly score, termed Statistical Leverage [127], was applied to quantify the anomaly score of individual traces. By identifying traces within the top 15% of anomaly scores as potential errors, we achieved a Recall score of 0.898 for these corruptions.

Misinformed Object. Misinformed objects, characterised by anomalous "OfferedAmount" values related to "CreditScore", were detected using linear regression analysis. A 90% confidence interval was established around the predictions generated by the regression model. Traces exhibiting "OfferedAmount" values falling outside this confidence interval were classified as potential misinformed objects, resulting in a Recall score of 0.999.

Transmogrifying Object. To detect the Transmogrifying Object imperfection, an Interquartile Range (IQR)-based outlier detection method was applied to the "RequestedAmount" attribute of the traces. Outliers were identified based on the standard IQR rule: values falling below ($Q1 - 1.5 \times IQR$) or above ($Q3 + 1.5 \times IQR$) were flagged as anomalous. This method achieved a Recall score of 0.884 in correctly identifying previously labelled transmogrified objects.

Lost Relative & Cuckoo's Egg. To detect these errors, we compared the object relationships between request IDs and permit IDs as recorded in the BPIC2020-PrepaidTravelCost log against the corresponding relationships present in the object table derived from the BPIC2020-PermitLog. Our comparison revealed two types of discrepancies: (i) *Lost Relatives*: 126 out of 2064 object relationships present in the event log were absent from the object table. (ii) *Cuckoo's Eggs*: 1874 instances exhibited matching permit IDs but were associated with different request IDs in the event log compared to the object table. The substantial number of identified errors likely stems from the temporal disparity between the creation or recording of the object table and the event data.

Lost Memory. A regression model was trained to predict whether the object ID of an event should be present (i.e., not missing). If the model predicted a non-missing value for an event where the actual "Offer" ID was missing, this was identified as a Lost Memory error. Employing this approach, we achieved a Recall score of 1.000 for these corruptions.

False Memory. The erroneous reassignment of an object ID resulted in process trace splitting followed by a subsequent merging with a trace associated with a different object ID, yielding two corrupted traces. To detect these anomalies, a control-flow anomaly score was calculated to identify unusual sequences of activities. Specifically, an information-theoretic measure for this control-flow anomaly score, termed Statistical Leverage, was applied to quantify the anomaly score of individual traces. By identifying and extracting the traces within the top 15% of anomaly scores, we achieved a Recall score of 0.987 for these corruptions.

6.3. User validation for pattern pervasiveness and importance

In order to further investigate the relevance of the OCED imperfection patterns presented in this paper, we conducted a survey within the process mining community.

6.3.1. Setup

The survey was made available online through the Microsoft Forms platform. It was advertised to process mining researchers, practitioners, software vendors, and students via email, LinkedIn, and our international academic network. There were two groups of questions in the survey: (1) *demographic questions* with multiple choice answers, including the category of participants (academic, practitioner, software vendor, etc.), the number of data sets they have analysed/pre-processed for process mining, the sectors where these data sets were from, their level of familiarity with object-centric event logs, etc.; and (2) *pattern questions*, where each pattern was introduced by its name, description, manifestation, and affect and this was followed by two questions with Likert-type answers (1–5), the first one asking about the level of pervasiveness of the pattern in the data sets the participant analysed/pre-processed for process mining and the second one asking about the level of importance they attach to the recognition of the pattern.

6.3.2. Results

A total of 24 process mining experts participated in the survey, with 21 being academics/researchers, 2 being professionals/practitioners, and 1 being a software vendor. All participants indicated that they have experience with data analysis; 67% had more than 5 years of experience, and 29% had between 2 and 5 years of experience. Furthermore,

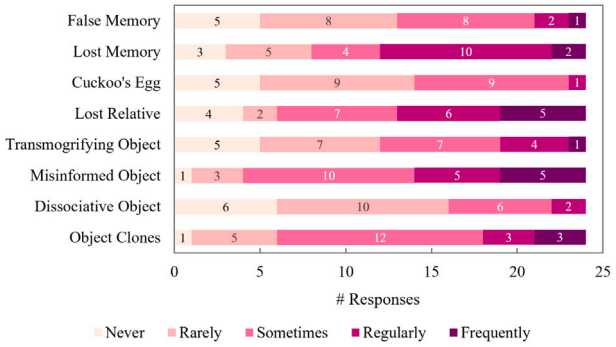


Fig. 14. Pervasiveness of the OCED imperfection patterns.

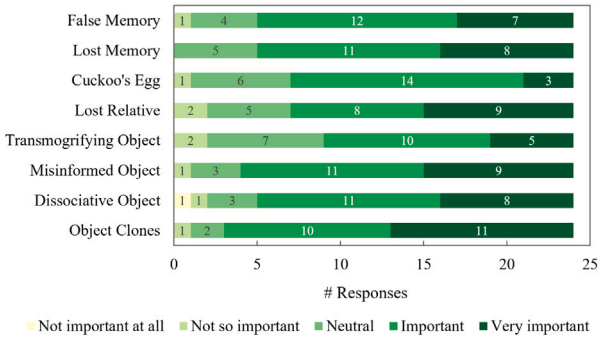


Fig. 15. Perceived importance of the recognition of the OCED imperfection patterns.

all but one of the participants had analysed data sets using process mining techniques so far; 46% had analysed more than 20 data sets, 33% had analysed 6–20 data sets, and 17% had analysed 2–5 data sets. Healthcare was the most pervasive domain for process mining, with 83% of the participants indicating that they have analysed data sets from this domain. Public sector (66%), finance/logistics (63%), and education (46%) were the other dominant domains for process mining. ProM (79%), PM4Py (75%), Disco (71%), and Celonis (42%) were the most popular tools or software packages for process mining among the respondents. 92% of the participants had been involved with data preprocessing (cleaning) for process mining, 25% had cleaned more than 20 data sets, 38% cleaned 6–20 data sets, and 25% cleaned 2–5 data sets. Finally, 92% of the participants knew about the object-centric notion of process event logs, with 50% being familiar and 42% being somewhat familiar.

Figs. 14, and 15 summarise the responses received for the pervasiveness and perceived importance of the recognition of OCED imperfection patterns in data sets. The patterns were reported to be highly pervasive as every pattern was observed by at least 18 participants (75%). On average, 26% of participants had noticed these patterns regularly or frequently. As shown in Fig. 15, the patterns were rated as important or very important by an average of 76% of participants. This shows that the process mining community confirms the importance of the recognition of these patterns in object-centric event data.

Figs. 16 and 17 present the survey results from a different angle, with a focus on participants and their experience (measured as the number of data sets they have analysed so far). For each of the 24 survey participants (P1–24) we show the average score they assigned to the pervasiveness of all patterns (in Fig. 16) and the average score they assigned to the importance of the recognition of all patterns (in Fig. 17). The purpose of this analysis was to see if there is a correlation between a participant's experience and the score they assign to the pervasiveness and the importance of the recognition of the patterns. In these charts the participants are sorted based on their experience from 0–1 data set analysed to more than 20 data sets analysed. The trend line in Fig. 16 shows that the more experience participants had, the more OCED imperfection patterns they observed in real-life data sets. Similarly, the trend line in Fig. 17 shows that more experienced participants had more awareness about the importance of the recognition of these patterns in real-life data sets.

Fig. 18 reports the Likert-type scores achieved for the pervasiveness and importance of object-centric imperfection patterns. Overall, the average pervasive score for all patterns was 2.78 with a standard deviation of 1.17. The patterns that were observed most frequently were Misinformed Object (M = 3.42, SD = 1.08), Lost Relative (M = 3.25, SD = 1.33) and Lost Memory (M = 3.13, SD = 1.20). On the other hand, Dissociative Object (M = 2.17, SD = 0.90), Cuckoo's Egg (M = 2.25, SD = 0.83), and False Memory (M = 2.42, SD = 1.04) were acknowledged as the least pervasive patterns by participants.

The results reported in Fig. 18 show good support for the importance of the identification of these patterns in event data, with an average score of 4.05 and a standard deviation of 0.85. Object Clones (M = 4.29, SD = 0.78), Misinformed Object (M = 4.17, SD = 0.80), and Lost Memory (M = 4.13, SD = 0.73) were recognised as the most important patterns to recognise, whereas, Transmogrifying Object (M = 3.75, SD = 0.88), Cuckoo's Egg (M = 3.80, SD = 0.71), and Lost Relative (M = 4.00, SD = 0.96) were perceived as the least important ones to recognise in event logs.

6.4. Discussion

The data validation described in Section 6.1 supports the relevance of the patterns in real-life data sets (DO1). We have observed most of the patterns in private real-life data sets accessible to the team. However, only the two patterns that are about missing data, i.e., Lost Relative and Lost Memory, were detectable in public data sets. This is because the detection of other patterns needed additional information to the event log; for instance, to detect Object Clones, we needed to know if the two objects with different object identifiers actually refer to the same object in reality. Such information was provided to us by the stakeholders for private data sets, but was not available for the public ones. Furthermore, while the patterns are generic, more evidence is provided from the healthcare data sets that the team has access to. We cannot release the data sets publicly due to their confidential nature. In future research, we would like to study data sets from other domains to strengthen the evidence base.

The results achieved by the prototypical detection techniques described in Section 6.2 confirm the feasibility of (semi-)automatic detection of OCED imperfection patterns in real-life event logs (DO4). We used various detection heuristics such as context-based similarity assessment, linear regression, and outlier analysis. We used key performance metrics, Recall, Precision, and Accuracy, to measure the success of the detection methods. We prioritised achieving a high Recall to minimise missed imperfections and ensure that the majority of true errors are captured. As such, all of our detection techniques achieved a high Recall (81% or higher). However, prioritising Recall might sometimes correspond to a moderately lower Precision. The precision achieved by the detection techniques was 56% or higher, and the accuracy was 68% or higher. There was one exceptionally low precision (13%) achieved by the context similarity-based approach used for detecting Object Clones. This result stems from the nature of the data set, where there is minimal variation in attribute values across traces. As a consequence, the similarity-based approach identified a large number of normal traces as object clones due to their shared attributes with actual imperfect objects. This led to a high rate of false positives, ultimately lowering the Precision.

The data sets used in the experiments with detection techniques were real-life event logs with object-centric nature, i.e., BPIC 2017

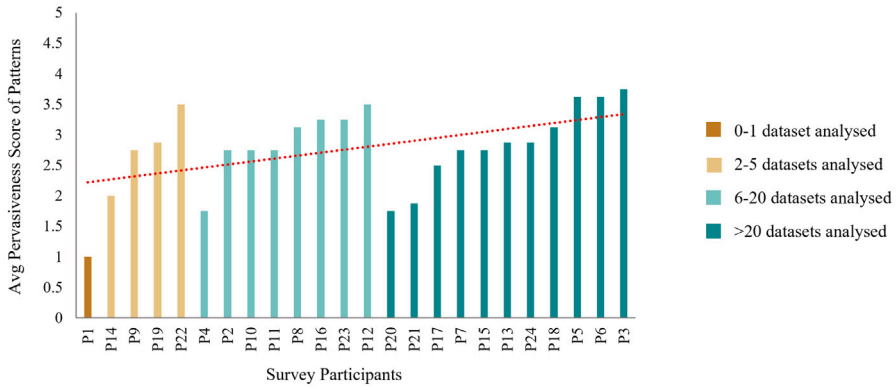


Fig. 16. Average pervasiveness of the OCED imperfection patterns per survey participant, ordered by the number of data sets analysed by the participant.

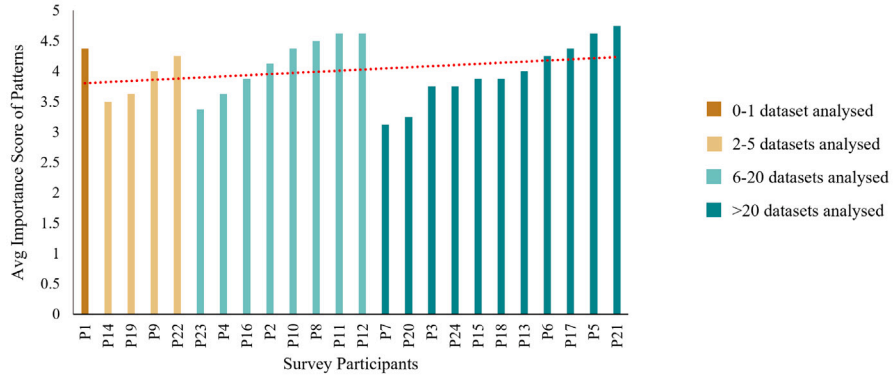


Fig. 17. Average perceived importance of the recognition of the OCED imperfection patterns per survey participant, ordered by the number of data sets analysed by the participant.

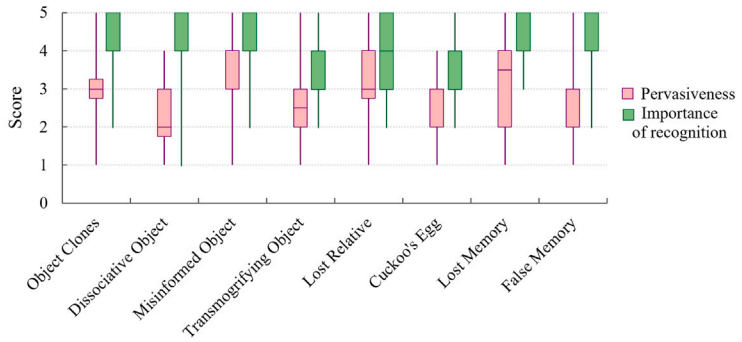


Fig. 18. Pervasiveness and importance of the recognition of the OCED imperfection patterns.

and BPIC 2020. Two of the eight OCED imperfection patterns were already present in the BPIC 2020 event log, however, for the other six patterns (i.e., Object Clones, Dissociative Object, Misinformed Object, Transmogrifying Object, Lost Memory, and False Memory patterns) we injected pseudo-real errors into the BPIC 2017 event log. Given that object-centric event logs are relatively new to the community, having access to object-centric logs with real errors with existing ground truth is challenging. Our expanded evaluation, which involved applying the detection techniques to multiple additional public real-life logs, confirmed the presence and provided frequency counts for all eight patterns in real-life datasets (Table 2), providing evidence for the prevalence and relevance of our patterns (DO1). As more real-life object-centric logs become publicly available, additional patterns could emerge, and further evaluations of existing patterns could be conducted.

The respondents to our user survey described in Section 6.3 represent a mature population of process mining researchers, as most of them have more than 5 years of experience in the field. The results show that most of these process mining researchers are familiar with the object-centric notion of event logs and have observed the OCED imperfection patterns in data sets from various domains (DO1). More significantly, due to the implications they can have on the reliability of process mining results, the detection of the OCED imperfection patterns is viewed as highly important by the experts (DO2). We can also see that the scores achieved for the importance of the patterns are higher than those achieved for their pervasiveness; this was expected

due to the emerging nature of the object-centric notion in process mining [2,54] and not many object-centric data sets being available to the researchers. We also found out that there is a direct relationship between a user's level of experience with real-life data sets and the frequency of the OCED imperfection patterns they observed. Similarly, more experienced users were more aware of the importance of the recognition of the OCED imperfection patterns due to the impact these patterns can have on the reliability of process mining results. While we collected responses from a rich population of process mining researchers, we acknowledge that more respondents from process mining practitioners could further support the pervasiveness, importance of the recognition, and implications of the patterns in practice. Finally, while a quantitative evaluation is important, the fact that every pattern has been observed by at least 18 participants and has been found important or very important by at least 15 participants is important in its own right. These absolute numbers are more important than the relative ones because it matters less whether some people have not observed certain patterns or find them less important. For each pattern, there is enough evidence that they are found to be important and that they do occur in practice. We acknowledge that the survey measured perceived relevance of the patterns among experts who self-reported familiarity with object-centric process mining but did not uniformly confirm hands-on experience with creating or analysing OCED logs. The survey was conducted in early 2023, when OCED is in its infancy, and the community may only have limited hands-on experience working with such logs. Future work can conduct another survey now that the community is more familiar with such logs.

In this study, the pervasiveness of the OCED imperfection patterns (DO1) is evaluated in two ways: one by directly searching for instances of these patterns in public/private real-world datasets, and the other by asking domain experts in the process mining community to comment on their pervasiveness in real-life datasets. It would be interesting to compare the results of these two experiments (reported in Table 2 and Fig. 18) and see how they complement one another. Based on the data, we see the Lost Relative pattern being the most pervasive pattern, and it is also the second most pervasive based on the survey. The least frequent pattern, according to the user survey, is Dissociative Object; however, we found evidence of it in three of the 10 real-life datasets. The Misinformed Object pattern is also observed in three out of 10 datasets, and it is the top-most pervasive pattern based on the survey. The Lost Memory pattern, on the other hand, is observed in two datasets, while being the third most common pattern as per the survey. Overall, these results together indicate that all eight OCED imperfection patterns characterised in this paper are pervasive in real-world datasets, as they are either (or both) directly observed in datasets or their pervasiveness in datasets is reported by domain experts. In the future, as more OCED event logs become available to the process mining community, we might see certain patterns become more pervasive than others, which can be studied using new experiments and surveys.

While some of the identified imperfection patterns may resemble issues known from relational database (RDB) contexts — such as missing relationships or inconsistent attributes — their manifestation in object-centric event data (OCED) is semantically distinct. Unlike traditional RDBs, OCED logs are designed to support dynamic, multi-object, and multi-perspective process analysis, where events can involve multiple objects of different types and where object–object and object–event relations are first-class citizens. This introduces new challenges such as ambiguous object–event bindings, inconsistent object lifecycles, and missing or conflicting object–object relations—issues that are not adequately addressed by existing event-centric or RDB-centric data quality frameworks. Our pattern collection is novel in that it systematically characterises these OCED-specific imperfections, grounded in a core metamodel aligned with OCED and OCEL 2.0 standards. To our knowledge, this is the first structured effort to define, categorise, and evaluate data quality issues that are intrinsic to the object-centric paradigm in process mining. This contribution fills a critical gap in the literature and provides a foundation for future detection, repair, and prevention techniques tailored to OCED.

7. Conclusion

This paper presents a pattern-based approach to characterise data quality problems that can occur in object-centric event logs. We propose an extendable collection of eight OCED imperfection patterns based on a core technology-independent metamodel that represents generic features of object-centric event logs. These patterns focus on the object-centric elements of event logs: objects, object attributes, object–object relations, and object–event relations. Each pattern is described by a definition, possible root-causes for its occurrence, and suggestions to detect and repair its existence in the log. The patterns are evaluated using a multi-prong approach, i.e., evidence-based, literature-based, empirical, and user-based evaluation methods, further strengthening our contribution. The evidence-based approach shows that the patterns do indeed occur in public and private real-life data sets. The literature-based evaluation provides evidence from the literature on the occurrence of OCED patterns in real-life data sets. In the empirical evaluation, we implemented a prototypical detection method for each of the imperfection patterns and validated it using real-life event logs (with pseudo-real errors). The results of this evaluation show that the technique can detect most of the instances of the OCED imperfection patterns, achieving a high Recall and Accuracy. The empirical evaluation serves as an initial step towards a dedicated tool support for (semi-)automated detection and/or repair of the patterns. Finally, the user-based evaluation is conducted within the process mining community to evaluate the pervasiveness and importance of the recognition of the patterns. The responses collected from 24 experts support the pervasiveness and, more significantly, the importance of recognising the patterns in real-life scenarios. The main focus of this paper was to develop an initial set of OCED imperfection patterns. It provides a theoretical foundation for systematic detection, quantification, and repair of the patterns as the logical next step. The techniques can be configured in a dedicated tool, similar to the PraeclarusPDQ open-source framework, which is a suite of detection and repair techniques organised based on case-centric event log imperfection pattern collection. Future work can also explore adding more patterns to our collection, as more object-centric elements are introduced.

CRediT authorship contribution statement

Sareh Sadeghianasl: Writing – review & editing, Writing – original draft, Visualization, Validation, Methodology, Investigation, Formal analysis, Data curation, Conceptualization. **Moe Thandar Wynn:** Writing – review & editing, Methodology, Conceptualization. **Robert Andrews:** Writing – review & editing, Writing – original draft, Validation, Methodology, Investigation, Conceptualization. **Wil M.P. van der Aalst:** Writing – review & editing, Methodology, Conceptualization. **Jonghyeon Ko:** Writing – review & editing, Writing – original draft, Visualization, Validation, Software, Investigation, Data curation.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

We acknowledge contributions made by Arthur ter Hofstede to the earlier version of this paper. Some of the data sets used in this study were made available by, and with the permission of, Queensland's Motor Accident Insurance Commission (MAIC).

Data availability

No data was used for the research described in the article.

References

[1] W. Van der Aalst, Getting the data, in: Process Mining: Data Science in Action, second ed., Springer, 2016, pp. 125–163.

[2] A.F. Ghahfarokhi, G. Park, A. Berti, W. van der Aalst, OCEL: a standard for object-centric event logs, in: New Trends in Database and Information Systems - ADBIS, in: Communications in Computer and Information Science, vol. 1450, Springer, 2021, pp. 169–175.

[3] H.M.W. Verbeek, J. Buijs, B. van Dongen, W. van der Aalst, XES, xesame, and prom 6, in: Information Systems Evolution - CAiSE Forum, in: LNBIP, vol. 72, Springer, 2010, pp. 60–75.

[4] W.M.P. van der Aalst, A. Berti, Discovering object-centric Petri nets, Fund. Inform. 175 (1–4) (2020) 1–40.

[5] W. van der Aalst, Object-Centric Process Mining: The Next Frontier in Business Performance, Tech. rep., Celonis, 2023, URL https://celon.is/OCPM-Whitepaper.

[6] W. van der Aalst, Object-centric process mining: Dealing with divergence and convergence in event data, in: International Conference on Software Engineering and Formal Methods, Springer, 2019, pp. 3–25.

[7] J.N. Adams, D. Schuster, S. Schmitz, G. Schuh, W.M.P. van der Aalst, Defining cases and variants for object-centric event data, in: International Conference on Process Mining (ICPM), IEEE, 2022, pp. 128–135.

[8] W.M.P. van der Aalst, Concurrency and objects matter! disentangling the fabric of real operational processes to create digital twins, in: International Colloquium on Theoretical Aspects of Computing (ICTAC), in: LNCS, vol. 12819, Springer, 2021, pp. 3–17.

[9] B. Knopp, M. Pourbafrani, W.M.P. van der Aalst, Discovering object-centric process simulation models, in: International Conference on Process Mining, ICPM, IEEE, Rome, Italy, 2023, pp. 81–88, http://dx.doi.org/10.1109/ICPM60904.2023.10271944.

[10] A.K.F. Christfort, A. Rivkin, D. Fahland, T.T. Hildebrandt, T. Slaats, Discovery of object-centric declarative models, in: International Conference on Process Mining, ICPM, IEEE, Lyngby, Denmark, 2024, pp. 121–128, http://dx.doi.org/10.1109/ICPM63005.2024.10680680.

[11] J.N. van Detten, P. Schumacher, S.J.J. Leemans, Discovering compact, live and identifier-sound object-centric process models, in: International Conference on Process Mining, ICPM, IEEE, Lyngby, Denmark, 2024, pp. 113–120, http://dx.doi.org/10.1109/ICPM63005.2024.10680659.

[12] G. Park, J.N. Adams, W.M.P. van der Aalst, Conformance checking and performance analysis using object-centric directly-follows graphs, in: A. Marrella, M. Resinas, M. Jans, M. Rosemann (Eds.), Business Process Management, BPM, in: LNBIP, vol. 526, Springer, Krakow, Poland, 2024, pp. 179–196, http://dx.doi.org/10.1007/978-3-031-70418-5_11.

[13] A. Gianola, M. Montali, S. Winkler, Object-centric conformance alignments with synchronization, in: G. Guizzardi, F.M. Santoro, H. Mouratidis, P. Soffer (Eds.), International Conference on Advanced Information Systems (CAiSE), in: LNCS, 14663, Springer, Limassol, Cyprus, 2024, pp. 3–19, http://dx.doi.org/10.1007/978-3-031-61057-8_1.

[14] T.K. Smit, H.A. Reijers, X. Lu, HOEG: a new approach for object-centric predictive process monitoring, in: G. Guizzardi, F.M. Santoro, H. Mouratidis, P. Soffer (Eds.), International Conference on Advanced Information Systems (CAiSE), in: LNCS, vol. 14663, Springer, Limassol, Cyprus, 2024, pp. 231–247, http://dx.doi.org/10.1007/978-3-031-61057-8_14.

[15] Y. Gerlach, A. Seeliger, T. Nolle, M. Mühlhäuser, Inferring a multi-perspective likelihood graph from black-box next event predictors, in: X. Franch, G. Poels, F. Gailly, M. Snoeck (Eds.), International Conference on Advanced Information Systems (CAiSE), in: LNCS, vol. 13295, Springer, Leuven, Belgium, 2022, pp. 19–35, http://dx.doi.org/10.1007/978-3-031-07472-1_2.

[16] K.A. Elyasi, H. van der Aa, H. Stuckenschmidt, Pgtnet: A process graph transformer network for remaining time prediction of business process instances, in: G. Guizzardi, F.M. Santoro, H. Mouratidis, P. Soffer (Eds.), International Conference on Advanced Information Systems (CAiSE), in: LNCS, vol. 14663, Springer, Limassol, Cyprus, 2024, pp. 124–140, http://dx.doi.org/10.1007/978-3-031-61057-8_8.

[17] Y. Bertrand, J.D. Weerdt, E. Serral, A novel multi-perspective trace clustering technique for IoT-enhanced processes: A case study in smart manufacturing, in: C.D. Francescomarino, A. Burattin, C. Janiesch, S. Sadiq (Eds.), International Conference on Business Process Management, BPM, in: LNCS, vol. 14159, Springer, Utrecht, The Netherlands, 2023, pp. 395–412, http://dx.doi.org/10.1007/978-3-031-41620-0_23.

[18] E.L. Klijn, D. Preuss, L. Imeri, F. Baumann, F. Mannhardt, D. Fahland, Event knowledge graphs for auditing: A case study, in: J.D. Smedt, P. Soffer (Eds.), Process Mining Workshops, ICPM, in: LNBIP, vol. 503, Springer, Rome, Italy, 2023, pp. 84–97, http://dx.doi.org/10.1007/978-3-031-56107-8_7.

[19] S. Suriadi, R. Andrews, A. ter Hofstede, M. Wynn, Event log imperfection patterns for process mining: Towards a systematic approach to cleaning event logs, Inf. Syst. 64 (2017) 132–150.

[20] C. dos Santos Garcia, A. Meincheim, E.R.F. Junior, M.R. Dallagassa, D.M.V. Sato, D.R. Carvalho, E.A.P. Santos, E.E. Scalabrin, Process mining techniques and applications - a systematic mapping study, Expert Syst. Appl. 133 (2019) 260–295, http://dx.doi.org/10.1016/J.ESWA.2019.05.003.

[21] J. Munoz-Gama, N. Martin, C. Fernández-Llatas, et al., Process mining for healthcare: Characteristics and challenges, J. Biomededical Inform. 127 (2022) 103994, http://dx.doi.org/10.1016/J.JBI.2022.103994.

[22] F. Mannhardt, M. de Leoni, H.A. Reijers, W.M.P. van der Aalst, Data-driven process discovery - revealing conditional infrequent behavior from event logs, in: E. Dubois, K. Pohl (Eds.), International Conference on Advanced Information Systems (CAiSE), in: LNCS, vol. 10253, Springer, Essen, Germany, 2017, pp. 545–560, http://dx.doi.org/10.1007/978-3-319-59536-8_34.

[23] N. Tax, N. Sidorova, W.M.P. van der Aalst, Discovering more precise process models from event logs by filtering out chaotic activities, J. Intell. Inf. Syst. 52 (1) (2019) 107–139, http://dx.doi.org/10.1007/S10844-018-0507-6.

[24] R. Andrews, F. Emamjome, A.H. ter Hofstede, H.A. Reijers, An expert lens on data quality in process mining, in: International Conference on Process Mining, ICPM, IEEE, 2020, pp. 49–56.

[25] K. Goel, S. Sadeghianasl, R. Andrews, A.H.M. ter Hofstede, M. Wynn, D.K. Geeganage, S.J.J. Leemans, J.M. McGree, R. Eden, A. Staib, R. Eley, R. Donovan, Digital health data imperfection patterns and their manifestations in an Australian digital hospital, in: T.X. Bui (Ed.), Hawaii International Conference on System Sciences (HICSS), ScholarSpace, 2023, pp. 3235–3244.

[26] R. Syed, R. Eden, T. Makasi, I. Chukwudi, A. Mamudu, M. Kamalpour, D. Kapugama Geeganage, S. Sadeghianasl, S.J. Leemans, K. Goel, et al., Digital health data quality issues: Systematic review, J. Med. Internet Res. 25 (2023) e42615.

[27] R. Andrews, S. Suriadi, C. Ouyang, E. Poppe, Towards event log querying for data quality: Let's start with detecting log imperfections, in: International Conference on Cooperative Information Systems (CoopIS), Springer, Valletta, Malta, 2018, pp. 116–134.

[28] R. Conforti, M.L. Rosa, A.H.M. ter Hofstede, A. Augusto, Automatic repair of same-timestamp errors in business process event logs, in: D. Fahland, C. Ghidini, J. Becker, M. Dumas (Eds.), International Conference on Business Process Management, BPM, in: LNCS, vol. 12168, Springer, Seville, Spain, 2020, pp. 327–345, http://dx.doi.org/10.1007/978-3-030-58666-9_19.

[29] S. Sadeghianasl, A. ter Hofstede, M. Wynn, S. Suriadi, A contextual approach to detecting synonymous and polluted activity labels in process event logs, in: International Conference on Cooperative Information Systems (CoopIS), in: LNCS, vol. 11877, Springer, 2019, pp. 76–94.

[30] S. Sadeghianasl, A.H.M. ter Hofstede, S. Suriadi, S. Turkay, Collaborative and interactive detection and repair of activity labels in process event logs, in: International Conference on Process Mining, ICPM, IEEE, 2020, pp. 41–48.

[31] D.A. Fischer, K. Goel, R. Andrews, C. van Dun, M. Wynn, M. Röglinger, Towards interactive event log forensics: Detecting and quantifying timestamp imperfections, Inf. Syst. 109 (2022) 102039.

[32] F. Zetzsche, R. Andrews, A.H. ter Hofstede, M. Röglinger, S.J. Schmid, M.T. Wynn, Case ID revealed HERE: Hybrid elusive case repair method for transformer-driven business process event log enhancement, Bus. Inf. Syst. Eng. 67 (3) (2025) 311–337.

[33] D. Fahland, M. Montali, J. Lebherz, W.M.P. van der Aalst, M. van Asseldonk, P. Blank, L. Bosmans, M. Brenscheidt, C. Di Ciccio, A. Delgado, D. Calegari, J. Peeperkorn, E. Verbeek, L. Vugs, M.T. Wynn, Towards a Simple and Extensible Standard for Object-Centric Event Data (OCED) - Core Model, Design Space, and Lessons Learned, Tech. rep., IEEE Task Force in Process Mining, 2023, URL https://www.tf-pm.org/upload/1728651303832.pdf.

[34] A. Berti, I. Koren, J.N. Adams, G. Park, B. Knopp, N. Graves, M. Rafiei, L. Liß, L.T.G. Unterberg, Y. Zhang, et al., OCEL (Object-Centric Event Log) 2.0 Specification, Tech. rep., RWTH Aachen University, 2023, URL https://www.ocel-standard.org/2.0/ocel20_specification.pdf.

[35] R.Y. Wang, V.C. Storey, C.P. Firth, A framework for analysis of data quality research, IEEE Trans. Knowl. Data Eng. 7 (4) (1995) 623–640, http://dx.doi.org/10.1109/69.404034.

[36] Y. Wand, R.Y. Wang, Anchoring data quality dimensions in ontological foundations, Commun. ACM 39 (11) (1996) 86–95.

[37] R.Y. Wang, D.M. Strong, Beyond accuracy: What data quality means to data consumers, J. Manage. Inf. Syst. 12 (4) (1996) 5–33.

[38] C. Batini, M. Scannapieco, Data quality: Concepts, methodologies and techniques, Data-Centric Systems and Applications, Springer, 2006, http://dx.doi.org/10.1007/3-540-33173-5.

[39] N. Laranjeiro, S.N. Soydemir, J. Bernardino, A survey on data quality: Classifying poor data, in: Pacific Rim International Symposium on Dependable Computing, PRDC, 2015, pp. 179–188, http://dx.doi.org/10.1109/PRDC.2015.41.

[40] ISO/IEC 25012. Software Engineering — Software Product Quality Requirements and Evaluation (SQuaRE) — Data Quality Model, ISO/IEC, International Standard, 2008.

[41] W. van der Aalst, A. Adriansyah, A.K.A. Medeiros, et al., Process mining manifesto, in: Business Process Management Workshops, BPM, in: LNBIP, Springer, 2011, pp. 169–194.

[42] R.S. Mans, W. Van der Aalst, R.J. Vanwersch, A.J. Moleman, Process mining in healthcare: Data challenges when answering frequently posed questions, in: Process Support and Knowledge Representation in Health Care, Springer, 2012, pp. 140–153.

[43] R.J.C. Bose, R.S. Mans, W. van der Aalst, Wanna improve process mining results - it's high time we consider data quality issues seriously, in: Computational Intelligence and Data Mining Symposium (CIDM), IEEE, 2013, pp. 127–134.

[44] M.O. Kherbouche, N. Laga, P.-A. Masse, Towards a better assessment of event logs quality, in: Symposium Series on Computational Intelligence, SSCI, IEEE, 2016, pp. 1–8.

[45] F. Fox, V.R. Aggarwal, H. Whelton, O. Johnson, A data quality framework for process mining of electronic health record data, in: International Conference on Healthcare Informatics, ICHI, IEEE, 2018, pp. 12–21.

[46] S. Sadeghianasl, A.H.M. ter Hofstede, M.T. Wynn, S. Turkay, T. Myers, Process activity ontology learning from event logs through gamification, IEEE Access 9 (2021) 165865–165880.

[47] S. Sadeghianasl, A.H.M. ter Hofstede, M.T. Wynn, S. Türkay, Humans-in-the-loop: Gamifying activity label repair in process event logs, Eng. Appl. Artif. Intell. 132 (2024) 107875, http://dx.doi.org/10.1016/J.ENGAPPAI.2024.107875.

[48] K. Busch, T. Kampik, H. Leopold, Xsemad: Explainable semantic anomaly detection in event logs using sequence-to-sequence models, in: A. Marrella, M. Resinas, M. Jans, M. Rosemann (Eds.), International Conference on Business Process Management, BPM, in: LNCS, vol. 14940, Springer, Krakow, Poland, 2024, pp. 309–327, http://dx.doi.org/10.1007/978-3-031-70396-6_18.

[49] S.J. van Zelst, J. Tai, M. Langenberg, X. Lu, Context-based activity label-splitting, in: C.D. Francescomarino, A. Burattin, C. Janiesch, S. Sadiq (Eds.), International Conference on Business Process Management, BPM, in: LNCS, vol. 14159, Springer, Utrecht, The Netherlands, 2023, pp. 232–248, http://dx.doi.org/10.1007/978-3-031-41620-0_14.

[50] R. Andrews, F. Emamjome, A. ter Hofstede, H. Reijers, Root-cause analysis of process-data quality problems, J. Bus. Anal. 5 (1) (2022) 51–75.

[51] S. Ghalibafan, S. Sadeghianasl, M.T. Wynn, ERooMiner: A data-driven approach for root cause detection of process data quality issues, in: C. Cappiello, O. Hartig, M. Sellami, A. Ouni (Eds.), International Conference on Cooperative Information Systems (CoopIS), in: LNCS, Springer, Marbella, Spain, 2025, pp. 33–51, http://dx.doi.org/10.1007/978-3-032-15538-2_3.

[52] G. Li, E.G.L. de Murillas, R.M. de Carvalho, W. van der Aalst, Extracting object-centric event logs to support process mining on databases, in: Information Systems in the Big Data Era - CAiSE Forum, in: LNBIP, vol.317, Springer, 2018, pp. 182–199.

[53] B. Xiu, G. Li, Y. Li, Discovery of object-centric behavioral constraint models with noise, IEEE Access 10 (2022) 88769–88786.

[54] A. Berti, W. van der Aalst, OC-pm: Analyzing object-centric event logs and process models, Int. J. Softw. Tools Technol. Transf. (2022) 1–17.

[55] A. Berti, W.M.P. van der Aalst, Extracting multiple viewpoint models from relational databases, in: International Symposium on Data-Driven Process Discovery and Analysis (SIMPDA), in: LNBIP, vol.379, Springer, 2019, pp. 24–51.

[56] W. van der Aalst, A. Berti, Discovering object-centric Petri nets, Fund. Inform. 175 (1–4) (2020) 1–40.

[57] D. Fahland, Process mining over multiple behavioral dimensions with event knowledge graphs, in: Process Mining Handbook, in: LNBIP, vol. 448, Springer, 2022, pp. 274–319.

[58] J.N. Adams, W. van der Aalst, Ocπ: Object-centric process insights, in: International Conference on Application and Theory of Petri Nets and Concurrency, in: LNCS, vol.13288, Springer, 2022, pp. 139–150.

[59] C. Di Ciccio, M. Montali, Declarative process specifications: Reasoning, discovery, monitoring, in: Process Mining Handbook, in: LNBIP, vol. 448, Springer, 2022, pp. 108–152.

[60] M. Pesic, H. Schonenberg, W.M.P. van der Aalst, DECLARE: full support for loosely-structured processes, in: IEEE International Enterprise Distributed Object Computing Conference - EDOC, IEEE Computer Society, 2007, pp. 287–300.

[61] W. van der Aalst, A. Artale, M. Montali, S. Tritini, Object-centric behavioral constraints: Integrating data and declarative process modelling, in: International Workshop on Description Logics, in: CEUR Workshop Proceedings, vol. 1879, CEUR-WS.org, 2017.

[62] A. Goossens, J.D. Smedt, J. Vanthienen, W.M.P. van der Aalst, Enhancing data-awareness of object-centric event logs, in: M. Montali, A. Senderovich, M. Weidlich (Eds.), Process Mining Workshops (ICPM), in: LNBIP, vl. 468, Springer, 2022, pp. 18–30.

[63] A. Buss, C. Kecht, W. Kratsch, M. Röglinger, S. Sadeghianasl, M.T. Wynn, From words to workflows: Extracting object-centric event logs from textual data, in: L. Pufahl, K. Rosenthal, S.E. na, S. Nurcan (Eds.), Intelligent Information Systems - CAiSE Forum and Doctoral Consortium, in: LNBIP, vol. 557, Springer, Vienna, Austria, 2025, pp. 37–44, http://dx.doi.org/10.1007/978-3-031-94590-8_5.

[64] A. Buss, C. Kecht, W. Kratsch, M. Röglinger, S. Sadeghianasl, M.T. Wynn, Process mining between the lines: Extracting object-centric event logs from textual data, Inf. Syst. 140 (2026) 102713, http://dx.doi.org/10.1016/J.IS.2026.102713.

[65] M. Basmer, M. Kabierski, K. Sahling, A. Patecka, S. Bala, J. Mendling, A classification of data quality issues in object-centric event data, in: A. Delgado, T. Slaats (Eds.), Process Mining Workshops (ICPM), Springer, 2025, pp. 311–323.

[66] D. Calvanese, S. Lukumbuzya, M. Montali, M. Simkus, Process mining with common sense, in: Proceedings of the International Workshop on BPM Problems To Solve before We Die (PROBLEMS) Co-Located with International Conference on Business Process Management (BPM), in: CEUR Workshop Proceedings, vol. 2938, CEUR-WS.org, 2021, pp. 45–50.

[67] A. Swevels, R.M. Dijkman, D. Fahland, Inferring missing entity identifiers from context using event knowledge graphs, in: C.D. Francescomarino, A. Burattin, C. Janiesch, S. Sadiq (Eds.), International Conference on Business Process Management, BPM, in: LNCS, vol. 14159, Springer, Utrecht, The Netherlands, 2023, pp. 180–197, http://dx.doi.org/10.1007/978-3-031-41620-0_11.

[68] O.P. Rotaru, M. Petrescu, A database integrity pattern language, Leonardo J. Sci. 5 (2004) 46–62.

[69] A.R. Hevner, S.T. March, J. Park, S. Ram, Design science in information systems research, MIS Q. 28 (1) (2004) 75–105, http://dx.doi.org/10.2307/25148625.

[70] K. Peffers, T. Tuunanen, M.A. Rothenberger, S. Chatterjee, A design science research methodology for information systems research, J. Manage. Inf. Syst. 24 (3) (2007) 45–77, http://dx.doi.org/10.2753/MIS0742-1222240302.

[71] T.A. Halpin, Object-role modeling (ORM/NIAM), in: Handbook on Architectures of Information Systems, in: International handbooks on information systems, Springer, 2006, pp. 81–103.

[72] P. van Bommel, A. ter Hofstede, T. van der Weide, Semantics and verification of object-role models, Inf. Syst. 16 (5) (1991) 471–495.

[73] A. ter Hofstede, T.P. van der Weide, Expressiveness in conceptual data modelling, Data Knowl. Eng. 10 (1993) 65–100.

[74] R.S. Mans, W. Van der Aalst, R.J. Vanwersch, Data quality issues, in: Process Mining in Healthcare: Evaluating and Exploiting Operational Healthcare Processes, Springer Briefs in Business Process Management, 2015, pp. 79–88.

[75] C. Alexander, S. Ishikawa, M. Silverstein, M. Jacobson, I. Fiksdahl-King, S. Angel, A Pattern Language - Towns, Buildings, Construction, Oxford University Press, 1977.

[76] M. Ebraheem, S. Thirumuruganathan, S. Joty, M. Ouzzani, N. Tang, Distributed representations of tuples for entity resolution, Proc. Very Large Data Bases Endow. (PVLDB) 11 (11) (2018) 1454–1467.

[77] M.A. McClellan, Duplicate medical records: A survey of twin cities healthcare organizations, in: American Medical Informatics Association (AMIA) Annual Symposium Proceedings, vol. 2009, 2009, p. 421, PMID: 20351892.

[78] E. Joffe, C.F. Bearden, M.J. Byrne, E.V. Bernstam, Duplicate patient records–implication for missed laboratory results, in: American Medical Informatics Association (AMIA) Annual Symposium Proceedings, vol. 2012, 2012, p. 1269, PMID:23304405.

[79] M. Weis, F. Naumann, U. Jehle, J. Lufter, H. Schuster, Industry-scale duplicate detection, Proc. Very Large Data Bases Endow. (PVLDB) 1 (2) (2008) 1253–1264.

[80] R. Andrews, S. Suriadi, M. Wynn, A. ter Hofstede, S. Rothwell, Improving patient flows at st. Andrew's war memorial hospital's emergency department through process mining, in: Business Process Management Cases, Digital Innovation and Business Transformation in Practice, in: Management for Professionals, Springer, 2018, pp. 311–333.

[81] S. Mudgal, H. Li, T. Rekatsinas, et al., Deep learning for entity matching: A design space exploration, in: International Conference on Management of Data (SIGMOD), ACM, 2018, pp. 19–34.

[82] J. Wang, T. Kraska, M.J. Franklin, J. Feng, Crowder: Crowdsourcing entity resolution, Proc. Very Large Data Bases Endow. (PVLDB) 5 (11) (2012) 1483–1494.

[83] C. Gokhale, S. Das, A. Doan, et al., Corleone: Hands-off crowdsourcing for entity matching, in: International Conference on Management of Data (SIGMOD), ACM, 2014, pp. 601–612.

[84] W.W. Cohen, Data integration using similarity joins and a word-based information representation language, Trans. Inf. Syst. 18 (3) (2000) 288–321.

[85] M.A. Yosef, J. Hoffart, I. Bordino, M. Spaniol, G. Weikum, AIDA: An online tool for accurate disambiguation of named entities in text and tables, Proc. Very Large Data Bases Endow. (PVLDB) 4 (12) (2011) 1450–1453.

[86] A. Saxena, M. Prasad, A. Gupta, et al., A review of clustering techniques and developments, Neurocomputing 267 (2017) 664–681.

[87] N. Jardine, C.J. van Rijsbergen, The use of hierarchic clustering in information retrieval, Inf. Storage Retr. 7 (5) (1971) 217–240.

[88] Herald and Times archive, Move to end day of jackal-like identity theft, 2007, https://www.heraldscotland.com/default_content/12780054.move-end-day-jackal-like-identity-theft/. (Accessed 4 April 2026).

[89] H. Kriegel, P. Kröger, J. Sander, A. Zimek, Density-based clustering, WIREs Data Min. Knowl. Discov. 1 (3) (2011) 231–240.

[90] M.A.P. e Silva, Dynamic integrity constraints definition and enforcement in databases: A classification framework, in: Integrity and Internal Control in Information Systems, in: IFIP Conference Proceedings, vol. 109, Chapman Hall, 1997, pp. 65–87.

[91] M. Wynn, E. Poppe, J. Xu, A. ter Hofstede, R. Brown, A. Pini, W. van der Aalst, ProcessProfiler3D: A visualisation framework for log-based process performance comparison, Decis. Support Syst. 100 (2017) 93–108.

[92] F. Mannhardt, M. de Leoni, H. Reijers, W. van der Aalst, Balanced multi-perspective checking of process conformance, Computing 98 (4) (2016) 407–437.

[93] F. Mannhardt, M. de Leoni, H. Reijers, Heuristic mining revamped: An interactive, data-aware, and conformance-aware miner, in: Proceedings of the BPM Demo Track and BPM Dissertation Award Co-Located with 15th International Conference on Business Process Modeling, in: CEUR Workshop Proceedings, vol. 1920, CEUR-WS.org, 2017.
[94] G. Levchuk, M. Jackobsen, B. Riordan, Detecting misinformation and knowledge conflicts in relational data, in: Signal Processing, Sensor/Information Fusion, and Target Recognition XXIII, vol. 9091, 2014, pp. 235–248.
[95] I. Dagan, B. Dolan, B. Magnini, D. Roth, Recognizing textual entailment: Rational, evaluation and approaches–erratum, Nat. Lang. Eng. 16 (1) (2010) 105.
[96] S. Krishnan, J. Wang, E. Wu, M.J. Franklin, K. Goldberg, Activeclean: Interactive data cleaning for statistical modeling, Proc. Very Large Data Bases Endow. (PVLDB) 9 (12) (2016) 948–959.
[97] RuleQuest Research Pty Ltd, Gritbot, 2020, https://www.rulequest.com/gritbot-info.html. (Accessed 4 April 2026).
[98] M. Mahdavi, F. Neutatz, L. Visengeriyeva, Z. Abedjan, Towards automated data cleaning workflows, in: Proceedings of the Conference on "Lernen, Wissen, Daten, Analysen", in: CEUR Workshop Proceedings, vol. 2454, CEUR-WS.org, 2019, pp. 10–19.
[99] Z. Abedjan, X. Chu, D. Deng, et al., Detecting data errors: Where are we and what needs to be done? Proc. Very Large Data Bases Endow. (PVLDB) 9 (12) (2016) 993–1004.
[100] Z. Kedad, E. Métais, Ontology-based data cleaning, in: Natural Language Processing and Information Systems, 6th International Conference on Applications of Natural Language To Information Systems, NLDB, in: LNCS, 2553, Springer, 2002, pp. 137–149.
[101] A.E. Johnson, T.J. Pollard, R.G. Mark, MIMIC-III clinical database (version 1.4). PhysioNet., 2016, http://dx.doi.org/10.13026/C2XW26.
[102] A.E. Johnson, T.J. Pollard, L. Shen, et al., MIMIC-III, a freely accessible critical care database, Sci. Data 3 (1) (2016) 1–9.
[103] A.P. Kurniati, E. Rojas, D. Hogg, G. Hall, O.A. Johnson, The assessment of data quality issues for process mining in healthcare using medical information mart for intensive care III, a freely available e-health record database, Health Inform. J. 25 (4) (2019) 1878–1893, PMID: 30488750.
[104] M.L. Astion, K.G. Shojania, T.R. Hamill, S. Kim, V.L. Ng, Classifying laboratory incident reports to identify problems that jeopardize patient safety, Am. J. Clin. Path. 120 (1) (2003) 18–26.
[105] J. Suchý, M. Suchy, M. Rosik, A. Valkova, Automate does not always mean optimize: Case study at a logistics company, in: Business Process Management Cases, Digital Innovation and Business Transformation in Practice, in: Management for Professionals, Springer, 2018, pp. 463–483.
[106] v. B.F., BPI challenge 2020. 4tu.ResearchData. Collection, 2020, http://dx.doi.org/10.4121/uuid:52fb97d4-4588-43c9-9d04-3604d4613b51.
[107] J. Elith, J.R. Leathwick, Species distribution models: Ecological explanation and prediction across space and time, Annu. Rev. Ecol. Evol. Syst. 40 (1) (2009) 677–697.
[108] A. Rogier, T. Donders, G.J. van der Heijden, T. Stijnen, K.G. Moons, Review: A gentle introduction to imputation of missing values, J. Clin. Epidemiol. 59 (10) (2006) 1087–1091.
[109] K. Khan, S. ur Rehman, K. Aziz, S. Fong, S. Sarasvady, A. Vishwa, DBSCAN: past, present and future, in: International Conference on the Applications of Digital Information and Web Technologies, ICADIWT, IEEE, 2014, pp. 232–238.
[110] P. Bonini, M. Plebani, F. Ceriotti, F. Rubboli, Errors in laboratory medicine, Clin. Chem. 48 (5) (2002) 691–698.
[111] M. Zaki, Scalable algorithms for association mining, IEEE Trans. Knowl. Data Eng. 12 (3) (2000) 372–390.
[112] W.J. Cook, A. Rohe, Computing minimum-weight perfect matchings, INFORMS J. Comput. 11 (2) (1999) 138–148.
[113] H. Tangmunarunkit, S. Decker, C. Kesselman, Ontology-based resource matching in the grid - the grid meets the semantic web, in: International Semantic Web Conference - ISWC, in: LNCS, vol. 2870, Springer, 2003, pp. 706–721.
[114] C.C. Aggarwal, An introduction to outlier analysis, in: Outlier Analysis, Springer, 2017, pp. 1–34.
[115] D. Dey, Entity matching in heterogeneous databases: A logistic regression approach, Decis. Support Syst. 44 (3) (2008) 740–747.
[116] R. Zass, A. Shashua, Probabilistic graph and hypergraph matching, in: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), IEEE Computer Society, 2008, pp. 1–8.
[117] R.M. Karp, U.V. Vazirani, V.V. Vazirani, An optimal algorithm for on-line bipartite matching, in: Proceedings of the 22nd Annual ACM Symposium on Theory of Computing, ACM, 1990, pp. 352–358.
[118] S. Khayatbashi, O. Hartig, A. Jalali, BPI challenge 2014 (OCEL), 2023, http://dx.doi.org/10.4121/7d097cec-7304-4b85-9e78-a3ca1cc44c40.v1.
[119] S. Khayatbashi, O. Hartig, A. Jalali, BPI challenge 2017 (OCEL), 2023, http://dx.doi.org/10.4121/6889ca3f-97cf-459a-b630-3b0b0d8664b5.v1.
[120] S. Khayatbashi, O. Hartig, A. Jalali, BPI challenge 2019 (OCEL), 2023, http://dx.doi.org/10.4121/46a7e15b-10c7-4ab2-988d-ee67d8ea515a.v1.
[121] A. Johnson, L. Bulgarelli, T. Pollard, S. Horng, L.A. Celi, R. Mark, MIMIC-IV, PhysioNet (2021) http://dx.doi.org/10.13026/s6n6-xd98, Version 1.0.
[122] R. Andrews, M.T. Wynn, K. Vallmuur, A.H.M. ter Hofstede, E. Bosley, M. Elcock, S. Rashford, Pre-hospital retrieval and transport of road trauma patients in queensland - a process mining analysis, in: F. Daniel, Q.Z. Sheng, H. Motahari (Eds.), Business Process Management (BPM) Workshops, in: LNBIP, vol. 342, Springer, Sydney, NSW, Australia, 2018, pp. 199–213, http://dx.doi.org/10.1007/978-3-030-11641-5_16.
[123] R. Andrews, M.T. Wynn, Shelf time analysis in CTP insurance claims processing, in: U. Kang, E. Lim, J.X. Yu, Y. Moon (Eds.), Trends and Applications in Knowledge Discovery and Data Mining - PAKDD 2017 Workshops, MLSDA, BDM, DM-BPM, Jeju, South Korea, May 23, 2017, Revised Selected Papers, in: LNCS, vol. 10526, Springer, 2017, pp. 151–162, http://dx.doi.org/10.1007/978-3-319-67274-8_14.
[124] M. Comuzzi, J. Ko, S. Lee, Predicting outpatient process flows to minimise the cost of handling returning patients: A case study, in: C.D. Francescomarino, R.M. Dijkman, U. Zdun (Eds.), Business Process Management (BPM) Workshops, in: LNBIP, 362, Springer, Vienna, Austria, 2019, pp. 557–569, http://dx.doi.org/10.1007/978-3-030-37453-2_45.
[125] X. Yang, G.K. Rajbahadur, D. Lin, S. Wang, Z.M. Jiang, Simclone: detecting tabular data clones using value similarity, ACM Trans. Softw. Eng. Methodol. 34 (1) (2025) 1–27.
[126] C. Bauer, G. King, Hibernate in Action, Manning Publications, 2005.
[127] J. Ko, M. Comuzzi, Detecting anomalies in business process event logs using statistical leverage, Inform. Sci. 549 (2021) 53–67.