*future internet*

MDPI

*Article*

# DataStream XES Extension: Embedding IoT Sensor Data into Extensible Event Stream Logs

**Juergen Mangler [1,*,†], Joscha Grüger [2,3,†], Lukas Malburg [2,3,†], Matthias Ehrendorfer [4,†], Yannis Bertrand [5,†], Janik-Vasily Benzin [1], Stefanie Rinderle-Ma [1], Estefania Serral Asensio [5] and Ralph Bergmann [2,3]**

[1] Department of Computer Science, School of Computation, Information and Technology, Technical University of Munich, 85748 Garching, Germany
[2] Artificial Intelligence and Intelligent Information Systems, University of Trier, 54296 Trier, Germany
[3] German Research Center for Artificial Intelligence (DFKI), Branch University of Trier, 54296 Trier, Germany
[4] Research Group Workflow Systems and Technology, Faculty of Computer Science, University of Vienna, 1090 Vienna, Austria
[5] Research Centre for Information Systems Engineering (LIRIS), KU Leuven, Warmoesberg 26, 1000 Brussels, Belgium
* Correspondence: juergen.mangler@gmail.com
† These authors contributed equally to this work.

**Abstract:** The Internet of Things (IoT) has been shown to be very valuable for Business Process Management (BPM), for example, to better track and control process executions. While IoT actuators can automatically trigger actions, IoT sensors can monitor the changes in the environment and the humans involved in the processes. These sensors produce large amounts of discrete and continuous data streams, which hold the key to understanding the quality of the executed processes. However, to enable this understanding, it is needed to have a joint representation of the data generated by the process engine executing the process, and the data generated by the IoT sensors. In this paper, we present an extension of the event log standard format XES called DataStream. DataStream enables the connection of IoT data to process events, preserving the full context required for data analysis, even when scenarios or hardware artifacts are rapidly changing. The DataStream extension is designed based on a set of goals and evaluated by creating two datasets for real-world scenarios from the transportation/logistics and manufacturing domains.

**Keywords:** process management; Industry 4.0; IoT data; process mining; XES



**Citation:** Mangler, J.; Grüger, J.; Malburg, L.; Ehrendorfer, M.; Bertrand, Y.; Benzin, J.-V.; Rinderle-Ma, S.; Serral Asensio, E.; Bergmann, R. DataStream XES Extension: Embedding IoT Sensor Data into Extensible Event Stream Logs. *Future Internet* **2023**, *15*, 109. https://doi.org/10.3390/fi15030109

Academic Editor: Paolo Bellavista

Received: 9 February 2023
Revised: 27 February 2023
Accepted: 9 March 2023
Published: 14 March 2023



## 1. Introduction

All companies rely on business logic, or more generic, process logic to accomplish business goals. Process logic describes the interaction between machines, humans, software (e.g., ERP, MES), resources (e.g., raw materials), and the environment in order to achieve a predefined business goal. Such process logic is executed by a process engine, which enacts and monitors all the rules contained in the processing logic. This process execution can rely on the Internet of Things (IoT) consisting of a network of (smart) machines and sensors. In this case, the process engine and the IoT work together to execute the processes. The engine orchestrates the process activities, using IoT actuators to automate process tasks, while IoT sensors and tags can be used to closely monitor the execution environment and involved resources [1–4]. We call such processes IoT-enhanced BPs.

To understand and improve the processes, the process execution data is stored in an event log, then can later be analyzed using a variety of process mining techniques [5,6]. However, IoT-enhanced BPs have special requirements on the data representation that cannot be fulfilled by the de-facto standard for storing event logs for process mining, eXtensible Event Stream (XES) [7]. When using XES, each observation (i.e., sensor values denoting, for example, temperature, vibration or humidity) by a sensor must be either

assignable to events or traces [8]. However, this may not be the case for observations. For example, a series of temperature readings representing the mean temperature of a timespan (e.g., 2 min), might span the boundaries of multiple events representing multiple short subsequent real-world tasks. But from a process event log perspective, IoT data (i.e., sensor value, sensor type, sensor configuration, trigger parameters) can be assigned to different context levels such as event, trace, or groups of instances from the same or different processes. For example, humidity might affect a single machining task, a whole process instance consisting of machining and quality assurance, as well as a group of related parallel machining tasks on a factory floor. The assignment is related to semantic process properties, as well as the nature of the collected data (static vs. dynamic, collection frequency, relation to the process, etc.). Depending on logging, knowledge about the executed processes and process models, and physical aspects such as the placement of sensors or their orientation, sensors can be directly assigned to individual events or traces or neither. In addition, IoT data is often ad-hoc, highly variable, contains data quality issues, and has varying degrees of semantic annotations [3,9], i.e., references to instances in domain ontologies.

In the absence of a unified, expressive standard for IoT-enriched event logs, various players in industry and academia are developing their own proprietary formats and database schemas. This results in many highly customized and not-interoperable data formats and procedural applications (examples include—as of 2023: Celonis Execution Management System, IBM Process Mining, Fluxicon Disco, Microsoft Process Advisor, Mehrwerk MPM Process Mining) dealing with process mining, i.e., runtime and ex-post analysis of data-streams to check the compliance with the business logic, search for the cause of errors and gain insights about bottlenecks and resource shortages.

In this paper, we present the DataStream XES extension for uniform representation of IoT-enriched event logs. The extension complements plain XES in a way that extensive IoT sensor data can be stored in events or traces, but also, independently of these concepts if the connection is not clear (yet). The foundations of this extension are based in the challenge C3 of the BPM&IoT Manifesto [4], which formulates the requirement of a "connection of analytical processes with IoT". In order to support the creation of analytical software to perform such analytical tasks, both ex-post and at runtime, the following 5 goals for the design of the DataStream extension are defined:

- Provide a well-defined set of named XES attributes to describe individual (sensor) events.
- Utilize well-established XES concepts such as lists to group the named attributes for simplified analysis.
- Establish a set of named XES attributes to store many (sensor) events per process event.
- Describe how to store large quantities of (sensor) events, which might occur between the start/end of a process event or a process instance (i.e., establishing a new XES BPAF lifecycle transition).
- Establish a set of named XES attributes to connect (sensor) events to groups of process tasks.

Subsequently these 5 goals are evaluated in two different real-world scenarios, from the transportation/logistics and manufacturing domains.

The extension adds nested attributes to XES, providing a vocabulary for storing IoT data streams with process logs. Thus, the extension can easily be integrated with existing process execution environments or log aggregation mechanisms, as exemplified in Section 4. The DataStream XES extension is intended to lay a foundation for process mining in IoT environments and to promote re-usability and interoperability.

The structure of the paper is as follows: In Section 2, we describe the theoretical basis for process mining in IoT and the related literature. Section 3 introduces the proposed DataStream XES extension to specify IoT-enriched event logs. In Section 4, we present application scenarios for IoT-enriched event logs in smart manufacturing and public transportation. Section 5 summarizes the results, lists advantages and limitations, and gives an outlook for future research directions.

## 2. Foundations and Related Work

The recent developments and technologies used in the *Industrial Internet of Things* (*IIoT*) [1] demand a more intelligent and interconnected process-based control of IoT devices [2]. To achieve a deeper integration with IoT environments in a process-oriented way, *Business Process Management* (*BPM*) methods can be applied for control and analysis purposes [4]. The benefit is that BPM can make use of the huge variety of IoT sensor data that can be used to improve analysis methods. The remainder of this section is structured as follows (see also Figure 1): Section 2.1 discusses relevant IoT terms, and their recognition in the BPM domain. Section 2.2 introduces related work regarding the integration of BPM and IoT in a more general context. Based on that, Section 2.3 describes how process mining techniques can be applied to IoT environments, including the presentation of typical application scenarios. Section 2.4 then describes related approaches that tackle data analysis problems, and provide datasets, which are related to the challenges and solutions described in this paper.

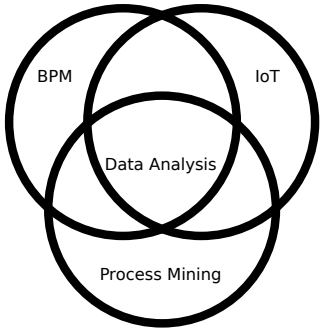


**Figure 1.** Related Work.

### 2.1. IoT

Dorsemaine et al. [10] define IoT as "Group of infrastructures interconnecting connected objects and allowing their management, data mining and the access to the data they generate."

Context in IoT is defined by Dey et al. [11] as: "any information that can be used to characterise the situation of an entity. An entity is a person, place, or object that is considered relevant to the interaction between a user and an application, including the user and applications themselves."

Serpanos [12] defines an IoT event as a time-value set, which contains: `key, value, destination, generation_time, release_time`. This in stark contrast to the BPM world, where sequences of events describe the lifecycle of a task in a process execution, and individual events can contain arbitrary attributes [13,14].

Furthermore, the IoT domain is characterized by a focus on devices topologies, and middleware that facilitates the interaction between various classes of devices in scenarios such as smart homes and smart cities [15]. The BPM domain on the other hand has strong focus on describing interactions between software components, hardware components and the environment, often avoiding layered architectures.

### 2.2. BPM and IoT

Often, existing work on IoT does not cover the integration of BPM concepts. However, various approaches from the BPM community investigating aspects regarding the integration of BPM with IoT exist and are discussed in different, recent surveys [16–18]. In the following, we present related work that deals with the integration of BPM in smart IIoT environments.

Chang et al. [16] present an overview of cloud BPM systems for (mobile) IoT devices. Process (execution) engines (PEs) are used as part of a middleware to control IoT devices at the edge. Baumgraß et al. [19] discuss the integration of complex events from IoT with

business processes in the context of smart logistics. Kammerer et al. [20] focus on this integration of sensor events from industrial production machines and their processing in a BPM context. Schönig et al. [21] deal mainly with data exchange and communication between IoT devices and execution engines, namely the aspect of abstraction and encapsulation of sensor events. Koot et al. [22] propose an architecture for IoT-enabled dynamic planning in the domain of smart logistics with an applications layer that serves the role of a PE executing business logic. In a literature study, Ramos Gutiérrez et al. [23] investigate which approaches, frameworks, and tools are available to integrate business processes and complex events in the logistic domain. Seiger et al. [24,25] present the *PROtEUS* system to execute self-adaptive cyber-physical workflows in IoT base smart home environments. They assume the existence of services to control IoT devices and collect data for event processing. In more recent work [26], they introduce a method for detecting process activity executions. The approach is based on sensor-actuator-activity patterns and activity signatures. In this context, they use a service-oriented architecture based on [2,27] to control a physical smart factory by a PE. *SmartPM* [28], on the other hand, supports adaptive processes based on AI planning in situations in which an adaptation of the current process instance is required. Based on this, Malburg et al. [29] present how control loops can be applied for process adaptation and recovery by using AI planning and semantics. Very similar to that is the approach by Ochoa et al. [30] in which they present an architecture for asset administration shell-based business processes. In this context, capabilities of resources are encapsulated in such a way that they can be executed in a PE. The architecture of the *SitOPT* system for situation-aware adaptive workflows in manufacturing is presented in [31]. It allows recognizing complex failure situations based on rules, leading to the execution of process variations to deal with the detected failures. Traganos [32] discusses an end-to-end printing process, relying on a PE on top of a middleware layer encapsulating the hardware and software. This work has been the basis for the *HORSE* framework [33], which deals mostly with logistics use-cases. Bordel Sánchez et al. [34] propose an 8-layer architecture to cover concerns reaching from low-level machine interactions up to decision-making by domain experts. Notably, they suggest a dedicated *Data Analytics* layer for sensor data processing. Kirikkayis et al. [35] present the *BPMNE4IoT* framework, which deals with the aspect that the current BPMN 2.0 standard has not been intended to fully meet the IoT characteristics and thus, is not well-suited in every IoT application scenario. For this purpose, they present a framework that supports modeling, executing, and monitoring IoT-driven processes. Bocciarelli et al. [36] analyzes IoT frameworks and ontologies that can be utilized to extend the current BPMN representation standard. In addition, they present an automated process for developing digital twins based on actual business processes.

The work presented in these papers complements all of these approaches, by giving them a means to uniformly represent the process and IoT data in a single data structure, that can be either used for long-term storage, or as the basis for data analysis.

### *2.3. Process Mining*

One technique to analyze IoT sensor data together with related event data is process mining. Process mining describes three analysis tasks. The most common is (i) process discovery. Discovery techniques take an event log and produce a process model depicting the process executed in the log [5,6]. The second task is (ii) conformance checking, which is used to verify the conformance of real process instances to a given a-priori model. The last analysis task is (iii) enhancement that uses an event log and the associated process model to identify bottlenecks and to improve the process accordingly [5]. All tasks require an event log as input.

Many proposals have been made in the past for storing event logs. The first was MXML, as a simple XML format for audit and trails in process-aware information systems [37]. XES, the current standard event log model, is also based on XML and widely used in both industrial and academic contexts [7,38]. An XES event log can contain one or more traces. Each trace represents a process instance. A trace, in turn, consists of the

sequence of executed activities, each represented by an event. Furthermore, event logs can store additional attributes, such as resources and data elements [39].

An XES attribute consist of (a) a data type represented by the qualified name of the XML element, (b) a key to denote the type of attribute (unique within its container), and (c) a value (see Listing 1). XES describes six types of attributes: string, date, int, float, boolean and id which have a value, as well as two additional attributes, container and list, which can hold arbitrary child attributes. All attributes can also be nested (even inside non-container and non-list attributes) [7].

**Listing 1.** Sample XES (XML serialization) with Trace, Events and Attributes.

```
<log xes.version="1.0"
xmlns="http://www.xes-standard.org"
xes.creator="cpee.org"
xes.features="nested-attributes">
<extension name="Concept" prefix="concept" uri="http://www.xes-standard.org/concept.xesext
    "/>
<extension name="Lifecycle" prefix="lifecycle" uri="http://www.xes-standard.org/lifecycle.
    xesext"/>
<extension name="Identity" prefix="identifier" uri="http://www.xes-standard.org/identity.
    xesext"/>
<extension name="Time" prefix="time" uri="http://www.xes-standard.org/time.xesext"/>
<global scope="trace">
<string key="concept:name" value="name"/>
</global>
<global scope="event">
<string key="concept:name" value="name"/>
<string key="lifecycle:transition" value="start"/>
<date key="time:timestamp" value="1970-01-01T00:00:00.000+00:00"/>
</global>
<string key="lifecycle:model" value="standard"/>
<string key="creator" value="cpee.org"/>
<string key="library" value="cpee.org"/>
<trace>
<string key="concept:name" value="Process 1"/>
<event>
<string key="concept:name" value="Task 1"/>
<string key="lifecycle:transition" value="start"/>
<date key="time:timestamp" value="1970-01-01T00:00:00.000+00:00"/>
<string key="name" value="Juergen"/>
</event>
<event>
<string key="concept:name" value="Task 2"/>
<string key="lifecycle:transition" value="start"/>
<date key="time:timestamp" value="1970-01-01T00:00:00.000+00:00"/>
<string key="name" value="Juergen"/>
/event>
</trace>
</log>
```

Since the requirements for event logs differ depending on the application and domain, XES can be extended. Standard XES extensions include the concept extension, which specifies a generally understood name for events, traces, or the log. In addition, the lifecycle extension can be used to specify different stages in the lifecycle of events and the time extension standardizes the specification of event timestamps [7]. XES also allows the definition of new data attribute types through the notion of extensions, thereby increasing the flexibility of the model.

Recently, the uptake of new technologies and the gain in maturity of the process mining field have increased the urge to create more powerful event log models. Multiple propositions that relax some assumptions of XES and allow for more flexibility in event data representation have been presented (e.g., [40,41]). Among them, a standard for *Object-Centric Event Logs* (*OCEL*) [41] has been developed to be more suitable for storing event data extracted from relational databases and is widely considered as the main challenger of XES today. OCEL replaces the strict notion of case with the concept of object, which generalizes it by allowing one event to be linked with multiple objects instead of a single case. This removes the necessity to "flatten" the event log during logging or ex-post log extraction based on relational databases, as is the case for XES by picking one case notion from the several potential case notions that often coexist in real-life processes. A second

noticeable difference with XES is the explicit inclusion of the concept of activity in OCEL, which is absent in XES [42].

When representing a process execution into the OCEL format, information about traces, events, or the overall sequence of steps is only available implicitly, by linking together the models of each object. This means that information is potentially lost, and ambiguities regarding the process model might appear. While theoretically, it is possible to define an object type corresponding to an overall case as understood in XES, it is non-standard and therefore not automatically analyzable. In this case, XES is still a better fit. So, despite its noteworthy flexibility and closer conceptualization to the real-world in case of business processes supported by relational databases, the missing case notion in the OCEL format can become a liability.

### *2.4. Data Analysis for BPM and IoT*

In recent publications the temporal and spatial aspects of sensor readings are used for automatically connecting them to process execution [43]. Both, the sensor readings as well as the business processes, are seen as an integral part of application scenarios and their analysis in IoT [2,44,45].

Multi-perspective process mining [46] has evolved to, for example, use tree structured process event logs containing time-series data outside the XES BPAF lifecycle, as presented in [47]. Also, the analysis of this time series data is used as described in [47] for detecting concept drifts during runtime. A survey on outcome-oriented predictive process monitoring presented in [48] compares different techniques.

Banham et al. [49] propose to perform data-aware process discovery with IoT-based attributes. A data petri net is discovered from two real-life event logs, and rules behind some decisions are mined based on IoT-derived attributes. The proposed framework requires abstracting the IoT data to integrate them in an XES event log.

For all of these approaches, datasets have been provided containing a wealth of context data in conjunction with process events. However, these datasets present slightly different granularity levels, slightly different formats, and slightly different semantics.

Wei et al. [50] recently proposed an approach named Amoretto to integrate IoT data into XES logs. They focus on a limited, well-established set of attributes from the BPM domain into process logs: physical object, location, time, identity, environment. They explicitly collect the data by adding collection tasks to existing process models, leading to a multitude of problems: (1) The data from the collection is indistinguishable from normal data flow from non-Amoretto event logs, so specialized analysis will be hard. (2) Their main context is the trace/process instance, if a sensor data collection is relevant for a particular task or set of tasks, it must be explicitly modeled as parallel to the task containing the business logic. This leads to complicated analysis as the tasks containing the business logic also have to have information about physical object, location, and environment to be connectable. (3) Unlike earlier work such as [51], they ignore IoT standardization efforts and classification. Finally, (4) when considering the real-world examples presented in this work with 50+ sensors, it becomes obvious that the resulting process models will violate all rules of good process modeling [52]. In contrast, the approach presented in our work, focuses on how IoT data can be connected to existing logs for unchanged process logs and process models, considering (a) intrinsic (process context) data, (b) extrinsic (non-process context), and (c) separate data, and focusing on connection to existing events, while acknowledging that many IoT events might occur while existing business logic is executed, both at instance and event level.

## 3. An XES-Extension for IoT-Enriched Event Logs

Based on our literature analysis presented in Section 2 we think, that XES is still the best starting point for representation and long-term storage of process data and all connected IoT data. As discussed above, (1) process and IOT data can have a different granularity (e.g., multiple sensor readings from multiple sensors per task), (2) IoT data

might not be explicitly connected to the process, and often has to be connected to the process data ex-post from heterogeneous sources, and (3) the data-sources for sensors are potentially evolving (e.g., additional sensors, new replacement sensors, sensor firmware updates leading to changed data structures).

Holding the extracted, transformed and aggregated data (i.e., after process mining, or after process execution) in a flexible, structured long-term storage format is imperative. This section discusses how XES can be extended to better support this goal.

XES is built around events. Each process activity execution can lead to a set of events in an XES log file, following the life-cycle (see Section 2) of the execution of that activity in a particular instance, i.e., each activity could lead to a *"start"* event, to a *"complete"* event, and to an arbitrary number of events in between, depending on the utilized life-cycle model.

Many XES log files just store one event per executed activity, thus sensor readings could be attached to this event. Other available logs, such as [53], expose a custom fine-grained life-cycle model, that anchors sensor readings to an event with special XES lifecycle:transition.

An execution of the model shown in Figure 2 leads to the XES log described in Listing 1. As mentioned in the XES Standard [7] (p. 5):

> "Log, trace, and event objects contain no information themselves. They only define the structure of the document. All information in an event log is stored in attributes. Attributes describe their parent element (log, trace, etc.). All attributes have a string-based key."

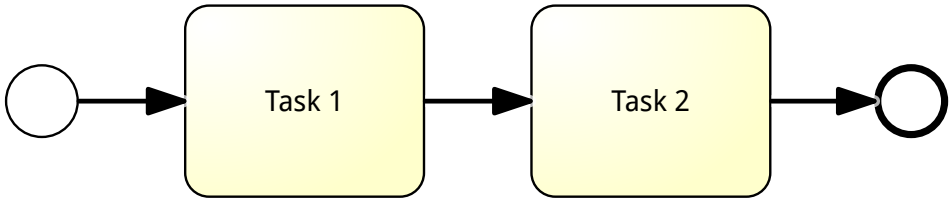


**Figure 2.** Example Process.

However, as we indicated before, XES cannot be directly used to store IoT-enhanced BP logs. Specifically, we distinguish three different cases (see Figure 3) where IoT data might be connected to process activities, according to which part of the process that data may be relevant for:

- **"Single Activity" Context**: A time-series of sensor readings from at least one sensor is connected to a single activity, e.g., when the activity represents the machining of a part, collected sensor data might describe various aspects, such as the throughput of coolant while machining, a discrete series of vibration readings, or a function (continuous data) describing the noise generation (volume). All sensor data can be assigned to a particular activity, being the data relevant between the start and the completion of the activity.
- **"Group of Activities" Context**: A time-series of sensor readings from at least one sensor is connected to a set of activities. This is especially relevant for environmental sensors, which for example span a multitude of production steps. For instance, temperature changes during several process activities might give insights into certain quality properties of a finished product but cannot be clearly attributed to a single step (e.g., significance of the difference between temperatures measured at the start of activity 1, and at the end of activity n).
- **"Trace" Context**: A time-series of sensor readings from at least one sensor is connected to a whole trace. This case is analogous to the "Group of Activities" case. This is for instance necessary when the sensor readings from a period before and/or after individual activities may be relevant for the process analysis. e.g., when enacting a chemical reaction, the characteristics of a warm-up phase might be important for the

outcome but might not be explicitly part of the process model, which only starts with adding ingredients.

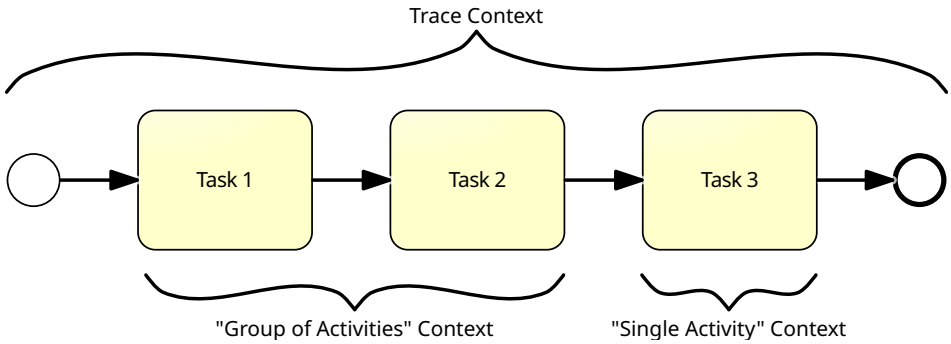


**Figure 3.** Different Contexts in Which IoT Data Can Be Collected.

Please note that being connected to "group of activities" or "traces" can also include that one sensor reading can be relevant for very different processes or process instances or activities in very different process or process instances. We assume that the readings are duplicated for these cases—we focus on single instances and their activities. As each reading can be uniquely identified, complex relationships between instances and processes will still be visible in the data.

In order to realize these three contexts, we extend XES as depicted in Figure 4.


**Figure 4.** XES + DataStream Metamodel Extension.

In the following, we will denote all attributes of our proposed extension with the prefix **stream:**, to increase the clarity of the description. We will furthermore assume that the **stream:** prefix is specified in an XES extension—see https://cpee.org/datastream/datastream.xesext (accessed on 8 February 2023).

The core of the extension is "`attribute<list>:point`", furthermore denoted as stream:point (see previous paragraph). It contains all the attributes that allow us to represent individual sensor values as XES artifacts. It is a list. Values include:

- **id**: uniquely identifies the sensor, e.g., if a gyro-sensor delivers orientation and angular velocity changes separately, the identifiers can be gyro/velocity and gyro/angular_velocity. On the other hand, if the sensor delivers a value pair, the identifier can be gyro.
- **source**: identify the source of a sensor value, e.g., a drilling machine is the source of many different sensor readings at all times. The source attribute allows grouping these values into groups that may belong together and, thus, make sense to be analyzed together. The source is optional.
- **timestamp**: A timestamp when the reading was taken. The timestamp is intended to be in ISO 8601 format, including milliseconds (YYYY-MM-DDTHH:mm:ss.sssZ) or microseconds (YYYY-MM-DDTHH:mm:ss.ssssssZ).
- **value**: The value delivered by the sensor. As sensors can deliver single values (float, int, strings) or complex data (pairs, triplets, deeply structured data, ...), we always assume this is stored as some serialized string representation.
- **meta**: A straightforward extension point, which allows us to specify an additional list of attributes, which might be important for custom data analysis purposes. Meta is optional.

### *3.1. Context, Grouping, and Nesting: Stream:Datacontext, Stream:Datastream, Stream:Point*

The second concept (see Figure 4) is the stream:datastream. It was introduced to group points for the **"Single Activity"** and **"Trace"** contexts. Its only (optional) attribute is name, which can be used to describe the purpose of the grouping.

If a set of stream:datastream is included directly in the level of the trace, all sensor:point attributes are meant to exist in the **"Trace"** context: they cannot be attributed to any event or group of events yet.

If a stream:datacontext exists at the trace level, the stream:datacontext has to group multiple events, and it has to contain at least one stream:datastream. This realizes the **"Group of Activities"** context. Multiple stream:datacontext attributes can exist at trace level, meaning that multiple groups exist.

If a stream:datastream exists at the event level, it has to contain at least one stream:point. Multiple stream:datastream can exist at the event level. While this does not change the meaning of all these points being connected to one event, its purpose might be to further structure the events, e.g., separating two different levels of importance for analysis purposes.

All stream:datacontext attributes might be nested. Nested sensor:datacontext attributes convey different layers of connection granularity. For example, some stream:point attributes might be grouped to a group (a) of 2 tasks, some other stream:point attributes might be connected to a group (b) of 2 different tasks. Then a third set of stream:point attributes might be connected to all tasks in groups (a) and (b), leading to a (c: (a) (b)) nesting, as depicted in Listing 2:

**Listing 2.** Sample XES (XML serialization) stream:datastream Nesting.

```
<trace>
<string key="concept:name" value="Process 1"/>
<list key="stream:datacontext">
<list key="stream:datastream">
<list key="stream:point">
<date key="stream:timestamp" value="2021-11-04T15:22:19.367+01:00"/>
<string key="stream:id" value="humidity"/>
<string key="stream:value" value="62.5"/>
</list>
</list>
[...]
<list key="stream:datacontext">
<list key="stream:datastream">
<list key="stream:point">
<date key="stream:timestamp" value="2021-11-04T15:22:22.369+01:00"/>
<string key="stream:id" value="pressure"/>
<int string key="stream:value" value="19"/>
</list>
[...]
</list>
<event>[...]</event>
<event>[...]</event>
[...]
</list>
<list key="stream:datacontext">
<list key="stream:datastream">
<list key="stream:point">
<date key="stream:timestamp" value="2021-11-04T15:22:28.369+01:00"/>
<string key="stream:id" value="temperature"/>
<int string key="stream:value" value="75.3"/>
</list>
[...]
</list>
<event>[...]</event>
<event>[...]</event>
[...]
</list>
</list>
</trace>
</log>
```

This leaves us with the special case of overlapping cases, where some stream:points are connected to tasks 1 and 2, where some other stream:points are connected to tasks 2 and 3. This case can only (XES being a tree structure) be solved by creating three stream:datastream attributes with some duplicated stream:point elements.

### *3.2. Task Lifecycle Extension: Stream/Data*

For long-running tasks, especially when adding IoT data at runtime, it is beneficial to add the IoT data immediately to the XES log, instead of waiting for the next event to occur.

For example, if IoT sensors deliver data during the execution of a task with a duration of 90 min, all data would occur in the "lifecycle:transition complete" event. If the XES log is processed at runtime, e.g., for runtime drift analysis such as in [47], this would prevent early availability of analysis results.

Thus, the introduction of a new lifecycle transition named **stream/data**, as shown in Listing 3, allows the immediate addition of data to the log. The event might optionally include the context (i.e., id of the task), or not include the context if the event exists at the trace or log level.

**Listing 3.** Sample XES (XML serialization) stream/data.

```
<trace>
<string key="concept:name" value="Process 1"/>
<event>
<string key="lifecycle:transition" value="start"/>
...
<event>
<event>
<string key="lifecycle:transition" value="stream/data"/>
<list key="stream:datastream">
...
</list>
...
<event>
...
<event>
<string key="lifecycle:transition" value="complete"/>
...
<event>
...
</trace>
</log>
```

### *3.3. Convenience and Size: Stream:Multipoint*

The final element introduced in Figure 4 is stream:multipoint. This concept is not necessary from a functional perspective, but allows reducing the size of the log file.

For example, when a set of sensor:point attributes all origin from the same sensor and the same source, and contain the same meta information, this information is duplicated all over and over. A sensor:multipoint (see Listing 4) allows us to group this redundant information for a set of points:

**Listing 4.** Sample XES (XML serialization) stream:multipoint.

```
<trace>
<string key="concept:name" value="Process 1"/>
<event>
<string key="concept:name" value="Task 1"/>
<string key="lifecycle:transition" value="complete"/>
<date key="time:timestamp" value="1970-01-01T00:00:00.000+00:00"/>
<string key="name" value="Juergen"/>
<list key="stream:datastream">
<string key="stream:name" value="Temperature"/>
<list key="stream:multipoint">
<string key="stream:id" value="keyence/mesurement"/>
<string key="stream:source" value="keyence"/>
<list key="stream:point">
<date key="stream:timestamp" value="2021-11-04T15:22:19.367+01:00"/>
<string key="stream:value" value="18"/>
</list>
<list key="stream:point">
<date key="stream:timestamp" value="2021-11-04T15:22:20.369+01:00"/>
<int string key="stream:value" value="19"/>
</list>
</list>
</list>
</event>
</trace>
</log>
```

Alternatively, it can be used to group according to timestamp if a set of sensor readings are taken at discrete points in time (see Listing 5):

**Listing 5.** Sample XES (XML serialization) stream:multipoint.

```
<trace>
<string key="concept:name" value="Process 1"/>
<event>
<string key="concept:name" value="Task 1"/>
<string key="lifecycle:transition" value="complete"/>
<date key="time:timestamp" value="1970-01-01T00:00:00.000+00:00"/>
<string key="name" value="Juergen"/>
<list key="stream:datastream">
<string key="stream:name" value="Temperature"/>
<list key="stream:multipoint">
<date key="stream:timestamp" value="2021-11-04T15:22:19.367+01:00"/>
<list key="stream:point">
<string key="stream:id" value="temperature"/>
<string key="stream:value" value="48.5371"/>
</list>
<list key="stream:point">
<string key="stream:id" value="pressure"/>
<string key="stream:value" value="12:30-1,12:31-2,3,4,5"/>
</list>
</list>
</list>
</event>
</trace>
</log>
```

## 4. Application Scenarios for IoT-Enriched Event Logs in Smart Manufacturing and Public Transportation

In order to evaluate the DataStream XES extension, we present and discuss real-world IoT-enriched event logs from smart factories and the public transportation domain (see Sections 4.1 and 4.2). The presented event logs have been created through cpee.org, which supports the DataStream XES extension to directly write logs. All presented application scenarios show how sensor data can be grouped, nested and embedded with ordinary XES logs as a basis for future data-oriented analysis tasks.

Examples in this section are displayed in the XES YAML (Yet Another Mark-up Language, https://yaml.org (accessed on 8 February 2023)) serialization because it is more compact and readable. YAML has the following properties: it relies on indentation for structure (like python), the data types are omitted, the key and value attributes directly result in key value pairs, e.g., the XML excerpt "`<string key=''stream:id'' value=''temperature''/>`" results in "`- stream:id: temperature`" in YAML.

After describing the application scenarios, we discuss the results derived from using the DataStream XES extension in the scenarios to create enriched event logs and describe use cases for process mining analysis (see Section 4.3).

### *4.1. Monitoring Public Transportation Delays Alongside Weather and Traffic Data in Vienna*

For efficient planning of public transportation services, knowledge about the operation of individual tram lines as well as information about effects which might influence its smooth execution, such as weather or traffic, is needed. The dataset (https://doi.org/10.5281/zenodo.7411234 (accessed on 8 February 2023)) used for this section provides an example for the collection of such data in Vienna, Austria using (1) the API of the "Wiener Linien" (the company providing public transport in Vienna), (2) a weather API, and (3) the TomTom navigation system to monitor traffic. Data from these different sources is then collected by enacting a process in an execution engine and stored in the log using the proposed XES DataStream extension format. Using this information, the correlation between delays of individual tramways with the weather and traffic conditions near the track of the observed line can be investigated. This can then lead to better planning for smart city scenarios concerning traffic control in general or suggestions leading to an improvement of the public transportation system. Collecting data for the before mentioned scenario is done by enacting a process model using the cpee.org process execution engine. The process retrieves the individual data for weather, traffic, or tram line(s) (per endpoint providing the data) in parallel and then waits until the next minute starts to repeat the data collection.

The process model in Figure 5 contains tasks that are responsible for the collection of data, which is signalized by the two curved lines on the top right side of these tasks. Such tasks contain a description of the data probes [43], which define how collected data is added to the log, as shown in Figures 6 and 7. These examples show how different kinds of data (in this case derived from the return value of the service invoked in this task) are extracted. It is possible to extract data in different ways, e.g., split data into multiple single data points (as shown in Figure 6) or create multiple data streams (as shown in Figure 7).

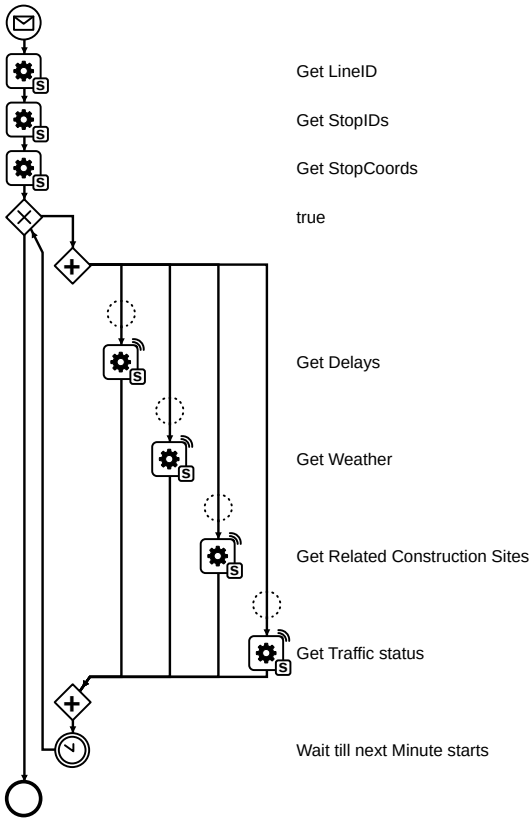


**Figure 5.** Process Model for Collecting Public Transportation Delay Data (Process model used for creating the dataset described in https://doi.org/10.5281/zenodo.7411234) (accessed on 8 February 2023).

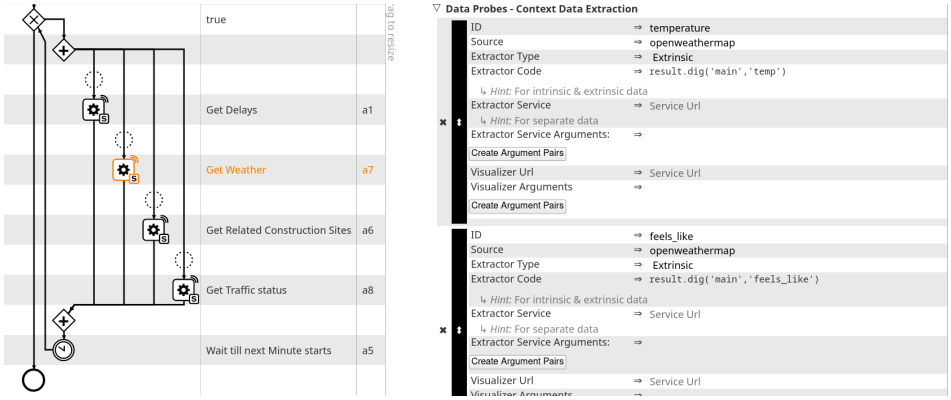


**Figure 6.** Data Probes for the "Get Weather" Task.

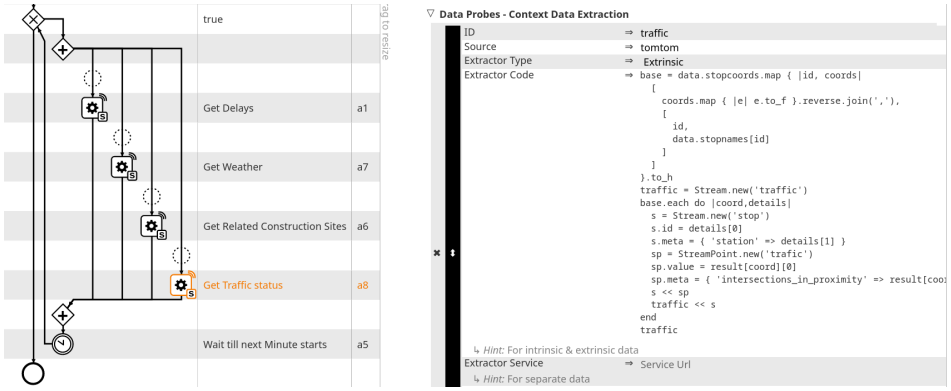


**Figure 7.** Data Probe(s) for the "Get Traffic Status" Task.

Corresponding snippets taken out of the created log are shown in Listings 6 and 7. Listing 6 contains a snippet of the weather data in the log as collected by the task shown in Figure 6. In this excerpt of the log, different stream:point elements (as specified in the data probes of the task) are contained in the stream:datastream element.

So, (nested) stream:datastream elements can be used as a grouping mechanism that conveys semantic cohesiveness. For example, in Listing 8, different stops identified by a name and an id, are children of a "traffic" datastream. Instead of providing a flat list of values, additional semantic depth can be expressed, which can be utilized for visualization or analysis.

**Listing 6.** Public Transportation Weather Data.

```
event:
concept:instance: 8968
concept:name: Get Weather
[...]
stream:datastream:
- stream:point:
stream:id: temperature
stream:value: 4.95
stream:timestamp: 2022-12-06 17:17:01.403889247 +01:00
stream:source: openweathermap
- stream:point:
stream:id: feels_like
stream:value: 0.56
stream:timestamp: 2022-12-06 17:17:01.403972224 +01:00
stream:source: openweathermap
- stream:point:
stream:id: pressure
stream:value: 1017
stream:timestamp: 2022-12-06 17:17:01.404046393 +01:00
stream:source: openweathermap
[...]
```

Listing 7 contains a snippet of the log which is created by the task shown in Figure 7. The "traffic" data stream contains information about the traffic at multiple stops. The structure of the collected data is defined by the data probe defined in the task. In contrast to the other example, multiple data streams (stream:datastream) are created which each include one data point (stream:point) while in the first example just one data stream exists which includes multiple data points.

**Listing 7.** Public Transportation Traffic Data.

```
event:
concept:instance: 8968
concept:name: Get Traffic status
[...]
stream:datastream:
- stream:name: traffic
- stream:source: tomtom
- stream:datastream:
- stream:name: stop
- stream:id: '5'
- stream:source: tomtom
- stream:meta:
station: Börse
- stream:point:
stream:id: trafic
stream:value: 0.6394647901326838
stream:timestamp: 2022-12-06 17:16:40.142570851 +01:00
stream:source: tomtom
stream:meta:
intersections_in_proximity: 71
- stream:datastream:
- stream:name: stop
- stream:id: '46'
- stream:source: tomtom
- stream:meta:
station: Schottentor U
- stream:point:
stream:id: trafic
stream:value: 0.5571382984201959
stream:timestamp: 2022-12-06 17:16:40.142576922 +01:00
stream:source: tomtom
stream:meta:
intersections_in_proximity: 94
[...]
```

### *4.2. Manufacturing and Measuring of a Chess Piece at Pilotfabrik TU Wien*

One goal in the manufacturing domain is the production of small lot sizes while still keeping a high degree of automation. To achieve this, it is possible to use a process execution engine enacting a process model consisting of several standardized subprocesses. These standardized subprocesses are then tailored to the currently produced part by invoking them with the parameters needed for the individual piece/use-case.

In the scenario discussed in Section 4.2 we focus on a process, where several components work together to produce and measure a chess piece. The components are: (1) a milling machine, (2) a robot for handling the part, and (3) a measuring machine (and an independent lift moving the part through the optical measuring machine) conducting an optical measurement of the part diameter. The process uses these components by (1) manufacturing the part in the milling machine, (2) taking the part out of the machine with the robot and putting it on the lift, (3) moving the lift down to measure the produced part optically, and (4) finally taking the part from the lift and putting it on a palette (see Figure 8).

During all of these steps, different data from the involved components is collected and stored in the logs, as proposed in the XES DataStream extension format. This data includes machining data such as the workload of the drive, the axis speed for different axis, and the actual speed and workload of the spindle as well as measuring data from the optical measurement. The dataset (https://doi.org/10.5281/zenodo.7477845 (accessed on 8 February 2023)) which is created adheres to the earlier described XES DataStream extension format provides the basis for analysis tasks such as process mining, prediction of part quality, or detection of broken tools.

As also described in Section 4.1 a process model is enacted by the process execution engine cpee.org. The process model includes tasks having data probes which define how to collect/extract data and attach it to the log (see Figures 6 and 7 for examples). In this scenario, machining data (see Listing 8) as well as measuring data (see Listing 9) are collected.

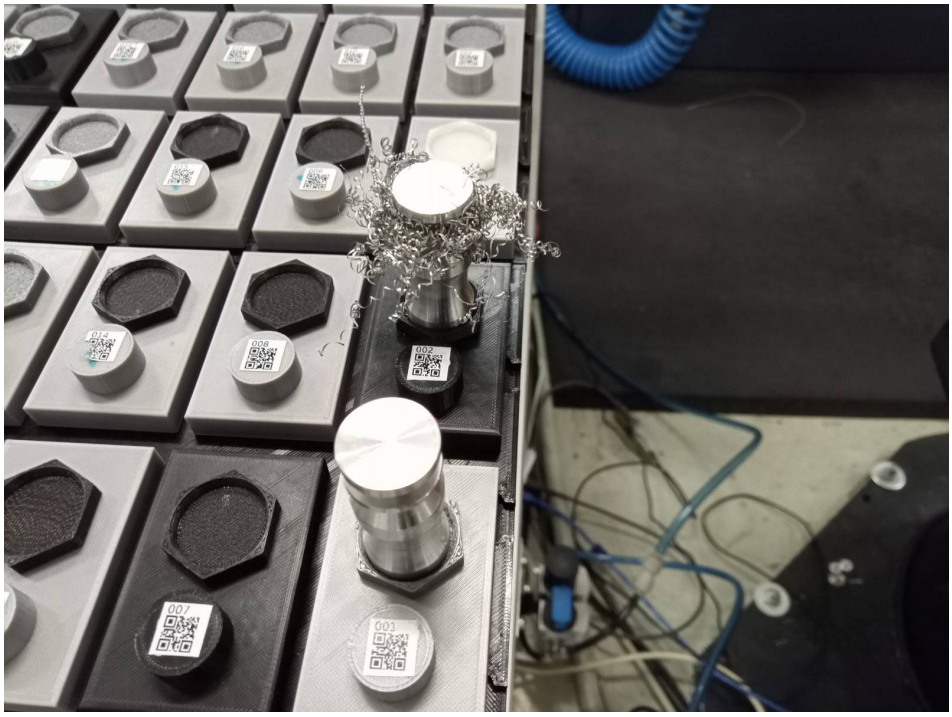

**Figure 8.** One Good and One Bad (With Chip) Chess Piece on the Palette (Figure available at https://doi.org/10.5281/zenodo.7477845 (accessed on 8 February 2023)).

**Listing 8.** Chess Piece Machining Data.

```
event:
concept:instance: 4129
concept:name: Fetch
[...]
stream:datastream:
- stream:name: MaxxTurn45
- stream:source: machine
- stream:point:
stream:id: State/progStatus
stream:value: 3
stream:timestamp: '2022-12-16T18:01:17.000+01:00'
stream:source:
proto: opcua
host: opc.tcp://192.168.10.59:4840
access: ns=2;s=/Channel/State/progStatus
[...]
- stream:point:
stream:id: Axes/Z/aaLeadP
stream:value: 405.5
stream:timestamp: '2022-12-16T18:01:18.000+01:00'
stream:source:
proto: opcua
host: opc.tcp://192.168.10.59:4840
access: ns=2;s=/Channel/MachineAxis/aaLeadP[u1,2]
- stream:point:
stream:id: Axes/X/aaTorque
stream:value: -2.028
stream:timestamp: '2022-12-16T18:01:18.000+01:00'
stream:source:
proto: opcua
host: opc.tcp://192.168.10.59:4840
access: ns=2;s=/Channel/MachineAxis/aaTorque[u1,1]
[...]
```

Both Listings 8 and 9 extract data and create a data stream consisting of different data points. This enables the collection of many individual values as shown in Listing 8 but also data collection in scenarios where a few or only one value is measured over and over again (as in the measuring example shown in Listing 9).

**Listing 9.** Chess Piece Measuring Data.

```
event:
concept:instance: 4153
concept:name: Fetch
[...]
stream:datastream:
- stream:name: keyence
- stream:source: machine
- stream:point:
stream:id: measurement
stream:value: 20.08
stream:timestamp: '2022-12-16T17:51:44.408+01:00'
stream:source:
proto: serial
access: "/dev/ttyUSB0 115200"
stream:meta: {}
- stream:point:
stream:id: measurement
stream:value: 20.09
stream:timestamp: '2022-12-16T17:51:44.423+01:00'
stream:source:
proto: serial
access: "/dev/ttyUSB0 115200"
stream:meta: {}
[...]
```

The stream points shown in Listing 9 represent a time series of measurements of the contour of the chess piece. Without the DataStream extension, these measurements would have been part of the log in a proprietary format as provided by the machine (as part of a PDF file). By representing the measurements in the DataStream format, data extraction and transformation steps can be avoided, and data for different measurement machines becomes readily comparable due to the uniform representation.

### *4.3. Discussion of Results*

For the scenarios described in Sections 4.1 and 4.2 the DataStream XES extension allowed to collect not only the process flow data but also IoT data collected during the process. Such context data contains important information, e.g., when performing a root cause analysis for process outcome properties such as part quality.

For example, in transportation data set (see Section 4.1) the traffic state is explicitly queried as part of the process model, leading to a value between 0 (no traffic flow) and 1 (free traffic flow). In a traditional process log all the sensor readings (from different crossings) in the vicinity would not be part of the data set but might yield crucial information for process improvement (in this case, e.g., changing the timing of traffic lights, or moving the station).

Another, even clearer example can be highlighted in the manufacturing data set (see Section 4.2) the task of producing a rook from the process point of view only results in "success" or "no success". The over twenty sensors that produced several thousand readings during the two-minute duration of machining, cannot easily be included in a traditional XES file, and were traditionally analyzed separately from the process. The XES DataStream extension provides a means to structure and store these readings in a common format.

Linking/integrating the data directly in the event log, can provide a common basis for future analysis tools that can perform a joint analysis of process and IoT data.

Limitations currently exist regarding the usability of the data structure in analysis tools, as there are no implementations to use it. Moreover, due to the large amounts of data in the IoT context, the logs can quickly become very large, which might overwhelm some existing process mining tools.

## 5. Conclusions and Future Work

In this paper, an extension to XES has been presented, allowing the joint representation of process event logs and IoT data related to the environment where these events occur. This enables the development of generic data pipelines, process mining approaches, and visualization tools for IoT event logs.

The extension identifies what is required from the IoT perspective to enable the use of BPM methods for IoT (cf. [4]). As it has been shown, there are special requirements for the data perspective in the IoT context, especially regarding IoT sensor data.

Section 1 defines goals which should be met by the proposed XES extension. By describing real-world application scenarios (see Section 4) and demonstrating how the goals are achieved using the DataStream XES extension proposed in this paper, it is proven that the approach fulfills the set requirements:

- *"Provide a well-defined set of named XES attributes to describe individual (sensor) events."* is achieved by defining the named XES attributes in the DataStream Metamodel shown in Figure 4 and using these attributes in the logs of the real-world application scenarios.
- *"Utilize well-established XES concepts such as lists to group the named attributes for simplified analysis."* is achieved by identifying different granularity levels such as `stream:datastream` and `point` in the DataStream Metamodel shown in Figure 4 and assign the named attributes to their corresponding levels. This leads to easier analysis as for example `stream:points` can share attributes of their parent `stream:datastream` as shown in Listings 7–9. This serves the purpose of data de-duplication for a set of `stream:point` attributes, thus reducing data cleaning effort.
- *"Establish a set of named XES attributes to store many (sensor) events per process event."* is achieved by allowing for multiple `stream:datastreams` and therefore also `stream:points` to be connected to one process event in the DataStream Metamodel shown in Figure 4. This is also shown in Listings 6–9 where multiple `stream:datastreams` and/or `stream:points` are present in one process event. Having a set of common attributes instead of a mix of different attributes with slightly different meaning, reduces data transformation effort, and provides context and meaning to the most basic shared concepts.
- *"Describe how to store large quantities of (sensor) events, which might occur between the start/end of a process event or a process instance (i.e., establishing a new XES BPAF lifecycle transition)."* is achieved by introducing the `stream/data` lifecycle transition. The `stream/data` lifecycle transition allows us to add IoT data to the XES log at any time therefore omitting the need to wait for the next process event to carry the data collected in the process until then. This is described in more depth in Section 3.
- *"Establish a set of named XES attributes to connect (sensor) events to groups of process tasks."* is achieved by introducing `stream/datacontext` (see Section `stream/datacontext`). This allows us to connect (sensor) data to process tasks, although it is not in the same granularity, for example when averaged sensor readings (e.g., temperature, humidity, ...) span multiple process tasks.

In the future, a complete event log of a factory shall be parsed, analysed and visualized based on the proposed extension format to support process refinement and root cause analysis, in order to promote process re-engineering with the goal of resilient [54] shop floor processes. Furthermore, tools for the visualization of DataStream-based event logs are to be developed and existing process mining approaches are to be adapted in such a way that they can also process the DataStream extension. Another point for future research is the incorporation of IoT data in event logs following other standards, such as OCEL, and more specifically the possibility of transposing the DataStream extension to these standards. Finally, we want to examine how semantic annotations can be further integrated and used during process analysis.

**Author Contributions:** Conceptualization, J.M., J.G., L.M., M.E. and Y.B.; methodology, J.M., J.G., L.M., M.E. and Y.B.; software, J.M. and J.G.; validation, J.M. and M.E.; investigation, J.M., J.G., L.M., M.E. and Y.B.; resources, S.R.-M., E.S.A. and R.B.; data curation, J.M. and M.E.; writing—original draft preparation, J.M., J.G. and L.M.; writing—review and editing, J.M., J.G., L.M., M.E., Y.B., J.-V.B., S.R.-M., E.S.A. and R.B.; visualization, J.M.; supervision, S.R.-M., E.S.A. and R.B.; project administration, S.R.-M., E.S.A. and R.B.; funding acquisition, S.R.-M., E.S.A. and R.B. All authors have read and agreed to the published version of the manuscript.

**Funding:** This work has been partly funded by the Austrian Research Promotion Agency (FFG) via the "Austrian Competence Center for Digital Production" (CDP) under the contract number 881843. This work has been supported by the Pilot Factory Industry 4.0, Seestadtstrasse 27, Vienna, Austria. This work is also partly funded by the Federal Ministry for Economic Affairs and Climate Action under grant No. 01MD22002C *EASY* and by the Ministry for Science and Health of Rhineland-Palatinate as part of the research college *AI-CPPS*.

**Data Availability Statement:** The datasets utilized in this work are openly available in the Zenodo open access repository at https://doi.org/10.5281/zenodo.7411234 and https://doi.org/10.5281/zenodo.7477845 (accessed on 8 February 2023).

**Conflicts of Interest:** The authors declare no conflict of interest.

## References

1. Sisinni, E.; Saifullah, A.; Han, S.; Jennehag, U.; Gidlund, M. Industrial Internet of Things: Challenges, Opportunities, and Directions. *IEEE Trans. Ind. Inf.* **2018**, *14*, 4724–4734. [CrossRef]
2. Seiger, R.; Malburg, L.; Weber, B.; Bergmann, R. Integrating process management and event processing in smart factories: A systems architecture and use cases. *J. Manuf. Syst.* **2022**, *63*, 575–592. [CrossRef]
3. Elsaleh, T.; Bermudez-Edo, M.; Enshaeifar, S.; Acton, S.T.; Rezvani, R.; Barnaghi, P. IoT-Stream: A Lightweight Ontology for Internet of Things Data Streams. In Proceedings of the 2019 Global IoT Summit (GIoTS), Aarhus, Denmark, 17–21 June 2019; IEEE: Piscataway, NJ, USA, 2019; pp. 1–6.
4. Janisch, C.; Koschmider, A.; Mecella, M.; Weber, B.; Burattin, A.; Di Ciccio, C.; Fortino, G.; Gal, A.; Kannengiesser, U.; Leotta, F.; et al. The Internet of Things Meets Business Process Management: A Manifesto. *IEEE Syst. Man Cybern. Mag.* **2020**, *6*, 34–44. [CrossRef]
5. van der Aalst, W.M.P. *Process Mining: Discovery, Conformance and Enhancement of Business Processes*; Springer: Berlin/Heidelberg, Germany, 2011.
6. Berti, A.; van Zelst, S.J.; van der Aalst, W.M.P. Process Mining for Python (PM4Py): Bridging the Gap Between Process- and Data Science. *arXiv* **2019**, arXiv:1905.06169.
7. Günther, C.W.; Verbeek, E. XEX Standard Definition—Version 2.0. 2014. Available online: https://www.xes-standard.org/_media/xes/xesstandarddefinition-2.0.pdf (accessed on 8 February 2023).
8. Bertrand, Y.; De Weerdt, J.; Serral Asensio, E. Assessing the suitability of traditional event log standards for IoT-enhanced event logs. In *LNBIP Post-Proceedings*; Springer: Cham, Switzerland, 2022.
9. Bermudez-Edo, M.; Elsaleh, T.; Barnaghi, P.; Taylor, K. IoT-Lite: A Lightweight Semantic Model for the Internet of Things. In Proceedings of the UIC-ATC-ScalCom-CBDCom-IoP-SmartWorld 2016, Toulouse, France, 18–21 July 2016; IEEE: Piscataway, NJ, USA, 2016; pp. 90–97.
10. Dorsemaine, B.; Gaulier, J.P.; Wary, J.P.; Kheir, N.; Urien, P. Internet of Things: A Definition & Taxonomy. In Proceedings of the NGMAST, Cambridge, UK, 9–11 September 2015; IEEE: Piscataway, NJ, USA, 2015; pp. 72–77.
11. Dey, A.K. Understanding and Using Context. *Pers. Ubiquitous Comput.* **2001**, *5*, 4–7. [CrossRef]
12. Serpanos, D.; Wolf, M. *Internet-of-Things (IoT) Systems*; Springer International Publishing: Berlin/Heidelberg, Germany, 2018.
13. Koschmider, A.; Janssen, D.; Mannhardt, F. Framework for Process Discovery from Sensor Data. In Proceedings of the EMISA, Kiel, Germany, 14–15 May 2020; pp. 32–38.
14. Tax, N.; Sidorova, N.; Haakma, R.; van der Aalst, W.M.P. Event Abstraction for Process Mining using Supervised Learning Techniques. *arXiv* **2018**, arXiv:1606.07283.
15. Aheleroff, S.; Xu, X.; Lu, Y.; Aristizabal, M.; Pablo Velásquez, J.; Joa, B.; Valencia, Y. IoT-enabled smart appliances under industry 4.0: A case study. *Adv. Eng. Inform.* **2020**, *43*, 101043. [CrossRef]
16. Chang, C.; Srirama, S.N.; Buyya, R. Mobile Cloud Business Process Management System for the Internet of Things: A Survey. *ACM Comput. Surv.* **2017**, *49*, 70:1–70:42. [CrossRef]
17. Bazan, P.; Estevez, E. Industry 4.0 and business process management: State of the art and new challenges. *Bus. Process Manag. J.* **2022**, *28*, 62–80. [CrossRef]
18. Fattouch, N.; Ben Lahmar, I.; Boukadi, K. IoT-aware Business Process: Comprehensive survey, discussion and challenges. In Proceedings of the 2020 IEEE 29th International Conference on Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE), Bayonne, France, 10–13 September 2020; IEEE: Piscataway, NJ, USA, 2020; pp. 100–105.
19. Baumgraß, A.; Botezatu, M.; Di Ciccio, C.; Dijkman, R.; Grefen, P.W.P.J.; Hewelt, M.; Mendling, J.; Meyer, A.; Pourmirza, S.; Völzer, H. Towards a methodology for the engineering of event-driven process applications. In Proceedings of the 13th International Workshops, Innsbruck, Austria, 31 August–3 September 2015; Springer: Berlin/Heidelberg, Germany, 2016; pp. 501–514.
20. Kammerer, K.; Pryss, R.; Hoppenstedt, B.; Sommer, K.; Reichert, M. Process-driven and flow-based processing of industrial sensor data. *Sensors* **2020**, *20*, 5245. [CrossRef]
21. Schönig, S.; Ackermann, L.; Jablonski, S.; Ermer, A. IoT meets BPM: A bidirectional communication architecture for IoT-aware process execution. *Softw. Syst. Model.* **2020**, *19*, 1443–1459. [CrossRef]

22. Koot, M.; Iacob, M.E.; Mes, M.R.K. A Reference Architecture for IoT-Enabled Dynamic Planning in Smart Logistics. In *Advanced Information Systems Engineering*; Lecture Notes in Computer Science; Springer Nature: Cham, Switzerland, 2021; Volume 12751, pp. 551–565.
23. Ramos Gutiérrez, B.; Reina Quintero, A.M.; Parody, L.; Gómez López, M.T. When business processes meet complex events in logistics: A systematic mapping study. *Comput. Ind.* **2023**, *144*, 103788. [CrossRef]
24. Seiger, R.; Huber, S.; Schlegel, T. Toward an execution system for self-healing workflows in cyber-physical systems. *Softw. Syst. Model.* **2018**, *17*, 551–572. [CrossRef]
25. Seiger, R.; Huber, S.; Heisig, P.; Aßmann, U. Toward a framework for self-adaptive workflows in cyber-physical systems. *Softw. Syst. Model.* **2019**, *18*, 1117–1134. [CrossRef]
26. Seiger, R.; Franceschetti, M.; Weber, B. An Interactive Method for Detection of Process Activity Executions from IoT Data. *Future Internet* **2023**, *15*, 77. [CrossRef]
27. Malburg, L.; Klein, P.; Bergmann, R. Semantic Web Services for AI-Research with Physical Factory Simulation Models in Industry 4.0. In Proceedings of the International Conference on Innovative Intelligent Industrial Production and Logistics 1st IN4PL, Budapest, Hungary, 2–4 November 2020; SciTePress: Setúbal, Portugal, 2020; pp. 32–43.
28. Marrella, A.; Mecella, M.; Sardiña, S. Intelligent Process Adaptation in the SmartPM System. *ACM Trans. Intell. Syst. Technol.* **2017**, *8*, 25:1–25:43. [CrossRef]
29. Malburg, L.; Hoffmann, M.; Bergmann, R. Applying MAPE-K control loops for adaptive workflow management in smart factories. *J. Intell. Inf. Syst.* **2023**, 1–29. [CrossRef]
30. Ochoa, W.; Larrinaga, F.; Pérez, A. Architecture for managing AAS-based business processes. *Procedia Comput. Sci.* **2023**, *217*, 217–226. [CrossRef]
31. Wieland, M.; Schwarz, H.; Breitenbucher, U.; Leymann, F. Towards situation-aware adaptive workflows: SitOPT—A general purpose situation-aware workflow management system. In Proceedings of the 2015 IEEE International Conference on Pervasive Computing and Communication Workshops (PerCom Workshops), St. Louis, MO, USA, 23–27 March 2015; IEEE: Piscataway, NJ, USA, 2015; pp. 32–37.
32. Traganos, K.; Vanderfeesten, I.; Grefen, P.W.P.J.; Erasmus, J.; Gerrits, T.; Verhofstad, W. End-to-End Production Process Orchestration for Smart Printing Factories: An Application in Industry. In Proceedings of the 2020 IEEE 24th International Enterprise Distributed Object Computing Conference (EDOC) 24th EDOC, Eindhoven, The Netherlands, 5–8 October 2020; IEEE: Piscataway, NJ, USA, 2020; pp. 155–164.
33. Traganos, K.; Grefen, P.W.P.J.; Vanderfeesten, I.; Erasmus, J.; Boultadakis, G.; Bouklis, P. The HORSE framework: A reference architecture for cyber-physical systems in hybrid smart manufacturing. *J. Manuf. Syst.* **2021**, *61*, 461–494. [CrossRef]
34. Bordel Sánchez, B.; Alcarria, R.; Rivera, D.S.d.; Sánchez-Picot, Á. Enhancing Process Control in Industry 4.0 Scenarios using Cyber-Physical Systems. *J. Wirel. Mob. Netw. Ubiquitous Comput. Dependable Appl.* **2016**, *7*, 41–64.
35. Kirikkayis, Y.; Gallik, F.; Winter, M.; Reichert, M. BPMNE4IoT: A Framework for Modeling, Executing and Monitoring IoT-Driven Processes. *Future Internet* **2023**, *15*, 90. [CrossRef]
36. Bocciarelli, P.; D'Ambrogio, A.; Panetti, T. A Model Based Framework for IoT-Aware Business Process Management. *Future Internet* **2023**, *15*, 50. [CrossRef]
37. van Dongen, B.F.; van der Aalst, W.M.P. A Meta Model for Process Mining Data. *EMOI-INTEROP* **2005**, *160*, 30.
38. van Dongen, B.F.; Shabani, S. Relational XES: Data Management for Process Mining. In Proceedings of the CAiSE Forum, Stockholm, Sweden, 8–15 June 2015.
39. van der Aalst, W.M.P. Data Science in Action. In *Process Mining*; Springer: Berlin/Heidelberg, Germany, 2018; pp. 3–23.
40. Popova, V.; Fahland, D.; Dumas, M. Artifact Lifecycle Discovery. *arXiv* **2013**, arXiv:1303.2554.
41. Ghahfarokhi, A.F.; Park, G.; Berti, A.; van der Aalst, W.M.P. OCEL: A Standard for Object-Centric Event Logs. In Proceedings of the 25th ADBIS, Tartu, Estonia, 24–26 August 2021; Springer: Berlin/Heidelberg, Germany, 2021. Communications in Computer and Information Science. Volume 1450, pp. 169–175.
42. Rebmann, A.; Rehse, J.R.; van der Aa, H. Uncovering Object-Centric Data in Classical Event Logs for the Automated Transformation from XES to OCEL. In *Business Process Management, Proceedings of the 20th International Conference, BPM 2022, Münster, Germany, 11–16 September 2022*; Lecture Notes in Computer Science; Di Ciccio, C., Dijkman, R., del Río Ortega, A., Rinderle-Ma, S., Eds.; Springer International Publishing: Cham, Switzerland, 2022; pp. 379–396. [CrossRef]
43. Ehrendorfer, M.; Mangler, J.; Rinderle-Ma, S. Assessing the impact of context data on process outcomes during runtime. In Proceedings of the International Conference on Service-Oriented Computing, Virtual, 22–25 November 2021; Springer: Cham, Switzerland, 2021; pp. 3–18.
44. Seiger, R.; Zerbato, F.; Burattin, A.; Garcia-Banuelos, L.; Weber, B. Towards IoT-driven Process Event Log Generation for Conformance Checking in Smart Factories. In Proceedings of the 2020 IEEE 24th International Enterprise Distributed Object Computing Workshop (EDOCW), Eindhoven, The Netherlands, 5–8 October 2020; IEEE: Piscataway, NJ, USA, 2020; pp. 20–26.
45. Bertrand, Y.; de Weerdt, J.; Serral, E. A Bridging Model for Process Mining and IoT. In *Process Mining Workshops*; Springer: Berlin/Heidelberg, Germany, 2022; Volume 433, pp. 98–110.
46. Mannhardt, F. Multi-perspective Process Mining. Ph.D. Thesis, Technische Universiteit Eindhoven, Eindhoven, The Netherlands, 2018.
47. Stertz, F.; Rinderle-Ma, S.; Mangler, J. Analyzing process concept drifts based on sensor event streams during runtime. In Proceedings of the 18th BPM, Seville, Spain, 13–18 September 2020; Springer: Berlin/Heidelberg, Germany, 2020; pp. 202–219.

48. Teinemaa, I.; Dumas, M.; Rosa, M.L.; Maggi, F.M. Outcome-Oriented Predictive Process Monitoring: Review and Benchmark. *ACM Trans. Knowl. Discov. Data* **2019**, *13*, 1–57. [CrossRef]
49. Banham, A.; Leemans, S.J.J.; Wynn, M.T.; Andrews, R. xPM: A Framework for Process Mining with Exogenous Data. In Proceedings of the Process Mining Workshops—ICPM 2021 International Workshops, Eindhoven, The Netherlands, 31 October–4 November 2021; Revised Selected Papers; Lecture Notes in Business Information Processing; Munoz-Gama, J., Lu, X., Eds.; Springer: Berlin/Heidelberg, Germany, 2021; Volume 433, pp. 85–97. [CrossRef]
50. Wei, J.; Ouyang, C.; ter Hofstede, A.H.; Moreira, C. AMORETTO: A Method for Deriving IoT-enriched Event Logs. *arXiv* **2022**, arXiv:abs/2212.02071.
51. Grüger, J.; Malburg, L.; Mangler, J.; Bertrand, Y.; Rinderle-Ma, S.; Bergmann, R.; Asensio, E.S. SensorStream: An XES Extension for Enriching Event Logs with IoT-Sensor Data. *arXiv* **2022**, arXiv:2206.11392.
52. Reijers, H.A.; Mendling, J. A Study Into the Factors That Influence the Understandability of Business Process Models. *IEEE Trans. Syst. Man Cybern.—Part A Syst. Hum.* **2011**, *41*, 449–462. [CrossRef]
53. Stertz, F.; Mangler, J.; Rinderle-Ma, S. Temporal Conformance Checking at Runtime based on Time-infused Process Models. *arXiv* **2020**, arXiv:2008.07262.
54. Aheleroff, S.; Huang, H.; Xu, X.; Zhong, R.Y. Toward sustainability and resilience with Industry 4.0 and Industry 5.0. *Front. Manuf. Technol.* **2022**, *2*, 951643. [CrossRef]