BROOM: Toolbox for IoT-Enhanced Process Mining*

Christian Imenkamp[1,*], Yannis Bertrand[2], Joscha Grüger[3,4], Lukas Malburg[3,4], Marco Franceschetti[5], Matthias Ehrendorfer[6] and Agnes Koschmider[1]

[1]*University of Bayreuth, Universitätsstraße 30, 95447 Bayreuth, Germany*
[2]*Ghent University, Tweekerkenstraat 2, 9000 Ghent, Belgium*
[3]*Trier University, Universitätsring 15, 54296 Trier, Germany*
[4]*German Research Center for Artificial Intelligence (DFKI), Behringstraße 21, 54296 Trier, Germany*
[5]*University of St. Gallen, Institute of Computer Science, 9000 St. Gallen, Switzerland*
[6]*Technical University of Munich, TUM School of Computation, Information and Technology, 85748 Garching, Germany*

Abstract
Internet of Things (IoT)-enhanced process mining faces interoperability challenges due to diverse event log formats. Addressing this gap, we introduce the CORE metamodel, a unified representation based on the Object-Centric Event Log (OCEL) standard. We also present *BROOM* (tool**B**ox fo**R** I**O**T-enhanced pr**O**cess **M**ining), a web-based toolbox that seamlessly transforms between log formats. *BROOM's* node-based interface simplifies attribute mapping, significantly improving accessibility and efficiency in IoT-enhanced process analysis.

Keywords
Process Mining, Internet of Things, OCEL

1. Introduction

The increasing integration of IoT technologies into business processes has led to the emergence of IoT-enriched event logs, as interconnected devices generate granular, time-stamped events. These logs extend traditional process execution data with contextual information from IoT devices such as sensors and actuators, allowing more sophisticated analysis and real-time process insights [1]. However, the lack of standardization among existing metamodels for IoT-enriched event logs, often based on XES (eXtensible Event Stream), has resulted in significant interoperability challenges. Event logs stored in different formats are typically not mutually compatible, limiting their reusability and hindering cross-tool import and exchange.

To address this challenge, we propose a common metamodel for IoT-enriched event logs named the *CORE* metamodel [2] that is based on the OCEL standard [3]. The *CORE* metamodel integrates the representational features of several already existing log formats into a unified model, thereby enabling interoperability across corresponding tools and datasets. The metamodel aims to serve as a foundational abstraction layer, allowing the seamless transformation of event data between heterogeneous log formats. *BROOM* complements existing data analytics and process mining platforms (e.g., RapidMiner). Its primary advantage lies in the high customizability and the ability to produce outputs adhering to the *CORE* metamodel. *BROOM* introduces three key innovations to address the limitations of the existing tools: (1) The native support for OCEL and therefore the OCEL ecosystem (i.e., using discovery algorithms for IoT-enhanced process data). (2) An extensible pipeline model for transforming,

Proceedings of the Best BPM Dissertation Award, Doctoral Consortium, and Demonstrations & Resources Forum co-located with 23nd International Conference on Business Process Management (BPM 2025), Seville, Spain, August 31st to 5th September, 2025
*Corresponding author.
†These authors contributed equally.
christian.imenkamp@uni-bayreuth.de (C. Imenkamp); yannis.bertrand@ugent.be (Y. Bertrand); grueger@uni-trier.de (J. Grüger); malburgl@uni-trier.de (L. Malburg); marco.franceschetti@unisg.ch (M. Franceschetti); matthias.ehrendorfer@tum.de (M. Ehrendorfer); agnes.koschmider@uni-bayreuth.de (A. Koschmider)
0009-0007-4295-1268 (C. Imenkamp); 0000-0002-6407-7221 (Y. Bertrand); 0000-0001-7538-1248 (J. Grüger); 0000-0002-6866-0799 (L. Malburg); 0000-0001-7030-282X (M. Franceschetti); 0000-0002-7739-9123 (M. Ehrendorfer); 0000-0001-8206-7636 (A. Koschmider)

manipulating and parsing of IoT-enhanced process data. (3) Semantic mapping of IoT-data to process-aware events.

Given the considerable number of IoT-enriched event logs and software tools currently available for different IoT-enriched event log formats, a toolbox for IoT-enhanced process mining appears to be promising; e.g., to convert existing logs into the common metamodel format. To this end, we have developed *BROOM*, a Python-based toolbox for IoT-enhanced process mining, which enables the bidirectional exchange of event logs between XES-based formats and the proposed OCEL-based *CORE* metamodel. A central challenge of a common metamodel format lies in the mapping of attributes across semantically and structurally divergent metamodels.

To make *BROOM* more accessible, we implemented a web-based user interface that guides users through the available functions, such as the transformation procedure. The interface provides a node-based editor that allows the user to construct a transformation pipeline by connecting predefined nodes.

2. CORE Metamodel for IoT-Enhanced Event Logs

The *CORE* metamodel [2] offers a unified framework for the representation of IoT-enhanced event logs by synthesizing core attributes and features from several pre-existing event log formats. It addresses the harmonization of heterogeneous data generated by IoT devices and process-aware information systems in the context of business processes (BPs). The metamodel has been designed to fulfill seven well-defined requirements that emerged from an extensive comparative evaluation of prior approaches and use cases across diverse application domains.

The *CORE* metamodel distinguishes between two central constructs: events and objects, adhering to the object-centric modeling paradigm established by OCEL 2.0. Events are further categorized into *IoT-Events*, representing physical observations from sensors, and *process events*, denoting transitions in the execution lifecycle of activities. Objects are likewise classified into *data source objects* (e.g., sensors, information systems), *business objects* (e.g., batches, tanks, customer orders), and *general objects* (e.g., subprocesses, resources). Key structural features include:

- Support for **different levels of data granularity**, enabling representation of low-level sensor observations and high-level business events.
- A **flexible case notion**, allowing events to be associated with multiple entities rather than a single process instance.
- Explicit representation of **semantic annotations** and metadata, improving the interpretability and reuse of logs.
- **Full traceability across abstraction layers**, including derivation links between raw sensor data, aggregated IoT-Events, and process events.
- **Compatibility with the OCEL 2.0** standard to take advantage of existing tools and ensure widespread adoption.

Compared to previously proposed models, such as DataStream [4], NICE [5], and CAIRO [6], the *CORE* metamodel offers several significant advantages:

- **Unification and Interoperability:** *CORE* merges and harmonizes features across models, facilitating interoperability and reducing fragmentation in IoT-enhanced process mining.
- **Model Expressiveness:** Through its object- and event-type taxonomies, *CORE* achieves comprehensive coverage of heterogeneous data sources and their interactions.
- **Tool Support and Extensibility:** Implementation within the OCEL 2.0 standard ensures compatibility with a growing ecosystem of related tools, while allowing future extensions.
- **Data Integration and Transformation:** The metamodel explicitly supports derivation chains, enabling transparent transformation and abstraction of sensor data into higher-level process information.

- **Cross-domain Applicability:** The generality and flexibility of the metamodel make it suitable for a wide range of domains, including manufacturing, healthcare, and smart environments.

In summary, the *CORE* metamodel provides a modular and extensible foundation for integrated representation, analysis, and exchange of IoT-enhanced event data, aligning with current standards and emerging requirements in process mining research.

3. Implementation

BROOM has been developed as a web application using Angular[1] for the frontend and Flask[2] as REST functionality. The implementation of the *CORE* metamodel is developed in Python@3.12. The source code of *BROOM* can be found at https://github.com/chimenkamp/IOT-PM-Suite for the frontend and https://github.com/chimenkamp/IOT-PM-Suite-Backend for the backend, and a deployed version can be accessed using the following link https://broom-iot-toolbox.onrender.com/.

3.1. Functionality

The goal of the implementation is to smoothly parse an arbitrary log into the *CORE* metamodel format. For this, *BROOM* relies upon a node-based user interface to intuitively create a parsing pipeline (see Figure 1). In general, the pipeline depends on the use case and dataset, but will subsequently perform the following steps:

- **Reading the data**: The source dataset can be of type *CSV*, *XML*, *YAML*, or *JSON*. Internally, all different data types are mapped into a data frame structure. This functionality is provided by the *Read File* node.
- **Data Processing**: *BROOM* offers multiple nodes to process the data (e.g., *Data Filter*, *Data Mapper*, *Column Selector*, *Attribute Selector*). For example, the *Column Selector* takes the *Raw Data* output of the *Read File* Node as an input and covert it into a *Series*. The *Data Filter* and *Data Mapper* (Input = Series, Output = Series) can apply filters to the series. The Attribute Selector can be used to select attributes from these series.
- ***CORE* Model Conversion**: Selected attributes are used to construct classes provided by the *CORE* metamodel. *BROOM* offers corresponding nodes: the *IoT-Event* node, which accepts *ID*, *Type*, *Timestamp*, and *Metadata* as inputs, all connectable to the *Attribute Selector* node; and the *Process Event*, which additionally requires an *Activity Label*. Objects are similarly created with *ID*, *Type*, *Class*, and *Metadata*, supported by utility nodes to generate unique IDs or select object classes.
- **Construction of the *CORE* Model**: Finally, outputs from *Process Event*, *IoT-Event*, and *Relationships* nodes are combined into a dataset in *CORE* format. The data can be displayed or exported in OCEL format via Output and Export nodes. Leveraging the object-centric process mining (OCPM) ecosystem also enables discovering object-centric process models by integrating IoT-Events, Process Events, and Event-To-Event Relationships directly in the browser.

3.2. Architecture of BROOM

BROOM's architecture is web-based and is therefore structured into two layers (i.e., Frontend and Backend), as shown in Figure 2. Users interact mainly with the site through the **Frontend**. The page layout is structured into two components (i.e., Sidebar and Editor). (1) The Sidebar incorporates the main functionalities and the collection of usable nodes. The user can drag and drop nodes from the sidebar into the editor component. Additionally, the sidebar also has the functions to Save/Load a

[1]https://angular.dev/
[2]https://flask.palletsprojects.com/en/stable/

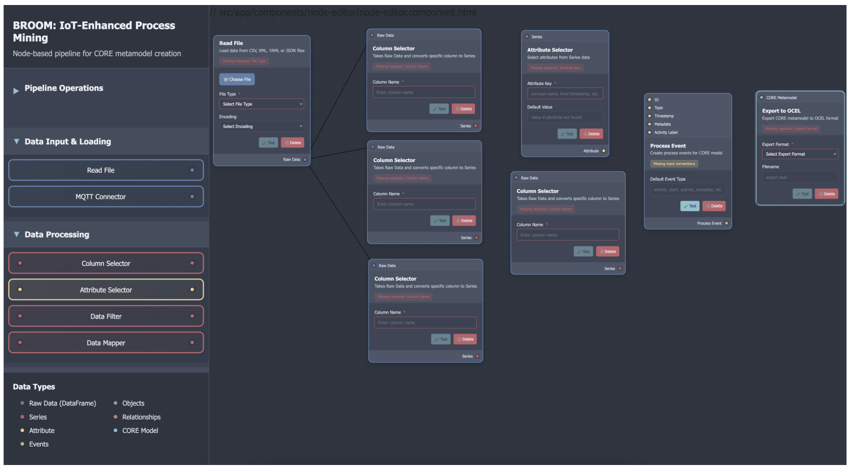

Figure 1: A partial view of *BROOM* including an exemplary pipeline

pipeline, upload a dataset to the server, or clear the editor. Mappings can be saved to a JSON file and later uploaded to restore the pipeline. Lastly, the sidebar contains a small legend containing the available port types (i.e., the connection between nodes). (2) A node will be displayed in the editor when a user drags and drops it to a certain position. The node consists of the outer and inner layer. The outer layer contains the input (left) and output (right) ports. Two ports can be connected by dragging from one output port to the input port of another node. Only ports of the same color can be connected, and one output port can be connected to multiple input ports. The inner layer of the node contains different UI elements to manipulate the behavior (e.g., the column identifier of the *Column Selector* node). In addition to the general element of the UI, the frontend also performs rudimentary validation tasks before the pipeline is forwarded to the backend for processing (i.e., starting with data and ending with the *CORE* metamodel, no dangling nodes, all nodes have the required inputs and outputs).

After validation, the pipeline is propagated to the backend via a REST service. The first component of the **Backend** is the REST-API, providing routes for posting datasets, pipelines, or pipeline results. Posting a pipeline definition triggers the *Pipeline Executor*, which first flattens the pipeline typologically into sequences of executable steps. Due to potentially unbounded node connections, pipelines may flatten into multiple, possibly overlapping sequences. During sequence processing, the *Pipeline Executor* loads the corresponding *Node Implementations*—classes describing node behavior—from the store. For example, the *Column Selector* node receives a DataFrame and a column identifier, returning the specific column as a series. The results are stored associated with node IDs. A node is triggered only when all its predecessors' results are available. Finally, once all predecessors of the *CORE* metamodel node are executed successfully, the required data is passed to the implementation layer, translating the *CORE* standard directly into OCEL.

4. Availability and Maturity

BROOM is a stable and production-ready tool, designed for use by researchers, practitioners, and educators. Its functionality has been validated against the DataStream extension for XES [4], and the CAIRO metamodel [6]. These tests confirm the suitability and comparability of the tool with existing metamodels. Additionally, the authors of these metamodels have evaluated *BROOM* and endorsed its usability and efficiency.

The tool is fully open source under the MIT license, making it suitable for both academic and commercial applications. It is available on GitHub at https://github.com/chimenkamp/IOT-PM-Suite for the frontend and https://github.com/chimenkamp/IOT-PM-Suite-Backend for the backend, with a live instance accessible at https://broom-iot-toolbox.onrender.com/. A "Getting Started" guide and example projects are provided to ensure smooth onboarding. Additionally, a video showcasing a simple

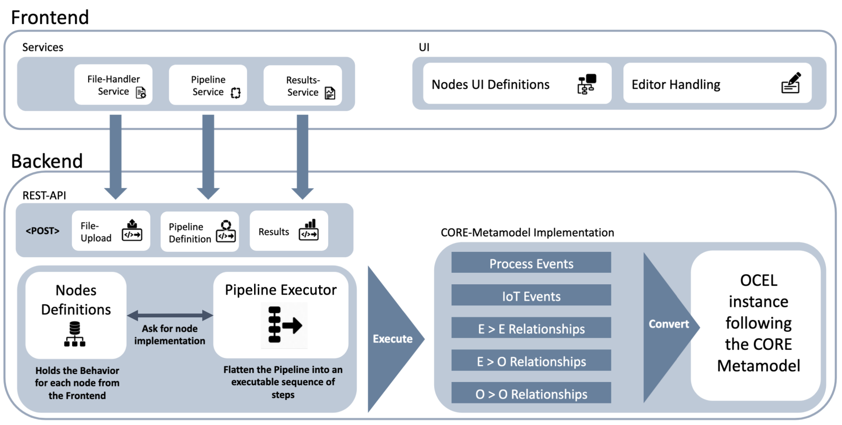

Figure 2: BROOM Architecture

pipeline can be found here https://github.com/chimenkamp/IOT-PM-Suite/tree/main/docs/

BROOM's mature state is reflected in its stable performance, complete feature set, and modular, extensible design. It supports a wide range of data formats and streaming inputs, and produces outputs compatible with object-centric process mining frameworks. Additionally, for use cases that cannot be modeled through the frontend, users can directly use *BROOM's* Python implementation to select the desired functionality.

Acknowledgments

This work received funding by the Deutsche Forschungsgemeinschaft (DFG), grant 496119880 and was supported by the Internet of Processes and Things (IoPT) community: https://zenodo.org/communities/iopt/about.

Declaration on Generative AI

During the preparation of this work, the author(s) used ChatGPT, Writefull, and LanguageTool in order to: Paraphrase and reword, improve writing style, and Grammar and spelling check. After using this tool/service, the author(s) reviewed and edited the content as needed and take(s) full responsibility for the publication's content.

References

[1] A. Koschmider, et al., Process Mining for Unstructured Data: Challenges and Research Directions, in: Modellierung 2024, Gesellschaft für Informatik e.V., Bonn, 2024, pp. 119–136.
[2] Y. Bertrand, C. Imenkamp, L. Malburg, M. Ehrendorfer, M. Franceschetti, J. Grüger, F. Leotta, J. Mangler, R. Seiger, A. Koschmider, S. Rinderle-Ma, B. Weber, E. Serral, An object-centric core metamodel for IoT-enhanced event logs, arXiv:2506.21300 (2025).
[3] 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, arXiv:2403.01975 (2024).
[4] J. Mangler, J. Grüger, L. Malburg, M. Ehrendorfer, Y. Bertrand, J.-V. Benzin, S. Rinderle-Ma, E. Serral Asensio, R. Bergmann, DataStream XES Extension: Embedding IoT Sensor Data into Extensible Event Stream Logs, Future Internet 15 (2023).
[5] Y. Bertrand, S. Veneruso, F. Leotta, M. Mecella, E. Serral, NICE: the Native IoT-centric event log model for process mining, in: ICPM, Springer, 2023, pp. 32–44.
[6] M. Franceschetti, R. Seiger, B. Weber, An event-centric metamodel for IoT-driven process monitoring and conformance checking, in: ICPM, Springer, 2023, pp. 131–143.