

Sebastian Böhm
Daniel Lübke

ZEUS 2024

16th ZEUS Workshop, ZEUS 2024,
Ulm, Germany, 29 February–1 March 2024
Proceedings

Volume Editors

Sebastian Böhm
University of Bamberg, Distributed Systems Group
An der Weberei 5, DE-96049 Bamberg

Daniel Lübke
Digital Solution Architecture GmbH

*Copyright ©2024 for the individual papers by the papers' authors.
Copyright ©2024 for the volume as a collection by its editors.
This volume and its papers are published under the Creative Commons License Attribution
4.0 International (CC BY 4.0).*

Preface

In February 2024, we had the pleasure to organize the 16th edition of the ZEUS Workshop planned in Ulm, Germany. This time, the workshop was held on-site again, giving us the chance to meet and discuss up-to-date research in person. We would like to thank all reviewers a lot for their work and ongoing support.

The workshop was generously sponsored by Camunda Services GmbH, Digital Solution Architecture GmbH, and envite consulting GmbH.

Ulm, February 2024

Sebastian Böhm
Daniel Lübke

Organization

Steering Committee

Oliver Kopp	Kopp Solutions
Johannes Manner	University of Bamberg
Robin Lichtenthäler	University of Bamberg
Stephan Haarmann	Camunda GmbH
Daniel Lübke	Digital Solution Architecture GmbH

Local Organizer

Lisa Arnold	Ulm University, Germany
Marius Breitmayer	Ulm University, Germany

Web Chair

Robin Lichtenthäler	University of Bamberg
Sebastian Böhm	University of Bamberg

Program Committee Chair

Sebastian Böhm	University of Bamberg
----------------	-----------------------

Program Committee

Saimir Bala	Vienna University of Economics and Business
Marius Breitmayer	University of Ulm
Achim D. Bruckner	University of Exeter
Jonas Cremerius	Hasso Plattner Institute, University of Potsdam
Stephan Fahrenkrog-Peterson	Humboldt-Universität Berlin
Manuel Fritz	University of Stuttgart
Georg Grossmann	University of South Australia
Lukas Harzenetter	University of Stuttgart
Thomas Heinze	German Aerospace Center
Pascal Hirmer	University of Stuttgart
Christoph Hochreiner	Compass Verlag
André van Hoorn	University of Hamburg
Martin Kabierski	Humboldt-Universität zu Berlin
Simone König	Mercedes-Benz AG, TU Munich
Jan Ladleif	Hasso Plattner Institute, University of Potsdam
Jörg Lenhard	SAP SE
Robin Lichtenthäler	University of Bamberg
Daniel Lübke	Digital Solution Architecture GmbH
Matteo Nardelli	University of Rome Tor Vergata
Adrian Rebmann	University of Mannheim
Fabiana Rossi	University of Rome Tor Vergata
Stefan Schulte	Vienna University of Technology
Jan Sürmeli	FZI Forschungszentrum Informatik, Karlsruhe
Maximilian Völker	Hasso Plattner Institut
Tom Lichtenstein	Hasso Plattner Institut
Stefan Winzinger	University of Bamberg

Sponsoring Institutions

Camunda Services GmbH
Digital Solution Architecture GmbH
envite consulting GmbH

Contents

Towards Process Mining on Kafka Event Streams <i>Maxim Vidgof</i>	1
Towards the Usage of Object-Aware Process Variants in Multiple Autonomous Organisations <i>Philipp Hehnle and Manfred Reichert</i>	9
Simulating Event Logs from Object Lifecycle Processes <i>Marius Breitmayer, Lisa Arnold, and Manfred Reichert</i>	14
Clounaq - Cloud-native architectural quality <i>Robin Lichtenthäler</i>	22
How good are you? An empirical classification performance comparison of large language models with traditional open set recognition classifiers <i>Alexander Grote, Anuja Hariharan, Michael Knierim and Christof Weinhardt</i>	27
A BPMN Profile for Test Case Execution Visualization <i>Daniel Lübke</i>	33
Towards Robustness of IoT devices in BPMNE4IoT <i>Pascal Schiessle, Yusuf Kirikkayis and Manfred Reichert</i>	41
Predictive Process Monitoring: An Implementation and Comparison of Student Performance Prediction <i>Lisa Arnold, Marius Breitmayer and Manfred Reichert</i>	47
A Literature Review on Reproducibility Studies in Computer Science <i>Tobias Hummel and Johannes Manner</i>	54
User-agent as a Cyber Intrusion Artifact: Detection of APT Activity using minimal Anomalies on the User-agent String Traffic <i>Badr-Eddine Bouhlal, Tim Sonnekalb, Bernd Gruner and Clemens-Alexander Brust</i>	63
Author Index	73

Towards Process Mining on Kafka Event Streams

Maxim Vidgof¹

¹Vienna University of Economics and Business (WU Wien), Welthandelsplatz 1, 1020 Vienna, Austria

Abstract

Process mining is a family of techniques to analyze business processes based on the event logs produced during their execution. While most process mining techniques rely on readily available event logs, extracting them poses challenges and limits their availability and usability. As event-driven architecture is gaining momentum, event streaming platforms like Apache Kafka can solve this problem. This paper proposes an architecture that allows using Kafka both as message broker for the running process as well as data store directly enabling process mining, including streaming process mining on the same dataset.

Keywords

Process mining, Event stream, Apache Kafka

1. Introduction

Process mining can help organizations to monitor processes, find bottlenecks and improvement potential. However, as information systems involved in the business processes are often backed by relational databases without a notion of event, extracting the data in a format suitable for process mining is challenging. With the emergence of event-driven architecture as new state-of-the-art for building loosely coupled information systems, opportunities arise to use event streams both for communication between the components of such systems and for process mining simultaneously.

In this paper, a data architecture for event streams is proposed. This architecture allows to structure event data in such way that it can be used both by automated services executing the process as well as by process mining tools, including online process monitoring. The rest of the paper is structured as follows: Section 2 provides the necessary background, Section 3 presents the proposed architecture, Section 4 describes related work and Section 5 concludes the paper.

2. Background

2.1. Process mining


Process mining is a family of techniques to analyze business processes based on event logs produced during their execution [1]. These techniques can further be divided into *discovery* – extracting process models from event logs, *conformance* – comparing an event log with the process model and detecting deviations, and *enhancement* – extending or improving an existing

ZEUS'24, February 29–March 1, 2024, Ulm, Germany

✉ maxim.vidgof@wu.ac.at (M. Vidgof)

🌐 <https://nm.wu.ac.at/nm/vidgof> (M. Vidgof)

🆔 0000-0003-2394-2247 (M. Vidgof)

 © 2024 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

S. Böhm and D. Lübke (Eds.): 16th ZEUS Workshop, ZEUS 2024, Ulm, Germany, 29 February–1 March 2024, published at <http://ceur-ws.org>

process model using information from an event log [2]. Event logs are collections of timestamped event records of execution of work items or other process-relevant events, recorded by BPMSs and enterprise systems such as CRM and ERP. Event logs are assumed to contain data related to a single process, and each event in the log must refer to a single process instance or case as well as to an activity. Events can also contain additional information such as resource or cost.

While process mining techniques rely on event logs, extracting them from the information systems supporting the process is not trivial. If a process is orchestrated by a BPMS, event data can be easily extracted in the right format [3]. In other cases, however, the only source of data are relational databases, whose primary task is to store the output of process activities rather than events themselves. Moreover, the unique case identifier that is required may not be stored across all systems, and in some cases business processes cannot have a single case identifier, which led to emergence of object-centric process mining [4].

2.2. Streaming Process Mining

Streaming process mining refers to a set of techniques and tools dealing with processing streaming event data [5]. As opposed to other process mining techniques that analyze static event logs, streaming process mining captures and analyzes events as they happen, in near-real time. Typical use cases include conformance checking to immediately detect violations and counteract timely, as well as process discovery. A major complication in streaming scenario lies in the limited amount of computational resources, especially memory, making impossible to employ traditional process mining techniques relying on complete event logs. Instead, different techniques like sliding window, problem reduction, offline (pre-)computation as well as hybrid approaches are applied, leading to efficient processing at the cost of data completeness.

2.3. Event Streaming and Apache Kafka

Event streaming is the practice of capturing data in real-time from different sources in form of streams of events, storing these streams durably for later retrieval, processing and reacting to these streams in real-time and retrospectively, as well as routing these events to specified destinations [6]. Event streaming allows to loosely couple different systems by using a pub/sub model. Apache Kafka¹ is an event streaming platform allowing to *publish* and *subscribe* to streams of events, *store* them durably and *process* them as they occur or retrospectively.

An *event* in Kafka has a *key*, *value*, *timestamp* and optional metadata. The events are organized and stored in *topics*. A topic can have multiple producers and multiple consumers. In contrast to traditional messaging systems, the events are not deleted after consumption. This allows multiple consumers to process the events at their own pace, as well as read the entire topic from the beginning. For each topic, the retention period after which the events are deleted can be set. Importantly, this retention period can also be set to infinity, allowing for permanent storage of events. To allow for scalability, topics are split into *partitions*, which are scattered across multiple brokers. Events with the same key are always written to the same partitions, and Kafka guarantees the order preservation for the events in the same partition.

¹<https://kafka.apache.org>

3. Proposed Architecture

This section presents the architecture for a Kafka-based system supporting process mining. Section 3.1 gives an overview of the system architecture. Section 3.2 describes the topic architecture, i.e. how events are split into topics. Section 3.3 describes the proposed serialization format of events in Kafka. Section 3.4 discusses the role of a BPMS in the proposed architecture, and Section 3.5 explains how process mining can be performed.

3.1. Overview

The proposed system architecture is shown in Figure 1. The central element is the *Kafka* event streaming platform that connects the other components. Importantly, Kafka serves as a communication backbone for the *Services* responsible for executing the tasks in the process. Depending on the environment where such system is deployed, there might already be a *BPMS* present, in which case it also should communicate with Kafka. Essentially, Kafka serves as the single source of truth for all traditional *process mining* components and for *streaming process mining*, which might have different data requirements.

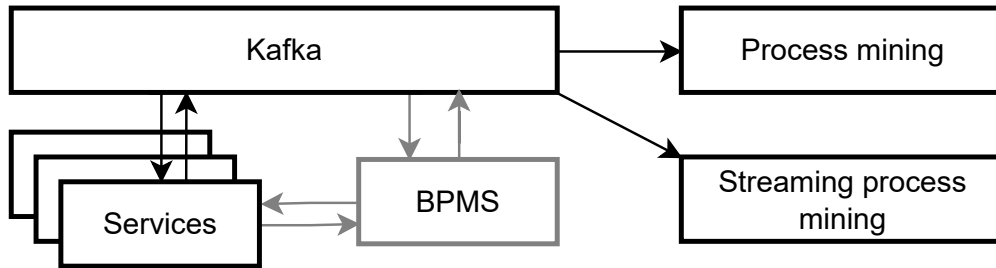


Figure 1: Proposed architecture.

3.2. Topic Architecture

In Kafka, event streams are stored in topics. The number and semantics of the topics is not predefined and should be selected according to the domain specifics. This consideration is crucial since Kafka only guarantees order preservation for events within one partition of the same topic. Thus, the events that need strict order, e.g. events within one case, should be stored in one partition. This is achieved by assigning the same *key* to all these events, e.g. a case ID. In this section, four strategies are proposed.

Case ID as topic. In this case, for each case that has started, a new topic has to be created in the broker. All further events referencing this case ID should be sent to this topic. Consumers should regularly poll the broker for new topics and subscribe to new topics that appear. Each case topic should have only one partition to ensure the event order within the case. Such architecture would, however, suffer from a number of other issues. First, the existing system supporting the process will have to be re-designed to accommodate for the new topics. Also, having such number of topics will introduce unnecessary overhead, especially as the number of

old inactive topics grows. Finally, while such architecture is very much oriented on traditional event logs with single case ID, object-centric logs are not supported.

Activity as topic. This architecture most closely resembles known microservice best practices. Each microservice (or other system) responsible for a task writes the corresponding events in its topic. The events are later picked up by the services relying on them. Case IDs can be used as event keys, allowing for simple partitioning. For object-centric processes, the object ID of some object involved in the activity can be used. A benefit of such strategy is the minimality of change necessary in the system if it already followed event-driven architecture. In the ideal case, the system can remain untouched (except the schema update, see Section 3.3) and additional consumers can be seamlessly added. Events having the same key (case or object ID) will benefit from consistent partitioning and total order within partitions. A downside of such system is, however, the partial view that every topic would have on the execution. The streams will have to be aggregated (using Kafka Streams API or processing outside of Kafka), and event correlation will be crucial. However, it might be solved with the event keys and timestamps provided by Kafka producer or broker.

Process as topic. Another approach is to store all events related to one business process in one topic. Case IDs will be used as keys for the traditional processes. The keys for the processes with no case ID are subject to discussion, however, it is reasonable to use the ID of some object and perform event correlation outside Kafka. The activity label must be included in the event value, and the consuming services will not only have to monitor the right topic but also make sure to only process events with the correct activity label. A clear benefit is that this setting is the closest to the desired output event log. Each topic will simply contain all executions of a process and can almost directly be used in process mining. A downside is the additional overhead on the (micro-)services that will have to process all events and filter them based on the activity label.

Single topic. A variation of the previous strategy, which can be especially useful in scenarios where borders of business processes are unclear. In this case, all events are put into the same topic. The event key for partitioning has to be defined using domain knowledge. A benefit of such strategy is the simplicity of implementation, as it only minimally uses Kafka as a universal message broker. However, in this setup the (micro-)services will suffer from even more filtering overhead. In addition, such single topic can quickly become a data swamp, especially if infinite retention is selected.

3.3. Serialization Format

Kafka itself is schema-less and stores the events as raw bytes, however, it is good practice to use common data exchange formats like JSON², protobuf³ or Avro⁴. In process mining, on the other hand, XES⁵ and OCEL [7] are accepted standards. XES is used for exchanging event logs having a case ID, and uses XML as serialization. However, recently JXES [8] was proposed to support serialization of XES event logs in JSON. OCEL is used for event logs not having a single

²<https://www.json.org/>

³<https://protobuf.dev/>

⁴<https://avro.apache.org/>

⁵<https://xes-standard.org/>

case ID, it offers JSON serialization⁶ included in the standard (along with the XML and SQLite variants).

This paper proposes using JXES and OCEL JSON as the event formats. It must be noted that OCEL objects cannot be stored in the event streams but rather will have to be reconstructed from the events that created or referenced used them at the event log construction stage.

The payload, or the value of the events should contain all the necessary event attributes. For the traditional logs, it is recommended to store at least `concept : name`, `case ID` attribute and `time : timestamp` of the producing application. While some of the attributes can be inferred from Kafka events depending on the chosen topic architecture, it improves the readability and unifies the approach regardless of the topic architecture. Note that the case ID on the event level has no standard attribute in XES, thus a custom one, e.g. `caseid` has to be used. Additional attributes should be stored in the same way as in XES event logs. For the object-centric processes, in addition, the referenced objects with their IDs and relevant attributes must be included in the event value. While not all of the stored attributes are necessary for the services processing the events, e.g. case ID, such services should be configured to simply copy these attributes to their output events.

3.4. Role of a BPMS

The proposed architecture can be used both in conjunction with a BPMS as well as without it. It can be distinguished between the following deployment scenarios.

BPMS-centric. In these scenarios, BPMS plays a central role in orchestrating a process. The events have a single case identifier, and external services are heterogeneous, requiring not only event-driven communication but also other methods, such as REST, Long polling and Java API. In these scenarios, a BPMS indeed serves as a single source of truth for process mining, and it would be challenging to meaningfully extract the process data from other sources. However, the events can be constantly extracted from the BPMS into Kafka to enable streaming process mining with minimal system changes.

BPMS-aided. If a BPMS is used to orchestrate a fully-automated process with event-driven services, this theoretically means both the BPMS and Kafka can be used as the primary event store. However, it may be still beneficial to use the proposed event serialization format and topic architecture to off-load data extraction tasks to Kafka. In addition, such scenarios can profit from streaming process mining if the proposed architecture is used.

No-BPMS. For other scenarios, where no BPMS is present, the proposed architecture becomes crucial for enabling process mining. If the process is composed of loosely coupled (micro-)services and systems, having a unified source of data is the only alternative to the complicated and error-prone process of extracting and correlating events (with possibly incomplete information) from heterogeneous internal storages of the respective systems.

3.5. Process Mining

Infinite retention allows to use Kafka both as message broker for the automated services as well as storage for historic data. The proposed architecture directly enables process mining on

⁶<https://www.ocel-standard.org/specification/formats/json/>

event streams. The last component that is needed for this is a Kafka consumer that connects to the respective topic(s) and extracts the events. If *activity as topic* architecture is chosen, this component also has to order the events using the timestamps provided by Kafka. The extraction of events is straightforward due to the standardized serialization format in the event streams. It must be noted that object-centric event logs might require additional event correlation efforts. The extracted event logs can be easily output as XES or OCEL logs or directly used by a (streaming) process mining tool.

4. Related Work

Apache Kafka has been used in projects related to various topics such as process mining, digital twins, IoT, and software development. However, these projects merely use Kafka as a middleware to exchange data between components. To the best of the author's knowledge, no paper shows explicit considerations about the format of the data and the strategy of using Kafka topics.

[9] uses Kafka as middleware for communication between components of a Digital Twins Cloud Platform. In [10], Kafka is a crucial part of a distributed architecture for real-time monitoring of Roll on/ Roll off (RoRo) terminals. This architecture combines Apache Kafka, Apache Spark⁷ and ProM⁸ platforms to perform process mining on the events recorded by the terminals. Kafka is used to transport the event logs in format of streams of events from their source – RoRo terminals – to Apache Spark cluster responsible for the online processing of the events. While it is mentioned that the data source are XES event logs and that the ProM framework is used for process mining, the exact specifics of the data exchange format and schema are not provided.

[11] proposes a framework that applies event streaming to environmental data. A major difficulty in this domain is the incompatibility of data producers and data consumers due to different data formats. The framework solves this problem by using Kafka Connect APIs to receive data from heterogeneous sources and perform lightweight transformations to bring the data into unified format and prepare them for further processing. In [12], Kafka is used to collect events from a CI/CD pipeline of an application. These events are then transformed and output in a format suitable for process mining. [13] presents an IoT-enriched event log for process mining research in smart factories. It uses Kafka to collect additional data from the machines in the smart factory. This data together with the domain ontology is then used to enrich the log produced by the Workflow Management System. It enabled correcting the timestamps and reconstructing unrecorded events.

Camunda process orchestration platform has already developed own Kafka connectors⁹ and provides examples of using them for process automation [14]. More broadly, [15] shows how a workflow engine can be used with event streaming platform. However, also here no recommendation on data architecture is provided.

⁷<https://spark.apache.org/>

⁸<https://promtools.org/>

⁹<https://docs.camunda.io/docs/components/connectors/out-of-the-box-connectors/kafka/>

5. Conclusion

Process mining can help organizations gain insights into their processes and find improvement potentials. Data extraction, however, poses significant difficulties as not all systems supporting business processes store the data in a format suitable for process mining. With the advent of event-driven architecture, where event streams are used to communicate between loosely coupled components, these difficulties can be solved by using data architecture suitable for process mining.

In this paper, a system architecture is proposed that uses Kafka event streams both as a platform for communication between systems responsible for the execution of a business process (e.g. BPMS, ERP systems and automated (micro-)services) and as a data storage for online and offline process mining. To this end, this paper proposes data architecture for Kafka topics, event serialization format, and a process mining component connecting the system to state-of-the-art process mining tools and techniques. In addition, it discusses the possible role of a BPMS in such setting.

Future work involves implementing the proposed architecture as well as evaluating its usefulness in simulated and real-life business processes.

References

- [1] M. Dumas, M. L. Rosa, J. Mendling, H. A. Reijers, *Fundamentals of Business Process Management*, Second Edition, Springer, 2018.
- [2] W. M. van der Aalst, *Process Mining: Data Science in Action*, Second Edition, Springer, 2016.
- [3] A. Berti, W. M. P. van der Aalst, D. Zang, M. Lang, An open-source integration of process mining features into the camunda workflow engine: Data extraction and challenges, in: C. D. Ciccio, B. Depaire, J. D. Weerdt, C. D. Francescomarino, J. Munoz-Gama (Eds.), *Proceedings of the ICPM Doctoral Consortium and Tool Demonstration Track 2020 co-located with the 2nd International Conference on Process Mining (ICPM 2020)*, Padua, Italy, October 4-9, 2020, volume 2703 of *CEUR Workshop Proceedings*, CEUR-WS.org, 2020, pp. 23–26. URL: <https://ceur-ws.org/Vol-2703/paperTD2.pdf>.
- [4] W. M. P. van der Aalst, Object-centric process mining: Dealing with divergence and convergence in event data, in: P. C. Ölveczky, G. Salaün (Eds.), *Software Engineering and Formal Methods - 17th International Conference, SEFM 2019, Oslo, Norway, September 18-20, 2019, Proceedings*, volume 11724 of *Lecture Notes in Computer Science*, Springer, 2019, pp. 3–25. URL: https://doi.org/10.1007/978-3-030-30446-1_1. doi:10.1007/978-3-030-30446-1_1.
- [5] A. Burattin, Streaming process mining, in: W. M. P. van der Aalst, J. Carmona (Eds.), *Process Mining Handbook*, volume 448 of *Lecture Notes in Business Information Processing*, Springer, 2022, pp. 349–372. URL: https://doi.org/10.1007/978-3-031-08848-3_11. doi:10.1007/978-3-031-08848-3_11.
- [6] Kafka 3.6 documentation, <https://kafka.apache.org/documentation/>, 2023. Accessed: 14.01.2024.

- [7] A. F. Ghahfarokhi, G. Park, A. Berti, W. M. P. van der Aalst, OCEL: A standard for object-centric event logs, in: L. Bellatreche, M. Dumas, P. Karras, R. Matulevicius, A. Awad, M. Weidlich, M. Ivanovic, O. Hartig (Eds.), *New Trends in Database and Information Systems - ADBIS 2021 Short Papers, Doctoral Consortium and Workshops: DOING, SIMPDA, MADEISD, MegaData, CAoNS, Tartu, Estonia, August 24-26, 2021, Proceedings*, volume 1450 of *Communications in Computer and Information Science*, Springer, 2021, pp. 169–175. URL: https://doi.org/10.1007/978-3-030-85082-1_16. doi:10.1007/978-3-030-85082-1_16.
- [8] M. B. S. Narayana, H. Khalifa, W. M. P. van der Aalst, JXES: JSON support for the XES event log standard, *CoRR abs/2009.06363* (2020). URL: <https://arxiv.org/abs/2009.06363>. arXiv:2009.06363.
- [9] G. I. Radchenko, A. B. A. Alaasam, A. Tchernykh, Micro-workflows: Kafka and kepler fusion to support digital twins of industrial processes, in: A. Sill, J. Spillner (Eds.), *2018 IEEE/ACM International Conference on Utility and Cloud Computing Companion, UCC Companion 2018, Zurich, Switzerland, December 17-20, 2018, IEEE, 2018*, pp. 83–88. URL: <https://doi.org/10.1109/UCC-Companion.2018.00039>. doi:10.1109/UCC-COMPANION.2018.00039.
- [10] M. A. Mhand, A. Boulmakoul, H. Badir, Scalable and distributed architecture based on apache spark streaming and prom6 for processing roro terminals logs, in: *Proceedings of the New Challenges in Data Sciences: Acts of the Second Conference of the Moroccan Classification Society*, 2019, pp. 1–4.
- [11] A. K. Akanbi, Estemd: A distributed processing framework for environmental monitoring based on apache kafka streaming engine, *CoRR abs/2104.01082* (2021). URL: <https://arxiv.org/abs/2104.01082>. arXiv:2104.01082.
- [12] A. F. Nogueira, M. Z. Rela, Monitoring a CI/CD workflow using process mining, *SN Comput. Sci.* 2 (2021) 448. URL: <https://doi.org/10.1007/s42979-021-00830-2>. doi:10.1007/s42979-021-00830-2.
- [13] L. Malburg, J. Grüger, R. Bergmann, An iot-enriched event log for process mining in smart factories, *CoRR abs/2209.02702* (2022). URL: <https://doi.org/10.48550/arXiv.2209.02702>. doi:10.48550/ARXIV.2209.02702. arXiv:2209.02702.
- [14] N. Deehan, Orchestration with camunda and kafka confluent cloud, <https://camunda.com/blog/2023/11/orchestration-camunda-kafka/>, 2023. Accessed: 14.01.2024.
- [15] B. Rücker, Process automation and apache kafka, <https://page.camunda.com/wp-process-automation-and-apache-kafka>, 2022. Accessed: 14.01.2024.

Towards the Usage of Object-Aware Process Variants in Multiple Autonomous Organisations

Philipp Hehnle^{1,*}, Manfred Reichert^{1,*}

¹*Institute of Databases and Information Systems, Ulm University, Germany*

Abstract

Managing similar software products and thereby reducing costs has been subject to research in the field of Software Product Line Engineering (SPLE). In our previous work, we applied the concepts of SPLE to activity-centric processes. Although research was conducted to manage variants of object-aware processes, there are still open challenges. In this paper, we address the challenge of managing object-aware process variants in autonomous organisations, which run their information systems separately.

Keywords

Business Process Management, Process Configuration, Process Variability, Software Reuse

1. Introduction

As a result of the right to self-administration, various German municipalities use slightly different digitised variants of the same business processes [1]. Existing approaches for managing process variants focus on the control flow rather than on the implementation level, i.e. in a process model, fragments may be added, moved, or removed [2] and gateways may be restricted (e.g. an OR gateway may be restricted to an AND gateway) [3]. To reduce costs when developing similar software products, the discipline of Software Product Line (SPL) Engineering emerged [4]. An SPL comprises a set of common core software artefacts from which individual software products may be derived, i.e. be configured by selecting and combining the software artefacts [5][6]. In our previous work [1], we applied the concepts of SPL Engineering to activity-centric process management to benefit from the advantages (e.g. reduced costs) in that a process-aware information systems may be derived from a set of software artefacts including a process model by selecting different implementations for the activities of that process model. Thereby, it is possible to derive various process-aware information systems with varying activity implementations.

Frequently, in activity-centric process management, data objects are under-specified. Besides, activity-centric process management lacks flexibility as users must not carry out activities in varying sequences. In contrast, data-centric process management [7] is a paradigm specifying unstructured or semi-structured processes in which data and processes are coupled tightly and the progress of a process is driven by data. The object-aware process management approach [8]


ZEUS'24: 16th Central European Workshop on Services and their Composition, February 29 - March 1, 2024, Ulm, Germany

*Corresponding author.

✉ philipp.hehnle@uni-ulm.de (P. Hehnle); manfred.reichert@uni-ulm.de (M. Reichert)

ORCID 0009-0006-0420-7000 (P. Hehnle); 0000-0003-2536-4153 (M. Reichert)

 © 2022 Copyright for this paper by its authors.

 CEUR Workshop Proceedings (CEUR-WS.org)

S. Böhm and D. Lübke (Eds.): 16th ZEUS Workshop, ZEUS 2024, Ulm, Germany, 29 February–1 March 2024, published at <http://ceur-ws.org>

implements the data-centric process management paradigm. There are object types and their corresponding object behaviours. The latter are specified in terms of lifecycle processes which comprise states and state transitions. At runtime, the object types are instantiated with their lifecycle processes. Activities with user forms are generated when a state is reached in which data is required and activities are completed (i.e. the process continues) when the required data becomes available (i.e. the user has entered the object attributes). Thus, the progress of a lifecycle process is data-driven and determined by changes of the object attributes. PHILharmonicFlows is a framework for object-aware process management. In [9], [10], and [11], an object-aware and process-aware information system (OPAIS) is presented as a proof-of-concept implementation of PHILharmonicFlows. While there have been approaches dealing with variants in activity-centric processes, only little research covers variants in OPAISs. The work in [12] deals with variants in one OPAISs. However, the same process can be operated in variants in different autonomous organisations that run their OPAIS separately. The management of process variants in separate OPAISs remains an open challenge, which is addressed in this paper.

2. Related Work

This section presents an approach to deal with variants in OPAISs as well as core concepts of SPL Engineering.

In [12], an approach is presented for coping with process variants in OPAISs. At design time, when evolving a process in an OPAIS (e.g. adding or removing attributes from an object or updating a lifecycle process) the changes are logged rather than propagated to the process in production. As soon as the changes need to be deployed to production, the logs created during design time can be replayed, i.e. propagating the changes to production. When developing various variants of a process, the base process containing the commonalities is copied by replaying the corresponding logs. Each copy of the base process is the starting point for a variant where the variant-specific changes are applied to. When the base process is altered, the changes can be applied to all variants by replaying the logs triggered by the changes to the base process. Certainly, this approach facilitates the development of variants and the propagation of common changes. However, the approach neglects that autonomous organisations that run process variants will most likely have physically independent hardware/OPAIS instances. Therefore, the logs need to be distributed among various OPAIS instances. Furthermore, the approach assumes that each variant is used once. However, multiple organisation might use the same variant each on their separate OPAIS instance. The logs should not be copied but referenced so that changes to a variant can be performed centrally and applied to all organisations that use the same variant. Moreover, an organisation might use a combination of variants instead of creating a new variant from scratch or use an outdated version of a variant while the variant has been updated centrally, i.e. different versions of a variant might be in production. During bug analysis, it becomes necessary to rebuild a specific version of a variant of an organisation.

Feature models [13] describe software products in terms of their features by outlining which features are mandatory, optional, and which features require or exclude each other.

Configuration management in SPL Engineering [14] deals with managing the versions of the common core artefacts as well as the derived software products over time. In contrast, version

control tools for configuration management in software engineering only manage one software product over time. In [15], a Divide & Conquer approach for configuration management is proposed that divides the challenge in sub-challenges and solves them with the use of existing version control tools from software engineering.

3. Research Direction

This section presents an approach to deal with variants in OPAIS of autonomous organisations. First, a real-world example is described which serves as motivation. Then, requirements are elicited based on the example. Finally, an approach satisfying these requirements is sketched.

During a cooperation with German municipalities, the process for checking the special parking permit application of craftspersons was studied. This process is pictured in a slightly adapted and simplified form compared to our previous work [1]. In German municipalities craftspersons may apply for a special parking permit which allows them to park in zones of the city where citizens have to pay or where parking is not permitted. The object-aware process for checking this application is depicted in Figure 1. The applicant must provide information about her company and the company's cars for which the parking permit shall be valid (cf. object types on the left-hand side). On the right-hand side, the lifecycle processes for the object types are shown. Different municipalities might need adaptations to the base process (cf. Figure 1) which results in variants. Some municipalities might require the applicant to submit pictures of the cars to prevent issuing parking permits for private cars. In other municipalities, information about the cars is not required at all. Then, the parking permit is valid for whatever car it is situated in. German municipalities are autonomous organisations, which run their information systems separately.

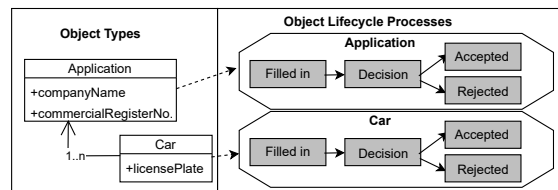


Figure 1: Object-Aware Process

Based on the example, the requirements are deduced for managing variants in OPAISs of autonomous organisations.

- R1:* Each organisation must be able to select a combination of adaptations in conjunction with the base process and build (i.e. derive) a process variant.
- R2:* Multiple organisations may derive the same process variant.
- R3:* Adaptions need to be managed centrally in that changes to them can be propagated easily to all process variants that are using these adaptations.
- R4:* Not every combination of adaptations may be valid. It needs to be evident what adaptations exclude each other.
- R5:* As the base process and the adaptations evolve, not every organisation may want to apply the updates.
- R6:* For bug analysis, it needs to be evident which organisation uses which version and adaptations of the base process in order to rebuild and analyse the corresponding variant.

Using concepts of SPL Engineering, namely feature models [13] and the Divide & Conquer approach for configuration management proposed in [15], it becomes possible to satisfy the

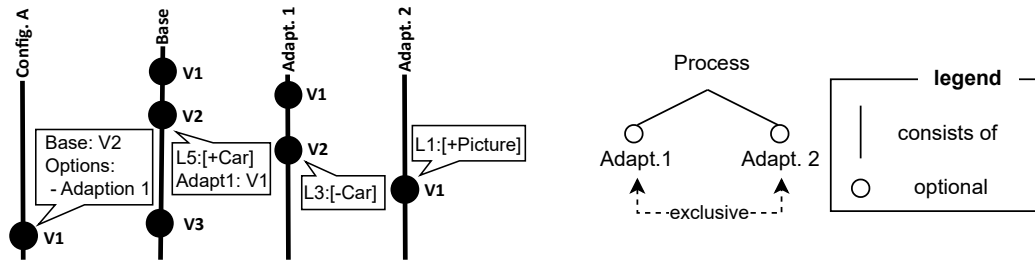


Figure 2: Configuration Management

Figure 3: Feature Diagram

requirements. The base process and every adaption are stored as logs each in a central repository under version control. Thereby, changes due to the evolution of the base process and the adaptations can be applied centrally. Every organisation has a configuration that stores the version of the base process and the selected adaptations. During derivation, the referenced logs are fetched and replayed in the organisation’s OPAIS instance. As each repository is under version control, older version of the logs can be used which allows the organisations to stay on older versions of the variants. Furthermore, for bug analysis, the configuration of each organisation is under version control as well. Consequently, to analyse a bug a specific version of a variant of an organisation can be restored and inspected. Figure 2 is an example for configuration management and shows the repositories and their version history over time. It reveals that Adaption 1 in Version V2 uses log L3 to remove the object type *Car*, whereas Adaption 2 in Version V1 uses log L1 to add the object type *Picture*. Obviously, Adaptions 1 and 2 cannot be co-selected as the picture object type needs a car object type it belongs to, which is removed in Adaption 1. Figure 3 imposes the base process with its adaptations in a feature model, which indicates that Adaptions 1 and 2 are mutually exclusive. Furthermore, Figure 2 represents the evolution of the base process model (viz. three versions) and the exemplary repository of one organisation, which stores a configuration (Config. A) to derive a process variant by specifying that the base process in version V2 and Adaptation 1 is selected.

4. Conclusion and Outlook

This paper deals with managing variants in OPAISs in that multiple autonomous organisations may derive process variants from a base process by selecting and combining adaption to the base process whereas different versions of both the base process and the adaptations may be used. Consequently, there may be a variety of process variants and versions that need to be tracked. This work presents an approach that applies concepts of SPL Engineering to OPAIS variants in order to keep track of the process variants and their versions.

In future work, we plan to assemble a tool chain that is able to automatically derive and keep track of process variants in OPAIS by using, extending, and integrating tools from SPL Engineering, and, where necessary, creating new tools. Furthermore, black-box-activities [9] implementing business logic as well as customised forms (i.e. opposed to generated forms) need to be considered in the future.

References

- [1] P. Hehnle, M. Reichert, Handling Process Variants in Information Systems with Software Product Line Engineering, in: 2023 IEEE 25th Conference on Business Informatics (CBI), 2023, pp. 1–10.
- [2] A. Hallerbach, T. Bauer, M. Reichert, Capturing variability in business process models: the Provop approach, *Journal of Software Maintenance and Evolution: Research and Practice* 22 (2010) 519–546.
- [3] W. M. P. van der Aalst, A. Dreiling, F. Gottschalk, M. Rosemann, M. H. Jansen-Vullers, Configurable Process Models as a Basis for Reference Modeling, in: *Business Process Management Workshops*, volume 3812 of *Lecture Notes in Computer Science*, Springer, 2006, pp. 512–518.
- [4] L. Bass, P. Clements, R. Kazman, *Software architecture in practice*, Always learning, 3. ed. ed., Addison-Wesley, 2013.
- [5] S. Deelstra, M. Sinnema, J. Bosch, Product derivation in software product families: A case study, *Journal of Systems and Software* 74 (2005) 173–194.
- [6] C. Kästner, S. Apel, Feature-Oriented Software Development, in: *Generative and Transformational Techniques in Software Engineering IV: International Summer School, GTTSE 2011*, Springer, 2013, pp. 346–382.
- [7] S. Steinau, A. Marrella, K. Andrews, F. Leotta, M. Mecella, M. Reichert, DALEC: A framework for the systematic evaluation of data-centric approaches to process management software, *Software & Systems Modeling* 18 (2019) 2679–2716.
- [8] V. Künzle, M. Reichert, PHILharmonicFlows: towards a framework for object-aware process management, *Journal of Software Maintenance and Evolution: Research and Practice* 23 (2011) 205–244.
- [9] C. M. Chiao, V. Kuenzle, K. Andrews, M. Reichert, A Tool for Supporting Object-Aware Processes, in: 2014 IEEE 18th International Enterprise Distributed Object Computing Conference Workshops and Demonstrations, 2014, pp. 410–413.
- [10] K. Andrews, S. Steinau, M. Reichert, A Runtime Environment for Object-Aware Processes, in: *International Conference on Business Process Management*, 2015, pp. 6–10.
- [11] S. Steinau, K. Andrews, M. Reichert, A Modeling Tool for PHILharmonicFlows Objects and Lifecycle Processes, in: *International Conference on Business Process Management*, 2017.
- [12] K. Andrews, S. Steinau, M. Reichert, Enabling Process Variants and Versions in Distributed Object-Aware Process Management Systems, in: *Information Systems in the Big Data Era*, Springer, 2018, pp. 1–15.
- [13] K. C. Kang, S. G. Cohen, J. A. Hess, W. E. Novak, S. Peterson, *Feature-Oriented Domain Analysis (FODA) Feasibility Study*, Technical Report, 1990.
- [14] P. Clements, L. Northrop, *Software product lines: Practices and patterns*, SEI series in software engineering, 7. print ed., Addison-Wesley, 2009.
- [15] C. W. Krueger, Variation Management for Software Production Lines, in: *Software Product Lines*, volume 2379 of *Lecture Notes in Computer Science*, Springer Nature, 2002, pp. 37–48.

Simulating Event Logs from Object Lifecycle Processes

Marius Breitmayer^{1,*}, Lisa Arnold¹ and Manfred Reichert¹

¹*Institute of Databases and Information Systems, Ulm University, Germany*

Abstract

Process Mining leverages event data to discover, analyze, and optimize business processes. Consequently, the availability and quality of event logs is crucial for the applicability as well as the development of process mining algorithms. However, obtaining such event logs from real-world scenarios is limited and costly. In this paper, we present an approach for simulating event logs from object lifecycle processes. Overall, the approach is capable of simulating event logs for object lifecycle processes that may be used to validate process mining algorithms.

Keywords

Event Log Simulation, Object-centric Processes, Object Lifecycle Processes, Flexibility

1. Introduction

Process mining is becoming more and more important for enterprises, as it provides a transparent and data-driven view on operational workflows while also offering insights into inefficiencies, bottlenecks as well as possible compliance issues. In general, process mining leverages event data to discover process models, check the conformance between models and event logs, or enhance the model [1]. In real-world scenarios, such event logs are often unavailable, inappropriate, or costly to obtain, consequently hindering the development and testing of new process mining algorithms [2].

In other domains, simulation provides a possible solution towards this problem by substituting or complementing real-world with simulated data [3, 4, 5].

In object-centric scenarios, the behavior of objects is defined in terms of object lifecycle processes. At runtime, however, objects may deviate from the behavior specified in object lifecycle processes due to flexibility [6]. Such deviations may include removing or adding both states and steps as well as executing them in a different order. Consequently, the simulation of event logs must account for such flexible behavior to ensure that the event logs are comparable to their real-world counterparts.

Overall, the simulation of event logs from object lifecycle processes presented in this paper facilitates the application and testing of process mining algorithms in object-centric scenarios by providing suitable event logs.

16th Central European Workshop on Services and their Composition, February 29– June 01, 2024, Ulm, Germany

*Corresponding author.

✉ marius.breitmayer@uni-ulm.de (M. Breitmayer); lisa.arnold@uni-ulm.de (L. Arnold);

manfred.reichert@uni-ulm.de (M. Reichert)

ORCID 0000-0003-1572-4573 (M. Breitmayer); 0000-0002-2358-2571 (L. Arnold); 0000-0003-2536-4153 (M. Reichert)



© 2022 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

The paper is structured as follows: Section 2 introduces object lifecycle processes. Section 3 describes the approach for simulating event logs. Section 4 evaluates the presented approach, and Section 5 discusses related work. Finally, Section 6 summarizes the paper and provides an outlook.

2. Fundamentals

In object-centric processes, the behavior of objects is specified in terms of a state-based object lifecycle process. Fig. 1 depicts the object lifecycle process for object *Submission*. An object lifecycle process comprises the states of the object as well as state transitions (see states *Edit*, *Submit*, *Rate*, and *Rated* in Fig. 1). Each state, in turn, comprises several steps (see *Exercise*, *E-Mail*, and *Files* in State *Edit* in Fig. 1) defining which object attributes are required before completing a state and transition to the next one. Note that we also support predicate steps to enable different execution paths through the object lifecycle process.

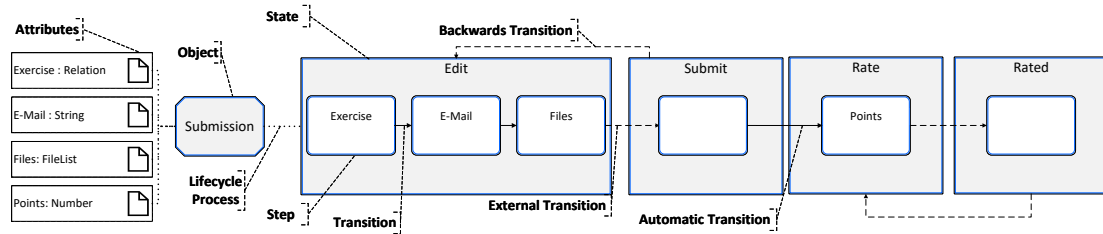


Figure 1: (simplified) Object Lifecycle Process Object Submission

During process execution, multiple objects (of the same or different type) may exist, and their instances are dynamically generated [7]. For each of these objects, their different object behavior may be observed on both the state level (i.e., how does an object change between its states), and the step level (i.e., how is data provided). Tab. 1 summarizes the guided, tolerated, and deviating behavior on both granularity levels.

On the state level, guided behavior corresponds to the object reaching its end state without returning to a previous state using a backwards transition. Tolerated behavior, in turn, includes backwards transitions. Deviating behavior on the state level is associated with skipping states (i.e., non-reachable) or reaching states not specified in the object lifecycle process model.

On the step level, guided behavior is observed if the steps of a state are provided in the order specified by the object lifecycle process. Tolerated behavior, in turn, only requires that all steps are provided (i.e., the order is not relevant). Deviating behavior on the step level includes skipping steps (i.e., not providing certain attributes) or providing additional steps in a particular state of the object lifecycle process model.

Table 1

Object Behavior on state and step level [8]

Level	Behavior	Description
State	Guided	The end state is reached without following any backwards transitions.
	Tolerated	The end state is reached, but backwards transitions were chosen.
	Deviating	The lifecycle transitions to a non-reachable or unspecified state during its execution.
Step	Guided	All steps of a lifecycle state are provided according to the pre-specified order.
	Tolerated	All steps have been filled prior to state completion.
	Deviating	Steps have been skipped or additional steps have been added.

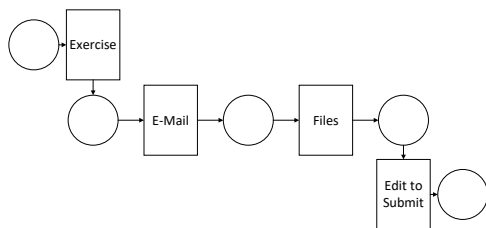
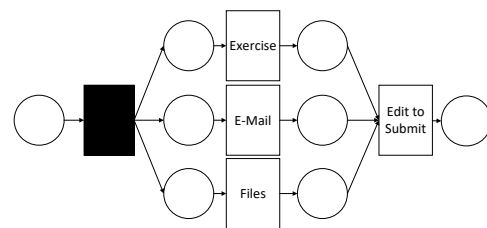
3. Event Log Simulation

The event log simulation must be capable of simulating guided, tolerated, and deviating behavior on both the state as well as the step level (see Tab. 1). To be more precise, simulated event logs must be able to represent all 3 behaviors regarding the state as well as the step level. Due to their well-defined syntax and replay capabilities, we decided to represent object lifecycle processes in terms of Petri nets [9].

3.1. Object Lifecycle Process Representation

We enable the simulation of different behavior by representing each object lifecycle process as two Petri nets. One of these Petri nets corresponds to the guided behavior. In other words, this Petri net only allows for the object lifecycle process to be executed exactly as modeled. The other Petri net represents the tolerated behavior by allowing for additional (tolerated) behavior. This behavior includes, for example, providing object attributes in an order different from the one specified by the object lifecycle process model, while also ensuring that all required attributes are provided. The representation of object lifecycle processes in terms of Petri nets enables the application of token-based simulation on both nets.

Figs. 2 and 3 depict two Petri nets generated for state *Edit* of object *Submission* (see Fig. 1). In a nutshell, we represent guided behavior by accounting for the sequence in which steps (and states) are modeled in the object lifecycle process, whereas we allow for tolerated behavior using an AND-split syntax within the state. Note that, we only depict the guided and tolerated nets for state *Edit* of object *Submission*, however, we concatenate the guided and tolerated Petri nets for each state to represent the state level and include backwards transitions if necessary.

**Figure 2:** Guided Petri Net State Edit**Figure 3:** Tolerated Petri Net State Edit

3.2. Trace Generation

After representing object lifecycle processes, we generate traces for individual objects. Alg. 1 describes the generation of traces in pseudocode. First, we check which transitions are enabled by the Petri net and check if the current marking equals the final marking. If not, we randomly choose an enabled transition, add it to the trace, and execute it. This updates the current marking. The trace generation terminates if no more enabled transitions exist, or the final marking is reached. In the latter case, the trace is added to the list of traces. Otherwise, the trace is discarded. This is repeated while the counter is smaller than the number of requested traces. In other words, we apply a token-based simulation in which tokens are propagated through the Petri net and the transitions passed are recorded as events in the event log. In a post-processing step, deviating behavior may be introduced to the generated event log by adding or removing corresponding events (i.e., writing (additional) attributes or transitioning to (additional) states).

Algorithm 1 Trace Generation

Require: PetriNet ▷ guided or tolerated net

```

1: visitedTransitions = []
2: marking ← copy(initialMarking)
3: while True do
4:   if not semantics.enabled_transitions(PetriNet, marking) then
5:     Break
6:   end if
7:   enabledTransitions ← semantics.enabled_transitions(PetriNet, marking)
8:   if marking == finalMarking then
9:     Break
10:  else
11:    transition ← randomElement(enabledTransitions)
12:  end if
13:  if transition.label is not None then
14:    visitedTransitions.append(transition)
15:  end if
16:  marking ← semantics.execute(transition, PetriNet, marking)
17: end while
18: if marking = finalMarking then
19:   allVisitedTransitions.append(visitedTransitions)
20:   counter ← counter + 1
21: end if

```

4. Evaluation

The evaluation of the presented approach for simulating event logs from object lifecycle processes is two-fold. First, we implemented a proof-of-concept prototype capable of simulating event logs from object lifecycle processes and applied it to two different scenarios (E-learning and Recruitment). Second, we checked the conformance of the simulated event logs with the models used for their generation and calculated resulting fitness values using alignments [8, 10].

On a scale from 1 (perfect fitness) to 0 (bad fitness), fitness measures to which degree a model allows for the behavior recorded in an event log. The proof-of-concept prototype as well as the event logs simulated during the evaluation are publicly available¹.

4.1. Conformance Checking

For every object in both the e-learning and recruitment scenario, we simulated 18 event logs (i.e., one log per possible configuration), each comprising 100 traces. In total, this results in 198 event logs from 11 different objects.

Tab. 2 describes the conformance categories into which the simulated traces are categorized for object *Submission* [8].

The event logs simulated for guided behavior configuration show the respective behavior in 100% of cases. Moreover, the event logs generated for the tolerated behavior show guided behavior in 17% of the cases and tolerated behavior in 83 % of cases. The inclusion of guided behavior within tolerated behavior is intended, as it generalizes guided behavior. Regarding deviating behavior (i.e., skipping or adding states and steps) all traces simulated with the respective configuration are correctly simulated with deviations.

Table 2

Categorization of Simulated Event Logs for Object *Submission*

	Guided Log	Tolerated Log	Deviating Log
Guided Behavior	100 %	17 %	0%
Tolerated Behavior	0%	83 %	0%
Deviating Behavior	0%	0%	100%

We further checked conformance of the simulated event logs regarding the state (i.e., how does the object instance change states?) and the step level (i.e., how are attributes written?). On the state level, this allows further verifying the deviations (e.g., additional states, unspecified backwards transitions, or state changes not allowed by the object lifecycle process model). We checked the conformance between the object lifecycle process model and the event logs simulated with additional states as well as steps in every trace on both the state and step level individually (see Tab. 3). In other words, the event log contained deviations on both the state and step level. Note that we only show the results for state *Edit* of object *Submission* for brevity.

Table 3

Object *Submission* on State and Step level

	State Level		Step Level (State Edit)	
	Average Fitness Value	% of traces	Average Fitness Value	% of traces
Guided Behavior	0.31	0 %	0.46	0 %
Tolerated Behavior	0.58	0 %	0.67	0 %
Deviating Behavior	NA	100 %	NA	100 %

Overall, the approach for generating event logs from object lifecycle processes is capable of reproducing all possible behavior based on the specification selected in the proof-of-concept implementation.

¹<https://cloudstore.uni-ulm.de/s/M4AmamPLWZHniZ6>

5. Related Work

The approach presented in this paper is related to the generation of event logs.

[11] presents the *PURpose-guided Log gEerator (PURPLE)* capable of generating event logs conforming to a process model (i.e., a BPMN model or Petri net), including noise (i.e., skipping, adding, or reordering events). In contrast, the presented approach considers granularity and uses data-driven object behavior specified in object lifecycle process models as input.

[12] introduce the *Data Aware event Log Generator (DALG)* that can generate event logs from a Petri net. DALG offers several capabilities for temporal constraints and writing variables, however, no capabilities regarding deviations in the event logs are available.

The *Straightforward Petri Net-based Event Log Generation* plugin available in ProM [13] generates event logs based on Petri nets using token-based simulation. In contrast to the presented approach, it does not differentiate between guided and tolerated behavior and does not generate deviating event logs.

Event log generation in the context of Internet of Things devices is presented in [2]. A Petri net is used to generate guided event logs as well as event logs containing noise. In contrast to the presented approach, different granularity levels are not supported.

[14] can generate event logs from BPMN models and is available in the process mining framework *ProM*. The presented approach leverages data-driven object lifecycle processes rather than activity-centric BPMN models.

Other approaches leverage relational databases to generate event logs based on the data available in the database [15]. In this approach, a case table with a unique case ID has to be chosen. Users then recommend other tables that could provide additional events. These tables, however, have to contain the case ID as foreign keys. The approach presented in this paper can produce guided, tolerated and deviating behavior on both state and step level and it may generate event logs of different objects as well as multiple instances of each object.

6. Summary & Outlook

This paper presented an approach for simulating event logs from object lifecycle processes, while considering the behavior of objects on both the state and the step level. The approach is capable of simulating guided, tolerated, and deviating object behavior and document the behavior in event logs using multiple Petri nets derived from object lifecycle processes. It has been evaluated using a publicly available proof-of-concept implementation which has been successfully applied to simulate event logs from different scenarios. Event logs simulated using the presented approach reliably conform with the specification set out, facilitating the development and testing of new process mining algorithms by providing suitable event logs for a variety of scenarios. In future work, we plan to extend the approach in multiple directions. First, we want to incorporate coordination constraints between objects of different types (i.e., object A may only reach state S_A if object B is in state S_B). Second, the actual object attribute values documented in the event log are currently generic placeholders. We plan to replace them with more realistic, process-specific values. Finally, we plan to provide the event logs in additional formats (e.g., OCEL [16] or XES [17]) instead of the currently used CSV-format.

References

- [1] W. M. P. van der Aalst, *Process Mining: Data Science in Action*, 2 ed., Springer, Heidelberg, 2016.
- [2] Y. Zisgen, D. Janssen, A. Koschmider, Generating synthetic sensor event logs for process mining, in: *International Conference on Advanced Information Systems Engineering*, Springer, 2022, pp. 130–137.
- [3] J. Chen, D. Chun, M. Patel, E. Chiang, J. James, The validity of synthetic clinical data: a validation study of a leading synthetic data generator (synthea) using clinical quality measures, *BMC medical informatics and decision making* 19 (2019) 1–9.
- [4] N. Patki, R. Wedge, K. Veeramachaneni, The synthetic data vault, in: *2016 IEEE International Conference on Data Science and Advanced Analytics (DSAA)*, 2016, pp. 399–410.
- [5] J. Tremblay, A. Prakash, D. Acuna, M. Brophy, V. Jampani, C. Anil, T. To, E. Cameracci, S. Boochoon, S. Birchfield, Training deep networks with synthetic data: Bridging the reality gap by domain randomization, in: *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, 2018, pp. 969–977.
- [6] K. Andrews, S. Steinau, M. Reichert, Enabling runtime flexibility in data-centric and data-driven process execution engines, *Information Systems* 101 (2021) 101447. URL: <https://www.sciencedirect.com/science/article/pii/S0306437919304995>. doi:<https://doi.org/10.1016/j.is.2019.101447>.
- [7] S. Steinau, K. Andrews, M. Reichert, Executing lifecycle processes in object-aware process management, in: *Data-Driven Process Discovery and Analysis, Lecture Notes in Business Information Processing*, Springer, 2017, pp. 25–44.
- [8] M. Breitmayer, L. Arnold, M. Reichert, Enabling conformance checking for object lifecycle processes, in: *Research Challenges in Information Science*, Springer International Publishing, Cham, 2022, pp. 124–141.
- [9] J. L. Peterson, Petri nets, *ACM Computing Surveys (CSUR)* 9 (1977) 223–252.
- [10] W. M. P. van der Aalst, A. Adriansyah, B. van Dongen, Replaying history on process models for conformance checking and performance analysis, *WIREs Data Mining and Knowledge Discovery* 2 (2012) 182–192.
- [11] A. Burattin, B. Re, L. Rossi, F. Tiezzi, Purple: a purpose-guided log generator, in: *Proceedings of the ICPM Doctoral Consortium and Demo Track 2022, CEUR Workshop Proceedings, CEUR-WS*, 2022.
- [12] D. Jilg, Generating synthetic procedural multi-perspective electronic healthcare treatment cases, 2022.
- [13] S. Vanden Broucke, J. Vanthienen, B. Baesens, Straightforward petri net-based event log generation in prom, SSRN 2489051 (2014).
- [14] A. A. Mitsyuk, I. S. Shugurov, A. A. Kalenkova, W. M. P. van der Aalst, Generating event logs for high-level process models, *Simulation Modelling Practice and Theory* 74 (2017) 1–16.
- [15] R. Andrews, C. G. J. van Dun, M. T. Wynn, W. Kratsch, M. Röglinger, A. H. ter Hofstede, Quality-informed semi-automated event log generation for process mining, *Decision Support Systems* 132 (2020) 113265.
- [16] A. F. Ghahfarokhi, G. Park, A. Berti, W. M. P. van der Aalst, Ocel: A standard for object-

- centric event logs, in: L. Bellatreche, M. Dumas, P. Karras, R. Matulevičius, A. Awad, M. Weidlich, M. Ivanović, O. Hartig (Eds.), *New Trends in Database and Information Systems*, Springer International Publishing, Cham, 2021, pp. 169–175.
- [17] Ieee standard for extensible event stream (xes) for achieving interoperability in event logs and event streams, IEEE Std 1849-2016 (2016) 1–50. doi:10.1109/IEEESTD.2016.7740858.

Clounaq - Cloud-native architectural quality

(Tool Demonstration)

Robin Lichtenthäler

Distributed Systems Group, University of Bamberg, An der Weberei 5, 96047 Bamberg, Germany

Abstract

Implementing cloud-native applications can be complex and challenging due to the breadth of the topic. A possibility to evaluate potential impacts from architectural characteristics on quality attributes could support the development and evolution of cloud-native applications. This work describes the Clounaq approach which aims to provide such a possibility. With the Clounaq tool software architectures can be modeled and evaluated based on a quality model which describes relationships between architectural characteristics and multiple quality aspects.

Keywords

cloud computing, cloud-native, quality evaluation, architectural model

1. Introduction

Cloud-native as a term has established itself in academia [1] and industry¹ for describing applications and tooling that take advantage of the characteristics of modern cloud environments [2, 3]. However, cloud-native covers a broad range of aspects impeding a clear and commonly accepted understanding of the term. Nevertheless, many benefits are associated with cloud-native. Thus, our motivation is how developers can be supported in developing software in a cloud-native way. Regarding the type of applications, our focus is on *enterprise applications* [4], that means web applications implemented by organizations to serve their customers or to satisfy internal need. Furthermore, our approach is focused on the design time. Although we acknowledge that an unambiguous measurement of the quality of an application is only possible at runtime, we argue that evaluations should be possible at design time already. When potential impacts of certain architectural characteristics on quality aspects can be evaluated and investigated at design time, problems can be avoided early on and an application can be designed according to cloud-native concepts from the beginning.

Our support for designing and evaluating applications is provided in the form of 1) A quality model that structures design-time architectural characteristics of cloud-native applications according to higher level quality aspects, and 2) a web-based tool which enables the modeling and evaluation of software architectures based on the quality model. We named our approach

16th Central European Workshop on Services and their Composition (ZEUS), February 29 – March 1, 2024, Ulm, Germany

✉ robin.lichtenthaeler@uni-bamberg.de (R. Lichtenthäler)

🌐 <https://www.uni-bamberg.de/pi/team/lichtenthaeler-robin/> (R. Lichtenthäler)

🆔 0000-0002-9608-619X (R. Lichtenthäler)



© 2024 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

📄 CEUR Workshop Proceedings (CEUR-WS.org)

¹<https://github.com/cncf/toc/blob/main/DEFINITION.md>

S. Böhm and D. Lübke (Eds.): 16th ZEUS Workshop, ZEUS 2024, Ulm, Germany, 29 February–1 March 2024, published at <http://ceur-ws.org>

Clounaq which is short for *Cloud-native architectural quality*. After a short overview of the quality model, we specifically present the implementation in this work. While working on it, several challenges had to be considered which we list here in a structured way:

- C1 Choosing a modeling approach and a suitable level of abstraction
- C2 Including support for the integration with other approaches and tools
- C3 Presenting information on modeling options and evaluation results in a meaningful way
- C4 Handling the complexity of both the quality model and the architectural models
- C5 Ensuring modifiability and extensibility of the tool

These challenges shaped the implementation of the approach and are the reasons for certain design decisions. Throughout this work we refer back to these challenges to highlight their causes or how they influenced the implementation. In section 2 we present the idea behind and the current state of the quality model. This serves as a foundation for the presentation of our tool in section 3, before we conclude our work in section 4.

2. Cloud-native Quality Model

The quality model for cloud-native software architectures² provides the conceptual foundation for our approach. Its initial formulation is based on literature [5] and parts have been validated empirically based on a survey [3]. In essence, the quality model includes *quality aspects* which represent higher level quality attributes, *product factors* which represent architectural characteristics whose prevalence in an architecture can be measured, and *impacts* which describe how the presence of a product factor impacts different quality aspects. An exemplary factor is *Service replication* which captures the characteristic of replicating service instances across infrastructure entities. A replication of service instances is considered to have a positive impact on *Time-behaviour*, because requests can be served from replicas closest to the respective client. And it is considered to have a positive impact on *Availability*, because even if one instance becomes unavailable, another replica can still serve requests.

To now evaluate an application based on this product factor, the extent of replication needs to be measurable. Therefore *measures* are used which can quantify characteristics of an application at a specified level of abstraction (C1). The values of measures are interpreted through *evaluations*. For the example of service replication *service replication level* (based on [6]) is a measure that quantifies the average level of service replication. A possible evaluation for this could be that a value of 1 means no replication and therefore no impact on availability and time-behaviour. And a value between 1 and 3 could mean a low replication and thus a slightly positive impact on availability and time-behavior.

This is merely one example from the quality model. Because it tries to cover the breadth of the topic of cloud-native applications and multiple quality aspects in combination, the quality model itself has a certain complexity (C4). Furthermore, the quality model is not completely specified yet regarding the quantitative measures and evaluations. The aim is to develop the model further together with the tool based on the application of our approach to different use cases.

²<https://r0light.github.io/cna-quality-model/>

3. Clounaq Tool

The Clounaq tool³ is the implementation of our quality evaluation approach. An overview of its main parts is provided in fig. 1 which is discussed more in detail in the following subsections.

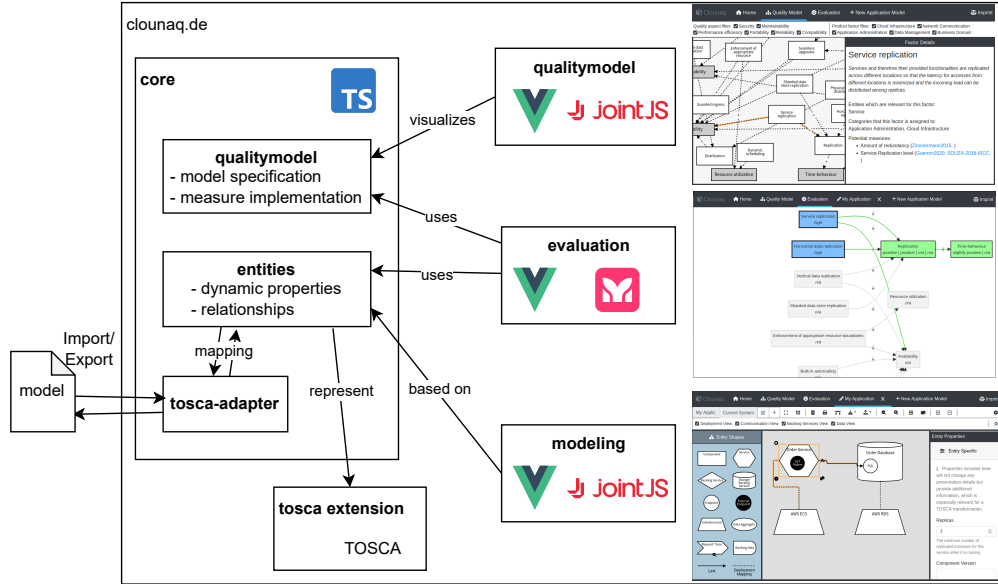


Figure 1: The internal design of the clounaq tool and corresponding views.

3.1. Functionality

The three main parts of the tool are accessible through different tabs (see right side of fig. 1):

Quality Model Tab This tab shows the quality model as described in section 2 with all product factors and quality aspects. Because of the complexity of the model (C4), it is possible to filter the shown quality aspects by their common high-level aspects or by different implementation aspects. Furthermore, to tackle C3, when a factor is selected, additional information is presented in a details section and related literature is linked.

Modeling Tab(s) For each created or imported software architecture model a new modeling tab is added. Therefore it is possible to work on multiple models at the same time. The initial modeling prototype which is now integrated in this tab was presented by Durr and Lichtenthaler [7]. In that work, we also considered different modeling options (C1) and decided to build on the TOSCA standard [8] because of its extensibility. Therefore we created a TOSCA profile that defines the different entity types required to model architectures of cloud-native applications and to evaluate them. With the decision for TOSCA it is in general possible to import models created with our tool also in other tools (C2) which support the TOSCA standard (e.g. Winery [9]). Currently, only an import and export for our TOSCA extension and for a custom JSON format is supported, but a mapping to formats of Infrastructure as Code (IaC)

³<https://clounaq.de>

tools could also be implemented. The modeling itself uses a graphical notation taking into account principles [10] for clarity and understandability (tackling C3). Additionally, also for the architectural models filters are available for temporarily focusing only on certain entities (C4).

Evaluation Tab In this tab, an architectural model can be selected for evaluation. For the product factors the values of specified measures are calculated and interpreted based on specified evaluations. With these evaluations, in turn, the quality aspects are evaluated and the results are presented with details so the user can comprehend their calculation (C3). Furthermore, either the perspective of product factors or of quality aspects can be selected (C4).

3.2. Implementation

The tool is implemented as a Single Page Application (SPA) using Typescript and Vue.js. It runs entirely in the browser of the user and does not have an additional backend. This design is intentional to simplify hosting (it is currently hosted using Github Pages) and support the reproducibility of our approach. Accordingly, it is open source and available at Github⁴, meaning that it can also be modified according to own needs (C5). Internally, the application is structured in four main parts represented by folders underneath `src` (see also fig. 1): In `core`, the entities are implemented as plain classes, but with a flexible properties attribute that is filled based on the underlying TOSCA extension. Therefore, properties can be specified only in TOSCA and the tool automatically adopts them. Furthermore, the quality model is specified in plain JSON and can be modified as such (C5) with changes being reflected in the tool. In `qualitymodel`, the quality model tab is implemented, solely depending on the specification of the model in `core`. Similarly, `evaluation` includes the code for the evaluation tab, depending solely on the code in `core` to evaluate a `System` entity based on the specified quality model and implemented measures and evaluations. Finally, `modeling` provides the modeling tab implementation based on JointJS. For each entity, a diagramming shape is specified and the properties editor dynamically includes the properties of each entity type.

3.3. Development Roadmap

The tool is in active development. While the modeling functionality and the visualization of the quality model are considered done, the evaluation functionality is being worked on: Specifically, additional measure calculations need to be defined and implemented. Where applicable, this also means additional properties need to be added to entities so that measures can be calculated. Finally, the evaluation tab should display all such relevant information in a structured way (C3).

4. Conclusion

This work presents the current state of the tool. However, it is refined in parallel with the overall approach. The tool enables the practical application of the approach, so that it can be tested and improved. Through quality evaluation results, useful features or required changes for the tool are uncovered. We therefore expect an ongoing development of both the approach and the tool when applying them to use cases, as we plan to do in future work.

⁴<https://github.com/r0light/cna-quality-tool>

References

- [1] N. Kratzke, P.-C. Quint, Understanding Cloud-native Applications after 10 Years of Cloud Computing - A Systematic Mapping Study, *Journal of Systems and Software* 126 (2017) 1–16. doi:10.1016/j.jss.2017.01.001.
- [2] D. Gannon, R. Barga, N. Sundaresan, Cloud-Native Applications, *IEEE Cloud Computing* 4 (2017) 16–21. doi:10.1109/mcc.2017.4250939.
- [3] R. Lichtenthäler, J. Fritzsche, G. Wirtz, Cloud-Native Architectural Characteristics and their Impacts on Software Quality: A Validation Survey, in: 2023 IEEE International Conference on Service-Oriented System Engineering (SOSE), IEEE Computer Society, Los Alamitos, CA, USA, 2023. doi:10.1109/SOSE58276.2023.00008.
- [4] T. Cerny, J. Svacina, D. Das, V. Bushong, M. Bures, P. Tisnovsky, K. Frajtek, D. Shin, J. Huang, On Code Analysis Opportunities and Challenges for Enterprise Systems and Microservices, *IEEE Access* 8 (2020) 159449–159470. doi:10.1109/access.2020.3019985.
- [5] R. Lichtenthäler, G. Wirtz, Towards a Quality Model for Cloud-native Applications, in: *Service-Oriented and Cloud Computing*, Springer, 2022, pp. 109–117. doi:10.1007/978-3-031-04718-3_7.
- [6] R. H. de Souza, P. A. Flores, M. A. R. Dantas, F. Siqueira, Architectural recovering model for distributed databases: A reliability, availability and serviceability approach, in: *Symposium on Computers and Communication (ISCC)*, IEEE, 2016. doi:10.1109/iscc.2016.7543799.
- [7] K. Dürr, R. Lichtenthäler, An evaluation of modeling options for cloud-native application architectures to enable quality investigations, in: 2022 IEEE/ACM 15th International Conference on Utility and Cloud Computing (UCC), IEEE, 2022. doi:10.1109/ucc56403.2022.00053.
- [8] OASIS, TOSCA Simple Profile in YAML Version 1.3, 2020. URL: <https://docs.oasis-open.org/tosca/TOSCA-Simple-Profile-YAML/v1.3/>, oASIS Standard.
- [9] O. Kopp, T. Binz, U. Breitenbücher, F. Leymann, Winery – A Modeling Tool for TOSCA-Based Cloud Applications, *Springer Berlin Heidelberg*, 2013, pp. 700–704. doi:10.1007/978-3-642-45005-1_64.
- [10] D. Moody, The “physics” of notations: Toward a scientific basis for constructing visual notations in software engineering, *IEEE Transactions on Software Engineering* 35 (2009) 756–779. doi:10.1109/tse.2009.67.

How good are you? An empirical classification performance comparison of large language models with traditional open set recognition classifiers

Alexander Grote^{1,*†}, Anuja Hariharan^{2,†}, Michael Knierim^{2,†} and Christof Weinhardt^{2,†}

¹FZI Research Center for Information Technology, Haid-und-Neu-Str. 10–14, 76131 Karlsruhe, Germany

²Karlsruhe Institute of Technology, Kaiserstraße 12, 76131 Karlsruhe, Germany

Abstract

The release of ChatGPT has led to an unprecedented surge in the popularity of generative AI-based Large Language Models (LLMs) among practitioners. These models have gained traction in business processes due to their ability to receive instructions in natural language. However, they suffer from hallucinations, which are generated texts that are factually incorrect. The issue of hallucinations also arises in text classification tasks, such as customer support ticket classification or intent classification for chatbots. In such scenarios, the user prompts the model to classify an incoming text into predefined categories. Furthermore, in real-world scenarios, it is common to encounter texts that do not fit into the predefined categories and it is unclear if current state-of-the-art LLM are capable of handling such scenarios and how they compare to existing classifiers that focus on these situations. In this paper, we propose a way to evaluate the classification performance of LLMs in an Open Set Recognition (OSR) scenario, where unseen classes can occur at inference time. The simulation consists of an empirical comparison between GPT4 and Gemini Pro, two state-of-the-art language models, and established OSR classifiers. The results would provide insights into how reliable large language models are for classification purposes and if they can replace existing OSR classifiers that typically require a decent amount of labelled data.

Keywords

Large Language Models, Open Set Recognition, Classification

1. Introduction

Since the release of ChatGPT in November 2022 [1], the adoption of Large Language Models (LLMs) in businesses has experienced significant growth [2]. Especially the ability to use natural language to interact with these models has allowed practitioners with little programming knowledge to harness the power of such systems in their daily operations. However, utilising LLMs comes at the risk of factually incorrect generated texts, also known as hallucinations [3]. Often, these hallucinations are undesired and, for example in the intent classification used in chatbot interactions, they might negatively impact customer service quality and potentially

16th ZEUS Workshop, ZEUS 2024, Ulm, Germany, February 29th - March 1st, 2024, Germany

*Corresponding author.

†These authors contributed equally.

✉ grote@fzi.de (A. Grote); anuja.hariharan@kit.edu (A. Hariharan); michael.knierim@kit.edu (M. Knierim); weinhardt@kit.edu (C. Weinhardt)

ORCID 0000-0001-7148-5138 (M. Knierim)



© 2024 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

harm the company’s reputation. This highlights the need of a robust system that is not only able to classify the customer intent correctly, but also detects out-of-distribution questions and replies accordingly. The functionality of a system that rejects out-of-distribution data points and classifies known patterns into existing categories has been widely studied under the term Open Set Recognition (OSR) [4]. In particular, deep learning based OSR classifiers, such as the OpenMax [5] or the DOC [6] algorithm, have shown an increased performance on OSR classification tasks [7]. Similarly, the zero- [8] and few-shot [9] abilities of LLMs have also been leveraged to solve these tasks. Due to the fast paced advancements in the realm of LLMs, it is unclear from a practitioner’s point of view how well state-of-the-art LLMs with zero- and few-shot strategies compare to established solutions from OSR, and how reliable they are in a production setting. This ultimately leads to the question of which approach to choose and how they compare against each other. In this work, we plan to provide not only insights into the classification accuracy but also into the ability to reject unknown instances by conducting an empirical analysis between these two research areas. We thereby give guidance for practitioners and an updated benchmark for the current state-of-the-art LLM classification performance.

2. Related Work

Generative Pre-trained Transformer (GPT) models represent a paradigm shift in Natural Language Processing (NLP) [10]. While these LLMs are typically pretrained in a self-supervised, task independent manner, they are known to be very good at NLP tasks, even without fine-tuning [11]. To use these models for classification tasks one can either fine-tune the model or use zero- and few-shot techniques for in-context-learning. Fine-tuning involves adjusting the weights of a pre-trained model for a particular task and, given a large dataset, supersedes the classification performance of zero- and few-shot strategies [12]. In contrast, zero- and few-shot learning methods utilise the capability of Large Language Models (LLMs) to categorise new data points effectively, even when they have encountered none or only a minimal number of examples from a specific class. Typically, zero- and few-shot strategies are combined with prompting strategies, such as ”Chain of Thought” [13] and ”Clue And Reasoning Prompting” [14], to further enhance the classification performance. Despite these strategies, Kocoń et al. [15] and Caruccio et al. [16] have demonstrated that the zero- and few-shot capabilities are worse than supervised machine learning models for classification tasks. In their analysis, however, they assumed a closed set scenario, which is an unrealistic assumption.

A more realistic scenario than traditional classification is Open Set Recognition [4]. It allows for unknown classes during inference and the classifier has an additional option to reject data points as unknown. If the incoming data point is not rejected as unknown, the classifier classifies the data point into a known class. Among the first OSR models were adapted Support Vector Machines [17, 18]. With the rise of neural networks, Bendale and Boulton [5] reformulated the final softmax layer to also estimate the probability of a data point being out-of-distribution. Similarly, Shu et al. [6] use a one-versus-rest classification layer to reduce the misclassifications in the open space, while Oza and Patel [19] utilise an autoencoder and its reconstruction loss to determine if a data point is novel.

3. Proposed Approach

To compare the performance of LLMs versus Open Set Recognition classifiers, we plan to setup an empirical evaluation as illustrated in Figure 1.

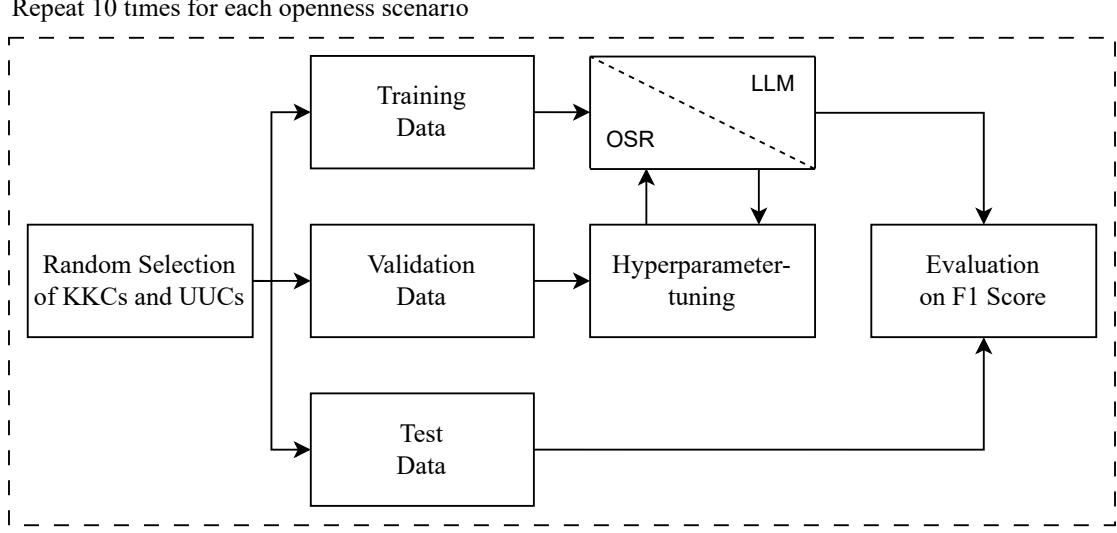


Figure 1: Machine learning workflow to compare conversational LLMs with OSR classifiers.

For our experiments, we will use four different text classification datasets. These datasets include the 20 Newsgroups dataset [20], the Yahoo! Answers dataset [21], the CLINC150 dataset [22] and the BANKING77 dataset [23]. While the first dataset consists of news articles and the second questions-answer pairs of certain categories, the last two represent intent classification tasks. All datasets have at least ten different classes or categories, based on which we will simulate an open set scenario. We follow the data splitting procedure of Geng et al. [7] to select the known and unknown classes for the open set simulation and repeat each simulation ten times to derive statistically meaningful results. Furthermore, we plan to exclude 0, 10, 20 and 30 % of all available classes from training and evaluate the classification for each scenario on the f1 score. The f1 score is a commonly used metric in classification problems, measuring the harmonic mean between precision and recall. However, in OSR scenarios, the unknown classes are typically not considered as an additional class when calculating the f1 score [7]. That is why we additionally distinguish between the f1 score classification performance on the known and unknown classes, providing further insights into the applicability of LLMs for open scenarios. In terms of conversational LLMs, we plan to use two state-of-the-art models, GPT4 [24] from OpenAI and Gemini Pro [25] from Google, with zero-shot and few-shot prompt configurations. When using a zero-shot configuration, we provide the LLM with only the category name and description, while in a few-shot setting, we also include examples of each category. Currently, it is not possible to create a custom, fine-tuned model from both of these two models. We then compare the results to the classification performance of the OpenMax [5] and DOC [6] classifiers. To speed up the training process of both OSR classifiers, we first transform the

incoming texts into meaningful embeddings using the most advanced text embedding provided by OpenAI [26] and then train a shallow neural network on the retrieved embeddings. The shallow neural network integrates either the OpenMax or the DOC architecture.

4. Conclusion

Generative AI models for text generation like ChatGPT have proven to be useful in a variety of tasks. In particular, they can be tasked to classify an incoming text into predefined categories. In this paper, we propose a study design, which compares the classification performance of state-of-the-art LLMs with existing classifiers for Open Set Recognition. The results of this study provide insights into the reliability of conversational LLMs and if they are a viable alternative for traditional classification systems.

References

- [1] OpenAI, Introducing ChatGPT, 2022. URL: <https://openai.com/blog/chatgpt>.
- [2] W. Hariri, Unlocking the Potential of ChatGPT: A Comprehensive Exploration of its Applications, Advantages, Limitations, and Future Directions in Natural Language Processing (2023). URL: <https://arxiv.org/abs/2304.02017>. doi:10.48550/ARXIV.2304.02017, publisher: arXiv Version Number: 6.
- [3] J. Li, X. Cheng, W. X. Zhao, J.-Y. Nie, J.-R. Wen, Halueval: A large-scale hallucination evaluation benchmark for large language models, in: Proceedings of the 2023 conference on empirical methods in natural language processing, 2023, pp. 6449–6464.
- [4] W. J. Scheirer, A. de Rezende Rocha, A. Sapkota, T. E. Boult, Toward Open Set Recognition, IEEE Transactions on Pattern Analysis and Machine Intelligence 35 (2013) 1757–1772. doi:10.1109/TPAMI.2012.256, conference Name: IEEE Transactions on Pattern Analysis and Machine Intelligence.
- [5] A. Bendale, T. E. Boult, Towards Open Set Deep Networks, in: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), IEEE, Las Vegas, NV, USA, 2016, pp. 1563–1572. URL: <http://ieeexplore.ieee.org/document/7780542/>. doi:10.1109/CVPR.2016.173.
- [6] L. Shu, H. Xu, B. Liu, DOC: Deep Open Classification of Text Documents, in: Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, Association for Computational Linguistics, Copenhagen, Denmark, 2017, pp. 2911–2916. URL: <https://aclanthology.org/D17-1314>. doi:10.18653/v1/D17-1314.
- [7] C. Geng, S.-J. Huang, S. Chen, Recent Advances in Open Set Recognition: A Survey, IEEE Transactions on Pattern Analysis and Machine Intelligence 43 (2021) 3614–3631. doi:10.1109/TPAMI.2020.2981604, conference Name: IEEE Transactions on Pattern Analysis and Machine Intelligence.
- [8] J. Wei, M. Bosma, V. Zhao, K. Guu, A. W. Yu, B. Lester, N. Du, A. M. Dai, Q. V. Le, Finetuned language models are zero-shot learners, in: International conference on learning representations, 2022. URL: <https://openreview.net/forum?id=gEZrGCozdqR>.
- [9] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan,

- P. Shyam, G. Sastry, A. Askeel, others, Language models are few-shot learners, *Advances in neural information processing systems* 33 (2020) 1877–1901.
- [10] T.-X. Sun, X.-Y. Liu, X.-P. Qiu, X.-J. Huang, Paradigm Shift in Natural Language Processing, *Machine Intelligence Research* 19 (2022) 169–183. URL: <https://link.springer.com/10.1007/s11633-022-1331-6>. doi:10.1007/s11633-022-1331-6.
- [11] K. S. Kalyan, A Survey of GPT-3 Family Large Language Models Including ChatGPT and GPT-4, *SSRN Electronic Journal* (2023). URL: <https://www.ssrn.com/abstract=4593895>. doi:10.2139/ssrn.4593895.
- [12] H. Liu, D. Tam, M. Muqeeth, J. Mohta, T. Huang, M. Bansal, C. A. Raffel, Few-shot parameter-efficient fine-tuning is better and cheaper than in-context learning, *Advances in Neural Information Processing Systems* 35 (2022) 1950–1965.
- [13] J. Wei, X. Wang, D. Schuurmans, M. Bosma, F. Xia, E. Chi, Q. V. Le, D. Zhou, others, Chain-of-thought prompting elicits reasoning in large language models, *Advances in Neural Information Processing Systems* 35 (2022) 24824–24837.
- [14] X. Sun, X. Li, J. Li, F. Wu, S. Guo, T. Zhang, G. Wang, Text Classification via Large Language Models (2023). URL: <https://arxiv.org/abs/2305.08377>. doi:10.48550/ARXIV.2305.08377, publisher: arXiv Version Number: 3.
- [15] J. Kocoń, I. Cichecki, O. Kaszyca, M. Kochanek, D. Szydło, J. Baran, J. Bielaniewicz, M. Gruza, A. Janz, K. Kanclerz, A. Kocoń, B. Koptyra, W. Mieleszczenko-Kowszewicz, P. Miłkowski, M. Oleksy, M. Piasecki, L. Radlinski, K. Wojtasik, S. Woźniak, P. Kazienko, ChatGPT: Jack of all trades, master of none, *Information Fusion* 99 (2023) 101861. URL: <https://linkinghub.elsevier.com/retrieve/pii/S156625352300177X>. doi:10.1016/j.inffus.2023.101861.
- [16] L. Caruccio, S. Cirillo, G. Polese, G. Solimando, S. Sundaramurthy, G. Tortora, Can ChatGPT provide intelligent diagnoses? A comparative study between predictive models and ChatGPT to define a new medical diagnostic bot, *Expert Systems with Applications* 235 (2024) 121186. URL: <https://linkinghub.elsevier.com/retrieve/pii/S0957417423016883>. doi:10.1016/j.eswa.2023.121186.
- [17] W. J. Scheirer, L. P. Jain, T. E. Boulton, Probability Models for Open Set Recognition, *IEEE Transactions on Pattern Analysis and Machine Intelligence* 36 (2014) 2317–2324. doi:10.1109/TPAMI.2014.2321392, conference Name: IEEE Transactions on Pattern Analysis and Machine Intelligence.
- [18] L. P. Jain, W. J. Scheirer, T. E. Boulton, Multi-class Open Set Recognition Using Probability of Inclusion, in: D. Fleet, T. Pajdla, B. Schiele, T. Tuytelaars (Eds.), *Computer Vision – ECCV 2014*, Lecture Notes in Computer Science, Springer International Publishing, Cham, 2014, pp. 393–409. doi:https://doi.org/10.1007/978-3-319-10578-9_26.
- [19] P. Oza, V. M. Patel, C2ae: Class conditioned auto-encoder for open-set recognition, in: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 2307–2316.
- [20] D. Lewis, Reuters-21578 text categorization collection, 1997. Text.howpublished: UCI Machine Learning Repository.
- [21] X. Zhang, J. Zhao, Y. LeCun, Character-level convolutional networks for text classification, *Advances in neural information processing systems* 28 (2015).
- [22] S. Larson, A. Mahendran, J. J. Peper, C. Clarke, A. Lee, P. Hill, J. K. Kummerfeld, K. Leach, M. A. Laurenzano, L. Tang, J. Mars, An evaluation dataset for intent classification and

- out-of-scope prediction, in: K. Inui, J. Jiang, V. Ng, X. Wan (Eds.), Proceedings of the 2019 conference on empirical methods in natural language processing and the 9th international joint conference on natural language processing (EMNLP-IJCNLP), Association for Computational Linguistics, Hong Kong, China, 2019, pp. 1311–1316. URL: <https://aclanthology.org/D19-1131>. doi:10.18653/v1/D19-1131.
- [23] I. Casanueva, T. Temčinas, D. Gerz, M. Henderson, I. Vulić, Efficient intent detection with dual sentence encoders, in: T.-H. Wen, A. Celikyilmaz, Z. Yu, A. Papangelis, M. Eric, A. Kumar, I. Casanueva, R. Shah (Eds.), Proceedings of the 2nd workshop on natural language processing for conversational AI, Association for Computational Linguistics, Online, 2020, pp. 38–45. URL: <https://aclanthology.org/2020.nlp4convai-1.5>. doi:10.18653/v1/2020.nlp4convai-1.5.
- [24] OpenAI, GPT-4 Technical Report (2023). URL: <https://arxiv.org/abs/2303.08774>. doi:10.48550/ARXIV.2303.08774, publisher: arXiv tex.version: 4.
- [25] Gemini Team, Gemini: A Family of Highly Capable Multimodal Models (2023). URL: <https://arxiv.org/abs/2312.11805>. doi:10.48550/ARXIV.2312.11805, publisher: arXiv tex.version: 1.
- [26] OpenAI, New and improved embedding model, 2022. URL: <https://openai.com/blog/new-and-improved-embedding-model>.

A BPMN Profile for Test Case Execution Visualization

Daniel Lübke^{1,2,*}

¹Digital Solution Architecture GmbH, Hannover, Germany

²Leibniz Universität Hannover, FG Software Engineering, Hannover, Germany

Abstract

Testing executable business processes, e.g., developed in BPMN 2.0 or WS-BPEL, is an important task within development projects. However, this task is labor-intensive due to the amount and scale of interactions of a test framework (e.g., BPELUnit) with the process under test. We want to help testers to debug test case failures by visualizing the test run information as a BPMN diagram. We developed a visual, BPMN-based notation for visualizing test runs, implemented a generation from BPELUnit test suites and execution logs to the newly defined format, and applied this in an industrial project. While no thorough validation has yet been performed, early results indicate better understandability of our notation compared to raw test logs.

Keywords

BPMN, Test Case, Visualization, Profile, BPELUnit

1. Motivation

Testing executable business processes is an important task in many digitization projects: errors in critical business processes are a huge risk for reputation and income of organizations. Therefore, unit testing, e.g., with BPELUnit [1], is often one activity to improve functional quality.

However, analyzing failing unit tests in executable business processes is often difficult, because they integrate many partners that potentially are performing activities in parallel. As such, test case logs containing successfully executed and failed activities alongside passed input and output messages become large and are hard to analyze. While test frameworks indicate which activity failed, further analysis is often required to pinpoint the underlying problem. This is especially true, when test cases are generated, e.g., by using combinatorial test design [2, 3] because errors in the test case configuration are more likely than with manually written tests.

We developed an approach to generate BPMN models, which help developers and testers to easier analyze failed test cases. It builds upon a test case subset for BPMN [4] and uses colors and documentation to make test case execution information better accessible.

This paper starts with a presentation of related work (section 2). Afterwards, the generation approach and underlying meta models are presented in section 3, which are implemented in a prototype application (section 4). At the end, a small validation in an industry project is

ZEUS'2024: 16th Central European Workshop on Services and their Composition, February 29 – March 01, 2024, Ulm, Germany

*Corresponding author.

✉ daniel.luebke@digital-solution-architecture.com (D. Lübke)

🌐 <https://www.digital-solution-architecture.com> (D. Lübke)

🆔 0000-0002-1557-8804 (D. Lübke)



© 2024 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

CEUR Workshop Proceedings (CEUR-WS.org)

S. Böhm and D. Lübke (Eds.): 16th ZEUS Workshop, ZEUS 2024, Ulm, Germany, 29 February–1 March 2024, published at <http://ceur-ws.org>

presented in section 5. Finally, conclusions are drawn and outlook on future work is made (section 6).

2. Related Work

Visualizing test cases has been a research area of interest for quite some time. For example, Cornelissen et al. [5] visualized execution traces of unit tests with UML Sequence Diagrams to help developers better understand the inner workings of software systems.

But visualization is not only concerned with the test execution per se but also with its results. For example, Opmanis et al. [6] plotted test result data over time, and Dzidic [7] performed a case study visualizing metrics of software tests as dashboards in the financial sector.

However, more similar to the goals of this research, studies have been concerned with visualizations to better detect faults: Jones et al. [8, 9] used coverage information for visualizing which parts of the software have been covered by which test cases. Similar work has been done by Koochakzadeh & Garousi [10]. They visualized dependencies in the program under test on class dependencies and implemented this as an Eclipse plug-in. Wes et al. [11] plotted test executions as Multivariate Visualizations to cluster test cases and ease reasoning about successfully implemented scenarios.

Existing tooling can also be used to present more data to testers. For example, there is a tool for the Camunda BPMN engine ¹ to visualize test coverage metrics as a graphical overlay over the process.

3. Definition, Meta Model and Generation

A BPMN Test Execution Visualization is a BPMN model that shows all activities executed or supposed to be executed during a test run, which conforms to the following constraints (derived from the constraints for model-driven testing of executable business processes [12]):

1. It shows test case execution in a BPMN model,
2. it uses uncollapsed pools for each logical party in the test case plus one collapsed pool for the process under test,
3. it only uses the following BPMN elements: tasks (user, service), events (catch/throw message, timer), and gateways (parallel),
4. its message flows must originate or terminate at the collapsed pool for the process under test,
5. it shows the status of each activity by color: red (error), yellow (interrupted), and green (okay),
6. it contains additional diagnostic information in the BPMN elements' documentations, e.g., received and sent message payload.

Figure 1 shows the relevant excerpts of the BPELUnit, BPELUnit log, and BPMN meta models required for generation of the BPMN models: A test case execution is mapped to one BPMN

¹<https://github.com/camunda-community-hub/camunda-process-test-coverage>

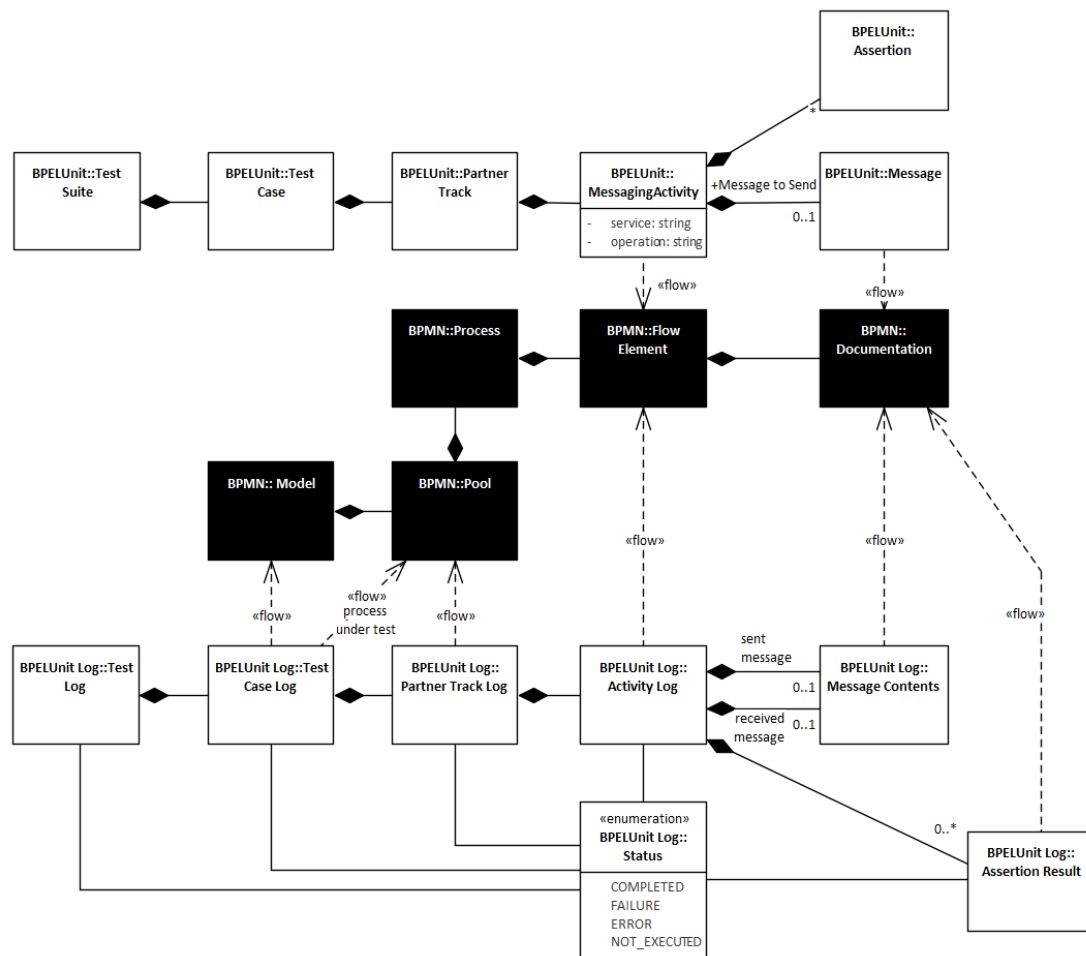


Figure 1: Meta Model and Generational Data Flows for Test Suite Information

collaboration diagram. Every mocked service, i.e., partner is represented by a pool. The process under test itself is represented by a collapsed pool. This means that this visualization technique will treat the process under test as a black box in contrast to other visualizations, e.g., those for test coverage.

If a mock sends a one-way message, a message throw event is added to its pool, if it receives a one-way message, a message catch event is added. If it initiates a two-way message exchange, a service task is added, and if it waits for a two-way message exchange a message catch event is added followed by a message throw event. If the mock waits for a specified duration, a timer event is added, and parallel branches within a partner is represented with parallel gateways. Because BPELUnit is block-structured, it is easy to also add the merging parallel gateway.

4. Prototype Implementation

Within this section we shortly describe some design considerations for the prototype application, which contains as much functionality as required for trying it in the industry project (see section 5). The prototype is available at <https://github.com/dluebke/bpelunit-viz>.

4.1. Targeted Tool Chain

Because we want to use colors for denoting tasks' error status, we were required to settle on one tool, because colors are not standardized in the diagram interchange format of the BPMN 2.0 specification. We settled on the Camunda Modeler because it allows us to show the documentation in a comparatively large text area in a side pane. Thus, developers can easily select BPMN elements of interest and directly see exchanged information.

4.2. XML Processing

We used Java's DOM and XPath API to read elements and attribute values from the BPELUnit log. We have planned to add the processing of the BPELUnit test suite itself as described above. However, the prototype does not yet resolve the original operation name as foreseen in the concept.

Internally, all metamodels are represented by model classes (BPELUnit, BPELUnit Log, BPMN 2.0). BPMN 2.0 can be serialized via custom serializers for the model classes.

4.3. Layouting

The resulting layout of a test case visualizaton is illustrated in Figure 2.

Because developers want to use the generated BPMN diagrams without further effort, we were required to layout the diagrams as well. We simply layouted all flow nodes in a pool horizontally. This approach works reasonably well because the diagrams contain no loops, few gateways, and are mostly sequential.

To reduce the average length of message-flows and number of line-crossings and thereby making the diagram easier to comprehend, we ordered the pools based on the number of contained flow nodes (events, tasks and gateways): The pool representing the process under test is placed in the middle. The pool containing the most flow nodes is placed on top of it, the pool with second most flow nodes is placed below it. The ordering progresses by adding the pools descendingly sorted by the number of contained flow nodes on top and below already layouted pools.

Some tuning was required due to the the large amount of message-flows and pools: Because nearly all task had straight vertical message flows to the process-under-test pool, most message flows overlapped. We resolved this problem by slightly offsetting the horizontal starting point within each pool.

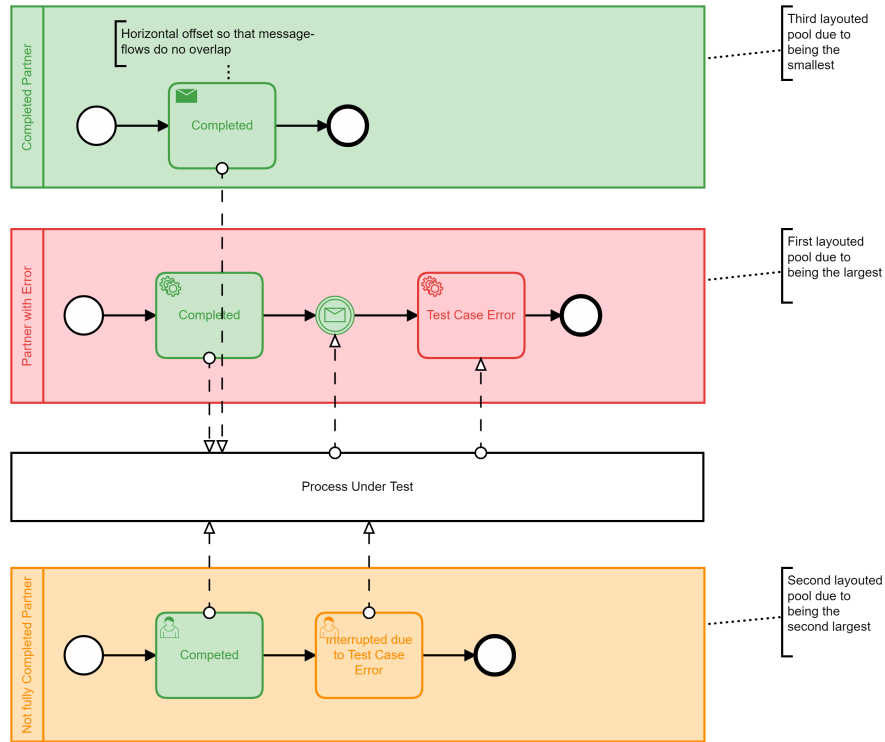


Figure 2: Example and Explanation of the Layout

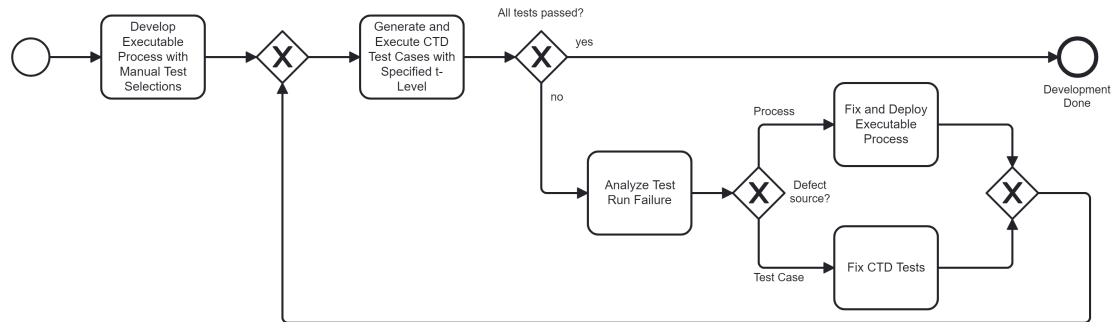


Figure 3: Development Process in the Industrial Project using CTD Test Generation

5. Evaluation: Application in Industry Project

For addressing quality concerns, Terravis [13] — a Swiss large-scale process integration platform — has invested much in efforts for improving testing. Due to many changes to its BPEL processes [14], extensive unit and integration tests based on BPELUnit [1, 15] have been developed. Especially, with the generation of such tests with Combinatorial Test Design (CTD) [2, 3] many

tests need to be debugged until the development of the generated test suites is completed. Similarly, after extending processes newly introduced bugs needed to be identified and tests adjusted.

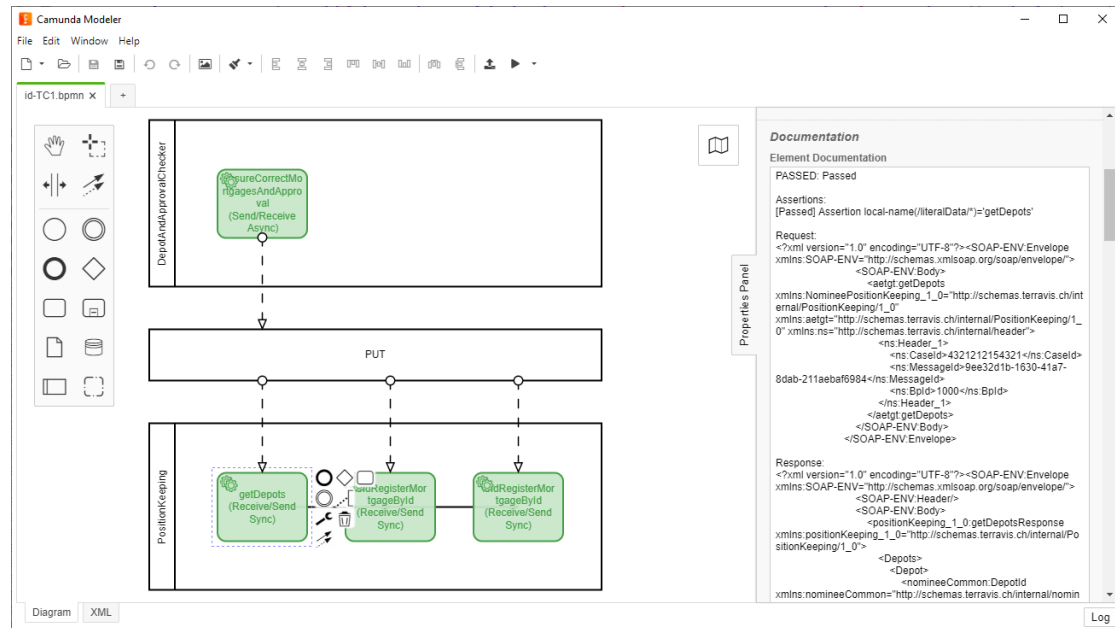


Figure 4: Generated BPMN test case visualization shown in Camunda Modeler with XML payload and assertion information in the right-hand property pane.

The project has so far used the visualization for 25 of its processes that are covered by CTD tests executed with BPELUnit. The largest BPELUnit log file is 48 MB large containing 328 test cases. The log file size is mainly influenced by the (large) XML messages being exchanged in the covered process. The visualization thus reduces the cognitive load required by developers because it filters out payload information initially although allowing users to access it via the elements' documentation as shown in Figure 4.

Feedback by developers was positive and they indicated that the visualization helped them to diagnose problems. However, further validation is required.

6. Conclusion and Outlook

In this paper we presented a way to visualize test cases for executable business processes and their executions as BPMN models. The proposed algorithm has been implemented in a tool and successfully applied in an industrial case. Consequently, BPMN can be used as a single language to model the business processes, model or visualize test cases (depending on the testing approaches), and visualize test execution results.

In future work we like to further validate the approach by conducting surveys and experiments for comparing log-based analysis with analysis performed using our models with regard to

effectiveness of finding and fixing problems in test cases and executable business processes.

We also made the prototype implementation publicly available so that everyone can use it and perform such studies with us or independently. We are happy to collaborate with others concerning research into optimizing the testing of executable business processes!

Acknowledgments

We like to thank all team members of the industrial case project Terravis for their participation.

References

- [1] P. Mayer, D. Lübke, Towards a BPEL Unit Testing Framework, in: TAV-WEB '06: Proceedings of the 2006 Workshop on Testing, Analysis, and Verification of Web Services and Applications, Portland, USA, ACM Press, New York, NY, USA, 2006, pp. 33–42. URL: <http://portal.acm.org/affiliated/citation.cfm?id=1145718.1145723&coll=ACM&dl=ACM&type=series&idx=1145718&part=Proceedings&WantType=Proceedings&title=International%20Symposium%20on%20Software%20Testing%20and%20Analysis&CFID=1483183&CFTOKEN=32880799#>. doi:<http://doi.acm.org/10.1145/1145718.1145723>.
- [2] T. Schnelle, D. Lübke, Towards the generation of test cases for executable business processes from classification trees, in: Proceedings of the 9th Central European Workshop on Services and their Composition (ZEUS) 2017, 2017, pp. 15–22.
- [3] D. Lübke, J. Greenyer, D. Vatlin, Effectiveness of Combinatorial Test Design with Executable Business Processes, in: D. Lübke, C. Pautasso (Eds.), Empirical Studies on the Development of Executable Business Processes, Springer, 2019, pp. 187–207.
- [4] D. Lübke, T. van Lessen, Modeling Test Cases in BPMN for Behavior-Driven Development (Extended Abstract), in: Proceedings of the EMISA Workshop 2016, 2016.
- [5] B. Cornelissen, A. van Deursen, L. Moonen, A. Zaidman, Visualizing testsuites to aid in software understanding, in: 11th European Conference on Software Maintenance and Reengineering (CSMR'07), 2007, pp. 213–222. doi:[10.1109/CSMR.2007.54](https://doi.org/10.1109/CSMR.2007.54).
- [6] R. Opmanis, P. Kikusts, M. Opmanis, Visualization of large-scale application testing results, Baltic Journal of Modern Computing 4 (2016) 34.
- [7] E. Dzidic, Data Visualization of Software Test Results : A Financial Technology Case Study, Master's thesis, KTH, School of Electrical Engineering and Computer Science (EECS), 2023.
- [8] J. A. Jones, M. J. Harrold, J. Stasko, Visualization of test information to assist fault localization, in: Proceedings of the 24th International Conference on Software Engineering, ICSE '02, Association for Computing Machinery, New York, NY, USA, 2002, p. 467–477. URL: <https://doi.org/10.1145/581339.581397>. doi:[10.1145/581339.581397](https://doi.org/10.1145/581339.581397).
- [9] J. Jones, Fault localization using visualization of test information, in: Proceedings. 26th International Conference on Software Engineering, 2004, pp. 54–56. doi:[10.1109/ICSE.2004.1317420](https://doi.org/10.1109/ICSE.2004.1317420).
- [10] N. Koochakzadeh, V. Garousi, Tecrevis: A tool for test coverage and test redundancy

- visualization, in: L. Bottaci, G. Fraser (Eds.), *Testing – Practice and Research Techniques*, Springer Berlin Heidelberg, Berlin, Heidelberg, 2010, pp. 129–136.
- [11] W. Masri, R. A. Assi, F. Zaraket, N. Fatairi, Enhancing fault localization via multivariate visualization, in: *2012 IEEE Fifth International Conference on Software Testing, Verification and Validation*, 2012, pp. 737–741. doi:10.1109/ICST.2012.166.
 - [12] D. Lübke, T. van Lessen, BPMN-based Model-Driven Testing of Service-based Processes, in: *Business Process Modeling, Development, and Support 2017*, 2017.
 - [13] W. Berli, D. Lübke, W. Möckli, Terravis – Large Scale Business Process Integration between Public and Private Partners, in: E. Plödereder, L. Grunske, E. Schneider, D. Ull (Eds.), *Lecture Notes in Informatics (LNI), Proceedings INFORMATIK 2014*, volume P-232, Gesellschaft für Informatik e.V., Gesellschaft für Informatik e.V., 2014, pp. 1075–1090.
 - [14] D. Lübke, Using Metric Time Lines for Identifying Architecture Shortcomings in Process Execution Architectures, in: *Software Architecture and Metrics (SAM)*, 2015 IEEE/ACM 2nd International Workshop on, IEEE, 2015, pp. 55–58.
 - [15] D. Lübke, *Test and Analysis of Service-Oriented Systems*, Springer, 2007, pp. 149–171.

Towards Robustness of IoT devices in BPMNE4IoT

Pascal Schiessle¹, Yusuf Kirikkayis¹ and Manfred Reichert¹

¹*Institute of Databases and Information Systems, Ulm University, 89081, Germany*

Abstract

Integrating the Internet of Things into Business Process Management has gained traction for several years to improve smart applications (e.g., smart homes, Industry 4.0). Different frameworks have been proposed to integrate IoT in all stages of the BPM lifecycle. However, current frameworks lack proper support for dealing with issues related to error handling, like sensor faults, fallback strategies, and redundancies. Therefore, it has to be assessed to what extent error handling should be included in the abstraction layer of IoT-aware processes to increase both robustness and reliability. This paper discusses five research questions on the challenges regarding the integration of error handling in existing frameworks throughout the BPM lifecycle.

Keywords

BPM, IoT, Error handling, IoT-aware business processes


1. Introduction


The Internet of Things (IoT) is an important part of digitization in the personal space, like smart homes or connected cars, as well as in the industrial space with Industry 4.0 and smart logistics. These complex systems combine a wide range of IoT devices to form highly automated systems, relying on sensors for collecting data from the physical world (e.g., temperature, air quality, humidity) as well as actuators (e.g., motors), for altering the physical world by either mechanical motion (e.g., movement or rotation) or environmental changes (e.g., light, humidity, pressure) [1, 2]. Utilizing data generated from IoT devices can improve IoT-aware business processes (BP) [3, 4]. Integrating IoT devices into Business Process Management (BPM) systems shows the potential of merging both areas [5, 6]. BPM is a paradigm to model, implement, execute, monitor, and analyze business processes [7]. Providing support for IoT capabilities in BPM systems can enhance the awareness of the physical world by connecting the digital to the physical world via IoT devices along the entire BPM life-cycle. Besides optimized business processes and better decision-making, the added IoT capabilities can be used to automate physical tasks, like turning on/off lights or tracking the location of physical devices [6]. Modeling IoT-aware business processes has been extensively studied in literature [2]. In this paper, we consider five research questions to assess the extent of IoT-related errors in an IoT-aware business process. Besides understanding common IoT errors, the goal is to investigate different strategies on how to include error handling into IoT-aware business processes to provide a set of recovery

16th Central European Workshop on Services and their Composition, 2024, Ulm, Germany

✉ pascal.schiessle@uni-ulm.de (P. Schiessle); yusuf.kirikkayis@uni-ulm.de (Y. Kirikkayis);
manfred.reichert@uni-ulm.de (M. Reichert)

ORCID 0000-0002-0793-4178 (P. Schiessle); 0000-0001-6536-0785 (Y. Kirikkayis); 0000-0003-2536-4153 (M. Reichert)

 © 2024 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

 CEUR Workshop Proceedings (CEUR-WS.org)

S. Böhm and D. Lübke (Eds.): 16th ZEUS Workshop, ZEUS 2024, Ulm, Germany, 29 February–1 March 2024, published at <http://ceur-ws.org>

strategies, and to study the effects these strategies have on the business process. We developed a holistic framework - BPMNE4IoT - for modeling, executing, monitoring, and logging IoT-aware processes [5]. The framework can already detect faulty sensors and actuators and visualize them in the BPMNE4IoT monitoring tool. However, the IoT-aware processes that can be built with our framework have not yet matched the needed resilience regarding IoT errors. More research is needed to not only detect, but also to cope with a wide range of errors, like failing sensors, sensor drifts, and stuck actuators. Considering the research questions presented in this paper, we plan to investigate possible improvements to the BPMNE4IoT framework to increase its resilience.

The paper is structured as follows. Section 2 outlines the overall problem statement and provides a real-world example. In Section 3, we define a set of research questions, that need to be answered in the context of the above problem statement. In 4, we present related work. Finally, Section 5, presents a summary and an outlook.

2. Problem Statement

Error handling is a multi-phase problem. Besides modeling, systems should be able to cope with errors during IoT-aware process execution. Here, we focus on approaches based on the combination of BPMN with IoT, however, other modeling approaches (e.g., Petri-nets or state machines) do exist. BPMN 2.0 does not provide the tools and notation to design IoT-aware processes [3, 5]. By introducing a holistic framework, i.e. BPMNE4IoT for modeling, executing, monitoring, and logging IoT-aware processes, the involved IoT devices can be incorporated with the business process. However, the framework contains a limited set of features to increase the robustness and reliability of IoT-related errors. We want to illustrate this based on the BPMN model, depicted in Figure 1.

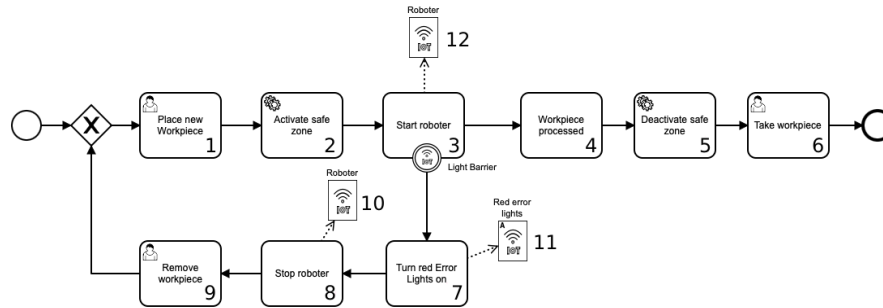


Figure 1: An example of an IoT-aware business process

In this scenario, different IoT-related errors may occur. To illustrate these errors, we focus on two scenarios—a faulty robot, which is an IoT object, comprising different types of sensors and actuators, and a faulty light barrier. For example, the robot might malfunction and refuse to start in step 3 of the IoT-aware process depicted in Figure 1. Currently, the process will be stuck forever in this step until a worker intervenes and manually handles the malfunction, alerted by the monitoring. Depending on the specific robot, calibrating or restarting the robot may fix the

error, however, this is often not supported by BPM systems, like BPMNE4IoT, which we use as an example. Here, an IoT device could include strategies to handle common errors automatically and report their state (e.g., re-calibration, restarting). Depending on the domain, this might decrease the amount of human intervention and the time consumed to treat the malfunction.

Note that there might also be life-threatening errors. In a second scenario, we consider the effects of a malfunctioning light barrier. When the robot is working on a piece and is standing still, waiting for a workpiece to cool down, the worker might believe the robot has finished its task and walk into the security zone. If the light barrier does not trigger, the robot is not aware of the human in range which could harm him significantly.

From these two scenarios we may conclude, robustness and reliability concerns need to be investigated for IoT-aware processes. Moreover, process execution opportunities to increase safety, reduce downtime, and increase availability.

3. Research Questions

Although BMPNE4IoT [5] offers a promising approach to model IoT devices in IoT-aware business processes, further research is needed. Currently, faulty sensors can be represented in the execution stage of the BPMN4IoT pipeline. However, BPMNE4IoT has not yet considered robustness and reliability issues regarding IoT devices. We plan to investigate if including IoT devices down to the hardware is the appropriate abstraction level for IoT-related error handling. Furthermore, we need to assess in future work, if changes to the handling of IoT devices in BPMNE4IoT are necessary to include and aggregate reliability information. The following research questions (RQs) must be explored to gain a deeper understanding.

RQ1: How can the appropriate abstraction level for a given IoT device be determined when integrating into IoT-aware BPs?

In BPMNE4IoT, devices can be represented either as a single device or as a group of devices or as a complex object with multiple types of devices. Defining error handling techniques is influenced by the access to the device. Low-level access to IoT devices in IoT-aware BPs enables more complex error handling by the process, however, this is not always desired [8]. Developing metrics to determine the appropriate abstraction level on a process level should be pursued.

RQ2: Can monitoring on an IoT device level contribute to identifying error-prone IoT devices across processes?

Monitoring IoT devices on a process level may provide domain experts with more detailed insights into which situations specific IoT devices produce errors. Currently, IoT devices in BPMNE4IoT are represented multiple times in the modeling stage (10 and 12 in 1), even though they are the same physical device. Connecting data from different IoT-aware BPs by a shared device pool may provide a better understanding of the reliability of specific IoT devices.

RQ3: Is the added complexity of IoT-related error handling over native BPMN 2.0 error handling techniques justifiable to enhance expressiveness?

Including IoT-related error handling in IoT-aware BPs could yield the benefit of a more descriptive modeling stage. Based on RQ1, implementation-specific logic can be pushed to the execution or device level by choosing a higher abstraction degree for the IoT device in the modeling stage.

RQ4: Which patterns for dealing with self-healing and recovery can be realized in a framework like BPMNE4IoT?

Currently, failing IoT devices lead to a stuck or terminated BP. Patterns to include self-healing capabilities, providing the ability to detect and compensate for IoT-related errors, are desired in BPMNE4IoT. It should be investigated which techniques (e.g., MAPE-K feedback loops) for dealing with self-healing can be incorporated into BPMNE4IoT.

RQ5: How can IoT device redundancies be integrated into IoT-aware BPs?

Especially in critical applications, redundant IoT devices are used to ensure the reliability of the system, however, they come at a price (e.g., device cost, and complexity). It should be investigated if IoT device redundancies should be incorporated on a BP level. Defining redundancies, receiving detailed insights while monitoring, and refining processes based on redundancy data linked to specific tasks could lead to an easier optimization of processes. However, it is open for debate which of these abilities should be controllable on a process level, especially due to redundancies being highly dependent on the context, software, and hardware.

4. Related Work

Several approaches have been proposed to include IoT-awareness into the BPM lifecycle [5, 9, 10, 2, 11, 12, 13]. uBPMN extends BPMN 2.0 by defining five task types, sensor task, reader task, image task, audio task, and collector task, to represent incoming and outgoing data streams as well as their respective context [14]. BPMNE4WSN specifically aligns with Wireless Sensor Networks (WSNs) [15] by introducing WSN task, WSN pool, and WSN performance annotations. BPMNE4CPS extends the BPMN 2.0 service task to provide dedicated support for web services, embedded services, and cloud services [16]. While these approaches enable IoT-aware processes, they do not specifically focus on robustness or reliability issues. Outside the BPM context, work was conducted to investigate IoT robustness and reliability [17, 18, 19, 20]. [21] provides a methodology for measuring task reliability by block reduction. For IoT-aware processes, [22] and [23] investigated the reliability of the BPMN extension relyBPMN when associating human and IoT-related information resource information for given tasks. [24] and [25] investigated the use of self-adaptive processes in distributed workflows. By using a modified MAPE-K feedback loop, known from self-driving frameworks to detect and compensate for errors in distributed processes. Finally, [26] proposes an approach to fault management for distributed sensor networks on the edge.

5. Conclusion

As discussed, robust and reliable IoT-aware processes should be pursued. As shown in two scenarios, handling IoT-related errors not only increases availability and decreases cost, but also increases safety by managing safety-related topics (e.g., redundancy). The paper introduced five research questions to assess and improve the extent of error handling of IoT-aware business processes in BPMNE4IoT. Existing approaches, like relyBPMN, do improve error handling in IoT-aware BPs for specific scenarios, however, a holistic approach is still missing.

References

- [1] M. Bauer, N. Bui, J. De Loof, C. Magerkurth, A. Nettsträter, J. Stefa, J. W. Walewski, Iot reference model, *Enabling Things to Talk: Designing IoT solutions with the IoT architectural reference model* (2013) 113–162. doi:10.1007/978-3-642-40403-0_7.
- [2] P. Valderas, V. Torres, E. Serral, Modelling and executing IoT-enhanced business processes through BPMN and microservices, *Journal of Systems and Software* 184 (2022) 111139. doi:10.1016/j.jss.2021.111139.
- [3] F. Martins, D. Domingos, Modelling IoT behaviour within BPMN Business Processes, *Procedia Computer Science* 121 (2017) 1014–1022. doi:10.1016/j.procs.2017.11.131, cENTERIS/ProjMAN/HCist 2017.
- [4] Y. Kirikkayis, F. Gallik, M. Reichert, IoTDM4BPMN: An IoT-Enhanced Decision Making Framework for BPMN 2.0, in: *2022 International Conference on Service Science (ICSS)*, 2022, pp. 88–95. doi:10.1109/ICSS55994.2022.00022.
- [5] Y. Kirikkayis, F. Gallik, M. Winter, M. Reichert, BPMNE4IoT: A Framework for Modeling, Executing and Monitoring IoT-Driven Processes, *Future Internet* 15 (2023). doi:10.3390/fi15030090.
- [6] C. Janiesch, A. Koschmider, M. Mecella, B. Weber, A. Burattin, C. Di Ciccio, G. Fortino, A. Gal, U. Kannengiesser, F. Leotta, F. Mannhardt, A. Marrella, J. Mendling, A. Oberweis, M. Reichert, S. Rinderle-Ma, E. Serral, W. Song, J. Su, V. Torres, M. Weidlich, M. Weske, L. Zhang, The Internet of Things Meets Business Process Management: A Manifesto, *IEEE Systems, Man, and Cybernetics Magazine* 6 (2020) 34–44. doi:10.1109/MSMC.2020.3003135.
- [7] M. Dumas, M. La Rosa, J. Mendling, H. A. Reijers, et al., *Fundamentals of business process management*, volume 2, Springer, 2018. doi:10.1007/978-3-642-33143-5.
- [8] O. Kopp, F. Leymann, D. Wutke, Fault handling in the web service stack, in: *8th Int. Conf. on Service-Oriented Computing -- ICSOC 2010*, Springer-Verlag, 2010. doi:10.1007/978-3-642-17358-5_21.
- [9] D. Domingos, F. Martins, Using BPMN to model Internet of Things behavior within business process, *International Journal of Information Systems and Project Management* 5 (2017) 39–51. doi:10.12821/ijispm050403.
- [10] S. Meyer, A. Ruppen, L. Hilty, "The Things of the Internet of Things in BPMN", in: A. Persson, J. Stirna (Eds.), *Advanced Information Systems Engineering Workshops*, Springer International Publishing, Cham, 2015, pp. 285–297. doi:10.1007/978-3-319-19243-7_27.
- [11] S. Schöning, L. Ackermann, S. Jablonski, A. Ermer, IoT meets BPM: a bidirectional communication architecture for IoT-aware process execution, *Software and Systems Modeling* 19 (2020) 1443–1459. doi:10.1007/s10270-020-00785-7.
- [12] C. Stoiber, S. Schöning, Patterns for IoT-based Business Process Improvements: Developing a Metamodel., in: *ICEIS* (1), 2022, pp. 655–666. doi:10.5220/0011059300003179.
- [13] Y. Cheng, S. Zhao, B. Cheng, X. Chen, J. Chen, Modeling and deploying IoT-aware business process applications in sensor networks, *Sensors* 19 (2019) 111.
- [14] A. Yousfi, C. Bauer, R. Saidi, A. K. Dey, uBPMN: A BPMN extension for modeling ubiquitous business processes, *Information and Software Technology* 74 (2016) 55–68. doi:10.1016/j.infsof.2016.02.002.

- [15] C. T. Sungur, P. Spiess, N. Oertel, O. Kopp, Extending bpmn for wireless sensor networks, in: 2013 IEEE 15th Conference on Business Informatics, IEEE, 2013, pp. 109–116. doi:10.1109/cbi.2013.24.
- [16] I. Graja, S. Kallel, N. Guermouche, A. H. Kacem, BPMN4CPS: A BPMN extension for modeling cyber-physical systems, in: 2016 IEEE 25th International Conference on Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE), IEEE, 2016, pp. 152–157. doi:10.1109/wetice.2016.41.
- [17] S. Sarkar, Chapter 11 - Internet of Things—robustness and reliability, in: R. Buyya, A. Vahid Dastjerdi (Eds.), Internet of Things, Morgan Kaufmann, 2016, pp. 201–218. doi:10.1016/B978-0-12-805395-9.00011-3.
- [18] W. Z. Khan, M. Y. Aalsalem, M. K. Khan, M. S. Hossain, M. Atiquzzaman, A reliable Internet of Things based architecture for oil and gas industry, in: 2017 19th International Conference on Advanced Communication Technology (ICACT), 2017, pp. 705–710. doi:10.23919/ICACT.2017.7890184.
- [19] P. Fraga-Lamas, T. M. Fernández-Caramés, M. Suárez-Albela, L. Castedo, M. González-López, A review on internet of things for defense and public safety, Sensors 16 (2016) 1644. doi:10.3390/s16101644.
- [20] P. K. Illa, N. Padhi, Practical Guide to Smart Factory Transition Using IoT, Big Data and Edge Analytics, IEEE Access 6 (2018) 55162–55170. doi:10.1109/ACCESS.2018.2872799.
- [21] A. Respício, D. Domingos, Reliability of BPMN Business Processes, Procedia Computer Science 64 (2015) 643–650. doi:10.1016/j.procs.2015.08.578, conference on ENTERprise Information Systems/International Conference on Project MANagement/Conference on Health and Social Care Information Systems and Technologies, CENTERIS/ProjMAN / HCist 2015 October 7-9, 2015.
- [22] D. Domingos, A. Respício, R. Martinho, Using resource reliability in bpmn processes, Procedia Computer Science 100 (2016) 1280–1288. doi:10.1016/j.procs.2016.09.243.
- [23] R. Martinho, D. Domingos, A. Respício, Evaluating the Reliability of Ambient-Assisted Living Business Processes., in: ICEIS (2), 2016, pp. 528–536. doi:10.5220/0005917005280536.
- [24] R. Seiger, S. Herrmann, U. Aßmann, Self-Healing for Distributed Workflows in the Internet of Things, in: 2017 IEEE International Conference on Software Architecture Workshops (ICSAW), 2017, pp. 72–79. doi:10.1109/ICSAW.2017.36.
- [25] R. Seiger, S. Huber, P. Heisig, U. Assmann, Enabling self-adaptive workflows for cyber-physical systems, in: Business-Process and Information Systems Modeling: 17th International Conference, BPMDS 2016, 21st International Conference, EMMSAD 2016, Held at CAiSE 2016, Springer, 2016, pp. 3–17. doi:10.1007/978-3-319-39429-9_1.
- [26] G. Di Modica, S. Gulino, O. Tomarchio, IoT fault management in cloud/fog environments, in: Proceedings of the 9th International Conference on the Internet of Things, 2019, pp. 1–4. doi:10.1145/3365871.3365882.

Predictive Process Monitoring: An Implementation and Comparison of Student Performance Prediction

Lisa Arnold^{1,*}, Marius Breitmayer¹ and Manfred Reichert¹

¹*Institute of Databases and Information Systems, Ulm University, Germany*

Abstract

Predictive monitoring can support students during the semester by motivating them if they are not performing well enough in a lecture or exercise. Furthermore, supervisor can create additional exercise sheets or can adapt their lectures and exercises to the needs (i.e. knowledge gaps) of the students. To realise this, three different regression algorithms (i.e. Neuronal Networks, Decision Trees, and Random Forest) are implemented to continuously predict further exercise points and the final grade during a semester. These algorithms are trained and tested based on student points and grades from previous semesters. A total of 17,136 predictions were determined, analysed, and compared. In several exercise sheets, all algorithms achieve between 91% and 96% correct predictions with a variance of 10% (i.e. up to 2.5 points). With 15% variance, 98.5% corrected results are possible. The prediction of grades with a variance of 0.3 (i.e. one grade level) with the Decision Tree and the Random Forest only achieves 32% to 35% correctness.

Keywords

predictive process monitoring, predictions, machine learning algorithms, business processes

1. Introduction

Some students are overwhelmed by the number of lectures and exercises at the beginning of their time at a university. Others underestimate the exams and do not invest enough effort and performance in the exercises and learning during the semester. *Predictive Process Monitoring* aims to predicting the results (e.g. grades) of a running and uncompleted business process (e.g. lecture during a semester) [1]. To motivate students to prepare for the upcoming exams during the semester, perform the exercises, and study the lecture material continuously we will predict their grades and points of upcoming exercises based on the already gained exercise points. Moreover, a supervisor of a lecture may provide additional exercises to support the students, when the algorithm predicts worse grades. For this, the regression algorithms Neuronal Networks, Decision Trees, and Random Forest are implemented and tested. To train and test the algorithm a data set of 334 students (i.e. 266 for training and 68 for testing) including their exercise points and grades is used. In total, 17,136 predictions are determined, evaluated, and compared with each other.

16th Central European Workshop on Services and their Composition, February 29 – March 1, 89081, Ulm, Germany


*Corresponding author.

✉ lisa.arnold@uni-ulm.de (L. Arnold); marius.breitmayer@uni-ulm.de (M. Breitmayer); manfred.reichert@uni-ulm.de (M. Reichert)

🆔 0000-0002-2358-2571 (L. Arnold); 0000-0003-1572-4573 (M. Breitmayer); 0000-0003-2536-4153 (M. Reichert)



© 2022 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

 CEUR Workshop Proceedings (CEUR-WS.org)

S. Böhm and D. Lübke (Eds.): 16th ZEUS Workshop, ZEUS 2024, Ulm, Germany, 29 February–1 March 2024, published at <http://ceur-ws.org>

The remainder of this paper is structured as follows. Section 2 explains the basics of the three machine learning algorithms. In Section 3 the initialisation and implementation of the algorithms is presented. Section 4 evaluates the results of the prediction and compares them with each other. Related work is discussed in Section 5 and Section 6 concludes the paper.

2. Background

In the context of this paper, the following three machine learning algorithms for the continuous prediction of grades and exercise points for students during a semester are implemented. The fundamentals of the three algorithms are explained in this section.

The **Neuronal Network** [2, 3] comprises a large number of artificial neurons and their connections to each other. These are designed to mimic the function of the human brain. The Neuronal Network exists of one input layer, at least one hidden layer, and one output layer. The input layer provides the Neuronal Network with the necessary information (i.e. input data). The input neurons process these input data and pass them to the next layer in a weighted manner. The hidden layer receives the information from the input layer and forwards the information from neuron to neuron to the output layer. The transport of information between two neurons of different layers is always weighted individually. The hidden layer is essentially a black box and not visible for users. The output layer is directly connected to the last hidden layer. The output neurons contain the resulting decision.

The **Decision Tree** [3, 4, 5] is a binary, hierarchical tree, in which each node has at most two children with one root node at the top. The Decision Tree is built on the data training set. The algorithm starts at the root node (i.e. at the top of the tree) and pass through the tree until a leaf (i.e. node without own children) is reached. In each node a decision is made which children node will be activated next. Decision trees can be categorised into classification trees, where the target variable can assume a discrete set of values, or regression trees, which are based on continuous values (i.e. real numbers). A node of a Decision Tree may consist of decision rules, several parameters (e.g. maximum depth of the tree), and/or eventually result values. The Decision Tree is individual for each data set. After executing the Decision Tree, the activated leaf node contains the resulting decision.

The **Random Forest** [3] consists of many Decision Trees (there is no limit to the number of Decision Trees). In general, the Random Forest defines an arbitrary number of Decision Trees and connect them with an independent root node. For the final result each Decision Tree is activated, and the individual results are aggregated (e.g. majority voting or averaging). Each of the Decision Trees result in different outputs. In addition to the parameters of the individual Decision Trees, the Random Forest requires further input variables (e.g. maximum number of individual trees into the Random Forest).

3. Implementation

Data set: For the prediction, the exercise points, and grades of 334 students from Ulm University, which are stored in the lecture 'databases' from the years 2018 until 2021 are used. The data set has been anonymised and does not contain any personal information. The maximum points

for each exercise sheet are different over the semester and compared to the previous semesters. To merge the data sheets a relative number is used in the form of percentage (i.e. 0-100%). Furthermore, the official grades ($x.0$, $x.3$, $x.7$ for $x = \{1, 2, 3\}$ and 4.0) are possible. Students, that failed the exam receive a 5.0 automatically. Some students have a grade bonus that may falsify the results as students who receive this are automatically one grade level better (expect for a 1.0 or a 5.0 for failing). To evaluate the results, the grade bonus will be subtracted from the grades.

Data Adjustments: The original data set is adjusted to uniform data types. When students failed or did not submitted an exercise sheet, different input variables (e.g. '0', NULL-Value '-', or a blank field) are given for the same context at the data set. Regarding the regression algorithms used, all non-numerical values are converted into equivalent numbers (i.e. real numbers), in this case into '0'. Additionally, students who had not submitted any exercise sheets, but had nevertheless taken part in the exam were deleted from the data set.

Data Distribution: The data set is split into training and test data set. With the training data set the models (e.g. Neuronal Network or Decision Tree) are generated and with the test data set the algorithms and its models are tested. In general, 80% of the data set is used for the training and the remaining 20% for testing. For the distribution of the data set three different methods are implemented and compared to avoid irregular distribution and avoid potential discrepancies.

First method takes the first 80% of the data set for testing and the remaining for training. With this method, the data set for training only consist of the students who took part in the year 2021. To avoid differences between the years the second method randomly assigned 80% for training and 20% for testing. The third method uses every 5th line for testing and the remaining for training. With the latter, a repeatable and reproducible method in comparison to the randomised method and a mixed distribution over the different years can be achieved.

Array Preparation: To develop the algorithms the feature (i.e. desired output) needs to be extracted from the data arrays (i.e. training and test data). For the example of this paper, exercise points from upcoming exercise sheets and the final grade will be predicted. The two latter define the features of the given context. For the training phase, the training array as well as the desired output (i.e. features) handed over to algorithm. For the subsequent testing, only the test array without the features is passed to the respective algorithms. The resulting predictions can then be compared to the actual values (i.e. the features).

3.1. Implementation of the Algorithm

The **Neuronal Network** uses the described input and feature arrays to predict their defined results (e.g. exercise points or grades). The number of neurons in the input layer is manifested in the variable *input_shape* and depends on the prediction (i.e. for the prediction of Exercise Sheet 5, the variable *input_shape* is set to 4 based on four previous sheets). In the hidden layer, the number of neurons is set to 200 and the activation function *ELU* is selected, as this function achieved the best results. Due to the small data set, the number of epochs (i.e. the number of passes through the training data set to train the network) is set to 900, which corresponds to the best results in manual tests. In the output layer, the number of neurons is set to 1 and either the exercise points or the grade may be predicted. To improve the manually tested Neuronal

Network, the function `GridSearch` (i.e. results in best fit values over all combinations of given parameters) and *callback* (i.e. stops the training when the best results have been achieved) is used. Through this improvement, the number of epochs is increased to 5000. In total, 1512 combination are tested.

A **Decision Tree** requires several parameters to be initialised. In this example, the four parameters *criterion*, which measures the quality of a split within the branches, *max_depth*, which specifies the maximum depth of the Decision Tree, *max_features*, which specifies the maximum number of features, and *random_state* for recovery a Decision Tree are selected in order to test the resulting predictions (i.e. exercise points and grades). For the parameter *criterion*, all four possible values (i.e. *squared_error*, *friedman_mse*, *absolute_error*, *poisson*) are tested. The *max_depth* and *max_features* accept fixed values, and a selection of values is tested. *Random_state* is used to achieve the same results when the algorithm is executed multiple times with the same parameters. The input array (e.g. *training_dataset*) depends on the feature (e.g. points of Exercise Sheet 4). To improve the manual tested combination of parameters, additionally a `GridSearch` is implemented. For this, the described parameters with its values are extended with the parameters *splitter*, which defines the rule to split at a node, *min_weight_fraction_leaf*, which defines the total size of weights within the sample to reach a leaf node, *min_samples_leaf*, which defines the minimum sample size (i.e. no further splitting is allowed), and *max_leaf_nodes*, which defines the maximum number of leaf nodes. In total, 1440 combination are tested.

Comparing the **Random Forest** with the Decision Tree they have the common parameters *max_depth*, *max_features*, *min_samples_leaf*, and *random_state*, but with difference values for each tree. In addition to them, the parameter *n_estimators*, which defines the maximum number of individual trees within the forest is required for the implementation. To improve the Random Forest again a `GridSearch` is implemented. For this, the described parameters are extended with the parameters *bootstrap* and *n_jobs*. If the parameter of *bootstrap* is set to *False*, the whole data set is used to train each tree, otherwise, the size for samples may be defined. The parameter *n_jobs* defines the number of jobs (i.e. functions) that can run in parallel. In total, 1620 combination are tested.

4. Evaluation and Results

For the comparison of the predicted results, **Actual Value** describes the points or grade a student received in the exercise sheet or exam. The parameter **Prediction** presents the results determined from the algorithms. The parameter **Variance** defines different values of possible deviations. First, no variance between the actual value and the prediction is acceptable (i.e. the variance is 0% for the exercise points and 0.0 for the grades). Second, a variance of 10% is acceptable for the exercise points and 0.4 for the grades and, third, 15% for the exercise points or 0.7 for grades are acceptable. Moreover, on average, an exercise sheet has between 20 and 25 points. An acceptable variance of 10% results in a total variance between 2 and 2.5 points. In addition, the variance of 4% for exercise points is also compared in the evaluation, which corresponds to a variance of one point per exercise sheet. The parameter **Match**, checks whether the defined variance is within the scope (i.e. *true*) or not (i.e. *false*).

4.1. Results

In total, the grades, and points of all 68 students are predicted for each sheet (i.e. Exercise Sheet 2 to 12) and their grade. In addition, these predictions are determined for each defined variance and each algorithm. In total, 17,136 predictions are determined, evaluated, and compared. In Tab. 1, the averages of correct answer (i.e. parameter *match* results in *true*) over each specified variance and algorithm for all Exercise Sheets 2 to 12 are shown. The percentage results (RES for short) of Tab. 1 are coloured in **red** for $RES < 30\%$, **orange** for $30\% \leq RES < 50\%$, **yellow** for $50\% \leq RES < 75.0\%$, **light green** for $75\% \leq RES < 90\%$ and **green** $RES \leq 90\%$.

Table 1

The percentages of correct prediction for each exercise sheet and algorithm over the specified variances.

Exercise Sheet	Neuronal Network				Decision Tree				Random Forest			
	0%	4%	10%	15%	0%	4%	10%	15%	0%	4%	10%	15%
2	0.0	57.4	86.8	92.6	1.5	77.9	94.1	95.6	0.0	76.5	94.1	95.6
3	0.0	47.1	82.4	95.6	0.0	48.5	82.4	91.2	0.0	47.1	80.9	91.2
4	0.0	60.3	92.6	97.1	0.0	58.8	95.6	95.6	0.0	79.4	94.1	97.1
5	0.0	60.3	94.1	97.1	1.5	57.4	88.2	98.5	0.0	64.7	91.2	97.1
6	0.0	36.8	77.9	89.7	0.0	33.8	77.9	88.2	0.0	42.6	76.5	88.2
7	1.5	25.0	52.9	82.4	0.0	25.0	52.9	77.9	0.0	25.0	60.3	73.5
8	0.0	35.3	70.6	92.6	0.0	29.9	77.9	89.7	0.0	36.8	72.1	89.7
9	0.0	41.2	80.9	91.2	0.0	42.6	67.6	89.7	0.0	42.6	82.4	86.8
10	0.0	27.9	67.6	80.9	0.0	32.4	67.6	88.2	0.0	36.8	79.4	86.8
11	0.0	17.6	36.8	57.4	0.0	13.2	41.2	64.7	0.0	23.5	54.4	72.1
12	0.0	4.4	11.8	19.1	2.9	8.8	11.8	27.9	0.0	8.8	25.0	36.8

Predicting 0% variance is close to impossible in any algorithm. This is because the points of the exercise sheets only allow whole or half points (i.e. x.0 or x.5). When transforming these points (between 0 and 25) into percentages (0 and 100) more than half of the percentage values are not addressed. In addition, the predictions allow decimal numbers with one decimal place. To predict the exact value when the points are converted is almost impossible. Exercise Sheet 11 and 12 also predict the points in most of the cases wrong. Considering the distribution of points across all sheets, it is remarkable that many students did not receive any points in the last two sheet (i.e. 0 points). The reason for this could be that the last exercise sheets deal with content that is no longer relevant to the exam or the students have completed their admission to the exam (i.e. students have achieved a minimum number of points across all exercise sheets). The Random Forest results the best predictions at 4% variance (i.e. variance of 1 point at the exercise sheet), however, these results are between 25.0% and 79.4%. In general, Exercise 7 achieves the worst results, while Exercise 4 achieves the best results. In Exercise Sheets 2 to 5, very positive results are achieved across all algorithms for 10% and 15% variance. The algorithms differ only minimally (i.e. up to 4.4%). In total, the best results are achieved by the Decision Tree with 98.5 correct predictions for 15% variance and 95.6% at 10%.

The predictions of the grades are capable of expansion. The Neuronal Network achieves only 29.4% with a variance of 0.3, and 48.5% with a variance of 0.7. The Decision Tree is slightly better with 32.4% with a variance of 0.3, and 57.4% with a variance of 0.7. The Random Forest

has a slightly better result than the Decision Tree with 35.3% with a variance of 0.3, and 61.8% with a variance of 0.7. For a variance of 0.0 the Neuronal Network predicts the best results with only 14.7% correctness (e.g. Decision Tree results in 10.3% and Random Forest results in 8.8%). There is no strict individual examination in the exercise sheets, i.e. students can help each other, read the script again and have no time pressure during the editing the exercise sheet. Additionally, the effort of learning directly before the exam is individual. All these factors are not considered at the current implementation and may be the reason for the inaccurate results.

5. Related Work

In [6], the performance (i.e. excellent, good, poorly, or fail) for around 400 students from the university of Bangladesh is predicted comparing eight different supervised algorithms (i.e. Support Vector Machines, Decision Tree, Multilayer Perceptron, Extra Tree, K-Nearest-Neighbour, AdaBoost, Logistic Regression, and Weighted Voting Classifier). In total, the Weighted Voting Classifier achieves the best results in predicting the performance category (i.e. excellent, good, poorly, or fail) of each student with 81.73% correctness. In [7], Neuronal Networks are used to predict the drop-out rates of students. The results are based on 2670 students at a Public University in Ecuador. Thereby, the input layer of the Neuronal Network consists of 11 neurons that are based on university factors (i.e. type of learning) and personal life decisions (e.g. pregnancy, born children and financial commitment). Two different kinds of Neuronal Networks were tried out: Multi-layer perception with 98.6% correctness and radial basis function with 98.1% correctness.

6. Conclusions

This paper implements three different regression algorithms (i.e. Neuronal Networks, Decision Trees, and Random Forest) to support and motivate students continuously during the semester by predicting their grades depending on their current performance (i.e. exercise points). To train and test the algorithms, a data set of 334 students with their exercise points and grades from previous semesters are used. Different variances (i.e. 0%, 4%, 10%, and 15%) for exercise points are tested and compared. With a variance of only 10% each algorithm may predict (for some but not all exercise sheets) a correctness between 91% and 96%. Predicting the exact points of upcoming exercise sheets (variance of 0%) none of the implemented algorithms works due to the transformation of points into percentages (i.e. finer range of points). On top, only a prediction of 35.3% with a variance of 0.3 and 61.8% with a variance of 0.7 using a Random Forest is possible. The results of the other two algorithms are slightly worse. In future work, the data set will be extended by collecting more data over the coming semesters. In addition, further algorithm will be implemented and tested to predict the grades more accurate. Moreover, other factors (i.e. learning type or financial commitment as student jobs) will be considered.

References

- [1] W. M. van der Aalst, J. Carmona, Process mining handbook, Springer Nature, 2022.

- [2] J. Moolayil, J. Moolayil, S. John, Learn Keras for deep neural networks, Springer, 2019.
- [3] A. Géron, Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow, " O'Reilly Media, Inc.", 2022.
- [4] B. Gupta, A. Rawat, A. Jain, A. Arora, N. Dhami, Analysis of various decision tree algorithms for classification in data mining, International Journal of Computer Applications 163 (2017) 15–19.
- [5] Y. Zhao, Y. Zhang, Comparison of decision tree methods for finding active objects, Advances in Space Research 41 (2008) 1955–1959.
- [6] M. S. Zulfiker, N. Kabir, A. A. Biswas, P. Chakraborty, M. M. Rahman, Predicting students' performance of the private universities of bangladesh using machine learning approaches, International Journal of Advanced Computer Science and Applications 11 (2020).
- [7] M. Alban, D. Mauricio, Neural networks to predict dropout at the universities, International Journal of Machine Learning and Computing 9 (2019) 149–153.

A Literature Review on Reproducibility Studies in Computer Science

Tobias Hummel^{1,*}, Johannes Manner¹

¹Distributed Systems Group, University of Bamberg, Germany

Abstract

Researchers expect a clear and well-documented experiment from industry experience reports and experimental research papers. All necessary configuration parameters, the source code, the experiment's machine configuration etc. should be documented in such a way that readers can interpret the results of these publication in detail. If this is the case, an interested reader is capable of redoing the experiments stated and verifying the results of others. This ability to properly interpret experiments and even reproduce them is a cornerstone of good scientific practice.

The experience from reading papers and consulting secondary studies reveals a different picture. A lot of papers are only partly interpretable since some information is missing. To understand the state of reproducibility in computer science, we conducted a Structured Literature Review (SLR) about reproducibility studies to list their motivations and challenges. These studies already tried to reproduce other research.

Two outcomes are of particular interest. First, the Information Retrieval (IR) domain is the role model w.r.t. reproducibility efforts. Most of the papers included in the SLR are from this domain. Second, publishers and conference formats start to create incentives by awarding badges for papers. Before the badges are awarded, the papers are checked for compliance with the rules of data submission and experiment reproducibility set by the conference respectively the publisher.

Keywords

Reproducibility, Replicability, Repeatability, Experiment Documentation

1. Introduction

“Reproducible Research in all sciences is critical to the advancement of knowledge. It is what enables a researcher to build upon, or refute, previous research allowing the field to act as a collective of knowledge rather than as tiny uncommunicated clusters” - [1, Cacho and Taghva p. 519].

This quote illustrates the importance of reproducibility of scientific work and describes an ideal state to gain knowledge. Nevertheless, the reality does not reflect this ideal state. In 2016, a survey with 1,576 respondents was published in Nature [2]. 90% of them stated that the scientific community in general is facing a reproducibility crisis. To understand this crisis, this paper tries to shed some light into this complex topic and states the current situation for

ZEUS2024: 16th Central European Workshop on Services and their Composition, Ulm, Germany, 2024.

*Corresponding author.

✉ tobias.hummel@stud.uni-bamberg.de (T. Hummel); johannes.manner@uni-bamberg.de (J. Manner)

🌐 <https://www.uni-bamberg.de/pi/team/manner-johannes/> (J. Manner)

🆔 0000-0002-7298-3574 (J. Manner)



© 2024 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

S. Böhm and D. Lübke (Eds.): 16th ZEUS Workshop, ZEUS 2024, Ulm, Germany, 29 February–1 March 2024, published at <http://ceur-ws.org>

reproducibility and related terms within the computer science domain. It furthermore highlights the efforts made towards publishing raw data and source code and shares some ideas on how to improve the current situation.

According to the Association for Computing Machinery (ACM) a measurement is reproducible if it *“can be obtained with stated precision by a different team using the same measurement procedure, the same measuring system, under the same operating conditions, in the same or a different location on multiple trials. For computational experiments, this means that an independent group can obtain the same result using the author’s own artifacts”*¹. Artifacts can be, for example, program code or datasets. In addition to reproducibility, the ACM lists two further often used terms at the mentioned website, namely repeatability and replicability. The following list briefly distinguishes the terms from each other:

- **reproducibility** - different team, same experimental setup
- **repeatability** - same team, same experimental setup
- **replicability** - different team, different experimental setup

However, in research these terms are sometimes used interchangeably [3]. Reproducibility and repeatability are even used the other way around [4]. There is also a standard called *Reproducibility Badging and Definitions* published by the National Information Standards Organization (NISO) [5]. There, they define further nuances of the three introduced terms above. Despite their depth, this standard being around for already three years is only referenced by 27 Google Scholar hits² when searching for the title of this publication. For this paper, we will stick to the ACM definitions introduced above due to their clarity and adaptation in the community. COLLBERG and others expected computer science to be in a special role compared to other disciplines: *“reproducing the work published in a systems conference or journal should be as simple as going to the authors’ website, downloading their code and data, typing ‘make,’ and seeing if the results correspond to the published ones”* [4, p. 1]. However, in a 30 minutes time frame they were able to retrieve and build the source code of only 32.3% of the papers analyzed. Other secondary studies, e.g. [6, 7, 8], confirm this issue. Only 3 out of 26 experiments are reproducible based on an assessment within the early Function as a Service (FaaS) research domain [6]. Another study concluded that a *majority* of 122 is not reproducible [7]. COUTURE and others [8] revealed that only 26% of 315 data projects published raw data.

All these studies should be a wake-up call for the computer science domain to publish raw data, source code and scripts. For without this background information a correct interpretation of results is not always possible for others which prevents the correct evaluation of the merit of a paper. This can be illustrated with an example from FaaS research: Two SLRs [9?] revealed papers reporting about unusual performance results when running a function on a cloud provider. Properly interpreting the original results based on the provider documentation showed that the measurements were due to a memory setting. It unintentionally assigned more than one CPU to the function. Without enough details about the experimental setup, it would not have been possible to falsify the misinterpretations of the original authors. As a consequence, conference chairs and publishers should incentivise researchers to make their experiments open to the public and enable others to reproduce experimental results.

¹<https://www.acm.org/publications/policies/artifact-review-and-badging-current>

²Google Scholar search was performed on 19th of February 2024.

Since we are interested in the current state of reproducibility studies, the objective of this paper is not to conduct a further reproducibility study but to analyze the already published studies by answering the following three research questions:

RQ1 What are publication trends for reproducibility studies in computer science?

RQ2 Why do authors try to reproduce the work of others?

RQ3 How successful are the reproducibility studies and which challenges do the authors encounter?

RQ1 focuses on the number of studies and visible trends. The reasons for performing such studies are questioned in **RQ2**. And the last question, **RQ3**, reveals success numbers on how many studies were able to reproduce the original work. One caveat here could be a publication bias, where reproducibility problems could be more likely to be published than success [10]. For upcoming reproducibility studies, we also look at challenges the included publications faced to share aggregated learnings.

The agenda of our paper is as follows: In Section 2, we shortly introduce the SLR methodology and present some numbers and an overview of included papers. Results for our three research questions are presented as subsections in Section 3. Section 4 concludes the paper with a short summary and an outlook to future work.

2. Methodology

We conducted a SLR to understand the current state of reproducibility studies in the computer science domain. A SLR is well suited for summarizing and synthesizing the current status on a topic in a fair way [10]. For the search phase, ACM Digital Library, IEEE Digital Library, DBLP and Google Scholar were used. While the first three libraries have a computer science focus, Google Scholar was chosen as a more general search engine with a large corpus. Therefore, we only used the first 50 entries at Google Scholar, sorted by relevance to get a manageable set of literature as already done by other SLRs [11, 12, 13]. Figure 1 summarizes the search process.

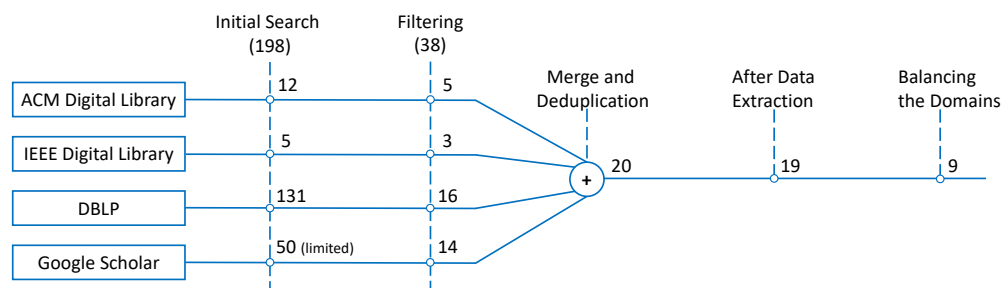


Figure 1: The initial search of the SLR was conducted on 23rd of May 2023 to identify reproducibility studies.

The search term required the phrase “reproducibility study” to be part of the title or abstract. As the search engines differ in their search options, the exact search strings differ [14]. In total,

198 entries were found in the initial search phase. The inclusion and exclusion criteria are based on the research questions and were applied to title, abstract and conclusions [14]. In cases of doubt, the full text was skimmed. The selection criteria were developed upfront and piloted on some studies as recommended by KITCHENHAM and CHARTERS [10]. Nevertheless, they evolved over the process. The inclusion criteria were as follows:

- Reproducibility studies in the area of computer science, other disciplines were excluded immediately.
- Publications in which an attempt is made to reproduce the findings of one or several prior publications.
- Publications by universities, public research institutes or industry.
- In addition to peer-reviewed publications, also preprints.

The following exclusion criteria were used:

- Publications in which the authors do not use or do not mention a specific software implementation.
- Publications that belong to the field of computational science rather than computer science, i.e. that are more concerned with the application of computer science in fields such as medicine.
- Publications in languages other than English (language is often used as an exclusion criterion [15, 16, 10]).

All raw data for the SLR are archived at Zenodo [17]. Inclusion and exclusion decisions as well as all other data related to the SLR can be found there. In case of exclusion, a reason is stated. Although required by KITCHENHAM and CHARTERS, the quality of the studies was not assessed for every paper [10]. It is assumed that in particular search engines from ACM, IEEE and DBLP have results with high quality. Otherwise, studies of questionable quality would have been excluded during the data extraction phase.

After the merge and deduplication step, 20 results remained. During data extraction, one paper turned out to be a replicability study and was thus excluded [18]. Within the remaining, 14 belonged to the information retrieval domain [19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32]. So as not to over-represent this field, out of these 14 papers the first four papers based on the first author names [19, 20, 21, 22] were chosen as examples from this computer science domain to discuss RQ2 and RQ3. Table 1 shows the remaining nine publications and the computer science sub-field to which they belong.

3. Results

3.1. RQ1: Publication Facts and Trends

The most obvious fact in our SLR investigation is that the information retrieval domain is a role model for reproducibility studies. 74% of our identified papers (14/19) are from this domain. The implementation of different ranking and retrieval algorithms is inherent to this computer science sub-field. For those an interpretation of results is only feasible by checking the raw data

Table 1

Publications included in the SLR by computer science sub-field.

Title	Ref	Sub-Field
A Comparison between Term-Independence Retrieval Models for Ad Hoc Retrieval	[22]	Information Retrieval
A Reproducibility Study of Question Retrieval for Clarifying Questions	[20]	
Cross-Domain Retrieval in the Legal and Patent Domains: A Reproducibility Study	[19]	
Cyberbullying Detection in Social Networks Using Deep Learning Based Models; A Reproducibility Study	[21]	
A Thorough Reproducibility Study on Sentiment Classification: Methodology, Experimental Setting, Results	[33]	Natural Language Processing
Reproducibility in Computational Linguistics: Is Source Code Enough?	[34]	
Examining the Reproducibility of Using Dynamic Loop Scheduling Techniques in Scientific Applications	[35]	Distributed Systems
IPAL: Breaking up Silos of Protocol-Dependent and Domain-Specific Industrial Intrusion Detection Systems	[36]	Security and Privacy
Machine Learning Based Invariant Generation: A Framework and Reproducibility Study	[37]	Software Verification

and experiment details. The same holds true for another little cluster, namely Natural Language Processing (NLP). Here, the research objectives are also highly dependent on input data and its processing.

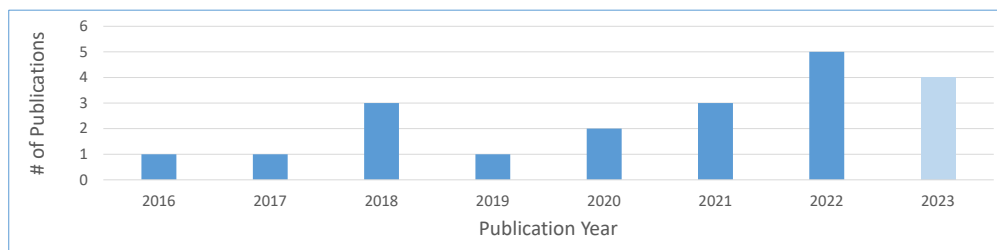


Figure 2: Number of reproducibility papers by publication year.

When we searched for literature, we did not limit the publication year. Nevertheless, the oldest paper in our set of filtered literature was published in 2016. Figure 2 shows the number of publications by their publication year. The bar for 2023 is colored orange since the search phase was conducted in May 2023, so further reproducibility studies might have been published in the rest of 2023. The distribution of publications over time shows that the number of papers which target reproducibility concerns has been rising over the last years.

Another facet of publication trends are the venues where papers get submitted and presented.

The *European Conference on Information Retrieval* is the top venue for reproducibility studies. A majority of the information retrieval papers were published there (9/14). One reason for this high number of papers is a special reproducibility track with a dedicated call for papers³. This shows that dedicated conference formats can support the reproducibility effort to bring experts together in one place.

To summarize the insights for **RQ1**: Information retrieval is the top domain for reproducibility research due to the domain specific challenges and supported by dedicated conference tracks. Reproducibility studies are a trending topic when looking at the rising publication numbers.

For answering the remaining research questions, we focus on the nine papers of Table 1.

3.2. RQ2: Reasons for Reproducibility

Five of the nine papers from Table 1, focus upon the reproducibility assessment [34, 20, 21, 35, 33]. In the other cases, the main contribution lies on other aspects but all publications included implicit or explicit statements why the reproducibility study was conducted. The motivations are summarized in the following list which answers **RQ2**, *Why do authors try to reproduce the work of others?*:

- Creating awareness for reproducibility [34, 20, 21, 35, 33].
- The analyzed paper is important for the research domain but the artifacts of the original study were not available [20].
- Starting point for own research [19, 21].
- Follow-up research after introducing a tool or framework to show that it works correctly [22, 37, 35, 36].

As an additional finding it could be shown that some studies mix terms and approaches. As introduced, we distinguish based on the ACM terms reproducibility, repeatability and replicability. Three of the publications analyzed in the SLR had a replicability study as an additional part [19, 21, 36]. That is, the authors first conducted a reproducibility study and then reused the setup for another dataset. There is a trade-off between an experimental setup as close as possible to the original setup, a reasonable use of resources, and the re-usability for other research questions. Nevertheless, the distinction is important since the motivation and results are influenced by the combination of team members and experimental setup.

3.3. RQ3: Success Ratios and Challenges

The success ratios for the nine included papers reveal a mixed picture. Four studies were successful, three were partly successful and two were not successful at all. The two biggest challenges were missing artifacts and lack of documentation.

Studies were considered successful when the measured values were close to the originally reported ones [21, 22, 33, 36]. For instance, DADVAR and others stated that “the majority of the reproduced results were within the standard deviation of the reference results” [21, p. 6]. NUNZIO and others report that they “have quite comparable results in terms of processing time”

³<https://ecir2023.org/calls/reproducibility.html?v=3.8>

and the “final scores differ from the original values by almost 2 percentage points” [33, p. 34]. For three publications, partly successful in this context means that the authors investigated several primary studies where some reproductions failed [34, 37, 35]. The two remaining studies failed in their efforts [19, 20].

One challenge of the studies was the availability of artifacts like source code or raw data. For three of the publications considered in this SLR the necessary artifacts were available [21, 22, 33] and the reproducibility for these studies succeeded. Regarding the other six, only incomplete artifacts were available.

Another big challenge was insufficient documentation of experimental setups. NUNZIO and others, for example, identified unclarities regarding the environment of the primary study. The hardware was not sufficiently described in the original publication, especially concerning the CPU and GPU as both could influence the execution time. In the original paper, there were two contradictory statements about the used GPU model. Additionally, the original authors trained one of their models in the cloud because they needed a more powerful environment. In comparison, the authors of the reproducibility study needed the cloud environment for two models which prevented the comparison of the training time for the additional model.

4. Conclusion

Reproducibility is a key requirement for science and has to be considered when starting a new research project. In reproducibility studies, researchers other than the original authors use a setup as close as possible and reasonable to reproduce and validate the work of others. Starting from this, the objective of this paper was to give an overview about reproducibility studies in computer science.

The publication trends showed that the information retrieval domain is a role model for other domains, supporting its researchers with dedicated conference tracks. Based on the number of publications per year, we saw that the reproducibility topic gains traction. Additionally, we could confirm other meta studies that showed that a lack of raw data and artifacts as well as an incomplete or missing documentation are the most serious challenges for good research.

For future work, we propose a follow-up study on reproducibility which should include additional search engines and incorporate further SLR tasks like snowballing⁴. We also plan to incorporate the feedback from reviewers to include the term “executable paper” which was a hyped term around 2011 as an additional search term. In addition, we want to broaden our scope by incorporating similar terms like “analysis” and “survey” paired with “reproducibility”, “replicability” and “repeatability” to also include publications which use another term but target “reproducibility studies” in the sense of this paper.

⁴Snowballing was already performed for this study and the raw results are already available at Zenodo. Due to time and space constraints, these snowballed publications were not assessed but published for follow-up research.

References

- [1] J. R. F. Cacho, K. Taghva, The state of reproducible research in computer science, in: Proc. of ITNG, 2020.
- [2] M. Baker, 1,500 scientists lift the lid on reproducibility, *Nature* 533 (2016) 452–454.
- [3] H. E. Plesser, Reproducibility vs. replicability: A brief history of a confused terminology, *Frontiers in Neuroinformatics* 11 (2018).
- [4] C. Collberg, T. Proebsting, A. M. Warren, Repeatability and benefaction in computer systems research - a study and a modest proposal, University of Arizona Technical Report 14 (2015).
- [5] NISO, Reproducibility Badging and Definitions: A Recommended Practice of the National Information Standards Organization, Technical Report niso-rp-31-2021, National Information Standards Organization (NISO), 2021.
- [6] J. Kuhlenkamp, S. Werner, Benchmarking FaaS Platforms: Call for Community Participation, in: Proc. of WoSC, 2018.
- [7] T. Kalibera, R. Jones, Rigorous benchmarking in reasonable time, in: Proc. of ISMM, 2013.
- [8] J. L. Couture, et al., A funder-imposed data publication requirement seldom inspired data sharing, *PLOS ONE* 13 (2018).
- [9] J. Manner, SeMoDe – simulation and benchmarking pipeline for function as a service, in: *Bamberger Beiträge zur Wirtschaftsinformatik und Angewandten Informatik*, 105, Otto-Friedrich-University, 2021.
- [10] B. Kitchenham, S. Charters, Guidelines for performing systematic literature reviews in software engineering, Keele University and University of Durham Technical Report (2007).
- [11] S. Kolb, On the Portability of Applications in Platform as a Service, Ph.D. thesis, Bamberg, 2019.
- [12] J. Manner, A structured literature review approach to define serverless computing and function as a service, in: Proc. of CLOUD, 2023.
- [13] D. Taïbi, et al., Patterns for serverless functions (function-as-a-service): A multivocal literature review, in: Proc. of CLOSER, 2020.
- [14] P. Brereton, et al., Lessons from applying the systematic literature review process within the software engineering domain, *Journal of Systems and Software* 80 (2007) 571–583.
- [15] J. Scheuner, P. Leitner, Function-as-a-service performance evaluation: A multivocal literature review, *Journal of Systems and Software* 170 (2020) 110708.
- [16] V. Yussupov, et al., A systematic mapping study on engineering function-as-a-service platforms and tools, in: Proc. of UCC, 2019.
- [17] T. Hummel, A Literature Review on Reproducibility Studies in Computer Science: Supporting Material, 2024.
- [18] R. F. G. Silva, K. Paixao, M. de Almeida Maia, Duplicate question detection in stack overflow: A reproducibility study, in: Proc. of SANER, 2018.
- [19] S. Althammer, S. Hofstätter, A. Hanbury, Cross-domain retrieval in the legal and patent domains: A reproducibility study, in: Proc. of ECIR, 2021.
- [20] S. Cross, G. Zuccon, A. Mourad, A reproducibility study of question retrieval for clarifying questions, in: Proc. of ECIR, 2023.
- [21] M. Dadvar, K. Eckert, Cyberbullying detection in social networks using deep learning

- based models; A reproducibility study, CoRR abs/1812.08046 (2018).
- [22] E. K. F. Dang, R. W. P. Luk, J. Allan, A comparison between term-independence retrieval models for ad hoc retrieval, *ACM Transactions on Information Systems* 40 (2021) 1–37.
 - [23] M. Färber, T. Klein, J. Sigloch, Neural citation recommendation: A reproducibility study, in: *Proc. of BIR*, 2020.
 - [24] M. Hendriksen, et al., Scene-centric vs. object-centric image-text cross-modal retrieval: A reproducibility study, in: *Proc. of ECIR*, 2023.
 - [25] J. Huang, et al., State encoders in reinforcement learning for recommendation: A reproducibility study, in: *Proc. of SIGIR*, 2022.
 - [26] W. Lajewska, K. Balog, From baseline to top performer: A reproducibility study of approaches at the TREC 2021 conversational assistance track, in: *Proc. of ECIR*, 2023.
 - [27] H. Li, et al., Improving query representations for dense retrieval with pseudo relevance feedback: A reproducibility study, in: *Proc. of ECIR*, 2022.
 - [28] J. Mackenzie, et al., Compressing inverted indexes with recursive graph bisection: A reproducibility study, in: *Proc. of ECIR*, 2019.
 - [29] R. Sequiera, L. Tan, Y. Zhang, J. Lin, Update delivery mechanisms for prospective information needs: A reproducibility study, in: *Proc. of CHIIR*, 2020.
 - [30] G. Silvello, et al., Statistical stemmers: A reproducibility study, in: *Proc. of ECIR*, 2018.
 - [31] S. Wang, S. Zhuang, G. Zuccon, Federated online learning to rank with evolution strategies: A reproducibility study, in: *Proc. of ECIR*, 2021.
 - [32] P. Yang, H. Fang, A reproducibility study of information retrieval models, in: *Proc. of ICTIR*, 2016.
 - [33] G. M. D. Nunzio, R. Minzoni, A thorough reproducibility study on sentiment classification: Methodology, experimental setting, results, *Information* 14 (2023) 76.
 - [34] M. Arvan, L. Pina, N. Parde, Reproducibility in computational linguistics: Is source code enough?, in: *Proc. of EMNLP*, 2022.
 - [35] F. Hoffeins, F. M. Ciorba, I. Banicescu, Examining the reproducibility of using dynamic loop scheduling techniques in scientific applications, in: *Proc. of IPDPS*, 2017.
 - [36] K. Wolsing, et al., Ipal: Breaking up silos of protocol-dependent and domain-specific industrial intrusion detection systems, in: *Proc. of RAID*, 2022.
 - [37] J. Haltermann, H. Wehrheim, Machine learning based invariant generation: A framework and reproducibility study, in: *Proc. of ICST*, 2022.

User-agent as a Cyber Intrusion Artifact: Detection of APT Activity using minimal Anomalies on the User-agent String Traffic

Badr-Eddine Bouhlal^{1,*}, Tim Sonnekalb¹, Bernd Gruner¹ and Clemens-Alexander Brust¹

¹German Aerospace Center (DLR), Institute of Data Science Jena, Germany

Abstract

The detection of attacks, especially persistent intrusions, relies on a combination of various artifacts. Despite being manipulable, the user-agent string, a component of HTTP headers, has proven to be a tool for triggering alerts, thereby enhancing detection capabilities. In this paper, we perform a review and analysis of existing malicious user agent strings. We gather relevant data from different sources of threat intelligence and present a dataset of user-agent strings associated with malicious activities gathered from real incident reports. We also propose a categorization of existing user-agent string anomalies with respect to their type (e.g., syntax) and their complexity degree.

Keywords

User-agent string (UAS), Advanced persistent threat (APT), Intrusion detection, Machine learning

1. Introduction

Intrusion detection is based on a multi-factor identification method that relies on multiple indicators. One of these indicators can be the HTTP User-Agent String (UAS) component. The initial function of the UAS is to allow the server to identify the request sender and disclose technical information such as the operating system and browser version. However, the UAS presents a considerable vulnerability in the web world because it is easily manipulated, making it a tool used to carry out code injection attacks. Nevertheless, in the context of a private and highly secure corporate network, the UAS can be used in addition to identifying the request sender as one of the intrusion detection factors.

The Cybersecurity and Infrastructure Agency (CISA) released a report in 2022 [1] containing recommendations that aim at helping organizations mitigating against advanced persistent threats (APT). The report recommend checking for anomalies that may be observed in UAS. These anomalies may directly affect the syntax of the UAS or may be related to other anomalies

*Corresponding author.

✉ badr-eddine.bouhlal@dlr.de (B. Bouhlal); Tim.Sonnekalb@dlr.de (T. Sonnekalb); Bernd.Gruner@dlr.de (B. Gruner); clemens-alexander.brust@dlr.de (C. Brust)

🆔 0009-0002-2860-3068 (B. Bouhlal); 0000-0002-0067-1790 (T. Sonnekalb); 0000-0002-4177-2993 (B. Gruner); 0000-0001-5419-1998 (C. Brust)



© 2024 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

CEUR Workshop Proceedings (CEUR-WS.org)

S. Böhm and D. Lübke (Eds.): 16th ZEUS Workshop, ZEUS 2024, Ulm, Germany, 29 February–1 March 2024, published at <http://ceur-ws.org>

that can be detected by using multiple identification factors, such as multiple authentication attempts with different UASs from the same IP addresses [1].

An anomaly may appear in several forms, and a malformed UAS can have multiple explanations: indicate the use of a vulnerability scanning tool (T1595 - Active Scanning) [2, 3], or be the sign of data exfiltration in the form of a legitimate string. It can also serve as a communication channel between malware and a command and control server (command and control attack). Communication via the application layer protocol is associated with several techniques used by APT groups attempting to exploit vulnerabilities in the HTTP/HTTPS protocol [4].

Challenges. The UAS does not follow a general format. While RFC 7231 [5] defines a general format for the presentation of the UAS many benign applications do not adhere to it [6]. This variability in UAS representation makes it difficult to conceive universal classification rules that remain effective over time. This situation presents significant challenges in differentiating legitimate UAS from malicious ones, as both representations initially consist of sequences of characters, numbers, and special characters. Another issue that arises when considering automation is determining the impact of irregular and sometimes anomalous user-agent that may not necessarily represent a threat which may increase the number of false alarms.

To the best of our knowledge, no dataset containing user-agent strings associated with malicious activity and presenting a detection artifact has been published. In addition, all existing studies concerning the detection of malicious traffic from user agents are mainly based on analysis of the network traffic, due to the absence of a malicious user-agent set. We summarize the key contributions of this paper as follows: (1) We introduce a dataset that consists of a collection of 1063 malicious UASs, which is publicly available under a CC-BY 4.0 License [7], (2) we perform a review and analysis of existing malicious UASs, and (3) we propose a categorization of the different UAS anomalies.

2. Related Work

In this section, we will present a series of studies that have focused on the user-agent string. Almost all of these studies aim to develop methods for distinguishing regular user-agent strings from malicious ones. Their approaches focus on developing parsing methods to make this distinction. However, we have noticed that they use data that cannot be directly linked to malicious traffic (a public database for this purpose does not exist, as far as we know). Furthermore, they focus mainly on syntax errors and do not consider cases of rarity. This rarity can hide minimal anomalies that are difficult to distinguish using a syntax parser, such as fake information, e.g., the pattern of the user-agent is correct, but the version of the browser used is fake. Zhang et al. [6] examined the UASs in malicious traffic, specifically malware. The authors found that one out of every eight instances of malware traffic contained suspicious user-agent in at least one of their HTTP requests. Currently, user-agent are still being analyzed manually. However, the analysis showed that there are multiple patterns that could be used to automatically classify user-agent anomalies. They also propose an automated technique for extracting user-agent anomalies and creating signatures for malware detection.

Kheir [8] applied a rule-based methodology using regular expressions for distinguishing abnormal and normal user-agent based on the fact that user-agent have a general and fixed

structure that enables their validation using regular expressions. The data used during this research was collected over two months from real traffic and contained 150 billion user-agent. Zhang et al. [6] stated that the causes of anomalies could be due to two factors, namely a malfunction during the encoding decoding of the user-agent or a malicious activity.

Zhang et al. [9] used a method that combines several steps to classify user-agent: firstly, it uses a parser based on a context-free grammar to classify them based on their representation, a standard UAS, a non-standard representation for non-standard UAS, and finally, for unrecognized representations. Then, the authors propose using an anomaly detection algorithm to separate benign UAS from malicious ones. Their study compared the Context-Free Grammar (CFG) with the User Agent parser based on regular expressions. They showed that the CFG is better suited for analyzing user-agent traffic due to its ease of adaptation and simplicity of comprehension.

Nandakumar et al. [10] presented a novel method of parsing the user-agent strings based on Multi-Headed Attention mechanism using transformer, the method is divided into two-step, first parse the UAS to gather the information related to the device and software, then correlate the extracted information with known related Common Vulnerabilities and Exposures (CVE).

3. Illustrative Incidents: Malicious User-Agent Case Studies

Cyber-attacks, especially those carried out for espionage purposes, are designed to persist without being detected in the network. Performed by well-trained teams (APTs) using complex methods and sometimes over several well-planned stages, these attacks are not necessarily easy to analyze and sometimes difficult to determine their consequences at first glance. However, every potential indicator of malicious activity on the network must be carefully analysed and considered. These indicators, also known as network artifacts, can vary from an IP address, an URI pattern or an UAS that has not previously been observed in a defined network environment, or one that appears to be out of the ordinary [11]. In this section, we provide a concise overview of some incident reports from cyber-attack campaigns, specifically focusing on cases where the UAS field deviates from the norm. This divergence serves as a crucial factor in uncovering potential threats within the network. We perform this review of real incident to be able to analyse the type of anomalies that can be considered as artifacts of detection within the UAS (cf. Subsection 3.2), and also to perform a categorization of them (cf. Subsection 4.2).

3.1. Real-life Incidents

Targeted Phishing Exploits Impacting Japanese and Taiwanese Organizations [12, 13]. APT groups have launched a mail fishing campaign with malicious word attachments targeting governmental organizations, finance, media, and high-tech sectors in Japan and Taiwan. The attack exploits a Microsoft Office EPS vulnerability CVE-2015-1701 [12], the exploit payload releases a binary, that includes an embedded sample of the IRONHALO malware. IRONHALO uses the HTTP protocol to fetch the payload from a command-and-control (C&C) server with hard-coded settings and a specific Uniform Resource Locator (URL) path [13]. This malware variant sends an HTTP request to a legitimate Japanese site with a malformed UAS with a syntax error: missing space between the different components of the UAS as shown in Figure 1a.

GET /syouyou/images/index.php HTTP/1.1 Accept: */* User-Agent: Mozilla/4.0 compatible; MSIE 8.0; Windows NT 6.1) Connection: Keep-Alive Host: www.<redacted>[.]com Cache-Control: no-cache	Registry Keys: HKLM\System\CurrentControlSet\Services\bmwappushservice URLs: https://is-cdn.edge.g18.dyn.usr-e12-as.akamaitechnology[.]Jcom/deploy/assets/css/main/style.min.css http://a17-h16.911.iad17.as.phnt-external.c15.goldenlines[.]Jnet/deploy/assets/css/main/style.min.css HTTP artifacts: "User-Agent : XXXXXXXXXXXXXXXXXXXX/5.0 (Windows NT 6.1 WOW64; Trident/7.0; AS; rv:11.0) like Gecko" "Proxy-Authorization : Basic [Data]" ~ [Data] Will contain the TDESS encrypted data to send
---	--

(a) IRONHALO HTTP GET request [13] (b) CopyKittens TDESS indicators of compromise [14]

Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/70
Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/70.0.3538.110 Safari/537.36
Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:63.0) Gecko/20100101 Firefox/63.0
Mozilla/5.0 (Windows NT 6.1; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/70.0.3538.110 Safari/537.36
Mozilla/5.0 (Macintosh; Intel Mac OS X 10_14_1) AppleWebKit/605.1.15 (KHTML, like Gecko) Version/12.0.1 Safari/605.1.15
Microsoft Office/14.0 (Windows NT 6.1; Microsoft Outlook 14.0.7162; Pro)
Microsoft Office/14.0 (Windows NT 6.1; Microsoft Outlook 14.0.7166; Pro)
Microsoft Office/14.0 (Windows NT 6.1; Microsoft Outlook 14.0.7143; Pro)
Microsoft Office/15.0 (Windows NT 6.1; Microsoft Outlook 15.0.4605; Pro)

(c) Fancy Bear (APT28) user-agent strings [15]

Figure 1: Anomalies remarked on the UASs based on real incident related to APT Traffic

CopyKittens [14]. A cyber espionage group that mainly targets strategic organizations such as governmental organizations (defense companies, research institutions, Ministry of defense, and large IT companies), using self-developed tools that are not necessarily publicly reported. The methods of attack are complex and varied. The report [14] describes the intrusion methodology and also a set of used malware's, for example, the TDESS which is a 64-bit .NET binary backdoor that communicates regularly with the command and control server, using basic authentication to receive new instructions. The incident analysis report presents various Indicators of compromise, among which is a malformed UAS [14], as shown in Figure 1b.

Russian GRU Conducting Global Brute Force Campaign to Compromise Enterprise and Cloud Environments [15]. Between mid-2019 and 2021, the Russian General Staff Main Intelligence Directorate (GRU) 85th Main Special Service Center (GTsSS), also known as Fancy Bear or APT28, used the Kubernetes cluster to launch large-scale anonymous brute force access attacks against government and private sector organizations, exploiting the CVE 2020-0688 and CVE 2020-17144 vulnerabilities in Microsoft Exchange, they used several protocols including HTTP. The campaign report publicly released by the NSA encompasses a detailed description of the techniques and tactics used as well as mitigation and detection methods including IP address lists and UASs, which "are crafted to appear consistent with those sent by legitimate client software. Some of the UASs delivered in the authentication requests are incomplete or truncated versions of legitimate UASs, offering the following unique detection opportunities [15]" (cf. Figure 1c) .

Bumblebee loader [16, 17]. A malware, employed in various campaigns by numerous threat actors, uses the Windows Management Instrumentation (WMI) framework to extract system details. Then, it establishes a connection with the C&C server at intervals of 25 seconds to receive commands to be executed. Logpoint [16] proposes two detection techniques within the proxy log files. Malware detection in the proxy log might be done either using the user agent and/or the URI. Hence, any existence of a user agent that matches the string *bumblebee* is

a strong sign of the persistence of this malware. Other versions of the malware uses different evasion techniques and might change the UAS. An undefined UAS with the same number of digits should also be suspicious and might refer to the persistence of *bumblebee* [16, 17].

LYCEUM middle east campaign [18]. An APT group that targets organizations in strategic sectors, including oil and gas. Campaigns were reported in South Africa in 2018 and in the Middle East in 2019. The group primarily uses password-sparing or brute-force attacks to gain access to an organization to obtain credentials. Then, using compromised accounts, they send spearphishing emails containing malicious attached Excel files containing the Danbot malware. Danbot malware is a first-stage access trojan (RAT) that uses DNS and the HTTP protocol for communication. The Danbot HTTP request contains two anomalies: An ampersand (&) after operating system values in the UAS (**Mozilla/5.0 (Windows NT 10.0; &) Gecko/20100101 Firefox/64.0**), and a misspelling of 'Encoding' in the accept-encoding header [18].

Quasar: Open-Source Remote Administration Tool [19]. Is a RAT open source used for Windows Operating systems, hosted publicly on GitHub, specially dedicated for being used for legitimate purposes. In addition, various APT threat groups are using Quasar to conduct cyber espionage campaigns. Quasar enables remote control, keylogging, file transfer and enables the user to collect information about the host system. During the client connection's setup, the client tries to determine its geolocation, including its Wide Area Network (WAN) IP address. This is achieved by sending an HTTP GET request to the Uniform Resource Locator (URL) `ip-api[.]com/json/` with the following User-Agent string: **Mozilla/5.0 (Windows NT 6.3; rv:48.0) Gecko/20100101 Firefox/48.0**. *"This User-Agent string mimics a Mozilla Firefox 48 browser running on Windows 8.1. This User-Agent string would likely stand out as unique in a corporate network environment, and its presence could be a high-confidence indication of Quasar activity"* as stated by the Cybersecurity and Agency [19].

3.2. Analysis of the Incidents

By analyzing the various real incidents presented in the previous section, we note that the irregularities in the HTTP traffic (in our case, UASs), represents especially syntax errors or rarity of occurrence, represent a serious sign of a potential threat.

Indeed, UAS patterns vary significantly and do not necessarily follow a general structure that could be generalized to all device types (software, hardware). However, some syntax anomalies could represent a "red flag" and should be carefully examined. These syntax errors vary from grave errors, where the user-agent does not match any pattern of a correct UAS, to non-defined strings, as in the case of Bumblebee, or the TeamTNT group, where they use the UAS field to execute an injection code operation **curl -referer \$REFERER -user-agent TNTcurl \$CURLPARA \$GETFROM -o \$PUTITTO** [20]. Another example of using the UAS as a tool of code injection is the recent exploit of the Log4j vulnerability by the APT35 [21]. In addition we mention also syntax errors such as misspellings, absences, or even the existence of strings that should not exist in the correct UAS (as in the cases of CopyKittens and LYCEUM).

Another category of anomalies concerns not the syntax of the UAS, where it may be completely accurate, but the existence of a rare UAS, that can be suspicious especially in corporate networks. The detection of such anomalies depends on the analysis of the overall traffic, whereas a simple

analysis of the syntax would not allow the detection, as in the case of Quasar malware and Fancy Bear (APT28). A rule-based detection method based on syntax would not be useful for the detection of fake UASs. The latter may take the form of a syntactically correct UAS. However, with a fake browser version, for example, the Metamorfo [22] malware uses a UAS with Mozilla/3.0, the existence of such a version 3.0 must be suspicious.

4. Dataset Construction

In this section we present the construction process of our dataset of UASs associated with malicious traffic, focusing on attacks related to APTs. An essential part of building this dataset is categorizing anomalies in UASs.

4.1. Data Sources and Data Collection

For creating the dataset, we relied on different sources. Firstly, security reports on real incidents (cf. Subsection 3.1) which are collected by reputable organizations such as MITRE ATT@CK [23] and the Cybersecurity and Infrastructure Security Agency (CISA) [24]. These reports permit valuable insights into the latest cyber threats, offering detailed information on tactics, techniques, and procedures (TTPs) employed by malicious actors. The second source of our data consisted of contributions published by security professionals and experts in security blogs and community forums, for example, Microsoft [25] and Cisco Talos [26]. Subsequently, the UASs identified as artifacts of malicious activity were obtained from the Sigma open-source rules. These entries are regularly identified and logged as part of blacklists¹. In addition, the dataset includes data gathered from the open-source project Apache Bad Bot Blocker².

4.2. User-Agent String Anomaly Categorization

Our categorization encompasses a wide range of anomalies observed in UASs, based on analyzing anomalies related to real-life incidents, we define the following two main categories.

4.2.1. The type of anomalies

The anomaly type, can be used as a detection hint and for this we define three different cases:

Syntax errors. These are the syntax anomalies that can be distinctive in a user-agent, for example, the missing space in the case of IRONHALO or the existence of an invalid string part, as is the case with CopyKittens [14] (xxxxxxxxxxxxxxxxxxxx/5.0 instead of **Mozilla/5.0**). In this case, the UAS has a correct pattern (format) but contains syntax errors.

Unknown string. In this category, we will classify all UASs that do not contain any pattern or part of a pattern of a correct UAS. These UASs are a collection of random strings (characters, digit or special characters) and may contain the name of the malware itself or any other string like the *bumblebee* for example.

¹Sigma - Generic Signature Format for SIEM Systems: <https://github.com/SigmaHQ/sigma#sigma---generic-signature-format-for-siem-systems>

²Apache ultimate bad bot blocker <https://github.com/mitchellkroga/apache-ultimate-bad-bot-blocker/tree/master>

Other. This category contains UASs without syntax anomaly, but are associated with a malware, APTs, or other malicious activities. Intrusion detection from these user agents may be due to their rarity in the traffic. They may stand out as anomalous in network traffic because of their rarity, or simply because their signatures should not exist in the specific network traffic.

4.2.2. The anomaly complexity degree

The second classification criterion distinguishes between high anomalies and low anomalies. This criterion is designed to evaluate the expected ability of a machine learning model to detect anomalous UASs at two levels of difficulty: the detection of a user-agent that differs totally syntactically from a normal pattern, and between the detection of a simple syntax anomaly. The two categories are defined as follows:

Low anomaly. Low anomaly formats represent user agent strings with minor or subtle deviations from normal (standard) formats. These may be a small syntax error, missing components, or the presence of unknown elements in an otherwise normal structure, as in the case of IRONHALO, where the anomaly is a missing space. In this category, we also include the UASs previously classified as *Other*.

High anomaly. Formats or patterns involving irregular, unusual, or entirely new structures in user agent strings that deviate significantly from regular formats. These anomalies can include random unknown strings, special characters, or unique identifiers with no correspondence with known, legitimate user agents, like the example of the **bumblebee**, where the string *bumblebee* do not represent a legitimate (or a part of) UAS. Another example is: **sample (unknown version) CFNetwork/596.5 Darwin/12.5.0 (x86_64) (iMac8%2C1)** [27], which includes many syntax errors and deviates systematically from a normal UAS. This UAS exhibits several anomalous characteristics, including elements like *sample (unknown version)*. Additionally, the presence of the identifier (*iMac8%2C1*) does not correspond to known UAS or a part of a legitimate UAS and the percentage sign is not typically part of a standard UAS.

5. Dataset Description and Usage

Description. During data collection, we systematically inspect malicious UASs and collect additional information: The *type of anomaly*, the *degree of complexity*, the *APT associated with the malicious UAS*, if the information is available, the *sources* (e.g., citations of the resources from where the UAS was gathered), and the *string pattern* that distinguishes between two cases: if the abnormal UAS matches the entire string (*match_string*) or if it involves a regular expression (*regex*). The Regular expression describes a string part that might be included in a syntactically correct UAS but might be a sign of malicious activities. The data is provided in a CSV format, each row represents a UAS with corresponding information. Table 1 depicts an example of the dataset structure and Table 2 shows the distribution of the malicious UASs per category.

Usage. The dataset contains two types of UAS entries that must be treated differently: *match_string*, which represents malicious UASs that can be used directly, and *regex*, which presents only string parts which must be applied to correct UASs to generate malicious ones, these regular expressions define the potential location of these parts within a correct UAS. In addition to the set of malicious UASs (abnormal), we provide a set of 1000 of the most frequently

UAS	Anomaly Type	Anomaly Complexity	Related Apt	Ressources	String Pattern
UAS-1	syntax error	low	APT34	URL of document/online resources	match_string

Table 1
Description of the dataset structure

Complexity degree	Type of anomaly		
	Unknown string	Syntax errors	Other
High anomaly	738	77	-
Low anomaly	-	34	214

Table 2
Distribution of the collected malicious UASs per category

seen user agents during November 2023, parsed by the Whatismybrowser API parser ³. These 1000 entries will be considered as a reference for normal user agents, containing no syntax anomalies. This allows using the dataset to train/test machine learning models on the detection of malicious UASs. We performed a similarity check between the two sets (normal and abnormal), and we noticed that there are eight common entries. An example entry is the malicious UAS representing the pattern used by the Google’s bot crawler **Mozilla/5.0 (compatible; Googlebot/2.1; +http://www.google.com/bot.html)**, normally legitimately used for indexing purposes. The explanation for this finding is the fact that some malwares use complicated evasion techniques and they might use the most common and widely used UAS patterns. Moreover, some malwares mimics the UAS of the system on which it is installed to blend into the network traffic and avoid to be detected, as in the case of the *FatDuke malware* used by APT29, also known as *Cozy Bear* [28]. Such UASs fall under the category *Other*, where the UAS does not exhibit any syntax errors. However, the detection artifacts might be due to their rarity of existence within a traffic of legitimate UASs. Another example is the *Quasar malware* that uses a legitimate UAS that mimics a *Mozilla Firefox 48* browser running on *Windows 8.1*, which could be suspicious, especially in corporate networks [19].

6. Conclusion and Future Work

This study highlights the significant role of the UAS as an indicator of attack detection, and especially persistent intrusions. We present a dataset of 1063 UAS associated with malicious activities, the majority of which were collected manually from real incident reports. These UASs exhibit anomalies in syntax or other aspects, providing a basis for detecting persistent activities within network traffic. As future work, we plan to further extend the dataset and use it as training and test set to evaluate different machine learning models on their capabilities to automatically detect anomalies related to UASs.

³Whatismybrowser <https://www.whatismybrowser.com/>

Acknowledgments

We extend our sincere gratitude to Whatismybrowser, Raape, Ulrich and Möbius, Max for their invaluable assistance in collecting UASs for this study.

References

- [1] Cybersecurity and Infrastructure Security Agency (CISA), Cybersecurity advisory: Impacket and exfiltration tool used to steal sensitive information from defense industrial base organization, 2022. URL: <https://www.cisa.gov/news-events/cybersecurity-advisories/aa22-277a>, accessed: April 15, 2023.
- [2] Center for Threat-Informed Defense, Microsoft azure security control mappings to mitre att&ck®, 2023. URL: <https://center-for-threat-informed-defense.github.io/security-stack-mappings/Azure/README.html>, accessed: April 20, 2023.
- [3] M. Attck, Active scanning, 2022. URL: <https://attack.mitre.org/techniques/T1595/>, accessed: April 20, 2023.
- [4] M. Attck, Application layer protocol: Web protocols, 2020. URL: <https://attack.mitre.org/versions/v7/techniques/T1071/001/>, accessed: April 21, 2023.
- [5] R. T. Fielding, J. F. Reschke, Hypertext transfer protocol (http/1.1): Semantics and content, 2014. URL: <https://www.rfc-editor.org/rfc/rfc7231>, accessed: April 11, 2023.
- [6] Y. Zhang, H. Mekky, Z.-L. Zhang, R. Torres, S. ju Lee, A. Tongaonkar, M. Mellia, Detecting malicious activities with user-agent-based profiles, *International Journal of Network Management* 25 (2015) 306 – 319. URL: <https://api.semanticscholar.org/CorpusID:13959125>.
- [7] B.-E. Bouhlal, Dataset of malicious user-agent strings, 2024. URL: <https://doi.org/10.5281/zenodo.10700806>. doi:10.5281/zenodo.10700806.
- [8] N. Kheir, Behavioral classification and detection of malware through http user agent anomalies, *Journal of Information Security and Applications* 18 (2013) 2–13. URL: <https://www.sciencedirect.com/science/article/pii/S2214212613000331>. doi:<https://doi.org/10.1016/j.jisa.2013.07.006>, sETOP’2012 and FPS’2012 Special Issue.
- [9] Y. Zhang, H. Mekky, Z.-L. Zhang, R. Torres, S.-J. Lee, A. Tongaonkar, M. Mellia, Detecting malicious activities with user-agent-based profiles, *International Journal of Network Management* 25 (2015) 306–319. URL: <https://onlinelibrary.wiley.com/doi/abs/10.1002/nem.1900>. doi:<https://doi.org/10.1002/nem.1900>.
- [10] D. Nandakumar, S. Murli, A. Khosla, K. Choi, A. Rahman, D. Walsh, S. Riede, E. Dull, E. Bowen, A novel approach to user agent string parsing for vulnerability analysis using mutli-headed attention, 2023. arXiv:2306.03733.
- [11] CSNP, Tryhackme - pyramid of pain room, Dec 5, 2022. URL: <https://www.csnp.org/post/tryhackme-pyramid-of-pain-room>, accessed: on: [02.01.2024].
- [12] R. W. Genwei Jiang, Dan Caselden, The eps awakens, 2015. URL: https://web.archive.org/web/20170613151054/https://www.fireeye.com/blog/threat-research/2015/12/the_eps_awakens.html, accessed: [15.12.2023].
- [13] F. T. I. Ryann Winters, The eps awakens - part 2, 2015. URL: <https://web.archive>.

- org/web/20151226205946/https://www.fireeye.com/blog/threat-research/2015/12/the-eps-awakens-part-two.html, accessed: [15.12.2023].
- [14] C. C. Security, Operation wilted tulip, 2017. URL: https://www.clearskysec.com/wp-content/uploads/2017/07/Operation_Wilted_Tulip.pdf, accessed: [15.12.2023].
 - [15] Cybersecurity, I. S. Agency, Russian gru conducting global brute force campaign compromise enterprise and cloud environments, 2021. URL: https://media.defense.gov/2021/Jul/01/2002753896/-1/-1/1/CSA_GRU_GLOBAL_BRUTE_FORCE_CAMPAIGN_UOO158036-21.PDF, accessed on [18.12.2023].
 - [16] Logpoint, Buzz of the bumblebee – a new malicious loader, <https://www.logpoint.com/wp-content/uploads/2022/05/buzz-of-the-bumblebee-a-new-malicious-loader-threat-report-no-3.pdf>, 2022. Accessed on [18.12.2023].
 - [17] K. Merriman, P. Trouerbach, This isn't optimus prime's bumblebee but it's still transforming, April 28, 2022. URL: <https://www.proofpoint.com/us/blog/threat-insight/bumblebee-is-still-transforming>, accessed on [18.12.2023].
 - [18] Secureworks, Lyceum takes center stage in middle east campaign, 2019. URL: <https://www.secureworks.com/blog/lyceum-takes-center-stage-in-middle-east-campaign>, accessed: on: [02.01.2024].
 - [19] Cybersecurity, I. S. Agency, Quasar open-source remote administration tool, 2019. URL: <https://www.cisa.gov/news-events/analysis-reports/ar18-352a>, accessed on [18.12.2023].
 - [20] David Fiser and Alfredo Oliveira, Tracking the Activities of TeamTNT: A Closer Look at Cloud-Focused Malicious Actor Group, . URL: https://documents.trendmicro.com/assets/white_papers/wp-tracking-the-activities-of-teamTNT.pdf, Accessed on: [18.12.2023].
 - [21] C. point, Apt35 exploits log4j vulnerability to distribute new modular powershell toolkit, <https://research.checkpoint.com/2022/apt35-exploits-log4j-vulnerability-to-distribute-new-modular-powershell-toolkit/>, 2022. Accessed on: [18.12.2023].
 - [22] Xiaopeng Zhang, Another Metamorfo Variant Targeting Customers of Financial Institutions, <https://www.fortinet.com/blog/threat-research/another-metamorfo-variant-targeting-customers-of-financial-institutions>, 2020. Accessed on: [18.12.2023].
 - [23] M. Corporation, MITRE ATTCK, Accessed: on: [02.01.2024]. URL: <https://attack.mitre.org/>.
 - [24] Cybersecurity and Infrastructure Security Agency (CISA), CISA, Accessed: on: [02.01.2024]. URL: <https://www.cisa.gov/>.
 - [25] Microsoft Corporation, Microsoft Security Blog, Accessed February 2024. URL: <https://www.microsoft.com/en-us/security/blog/>.
 - [26] Cisco Talos, Cisco Talos Blog, Accessed February 2024. URL: <https://blog.talosintelligence.com/>.
 - [27] Unit 42 Palo Alto Networks, Unit 42: Xagentosx – sofacy's xagent macos tool, 2017. URL: <https://unit42.paloaltonetworks.com/unit42-xagentosx-sofacys-xagent-macos-tool/>, accessed: on: [19.02.2024].
 - [28] T. D. ESET: Matthieu Faou, Mathieu Tartare, Operation ghost the dukes aren't back they never left, 2019. URL: https://web-assets.esetstatic.com/wls/2019/10/ESET_Operation_Ghost_Dukes.pdf, accessed: on: [05.01.2024].

Author Index

Arnold, Lisa, 14, 47

Bouhlal, Badr-Eddine, 63

Breitmayer, Marius, 14, 47

Brust, Clemens-Alexander, 63

Grote, Alexander, 27

Gruner, Bernd, 63

Hariharan, Anuja, 27

Hehnle, Philipp, 9

Hummel, Tobias, 54

Kirikkayis, Yusuf, 41

Knierim, Michael, 27

Lichtenthäler, Robin, 22

Lübke, Daniel, 33

Manner, Johannes, 54

Reichert, Manfred, 9, 14, 41, 47

Schiessle, Pascal, 41

Sonnekalb, Tim, 63

Vidgof, Maxim, 1

Weinhardt, Christof, 27