John Krogstie
Stefanie Rinderle-Ma
Gerti Kappel
Hend*erik* A. Proper (Eds.)

LNCS 15701

Advanced Information Systems Engineering

37th International Conference, CAiSE 2025
Vienna, Austria, June 16–20, 2025
Proceedings, Part I

1 Part I

CAiSE '25

Springer

Lecture Notes in Computer Science **15701**

Founding Editors

Gerhard Goos
Juris Hartmanis

The series Lecture Notes in Computer Science (LNCS), including its subseries Lecture Notes in Artificial Intelligence (LNAI) and Lecture Notes in Bioinformatics (LNBI), has established itself as a medium for the publication of new developments in computer science and information technology research, teaching, and education.

LNCS enjoys close cooperation with the computer science R & D community, the series counts many renowned academics among its volume editors and paper authors, and collaborates with prestigious societies. Its mission is to serve this international community by providing an invaluable service, mainly focused on the publication of conference and workshop proceedings and postproceedings. LNCS commenced publication in 1973.

John Krogstie · Stefanie Rinderle-Ma ·
Gerti Kappel · Hend*erik* A. Proper
Editors

Advanced Information Systems Engineering

37th International Conference, CAiSE 2025
Vienna, Austria, June 16–20, 2025
Proceedings, Part I

Springer

Editors
John Krogstie
NTNU – Norwegian University of Science and Technology
Trondheim, Norway

Gerti Kappel
TU Wien
Vienna, Austria

Stefanie Rinderle-Ma
Technische Universität München
Garching bei München, Germany

Hend*erik* A. Proper
TU Wien
Vienna, Austria

ISSN 0302-9743 ISSN 1611-3349 (electronic)
Lecture Notes in Computer Science
ISBN 978-3-031-94568-7 ISBN 978-3-031-94569-4 (eBook)
https://doi.org/10.1007/978-3-031-94569-4

This Springer imprint is published by the registered company Springer Nature Switzerland AG
The registered company address is: Gewerbestrasse 11, 6330 Cham, Switzerland

Preface

This volume (of two) of the Lecture Notes in Computer Science series contains the first part of the proceedings of the 37th International Conference on Advanced Information Systems Engineering – CAiSE 2025, held in Vienna, Austria, during June 16–20. The CAiSE conference series was started in Sweden, and its first two editions were organized in Stockholm (1989 and 1990). Since then, it has been held in 21 countries, mostly in Europe but also in locations on other continents (Canada, Australia, and Tunisia). Despite its roots in Europe, soon after its inception, CAiSE became the flagship international conference on information systems engineering, attracting some of the most original and scientifically rigorous articles in the field. Furthermore, it has always been an attractive event for early-career researchers as well as practitioners. After almost four decades of continuous organization, CAiSE continues to attract articles that address fundamental and timeless problems in the field.

As a conference attuned to new theoretical and societal challenges, each year a new focal topic is chosen as the theme of the conference. CAiSE 2025 was organized with a special emphasis on the theme of **Bridging Silos**. Engineering real-world information systems requires a coherent design, encompassing human, organizational, economic, societal, and technological aspects. Information systems are utilized in increasingly diverse contexts such as business process management, geographical information systems, and digital twins. At the same time, the discipline continually evolves with trends in data science, machine learning, process mining, blockchain, mobile computing, sustainability, new regulations, cyber warfare, and military conflicts all influencing its development. Each application context and emerging trend can lead to specializations within information systems engineering, necessitating various research traditions and needs. While beneficial, these specializations risk creating silos within the field of information systems engineering. Thus, the CAiSE conference, the premier event for this discipline, aims to prevent such fragmentation.

CAiSE 2025 attracted a broad spectrum of classical and modern topics in Information Systems Engineering, including several papers directly focused on this year's theme. In total, we received 286 abstract submissions, 248 of which materialized as full submissions. We then desk rejected 19 papers due to issues related to paper formatting and length but, mostly, papers that we independently judged to be out of the scope of the conference. The remaining papers were then sent for independent reviews, each being allocated to two members of the international Program Committee. After the first phase of reviews, papers that received two negative informative reviews were not admitted to the second phase of reviews – 120 papers were rejected in this first phase. In the second phase of reviews, the remaining 110 papers were allocated to a third reviewer, who was necessarily a member of the conference Program Board (a body of senior scholars from the community, representing different research areas and geographical locations, the Program Committee chairs of the previous and succeeding editions of CAiSE, as well as members of the conference Steering Committee). After the second phase of reviews, the submissions for which the three reviewers converged to a unanimous negative judgement were rejected (37 submissions). The remaining submissions (73) were then assigned to yet another member of the Program Board for discussions and meta-reviews. In an intense discussion period, these latter members of the Program Board moderated the discussions between the reviewers of the paper (the two Program Committee members and the reviewing Program Board Member). They then summarized and explained the recommendations for each submission in a meta-review consolidating the outcome of those discussions. As an outcome of this discussion and meta-reviewing process, we were able to decide on 29 papers: 16 which were directly accepted (these are papers that received three positive and informative reviews plus a recommendation from the discussant/meta-reviewer to accept without conditions); 13 papers were directly rejected (these were papers with three negative and informative final reviews plus a recommendation from the discussant/meta-reviewer to reject without an invitation to the Forum). After these different phases of reviewing, meta-reviewing, and online discussions, 44 papers (18% of the original submissions) were discussed in a face-to-face meeting organized in February 2025 at the Technische Universität Wien (TU Wien) in Austria. In that meeting, besides the Program Chairs and Program Board members, participated representatives of the General Chairs of the conference, the Forum Chairs, and the Workshop Chairs. After two days of intensive and informative discussions, 19 submissions were directly accepted based on the novelty of their work and the depth of contributions to the advancement of the state of the art. Furthermore, 17 submissions were accepted conditionally, subject to specific improvements the co-authors were able to address in a minor revision checked by a gatekeeper – in the vast majority of cases, a member of the Program Board who was involved in the reviewing or meta-reviewing of the paper. The outcome is presented in this volume, comprising 35 scientifically rigorous and innovative papers – a 14% acceptance rate. In addition, we recommended 16 submissions for consideration by the CAiSE Forum.

As a tradition at CAiSE, the conference program started with workshops as well as two traditional satellite conferences (EMMSAD and BPMDS). The main conference included sessions involving Research Papers, Keynotes, Tutorials, the Forum – for discussion triggering visionary work – the Doctoral Consortium, Project Exhibitions, as well as Journal-First papers. In addition, the program included a panel discussion. This volume collects the Research Papers of the conference, as well as companion abstracts for the keynote talks and the tutorials. The 35 accepted papers together with 9 accepted Journal-First papers were grouped into the following fourteen topical sessions: Modelling with LLM; Security; Sustainability; Chatbots and social networks; Process monitoring; IS development and usage; Pre-processing and forecasting; Comprehension, explanation and recommendation; Process discovery; Systems architecture and privacy; Conformance checking; Cloud systems; Extended process modelling; Ontologies and knowledge graphs.

The three invited keynote presentations were: *"Engineering the (Information) System of Life"* by Oscar Pastor from the Universidad Politécnica de Valencia; *"Reflections on the Early History of Information Systems Development Methodologies"* by Rudy Hirschheim, from Louisiana State University; and *"Business Transformation Management in the age of AI and agents"* by Gero Decker, from SAP Berlin. On behalf of the information systems engineering community, we would like to thank our keynote speakers for their time dedicated to preparing profound, timely, and insightful talks. After an open call for tutorials, we accepted the following two proposals: *"Empowering Development and Use of DSMLs with the FMML^{X} and the XModeler^{ML}"* by Ulrich Frank from the University of Duisburg-Essen; and *"Addressing Vulnerabilities in Information Systems Engineering"* by Avi Shaked from the University of Oxford.

As the editors of both CAiSE 2025 volumes, we would like to express our immense gratitude to:all authors who chose CAiSE as a forum for submitting their research contributions; the members of the Program Committee for their professional and timely work; the members of the Program Board for their dedication and expertise in helping to shape an essential part of the final program of the conference; all the chairs of the conference for selecting an interesting program that complements the contributions collected here; the Organization Committee for professionally taking care of all the multiple details of running an international and sufficiently large conference like CAiSE, but also for bringing the conference to the fantastic city of Vienna in Austria.

John Krogstie
Stefanie Rinderle-Ma
Gerti Kappel
Hend*erik* A. Proper

April 2025

Organization

General Chairs

Gerti Kappel	TU Wien, Austria
Hend*erik* A. Proper	TU Wien, Austria

Program Committee Chairs

John Krogstie	Norwegian University of Science and Technology, Norway
Stefanie Rinderle-Ma	Technical University of Munich, Germany

Local Organization Chairs

Dominik Bork	TU Wien, Austria
Angela Edlinger	TU Wien, Austria

Forum Chairs

Kristina Rosenthal	Hochschule Niederrhein University of Applied Sciences, Germany
Luise Pufahl	Technical University of Munich, Germany

Workshop Chairs

Janis Grabis	Riga Technical University, Latvia
Yves Wautelet	KU Leuven, Belgium

Journal-First Chairs

Barbara Weber	University of St. Gallen, Switzerland
Janis Stirna	Stockholm University, Sweden

Tutorial and Panel Chairs

Henrik Leopold	Kühne Logistics University, Germany
Tiago Prince Sales	University of Twente, The Netherlands

Doctoral Consortium Chairs

Selmin Nurcan	Université Paris 1 Panthéon-Sorbonne, France
Sergio España	Universitat Politècnica de València, Spain

PhD Award Chairs

Giancarlo Guizzardi	University of Twente, The Netherlands
Pnina Soffer	University of Haifa, Israel

Research Projects Exhibition Chairs

Claudenir Fonseca	University of Twente, The Netherlands
Mattia Fumagalli	Free University of Bozen-Bolzano, Italy

Proceedings Chair

Marianne Schnellmann	TU Wien, Austria

Web Chair

Aleksandar Gavric	TU Wien, Austria

Publicity and Social Media Chairs

Asif Gil	University of Technology Sydney, Australia
Jaap Gordijn	Vrije Universiteit Amsterdam, The Netherlands
João Paulo Almeida	Federal University of Espírito Santo, Brazil
Roman Lukyanenko	University of Virginia, USA

Program Board

Arnon Sturm	Ben-Gurion University of the Negev, Israel
Barbara Pernici	Politecnico di Milano, Italy
Barbara Weber	University of St. Gallen, Switzerland
Flavia Santoro	Universidade do Estado do Rio de Janeiro, Brazil
Giancarlo Guizzardi	University of Twente, The Netherlands
Hajo Reijers	Utrecht University, The Netherlands
Iris Reinhartz-Berger	University of Haifa, Israel
Jan Mendling	Humboldt-Universität zu Berlin, Germany
Janis Stirna	Stockholm University, Sweden
Jelena Zdravkovic	Stockholm University, Sweden
João Paulo Almeida	Federal University of Espírito Santo, Brazil
Johann Eder	University of Klagenfurt, Austria
Jolita Ralyté	University of Geneva, Switzerland
Lidia Fuentes	Universidad de Málaga, Spain
Manfred Reichert	University of Ulm, Germany
Maribel Yasmina Santos	University of Minho, Portugal
Marta Indulska	University of Queensland, Australia
Mathias Weske	University of Potsdam, Germany
Monique Snoeck	KU Leuven, Belgium
Oscar Pastor	Universitat Politècnica de València, Spain
Paolo Giorgini	University of Trento, Italy
Pierluigi Plebani	Politecnico di Milano, Italy
Pnina Soffer	University of Haifa, Israel
Renata Guizzardi	University of Twente, The Netherlands
Selmin Nurcan	Paris 1 Panthéon-Sorbonne University, France
Shazia Sadiq	University of Queensland, Australia
Wolfgang Maaß	Saarland University, Germany
Xavier Franch	Universitat Politècnica de Catalunya, Spain

Program Committee

Adela Del Río Ortega	University of Seville, Spain
Alejandro Maté	University of Alicante, Spain
Alessandro Gianola	Universidade de Lisboa, Portugal
Andrea Burattin	Technical University of Denmark, Denmark
Andrea Marrella	Sapienza University of Rome, Italy
Andreas L. Opdahl	University of Bergen, Norway
Andrey Rivkin	Technical University of Denmark, Denmark
Anna Bernasconi	Politecnico di Milano, Italy

Antonio Ruiz-Cortés	University of Seville, Spain
Arik Senderovich	York University, Canada
Artem Polyvyanyy	University of Melbourne, Australia
Ashwin Viswanathan Kannan	Oklahoma State University, USA
Bernhard Thalheim	Christian-Albrechts-Universität zu Kiel, Germany
Boualem Benatallah	UNSW Australia, Australia
Camille Salinesi	Paris 1 Panthéon-Sorbonne University, France
Chiara Di Francescomarino	DISI - University of Trento, Italy
Claudenir Fonseca	University of Twente, The Netherlands
Claudio Di Ciccio	Utrecht University, The Netherlands
Corentin Burnay	University of Namur, Belgium
Cristina Cabanillas	Universidad de Sevilla, Spain
Devis Bianchini	University of Brescia, Italy
Dominik Bork	TU Wien, Austria
Emanuele Laurenzi	University of Applied Sciences and Arts Northwestern Switzerland, Switzerland
Ernest Teniente	Universitat Politècnica de Catalunya, Spain
Fabrizio Maria Maggi	Free University of Bozen-Bolzano, Italy
Felix Mannhardt	Eindhoven University of Technology, The Netherlands
Fernanda Baião	PUC-Rio, Brazil
Georg Grossmann	University of South Australia, Australia
Han van der Aa	University of Vienna, Austria
Hans-Georg Fill	University of Fribourg, Switzerland
Haralambos Mouratidis	University of Essex, UK
Henrik Leopold	Kühne Logistics University, Germany
Ingo Weber	Technical University of Munich, Germany
Istvan David	McMaster University, Canada
Jaap Gordijn	Vrije Universiteit Amsterdam, The Netherlands
Janis Grabis	Riga Technical University, Latvia
Jennifer Horkoff	Chalmers University of Technology, Sweden
Johannes De Smedt	KU Leuven, Belgium
João Araújo	Universidade NOVA de Lisboa, Portugal
Julio Cesar Leite	UFBA, Brazil
Kate Revoredo	Humboldt-Universität zu Berlin, Germany
Kurt Sandkuhl	University of Rostock, Germany
Leonardo Montecchi	NTNU IDI, Norway
Luca Piras	Middlesex University, UK
Luiz Olavo Bonino Da Silva Santos	University of Twente, The Netherlands
Luís Ferreira Pires	University of Twente, The Netherlands
Manfred Jeusfeld	University of Skövde, Sweden

Manuel Wimmer	Johannes Kepler University Linz, Austria
Marcela Ruiz	Zurich University of Applied Sciences, Switzerland
Marcelo Fantinato	University of São Paulo, Brazil
Maria Teresa Gómez López	University of Seville, Spain
Marite Kirikova	Riga Technical University, Latvia
Marlon Dumas	University of Tartu, Estonia
Massimiliano Leoni	University of Padua, Italy
Massimo Mecella	Sapienza University of Rome, Italy
Matthias Weidlich	Humboldt-Universität zu Berlin, Germany
Mattia Fumagalli	Free University of Bozen-Bolzano, Italy
Minseok Song	Pohang University of Science and Technology, South Korea
Oscar Diaz	University of the Basque Country, Spain
Palash Bera	Saint Louis University, USA
Paolo Giorgini	University of Trento, Italy
Paul Grefen	Eindhoven University, The Netherlands
Pedro Paulo F. Barcelos	University of Twente, The Netherlands
Peter Fettke	DFKI, Germany
Raimundas Matulevicius	University of Tartu, Estonia
Sagar Sunkle	Tata Consultancy Services, India
Sander Leemans	RWTH Aachen University, Germany
Sareh Sadeghianasl	Queensland University of Technology, Australia
Schahram Dustdar	TU Wien, Austria
Sebastian Link	University of Auckland, New Zealand
Seppe Van den Broucke	KU Leuven, Belgium
Shang Gao	Ørebro University, Sweden
Simon Hacks	Stockholm University, Sweden
Sobah Abbas Petersen	NTNU IDI, Norway
Stijn Hoppenbrouwers	HAN University of Applied Sciences, The Netherlands
Tiago Prince Sales	University of Twente, The Netherlands
Tong Li	Beijing University of Technology, China
Tony Wasserman	Carnegie Mellon Silicon Valley, USA
Veda Storey	Georgia State University, USA
Veruska Zamborlini	Universidade Federal do Espírito Santo, Brazil

Steering Committee Chairs

John Krogstie	Norwegian University of Science and Technology, Norway
Monique Snoeck	KU Leuven, Belgium
Xavier Franch	Universitat Politècnica de Catalunya, Spain

Advisory Board

Arne Sølvberg	Norwegian University of Science and Technology, Norway
Barbara Pernici	Politecnico di Milano, Italy
Colette Rolland	University of Paris 1 Pantheon-Sorbonne, France
Johann Eder	University of Klagenfurt, Austria
Oscar Pastor	Universitat Politècnica de València, Spain

Steering Committee

Barbara Weber	University of St. Gallen, Switzerland
Camille Salinesi	University of Paris 1 Pantheon-Sorbonne, France
Carlos Cetina	Universidad San Jorge, Spain
Eric Yu	University of Toronto, Canada
Ernest Teniente	Universitat Politècnica de Catalunya, Spain
Flavia Santoro	UERJ - Universidade do Estado do Rio de Janeiro, Brazil
Frederik Gailly	Ghent University, Belgium
Geert Poels	Ghent University, Belgium
Giancarlo Guizzardi	University of Twente, The Netherlands
Hajo Reijers	Utrecht University, The Netherlands
Haralambos Mouratidis	University of Essex, UK
Iris Reinhartz-Berger	University of Haifa, Israel
Janis Stirna	Stockholm University, Sweden
Jelena Zdravkovic	Stockholm University, Sweden
Marcello La Rosa	University of Melbourne, Australia
Marta Indulska	University of Queensland, Australia
Massimo Mecella	Sapienza University of Rome, Italy
Paolo Giorgini	University of Trento, Italy
Pnina Soffer	University of Haifa, Israel
Shazia Sadiq	University of Queensland, Australia

Forum Program Committee

Abel Armas Cervantes	University of Melbourne, Australia
Agnes Koschmider	University of Bayreuth, Germany
Andrea Marrella	Sapienza University of Rome, Italy
Ben Roelens	Open University of the Netherlands, The Netherlands
Christophe Feltus	Institute of Science and Technology, Ireland
Cinzia Cappiello	Politecnico di Milano, Italy
Cristina Cabanillas	Universidad de Sevilla, Spain
Drazen Brdjanin	University of Banja Luka, Bosnia and Herzegovina
Elena Kornyshova	Conservatoire National des Arts et Métiers, France
Evangelia Kavakli	University of the Aegean, Greece
Hans Weigand	Tilburg University, The Netherlands
Hans-Georg Fill	University of Fribourg, Switzerland
Henrik Leopold	Kühne Logistics University, Germany
Irene Vanderfeesten	KU Leuven, Belgium
Janis Grabis	Riga Technical University, Latvia
Janne J. Korhonen	Aalto University, Finland
Jose Ignacio Panach Navarrete	Universitat de València, Spain
Jānis Kampars	Riga Technical University, Latvia
Lawrence Chung	University of Texas at Dallas, USA
Manuel Resinas	University of Seville, Spain
Manuel Wimmer	Johannes Kepler University Linz, Austria
Marite Kirikova	Riga Technical University, Latvia
Martin Henkel	Stockholm University, Sweden
Mattia Salnitri	Politecnico di Milano, Italy
Maya Daneva	University of Twente, The Netherlands
Michael Fellmann	University of Rostock, Institute for Computer Science, Germany
Mohamad Gharib	University of Tartu, Estonia
Oscar Pastor	Universitat Politècnica de València, Spain
Patricia Martin-Rodilla	Instituto de Estudios Gallegos, Spain
Raimundas Matulevicius	University of Tartu, Estonia
Sergio de Cesare	University of Westminster, UK
Simon Hacks	Stockholm University, Sweden
Stefan Strecker	University of Hagen, Germany
Steven Alter	University of San Francisco, USA
Sybren de Kinderen	Eindhoven University of Technology, The Netherlands

Sérgio Guerreiro	University of Lisbon, Portugal
Tony Clark	Aston University, UK
Yves Wautelet	KU Leuven, Belgium

Additional Reviewers

Abasi-Amefon Affia-Jomants
Ada Slupczynski
Adam Banham
Adam Burke
Aleksandar Gavric
Alfonso Marquez-Chamorro
Anne Gutschmidt
Arsalan Ghasemi
Benjamin Nast
Carlos Müller
David Mosquera
Deepika Gopukumar
Diana Malakhova
Dominic Detering
Evellin Cardoso
Gabriel Morais
Gal Engelberg
Henryk Mustroph
Ijeoma Faustina Ekeh
Illr Murturi
Jan Niklas van Detten
José Antonio Parejo Maestre
José María García
José Miguel Horcas Aguilera
Karamjit Kaur
Maider Azanza
Mariia Bakhtina
Maximilian König
Maya Sappelli
Mónica Pinto
Mubashar Iqbal
Pablo Fernandez
Qian Chen
Quim Motger
Rainer Weinreich
Rebecca Morgan
Sandip Kumar Sarkar
Savandi Kalukapuge
Simone Agostinelli
Stefan Klikovits
Thomas Ricardo Pathe
Tian Li
Tom Lichtenstein
Victoria Sonnemans
Vitor Gaboardi dos Santos
Vjatšeslav Antipenko
Wenjun Zhou
Zsolt Kardkovacs

Abstracts of the Invited Talks

Engineering the (Information) System of Life

Oscar Pastor

Universitat Politècnica de València, Spain
opastor@dsic.upv.es

The theme of this year's CAiSE Conference is "Bridging Silos", which invites us to consider Information Systems Engineering from a cohesive design perspective that integrates multiple aspects. Having in mind the grand challenge of deciphering the language of life, this keynote takes up the challenge of "Bridging Silos" by exploring how to engineer the (information) system of life, connecting information systems engineering with life engineering, and showing how a well-structured application of AI technologies can make that challenge reachable, combining fast and slow thinking.

To describe life as a system for an audience of information systems engineers requires us to focus on the principles of design and operation that are common between technological systems and life itself. Life can be interpreted as an open, adaptable, and evolving system, where connections, data, and processes not only determine survival but also the quality of results. Recognizing these analogies helps engineers reflect on the nature of the systems they design and their impact on life itself.

Going further, life as an information system can be seen as a distributed, interactive, and adaptive system that continuously processes data for decision-making and evolution, leveraging connections and feedback in a dynamic environment. Decoding the human genome is a challenge that fits perfectly with the analogy of life as an information system.

The keynote will explore the Human Genome as the "Source Code" of Life's Information System, looking at the human genome as the "instruction manual" of life's system, and focusing on how to understand and manipulate it, a capability never as close for humanity as it starts to be now. Relating it to an information system perspective allows us to understand how nature stores, processes, and uses data for survival, adaptation, and evolution. The keynote will connect biological and technological concepts, showing that the intelligent design behind the genome and modern systems have more in common than it seems.

Reflections on the Early History of Information Systems Development Methodologies

Rudy Hirschheim

Louisiana State University, USA
rudy@lsu.edu

Information systems development (ISD) methodologies have been at the core of the information systems (IS) domain since its inception. The challenges of understanding the evolution of this domain is their fast change, enormous variety, and increasing embedding of the "how" of ISD into development contexts. Yet knowing how IS are designed and implemented, and how this knowledge is passed on to future developer generations, is centrally important.

In this presentation, we will explore how and why ISD methodologies have changed over time, and what likely explains their enormous variety. Generally, ISD methodologies can be defined as: "evolving, shared, explicit, cognitive, and normative knowledge systems that provide answers to how to develop an information system". As institutionalized knowledge artifacts they are created and sustained collectively across many communities (academic/practice) to support ISD. They do so by typifying and standardizing development activities and outputs, and offering heuristics to address development problems. The first publicly available ISD methodologies emerged in the 1960s. Since then, they have proliferated into hundreds, perhaps thousands, even though most invented forms of the major methodologies (waterfall, etc.) are still in use.

This talk will present a historical analysis of the 70-year evolution of ISD methodologies focusing particularly on the early history of ISD methodology development and use.

based on a working paper entitled: "Information Systems Development (ISD) Methodology Evolution -- A Historical Analysis of Drivers and Outcomes", Jaana Porra, Rudy Hirschheim, Kalle Lyytinen & Frank Land.

Business Transformation Management in the age of AI and Agents

Gero Decker

SAP Signavio, Germany
gero.decker@sap.com

Without a doubt, process management, enterprise architecture and all the other disciplines within information systems engineering have helped companies drive efficiency and adapt to change, with tremendous impact over the past decades. But - now that Gen AI has arrived and excitement around it has reached an all-time high - are the existing disciplines still relevant for the future at all? Won't some artificial superintelligence magically solve all challenges a company faces? Or - if they still have a role to play - how is that different from their role in the past, how do they have to evolve?

Gero Decker, founder and CEO of SAP Signavio, will shed light on what he observes and the bets that SAP is making in the area of Business Transformation Management.

Contents – Part I

Pre-processing and Forecasting

Contents – Part II

Conformance-Checking

Cloud Systems

Extending Process Modelling

Ontologies and Knowledge Graphs

Tutorials

Modelling with LLM

Leveraging LLMs for Domain Modeling: The Impact of Granularity and Strategy on Quality

Iris Reinhartz-Berger[1](✉), Syed Juned Ali[2], and Dominik Bork[2]

[1] University of Haifa, Haifa, Israel
iris@is.haifa.ac.il
[2] TU Wien, Business Informatics Group, Vienna, Austria
{syed.juned.ali,dominik.bork}@tuwien.ac.at

Abstract. The information systems engineering community is increasingly exploring the use of Large Language Models (LLMs) for a variety of tasks, including domain modeling, business process modeling, software modeling, and systems modeling. However, most existing research remains exploratory and lacks a systematic approach to analyzing the impact of prompt content on model quality. This paper seeks to fill this gap by investigating how different levels of description *granularity* (whole text vs. paragraph-by-paragraph) and modeling *strategies* (model-based vs. list-based) affect the quality of LLM-generated domain models. Specifically, we conducted an experiment with two state-of-the-art LLMs (GPT-4o and Llama-3.1-70b-versatile) on tasks involving use case and class modeling. Our results reveal challenges that extend beyond the chosen granularity, strategy, and LLM, emphasizing the importance of human modelers not only in crafting effective prompts but also in identifying and addressing critical aspects of LLM-generated models that require refinement and correction.

Keywords: Domain modeling · Conceptual modeling · LLM · Generative AI · UML

1 Introduction

Domain modeling plays a central role in designing and developing complex systems that address diverse human, organizational, and technological needs. Over time, the discipline has witnessed significant advancements in supporting or automating various modeling tasks. Among these, Large Language Models (LLMs) have recently emerged as powerful tools for natural language understanding and generation, offering promising applications in domain modeling [8,19]. By leveraging their ability to process and analyze large volumes of textual data, LLMs hold the potential to assist in generating accurate and comprehensive models.

J. Krogstie et al. (Eds.): CAiSE 2025, LNCS 15701, pp. 3–19, 2025.
https://doi.org/10.1007/978-3-031-94569-4_1

Effective modeling requires not only domain knowledge but also methodological rigor. In our previous study [1], we demonstrated that novice modelers (namely, IS students) tend to adopt one of two primary strategies: starting from lists of model elements (e.g., classes) or directly constructing models. Additionally, they often rely on complete textual descriptions rather than incrementally building models from individual paragraphs. These tendencies raise critical questions about how different modeling strategies and granularity levels influence the quality of the resulting models. In this context, model quality is assessed in terms of semantic properties, particularly *requirements satisfaction* and *redundancy*, and syntactic properties, particularly *syntactic correctness* [14,15].

To explore the impact of granularity and modeling strategy on the quality of the LLM-generated domain model, we used two state-of-the-art LLMs, GPT-4o (GPT for short) and Llama-3.1-70b-versatile (Llama for short), to assist with modeling tasks across three distinct application domains. The research is guided by the following research questions:

[RQ1] To what extent does the description ***granularity*** (*whole text* vs. *paragraph-by-paragraph*) affect the quality of LLM-generated domain models?

[RQ2] To what extent do different modeling ***strategies*** (*model-based* vs. *list-based*) influence the quality of LLM-generated domain models?

[RQ3] Are there differences in the observations noted in RQ1 and RQ2 across
- **[RQ3.1]** different ***LLMs*** (*GPT* and *Llama*)?
- **[RQ3.2]** different ***application domains***?
- **[RQ3.3]** different ***tasks*** (Use Case Modeling and Class Modeling)?

The remainder of this paper is structured as follows. Section 2 provides a brief overview of related work on domain modeling and LLMs. Section 3 describes the research methodology, including the experimental design and evaluation metrics. Section 4 presents the results and discusses their implications as well as threats to validity. Finally, Sect. 5 concludes the paper by summarizing key findings and outlining directions for future research.

2 Related Work

Next, we review key works related to automated domain modeling in general and specifically using LLMs. We also discuss prompting techniques in this context.

2.1 Automated Domain Modeling

Automated domain modeling has been an active research area even before the emergence of LLMs. Traditionally, these methods use textual descriptions of domains and apply statistical or rule-based methods combined with natural language processing (NLP). Such methods have been used to directly derive complete domain models [10,17,18,24], or provide modeling assistance or recommendations during the modeling process [4,20]. Burgueno et al. [4], for example, use vector representations, i.e., word embeddings, to capture the lexical and semantic information from textual documents to suggest model elements for a given partially completed model. Several other approaches combine NLP and machine learning techniques to automate the model creation process [18,24].

Rule-based methods use manually designed grammatical templates to extract domain models from textual descriptions. Robeer et al. [17], for example, present an algorithm with 23 heuristics to automatically identify model elements from user stories. Herchi et al. [11] combine NLP techniques like sentence splitting, tokenization, and syntactic parsing to decompose the input text and then use linguistic rules (e.g., all nouns are converted to entity types) to extract UML concepts. Jahan et al. [12] present a rule-based approach for automated domain modeling and report that their approach effectively produces relevant, simplified diagrams for straightforward user stories, whereas the LLM tends to create more complex diagrams that can go beyond the simplicity of the original user stories.

2.2 LLM-Based Domain Modeling

With the advancement of LLMs across various tasks, their application in domain modeling has gained significant attention. Table 1 compares studies in this area, emphasizing the granularity they address, whether they explicitly support refinement or updates to outcomes from prior interactions, the prompt content, and the expected modeling artifacts. Prompt content may encompass task descriptions, domain descriptions, and/or format specifications for the syntax of the expected output.

Fill et al. [9] conducted experiments with GPT-4 for creating ER models, UML class diagrams, and BPMN models. They concluded that very large model parts can be correctly generated by ChatGPT. However, modeling experience is still required to validate the results. Bajaj et al. [3] concluded that GPT-3 outperforms classical tools commonly used in practice for extracting use cases. Chen et al. [8] present a comparative analysis of GPT-4 and GPT-3.5 for automated domain modeling, utilizing various prompt engineering techniques on a dataset comprising ten diverse domain modeling examples. Each example was accompanied by a reference solution created by modeling experts. The authors use the task, domain, and expected output format descriptions to create class diagrams. They conclude, that, while the LLMs demonstrate impressive domain understanding capabilities, they are still impractical for fully automated domain modeling. They further report that LLMs offer the lowest performance in identifying relationships compared to their performance with classes and attributes.

In another work, Chen et al. [6] explore the use of GPT-4 for creating goal models and report that the amount of domain information in the textual description has a limited effect on the responses of GPT-4. The responses have to be evaluated carefully as many elements generated by GPT-4 may be either incorrect or rather generic. Further, the authors conclude that immediate interactive feedback can improve the syntax and semantics of the goal model and expand the initial draft for simple requests. In [7], the authors compare the effectiveness of prompt engineering and fine-tuning for domain-specific modeling tasks.

Their findings reveal that approaches focusing on prompt improvements outperform fine-tuning-based methods, even without explicit training on the dataset. Furthermore, the performance gap between prompting and fine-tuning becomes wider when the training dataset is small.

Table 1. Comparison of existing studies in LLM-based domain modeling

Paper	Granularity	Refinement	Prompt Content	Modeling Artifact
[9]	Whole Text	Not Supported	Task, Domain, Format	Class Diagrams, ER Diagrams, BPMN Models
[3]	Whole Text	Not Supported	Task, Domain	Use Case Diagrams
[8]	Whole Text	Not Supported	Task, Domain, Format	Class diagrams
[6]	Whole Text	Partially	Task, Domain, Format	Goal Models
[7]	Concept Names	Not Supported	Task, Concept Names	Taxonomy Graphs
[2]	Single User, Story Paragraph	Not Supported	Task, User Story	Class Diagrams
[5]	Whole Text, Modeling Element	Not Supported	Task, Domain	Class diagrams
[19]	Whole Text	Not Supported	Task, Domain	Class Diagrams
Ours	Whole Text, Paragraphs	Supported	Task, Domain, Format	Use Case Diagrams, Class Diagrams

Arulmohan et al. [2] use GPT-3.5 to extract domain models from user stories. The authors apply OpenAI's prompt engineering techniques[1] to create the prompts for the LLM. They separately extract the concepts and relationships and report that while LLMs demonstrate impressive performance, traditional NLP techniques outperform them in this task. Camara et al. [5] also assess GPT-3.5 in an interactive mode using prompts enriched with OCL constraints. They report that GPT-3.5 struggles to handle models larger than 8–10 classes, but performs well with syntax. However, it faces challenges with model semantics. Most notably, the authors report that multiple iterations with explicit requests for modification are required to align the model with the user's intent. Thus, developing a model usually involves an ongoing dialogue with ChatGPT rather than a simple request-response interaction.

In a recent work, Silva et al. [19] proposed a framework to break the task of model generation into separated tasks of class, attribute, and relationship generation using a tree-of-thought framework that allows an LLM to explore several possibilities in the solution space and then choose the best alternative. Their approach performs well to accurately predict the classes, but, similar to the findings in [8], struggles with relationships. Moreover, they do not provide a comparative analysis with existing works rendering the generalizability of their approach unclear.

[1] https://platform.openai.com/docs/guides/gpt-best-practices.

2.3 Prompting Techniques for Domain Modeling

White et al. [21,22] provide a catalog of prompt engineering techniques that have been applied to solve common problems when interacting with LLMs. The authors conclude that these prompt patterns significantly enrich the capabilities that can be achieved in a conversational LLM. Furthermore, they conclude that prompt patterns are generalizable to many different domains. Kim et al. [13] formulate prompt templates and conduct comprehensive experiments to assess the impact of in-context examples on LLM-based evaluation. Their experiments reveal that providing clear and straightforward instructions akin to those explained to humans proved to be more effective compared to unstructured and unclear prompts.

2.4 Synopsis

Based on the analysis of existing related studies, we identify the following key observations. First, the quality of the generated domain models is largely influenced by the prompt. Iterative improvements to the initial results are always required but are not explicitly supported in existing works. Second, most studies do not consider the granularity of the domain description. Typically, the entire text is used to generate the models without breaking it into smaller pieces that can fit better within the content window and be incrementally used to build the final domain model. Some works, e.g., [2], do break the task into sub-tasks of generating the classes and relationships separately. However, they do so with the entire domain description. Moreover, they do not combine the domain models generated from each (story) paragraph to produce a consolidated model.

To address these gaps, this study focuses on the granularity of domain descriptions used in input prompts and the strategies employed for model generation. Accordingly, explicit support for model improvements through update operations is incorporated into the incremental generation process.

3 Research Methodology

3.1 Experimental Design and Scenarios

Figure 1 provides a high-level overview of the experimental design, which involves eight scenarios, generated from three binary variables: granularity (whole text vs. by-paragraph), modeling strategy (model-based vs. list-based), and task. In the context of granularity, “whole text” refers to the complete domain description provided in its entirety, without any segmentation or breakdown. This is the full context given to the model in one go. “By-paragraph” means the domain description is split into individual paragraphs, and each paragraph is treated as a separate input to the model. The idea is to process and analyze each part sequentially or independently. Each paragraph typically focuses on a specific sub-topic or aspect of the domain. In the case of strategy, while we note that different techniques or instructions can be provided to the LLM about *how* to construct a model or list, in this work, we utilize the LLMs *understanding* or *capability* of constructing a list or a model from the domain description by providing straightforward instructions as shown in Table 3. We utilized two common domain modeling tasks: functional modeling via use case diagrams (UCD) and structural modeling via class diagrams (CD). The eight scenarios are summarized in Table 2, while Fig. 2 shows the required interactions with the LLMs for each scenario. For scenarios 5 to 8, we instructed the LLMs to first generate separate lists of actors and use cases for use case modeling, and lists of classes for class modeling. These lists were subsequently used as the basis for model generation.

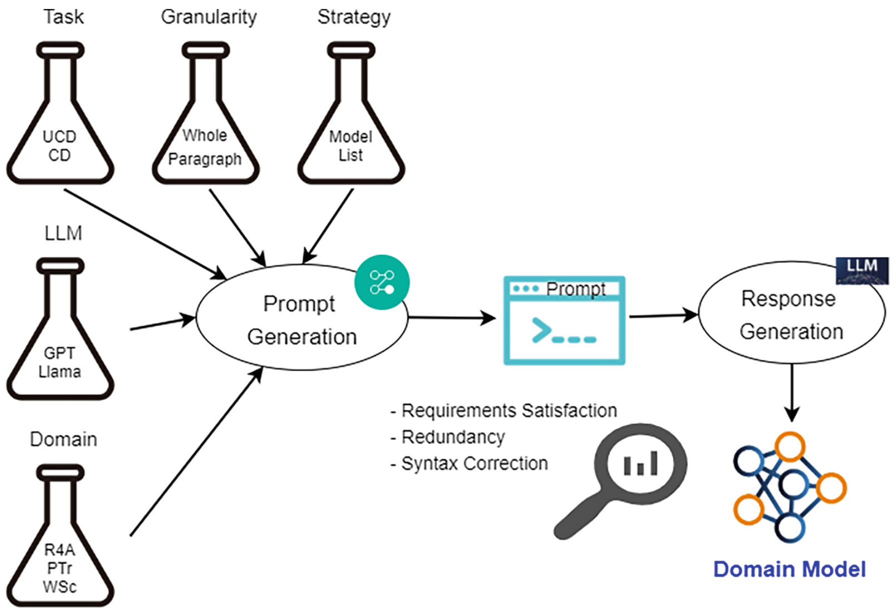


Fig. 1. High-level overview of the experimental design

The scenarios were applied to three distinct application domains using two LLMs: GPT-4o and Llama-3.1-70b-versatile. The investigated application domains were: R4A (Rating for All) for viewership data analytics, PTr (Perfect Trip) for tourism management, and WSc (Witchery School) for school acceptance management. To avoid introducing additional confounding variables, the domains were kept to a comparable size and described using seven paragraphs of text each, referring to different aspects of the domains. The entire experimental material can be found in [16].

3.2 Templates and Tasks

Building on our findings in [1], we employed a set of five templates designed to support the key tasks of domain modeling: *Create List* and *Update List* were used to respectively create and incrementally update a list of elements, such as use cases, actors, or classes; *Generate Model* was used to generate a model (e.g., use case or class diagram) from the previously created and potentially updated lists; and *Create Model* and *Update Model* facilitated the direct creation and incremental refinement of models. In all scenarios, we requested that the models be presented in the common format of PlantUML, while we did not provide

Table 2. The experimental scenarios

Scenario	Granularity	Strategy	Task	# of prompts
1	Whole	Model	UCD	1
2	Whole	Model	CD	1
3	By-paragraph	Model	UCD	7
4	By-paragraph	Model	CD	7
5	Whole	List	UCD	3
6	Whole	List	CD	2
7	By-paragraph	List	UCD	15
8	By-paragraph	List	CD	8

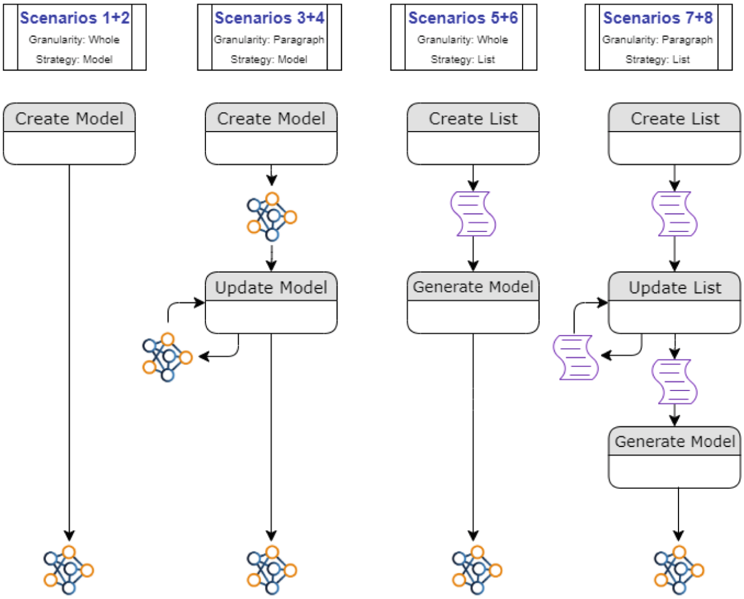


Fig. 2. The required LLM interactions for the different experimental scenarios

a concrete format for the lists. Table 3 provides an overview of the templates, including their descriptions and format.

Table 3. Templates Used for IS Modeling Tasks

Template	Description	Prompt
Create List	Create a list of elements (e.g., use cases, actors, classes) based on the description.	Create a list of <*elements*> from the following description. The description: <*desc*>
Update List	Modify or refine an existing list of elements to incorporate new information or correct errors expressed in the concern.	Modify the list of <*elements*> to address the following concern. The concern: <*conc*>
Generate Model	Create a model (e.g., use case diagram or class diagram) from the list of elements generated for previous prompts.	Generate a <*model*> from the list of <*elements*> generated in the previous prompt. Present the response in the following format: PlantUML.
Create Model	Create a model directly from a given description.	Create a <*model*> from the following description. Present the response in the following format: PlantUML. The description: <*desc*>
Update Model	Modify an existing model to address a specific concern.	Modify the <*model*> to address the following concern. Present the response in the following format: PlantUML. The concern: <conc>

Figure 3 illustrates the model generated by GPT for Scenario 4 in the Perfect Trip (PTr) domain. The LLM (GPT in this case) was asked to create and refine (update) a class diagram based on the sequential input of each paragraph from the PTr description. The resulting model, which considers all seven paragraphs, effectively identifies core relevant classes such as *User*, *Member*, *Trip*, *Place*, *Visit*, *Opinion* (interpreting Review), and *Recommendation*. However, it also includes several irrelevant classes derived from the description. These classes, while mentioned in the text, pertain more to the system's functionality and are better suited for inclusion in a use case diagram rather than a class diagram. Examples of such misclassified elements include *MZ System*, *VP of Content*, *VP of Culture*, and *World Tourism Organization*, which are actors interacting with the system, as well as *Search*, which represents a potential use case. The model has further inaccuracies regarding attributes, associations, and missing classes.

3.3 Evaluation Procedure

The final output for each scenario, representing the last modeling step, was evaluated by two undergraduate students who had previously served as teaching assistants (TA) in an IS modeling course. Each TA assessed the outputs of one

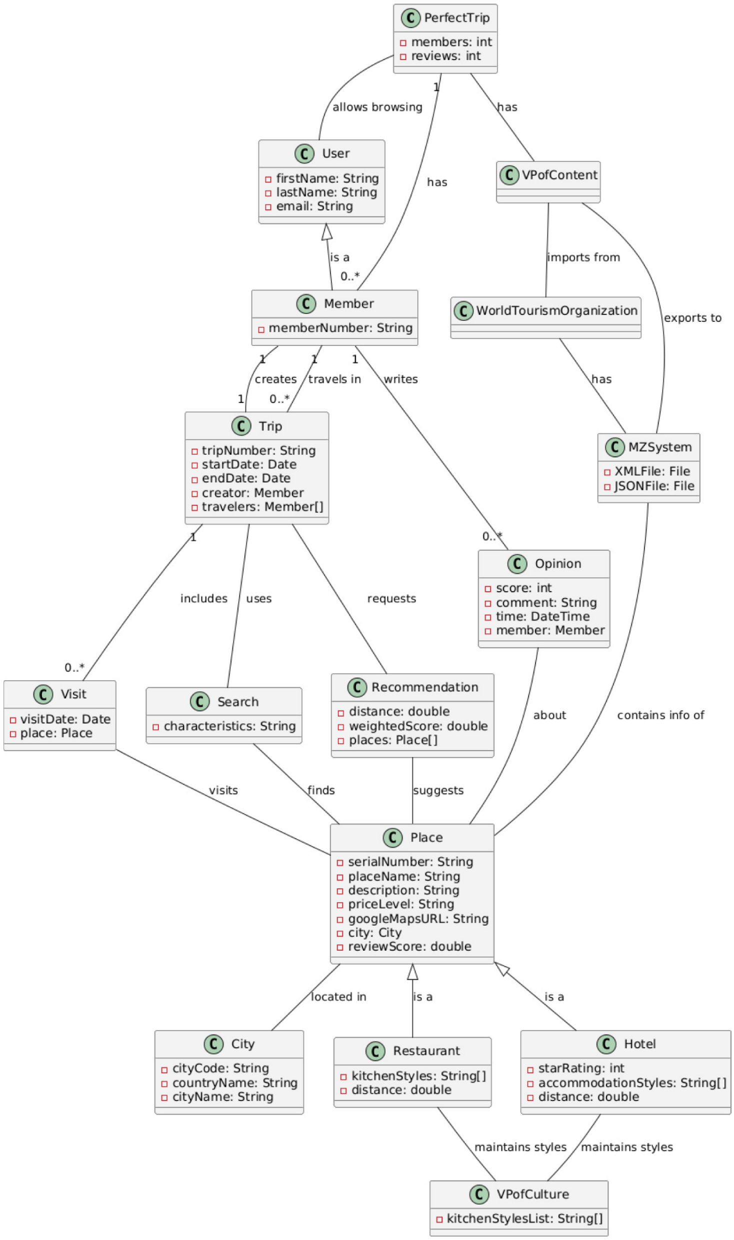


Fig. 3. A model generated by GPT for Scenario 4 in the PTr domain.

to two application domains. The evaluation criteria included both semantic and syntactic criteria:

– **Requirements Satisfaction**: The extent to which the outcomes align with the requirements outlined in the provided descriptions, including both *correctness* and *completeness*.
– **Redundancy**: The identification of unnecessary elements or relationships that were not mentioned in the descriptions. This can be partially associated with LLMs' *hallucination*.
– **Syntactic Correctness**: Identification of errors in the syntax of the model (UCD or CD) or in the PlantUML format.

To assess satisfaction of requirements, we first extracted model fragments from the descriptions, rather than generating complete reference solutions. This approach provides flexibility and allows for variations in the solutions. The extracted model fragments were of three types: use case fragment (including its actors), class fragment (including its attributes and operations), and association fragment (including the associated classes and potential attributes of the association class). Table 4 exemplifies part of the PTr description that is relevant to the creation of a class diagram, along with the expected model fragments.

Table 4. Examples of Specification and Expected Fragments for the Perfect Trip Domain

Specification	Expected Fragments
Perfect Trip's repository includes information on places of interest in different cities of the world. For each place, the file contains a unique serial number, the name of the place, a description, the price level (high/medium/low), the corresponding landmark on Google Maps expressed as a URL, and the city in which the place is located.	**class**.Place; attribute.serialNumber; attribute.placeName; attribute.description; attribute.level; attribute.linkToGoogleMaps; operation.calcWeightedScores **class**.PriceLevel; value.high; value.medium; value.low **association**.City-Place
City codes are unique and composed of the country number and a distinctive combination of three letters representing the city's name. For example, the city code for Tel Aviv is 972TLV, where 972 denotes the State of Israel's code.	**class**.City; attribute.cityCode; attribute.cityName **class**.Country; attribute.countryCode; attribute.countryName **aggregation**.Country-City
Restaurants and hotels are special kinds of places.	**class**.Hotel; inherits.Place **class**.Restaurant; inherits.Place
For each restaurant, the relevant kitchen styles should be kept.	**class**.Kitchen style; attribute.styleNumber; attribute.styleName **association**.Kitchen style-Restaurant
For each hotel, besides the previously mentioned details, we aim to include its star rating (ranging from 0 to 5), supported accommodation styles (AI - all-inclusive, BB - bed & breakfast, HB - half board, FB - full board and RO - room only). Note that a hotel may offer some or all accommodation styles.	**class**.Hotel; attribute.starRating; attribute.AI; attribute.BB; attribute.HB; attribute.FB; attribute.RO

With the extracted model fragments in hand, the evaluators were asked to assess the final models generated by the LLMs by scoring them against the various requirements and providing detailed comments on their assessments. To ensure the quality of the evaluations, two of the authors of this paper reviewed a sample of four models each (eight in total) and approved the TAs' assessments. The evaluation of the model in Fig. 3, based on the *partial list* of requirements outlined in Table 4, identified a missing enumeration class for the price level. Overall, the model achieved a requirements satisfaction score of 71.2%, a redundancy rating of 1 (very high), and a syntactic correctness score of 0 (no issues identified).

4 Results

The results were aggregated, such that each outcome (i.e., scenario per LLM) got a score on each evaluation criterion and analyzed to determine the impact of the experimental factors on model quality. As each criterion respond to multiple RQs, we indicate the related RQ in brackets within the subsequent text.

4.1 Requirements Satisfaction Results

Our scores for requirements satisfaction results for all experimental factors range from 15.4% to 98.6% as shown in Fig. 4. This wide range highlights the variability in LLM performance under different conditions. The Kolmogorov-Smirnov tests confirmed that the data adhered to normality assumptions. Consequently, t-tests were performed, showing no significant differences in mean scores for granularity (by-paragraph vs. whole, $p = 0.6409$) [**RQ1**], modeling strategy (list vs. model, $p = 0.3472$) [**RQ2**], or LLM (GPT vs. Llama, $p = 0.7035$) [**RQ3.1**]. ANOVA tests on domain (WSc, PTr, R4A) showed no significant effect on the requirements satisfaction score ($p = 0.8794$) [**RQ3.2**]. However, a significant difference was found between tasks (CD vs. UCD, $p = 0.0013$), with UCD showing higher mean scores [**RQ3.3**].

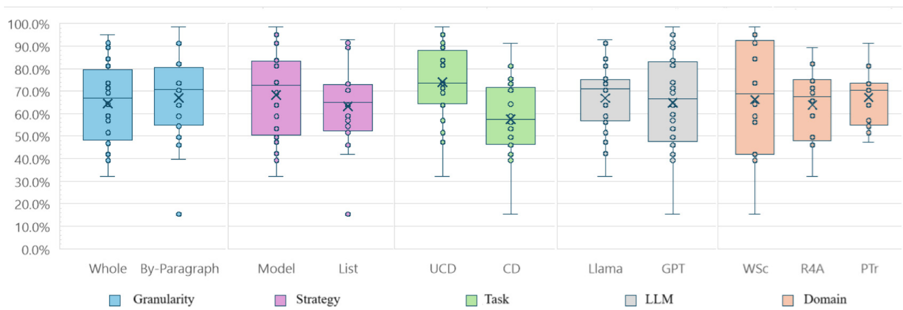


Fig. 4. Results of Requirements Satisfaction

A deeper analysis of requirements with low average scores (below 70%) across all scenarios or high standard deviations (above 33%) highlighted several constructs that posed challenges for the LLMs. In use case modeling, the most prominent issues were identifying *include* and *extend* dependencies and misinterpreting system boundaries. The latter often led to missing actors, use cases, or associations between specific actors and use cases [**RQ3.3**]. In class modeling, the LLMs struggled particularly with association classes, inheritance relationships, and enumeration types. These findings are aligned with other studies in the literature, as reviewed in Sect. 2.2. Common errors included overlooking or misplacing attributes of associations, incorrectly identifying super- and subclasses, and mislocating attributes of super-classes. Additionally, attributes of enumeration types were frequently misidentified as string attributes.

4.2 Redundancy and Syntactic Correctness Results

Wilcoxon tests revealed significant differences in Redundancy between granularity levels ($Z = -2.4392$, $p = 0.0186$) [**RQ1**] and tasks ($Z = -2.3629$, $p = 0.0223$) [**RQ3.3**]. Granularity at the by-paragraph level resulted in higher redundancy compared to the whole-description approach. This observation can be explained by the partial context provided to the LLM in the by-paragraph granularity, which "encourages" it to supplement or invent information that may already exist or be introduced differently in subsequent paragraphs. The segmented processing may lead the LLM to produce redundant content to ensure comprehensiveness.

For tasks, class modeling (CD) exhibited greater redundancy than use case modeling (UCD). This can be attributed to the inherently abstract and structural nature of CD tasks, which often involve defining and describing relationships, hierarchies, and attributes in detail. To ensure the correctness of these complex constructs, the LLM may over-communicate or reiterate information. In contrast, UCD tasks tend to have more explicit requirements, providing clearer guidance and reducing opportunities for redundancy [**RQ3.3**].

Syntactic errors were observed across all combinations of Granularity, Strategy, LLM, and Domain. Many of these errors stemmed from the LLM's difficulty in distinguishing between information relevant to use case modeling (UCD) and class modeling (CD). Consequently, some outputs improperly combined classes with use cases or actors, resulting in a violation of the PlantUML syntax. The most extreme instance occurred in Scenario 3, where Llama produced a model containing only classes and actors, despite the task being to create a use case diagram. Additional syntactic errors included outputs where actors lacked corresponding use cases and vice versa, as well as comments in parentheses being misinterpreted as operations rather than attributes. These issues highlight the LLM's struggle to maintain strict adherence to modeling conventions and emphasize the need for post-processing mechanisms to improve the generated models.

4.3 Discussion and Implications

In the following, we report on the implications of our findings to LLM-based domain modeling.

The severe limitations observed in LLM-generated outcomes regarding requirements satisfaction, redundancy, and syntactic correctness emphasize the need for active human involvement in the domain modeling process. Specifically, LLMs struggle with accurately identifying relationships, such as *include* and *extend* dependencies in use case diagrams, and *associations*, *inheritance* relations, and *association classes* in class diagrams. These challenges align with findings from other recent studies [5,8,19], highlighting that, despite their potential, LLMs are not yet capable of autonomously generating high-quality domain models. Consequently, practitioners must act as both facilitators and validators to align models with the intended requirements. This reliance on human involvement not only underscores the limitations of current LLMs but also calls for more targeted research for developing LLM-assisted (rather than LLM-generated) domain modeling approaches.

> **Implication I:** LLM-assisted, rather than LLM-generated, domain modeling methods should be developed to enable iterative interactions with the LLM and incorporate validation processes to enhance model accuracy and ensure alignment with requirements.

The findings also suggest that while by-paragraph granularity does not significantly affect requirements satisfaction, it does increase redundancy, raising concerns about LLMs' ability to handle large, complex domain descriptions. This highlights the need for improved methods that support engineering prompts for real-world domain descriptions, which are typically multifaceted and more extensive. Our findings further show that if the descriptions include information relevant to different aspects of the domain, such as the domain functionality and structure, the LLMs often become "confused" and include incorrect elements (e.g., placing classes in use case diagrams or actors in class diagrams). Additionally, they sometimes attempt to enforce the creation of model elements, like representing actors or use cases as classes, as shown in the example in Fig. 3.

> **Implication II:** Domain descriptions can be split and provided sequentially to LLMs, which may have a marginal effect on requirements satisfaction but could significantly impact redundancy. This requires careful consideration of how the descriptions are split to minimize redundancy and maintain model correctness.

The lack of significant differences in the mean scores for the strategy variable (list-based vs. model-based) suggests that the choice of modeling strategy does not substantially impact the correctness of LLM-generated models. This finding implies that both strategies are similarly effective in guiding LLMs to produce correct domain models. However, this result also highlights the flexibility of LLMs in adapting to different approaches without significant performance variation, which could be advantageous for modelers with varying preferences or expertise. Nonetheless, future research could explore other strategies and different qualities, such as usability, scalability, or time efficiency.

Implication III: Model-based and list-based modeling strategies result in models of similar quality, allowing modelers to choose the approach that best suits their preferences and expertise.

4.4 Threats to Validity

The discussion of threats to the validity of this research is aligned with the major threat categories introduced by Wohlin et al. [23]. With respect to *conclusion validity*, our results are limited to the scope and extent covered by our scenarios. In total, we focused on eight scenarios (cf. Table 2) which we applied to three distinct application domains using two LLMs. In total, this resulted in 48 models being created through the execution of 132 prompts.

Construct validity threatens the validity of our assessment of the LLM-generated models, as the evaluation was primarily conducted by graduate students. To address this, we briefed the students, who were already experienced in evaluating modeling tasks through their role as teaching assistants. We further prepared them by providing expected model segments, offering feedback on their initial evaluations, and guiding them on any questions they had. Additionally, two authors of this paper reviewed and approved samples of their evaluations to ensure the quality of the assessment.

We do not see a strong human-focused threat regarding the *internal validity* as we prompted the LLMs systematically with identical prompts. Unlike many other studies, we did not rely on human modelers or prompters.

Finally, to address *external validity* concerning the generalizability of our findings, we used three different application domains to ensure that the results are not incidental. This was intended to enhance the potential transferability of our conclusions. The chosen domains were common and expected to be familiar to LLMs. Furthermore, the fact that we used two LLMs, amongst the vast array of LLMs from different vendors, can be an external validity threat in our work. However, we chose two state-of-the-art LLMs to mitigate this threat. Moreover, while we acknowledge that the rapid development of LLMs poses a threat that our results will soon become outdated, it is important to emphasize that advances in LLMs do not necessarily translate into corresponding improvements in domain modeling. The primary focus of LLM development lies in enhancing general reasoning capabilities, rather than specifically targeting domain modeling or model-driven engineering. Although improved reasoning can support domain modeling tasks, existing studies have shown that even the substantial leap from

GPT-3.5 to GPT-4 yielded comparable results and revealed similar limitations in this context [8,19]. This suggests that achieving qualitatively different results may require training LLMs specifically for domain modeling tasks.

5 Conclusion

The use of LLMs for domain modeling is still in its early stages, with initial, explorative results. A systematic and maturated approach to utilizing LLMs in modeling is still greatly needed. In this paper, we presented our investigation into the impact of different levels of granularity (whole text vs. by-paragraph), modeling strategies (model-based vs. list-based), and tasks (use case modeling vs. class modeling) on the quality of LLM-generated domain models, with a particular focus on semantic and syntactic correctness. Our results demonstrated that state-of-the-art LLMs produce domain models of varying quality, with significant findings regarding the impact of granularity on redundancy (by-paragraph resulted in more redundancies) and the effect of task on requirements satisfaction (class diagrams resulted in less satisfied requirements). The value of this paper goes beyond these experimental findings, by providing a comprehensive discussion and raising implications for LLM-assisted domain modeling. These research challenges must be addressed before LLMs can be considered co-modelers in the domain modeling process – similar to the role Copilot plays in software engineering.

Future research includes improving LLM-assisted domain modeling methods to handle larger and more complex domain descriptions. This requires developing more effective strategies for interactions and integrating robust validation processes to enhance model quality. Furthermore, exploring the integration of LLMs with human modelers in a collaborative co-creation environment will be crucial for advancing LLM-assisted modeling. Research into new prompt engineering techniques that minimize redundancy and improve model correctness will also be key to ensuring that LLMs can be effectively applied in real-world domain modeling scenarios. In addition, we plan to explore the assistance of LLMs in behavioral modeling tasks, such as generating state diagrams, to assess their potential beyond functional and structural modeling. Finally, we plan to explore the process of LLM-assisted domain modeling by analyzing both intermediate and final generated models, as well as the paths explored during the modeling process.

Acknowledgment. The authors would like to sincerely thank Neta Mandelbaum and Matan Meirovitz for their invaluable assistance in evaluating the LLM-generated domain models.

References

1. Ali, S.J., Reinhartz-Berger, I., Bork, D.: How are LLMs used for conceptual modeling? an exploratory study on interaction behavior and user perception. In: Maass, W., Han, H., Yasar, H., Multari, N.J. (eds.) Conceptual Modeling - 43rd International Conference, ER 2024, Pittsburgh, PA, USA, 28-31 October 2024, Proceedings. LNCS, vol. 15238, pp. 257–275. Springer (2024). https://doi.org/10.1007/978-3-031-75872-0_14
2. Arulmohan, S., Meurs, M.J., Mosser, S.: Extracting domain models from textual requirements in the era of large language models. In: 2023 ACM/IEEE International Conference on Model Driven Engineering Languages and Systems Companion (MODELS-C), pp. 580–587. IEEE (2023)
3. Bajaj, D., Goel, A., Gupta, S., Batra, H.: MUCE: a multilingual use case model extractor using GPT-3. Int. J. Inf. Technol. **14**(3), 1543–1554 (2022)
4. Burgueño, L., Clarisó, R., Gérard, S., Li, S., Cabot, J.: An NLP-based architecture for the autocompletion of partial domain models. In: Advanced Information Systems Engineering - 33rd International Conference, CAiSE 2021, Australia, 2021. vol. 12751, pp. 91–106. Springer (2021). https://doi.org/10.1007/978-3-030-79382-1_6
5. Cámara, J., Troya, J., Burgueño, L., Vallecillo, A.: On the assessment of generative AI in modeling tasks: an experience report with ChatGPT and UML. Softw. Syst. Model. **22**(3), 781–793 (2023)
6. Chen, B., t al.: On the use of GPT-4 for creating goal models: an exploratory study. In: 2023 IEEE 31st International Requirements Engineering Conference Workshops (REW), pp. 262–271. IEEE (2023)
7. Chen, B., Yi, F., Varró, D.: Prompting or fine-tuning? A comparative study of large language models for taxonomy construction. In: 2023 ACM/IEEE International Conference on Model Driven Engineering Languages and Systems Companion (MODELS-C), pp. 588–596. IEEE (2023)
8. Chen, K., Yang, Y., Chen, B., López, J.A.H., Mussbacher, G., Varró, D.: Automated domain modeling with large language models: a comparative study. In: 2023 ACM/IEEE 26th International Conference on Model Driven Engineering Languages and Systems (MODELS), pp. 162–172. IEEE (2023)
9. Fill, H.G., Fettke, P., Köpke, J.: Conceptual modeling and large language models: impressions from first experiments with ChatGPT. Enterprise Model. Inf. Syst. Archit. (EMISAJ) **18**, 1–15 (2023)
10. Francu, J., Hnetynka, P.: Automated generation of implementation from textual system requirements. In: Software Engineering Techniques - Third IFIP TC 2 Central and East European Conference, CEE-SET, Czech Republic, 2008. LNCS, vol. 4980, pp. 34–47. Springer (2008). https://doi.org/10.1007/978-3-642-22386-0_3
11. Herchi, H., Abdessalem, W.B.: From user requirements to UML class diagram. CoRR abs/1211.0713 (2012). http://arxiv.org/abs/1211.0713
12. Jahan, M., et al.: Automated derivation of UML sequence diagrams from user stories: unleashing the power of generative AI vs. a rule-based approach. In: Proceedings of the ACM/IEEE 27th International Conference on Model Driven Engineering Languages and Systems, pp. 138–148 (2024)
13. Kim, J., et al.: Which is better? Exploring prompting strategy for LLM-based metrics. arXiv preprint arXiv:2311.03754 (2023)

14. Moody, D.L., Sindre, G., Brasethvik, T., Sølvberg, A.: Evaluating the quality of information models: empirical testing of a conceptual model quality framework. In: Clarke, L.A., Dillon, L., Tichy, W.F. (eds.) Proceedings of the 25th International Conference on Software Engineering, 3-10 May 2003, Portland, Oregon, USA, pp. 295–307. IEEE Computer Society (2003). https://doi.org/10.1109/ICSE.2003.1201209
15. Nelson, H.J., Poels, G., Genero, M., Piattini, M.: A conceptual modeling quality framework. Softw. Qual. J. **20**(1), 201–228 (2012). https://doi.org/10.1007/S11219-011-9136-9
16. Reinhartz-Berger, I., Ali, S.J., Bork, D.: Leveraging LLMs for domain modeling: the impact of granularity and strategy on quality - Online supplementary material (2025). https://doi.org/10.5281/zenodo.15192220
17. Robeer, M., Lucassen, G., van der Werf, J.M.E.M., Dalpiaz, F., Brinkkemper, S.: Automated extraction of conceptual models from user stories via NLP. In: 24th IEEE International Requirements Engineering Conference, RE China, 2016, pp. 196–205. IEEE Computer Society (2016). https://doi.org/10.1109/RE.2016.40
18. Saini, R., Mussbacher, G., Guo, J.L.C., Kienzle, J.: Machine learning-based incremental learning in interactive domain modelling. In: 25th International Conference on Model Driven Engineering Languages and Systems, MODELS, Canada, 2022, pp. 176–186. ACM (2022). https://doi.org/10.1145/3550355.3552421
19. Silva, J., Ma, Q., Cabot, J., Kelsen, P., Proper, H.A.: Application of the tree-of-thoughts framework to LLM-enabled domain modeling. In: Maass, W., Han, H., Yasar, H., Multari, N.J. (eds.) Conceptual Modeling - 43rd International Conference, ER 2024, Pittsburgh, PA, USA, 28-31 October 2024, Proceedings. LNCS, vol. 15238, pp. 94–111. Springer (2024). https://doi.org/10.1007/978-3-031-75872-0_6
20. Weyssow, M., Sahraoui, H.A., Syriani, E.: Recommending metamodel concepts during modeling activities with pre-trained language models. Softw. Syst. Model. **21**(3), 1071–1089 (2022). https://doi.org/10.1007/S10270-022-00975-5
21. White, J., et al.: A prompt pattern catalog to enhance prompt engineering with ChatGPT. arXiv preprint arXiv:2302.11382 (2023)
22. White, J., Hays, S., Fu, Q., Spencer-Smith, J., Schmidt, D.C.: ChatGPT prompt patterns for improving code quality, refactoring, requirements elicitation, and software design. arXiv preprint arXiv:2303.07839 (2023)
23. Wohlin, C., Runeson, P., Höst, M., Ohlsson, M.C., Regnell, B., Wesslén, A.: Experimentation in Software Engineering. 2nd edn. Springer (2024). https://doi.org/10.1007/978-3-662-69306-3
24. Yang, S., Sahraoui, H.A.: Towards automatically extracting UML class diagrams from natural language specifications. In: 25th International Conference on Model Driven Engineering Languages and Systems: Companion Proceedings, MODELS, Canada, October 2022, pp. 396–403. ACM (2022). https://doi.org/10.1145/3550356.3561592

AI-Based Requirements Analysis Assistant that Applies Explicit Knowledge and Includes Humans in the Loop

Steven Alter(✉)

University of San Francisco, 2130 Fulton St., San Francisco 94117, USA
alter@usfca.edu

Abstract. This exploratory paper builds on the EMMSAD 2024 paper "Could a Large Language Model Contribute Significantly to Requirements Analysis?" Eight versions of each of three LLM prompts (for system structure, analysis, and recommendations) were applied to three 3000+ word case studies. Those versions expressed different "treatments" including a control with no RAG augmentation, a version with RAG augmentation based on an analysis template used by MBA and EMBA students, and six other versions based on theoretical approaches such as activity theory, a BPM design space, work system principles, and so on. The LLM responses were somewhat reliable for summarizing system structure, less reliable for summarizing an analysis, and often generic and impractical for recommendations because the LLM did not understand contexts.

This new paper adds three new capabilities: 1) RAG augmentation using a knowledge base consisting of "knowledge objects" built on work system theory, 2) application of that knowledge base using chain-of-thought reasoning, 3) inclusion of direct feedback from analysts during an analysis process in order to correct errors and to extend the prompt in new directions. Examples are used to illustrate results from applying those capabilities to 3 disparate case studies.

Keywords: Large Language Model · Requirements Analysis · Work System Theory · Facets of Work

1 Applying LLMs Despite their Known Shortcomings

As noted by [1], the enormous wave of attention to large language models (LLMs), especially ChatGPT, combines enthusiasm, fear, and initial applications in many organizations. LLM-based chatbots often produce grammatical text that is reasonably well-structured and approximately correct [2]. Their shortcomings include lack of common sense and real-world knowledge despite being "excellent at processing longer sequences of data –like text– by using self-attention processes that enable the model to focus on different areas of the input" [3]. They tend to fabricate facts or ideas (to "hallucinate"), as is discussed widely. The idea of "prompt engineering" arose from the common difficulty in specifying "prompts" (instructions for an LLM) specific enough to obtain adequate

J. Krogstie et al. (Eds.): CAiSE 2025, LNCS 15701, pp. 20–36, 2025.
https://doi.org/10.1007/978-3-031-94569-4_2

responses. On the other hand, research published between 2023 and early 2025 demonstrates that LLMs can be applied carefully to professional tasks in consulting, software development, and other professional fields.

Impetus and Relevance to Requirements Analysis (RA). This paper builds on results from [1] which attempted to use ChatGPT to generate reasonably good first cuts at management briefings similar to over 700 briefings produced by (mostly) MBA and EMBA students using versions of a systems analysis template for business students (e.g., [4]). Those preliminary, management-oriented briefings addressed the initial aspects of RA by focusing on requirements for specific work systems rather than technical requirements or documentation of the hardware or software that those work systems might use. Using an LLM to produce management-oriented briefings of that type might contribute to RA in real world situations. Later stages of RA would build on that basis to produce rigorous and internally consistent documentation of process flows (if processes are highly structured), business rules, detailed information and technology requirements, and specification of constraints that may apply.

Limitations of LLM usage results described in [1] suggest that an AI-based RA assistant should incorporate three additional types of capabilities to minimize or correct confusions and errors: 1) use of an organized knowledge base related to information systems, 2) application of that knowledge base in chain-of-thought reasoning, whereby the prompt would subdivide the LLM's "thinking" into a series of connected steps that lead to a cumulative result, 3) inclusion of direct feedback from analysts during an analysis process in order to correct errors and to extend the prompt in new directions.

Research Goal and Approach. The goal is to demonstrate an LLM-based method for supporting preliminary stages of IS requirements analysis for problematic IT-enabled work systems in organizations. This paper's new approach combines the speed and linguistic capabilities of LLMs with a specified subset of knowledge about information systems in organizational settings and guidance by a human analyst. The knowledge is built around work system theory (WST – Fig. 1) [5].

- WST is expressed briefly as a series of (mostly) RDF-like statements about work system-related entities and relationships.
- An IS-related knowledge graph (ISKG) consists of knowledge objects (KOs) from work system theory (WST) and its extensions. The ISKG is expressed as a series of short sentences that could be incorporated into a knowledge graph. The KOs in the ISKG include concepts, generalizations, and methods or tools such as analysis templates related to facets of the elements of the work system framework. The concepts include frequently relevant characteristics, evaluation criteria, and phenomena. The generalizations include principles, frameworks, and theories.
- A multi-stage prompt uses chain-of-thought reasoning to instruct an LLM to apply the ISKG while answering a series of queries about a work system supported by an information system and while permitting an analyst to provide corrections and introduce additional prompts along the way.
- The LLM creates a semi-structured overview (SSO) that incorporates its responses as corrected or improved interactively by the analyst. Calling this an SSO emphasizes its form as human-readable text rather than diagrams or code. Prescribing a structured version would have limited the current exploratory effort.

The proposed RA assistant is designed to produce an SSO that can support early stages of RA, such as describing (documenting textually) the current system and identifying its strengths and weaknesses and other relevant factors. Those early stages matter because subsequent RA stages build on them. The trial version of the RA assistant is meant as an exploratory step toward a more complete RA assistant based on AI. A more capable assistant might address engineering questions related to performance expectations, software scope, quality constraints, assumptions, priorities, conflicts between requirements, and internal inconsistencies within the analysis. The new approach can be used in conjunction with existing tools and does not attempt to replace them. Additional trials and adjustments could generate new RA methods and tools that help analysts organize what is known or believed about a specific situation and identify opportunities, issues, and challenges that might otherwise be ignored.

Organization. Section 2 identifies recent research that inspired this effort and the approach that it takes. Section 3 summarizes WST as the core of the broader work system perspective. Section 4 uses the idea of facets as a path for creating an ISKG that expands outward from WST to include many knowledge objects (KOs) that are potentially useful for analyzing and designing work systems. Section 5 summarizes key results of applying the current ISKG in a multi-stage prompt that is adjustable for use in different situations. That approach is applied to three disparate case studies. Section 6 summarizes conclusions to date and suggests areas for future work.

Data used in this Research. Use the following link to access the ISKG plus cases, prompts, complete interactions with ChatGPT, and summaries produced by ChatGPT: https://www.dropbox.com/scl/fo/g1pzxsiwehqbg5pl3yslo/AJ0qE2d1LTd9KZ9oColIzU4?rlkey=7i6veoq0k9cnzepvl7zyx36ij&st=pcb7qm2b&dl=0.

Relation to "Bridging Silos" – The CAISE 2025 Theme. Preliminary stages of requirements analysis inherently bridge gaps between situated business knowledge and issues versus IT knowledge and issues. In contrast, rigorous IS engineering techniques and methods place less emphasis on business, user, and usage issues that matter greatly.

Acronyms. These are used to avoid confusion about terms that might be interpreted in different ways: knowledge object (KO), IS knowledge graph (ISKG), large language model (LLM), requirements analysis (RA), semi-structured overview (SSO), work system method (WSM), work system perspective (WSP), work system theory (WST).

2 Recent Research that Inspired the Current Effort

Results presented in [1] led to wondering whether an explicit representation of knowledge combined with user interactions would be more effective than any of the eight treatments used in that research. Other recent research that inspired or influenced this paper's approach starts with applications in consulting work and programming and also includes research concerning IS-related knowledge, KGs, and LLMs.

Experimental uses of LLMs by Employed Consultants. An experiment involving 758 individual-contributor consultants for a leading consulting firm suggests that LLM applications can improve performance in professional work. LLMs were applied to "realistic consulting tasks within the [current] frontier of AI capabilities. Consultants using AI were significantly more productive (they completed 12.2% more tasks on average, and completed tasks 25.1% more quickly), and produced significantly higher quality results (more than 40% higher quality compared to a control group)." [6].

Using LLMs for Software Development. Increasingly important research and practice in a related area involves the use of LLMs to support coding. [7] focuses on code understanding by using an LLM-based conversational user interface that allows open-ended prompts and permits four high-level requests without requiring the user to write prompts. Those requests involve explaining a highlighted section of code, providing details of API calls, explaining domain specific terms, and providing usage examples for the API. [8] uses a code synthesis evaluation framework to rigorously benchmark the functional correctness of LLM-synthesized code. [9] evaluates code snippets generated by ChatGPT in terms of correctness, complexity, and security. [10] uses a code generation framework inspired by three approaches: waterfall, test-driven development, and scrum. "Each model assigns LLM agents specific roles such as requirement engineer, architect, developer, tester, and scrum master…. Through collaborative efforts utilizing chain-of-thought and prompt composition techniques, the agents continuously refine themselves to enhance code quality." The current research applies chain-of-thought but applies it to description and analysis rather than coding.

Applying "Progressive Prompting" when Using an LLM for Programming. Using LLMs for producing code based on rigorous specifications is quite different from using LLMs for producing those requirements. That is apparent from [11], which uses "progressive prompting" to assist in generating code based on system requirements. The LLM in [11] incrementally interprets previously produced specifications to extract functional requirements, create object-oriented models, and generate unit tests and code based on the object-oriented designs. The required inputs are based on understandings of the situation by business and IT professionals. Those components include a project glossary, a statement of the project's goals and expected outcomes, and detailed use cases. The current research enables progressive prompting to increase the quality of description and analysis rather than coding.

Informing an Enterprise Knowledge Graph. [12] integrated the work system framework into an RDF-based knowledge graph that includes operational data. It illustrates potential benefits of representing work systems as knowledge graphs linked to operational data that can be subjected to semantic queries and deductive reasoning. The current research adapts the use of a knowledge graph to bridge a gap between a work system conceptualization and situational specifics.

Using LLMs in Conjunction with Knowledge Graphs. [13] presents many examples of recent research illustrating different ways in which LLMs are being used in conjunction with knowledge graphs. Many of its 80 references are dated between 2021 and 2024. Those references concern trends and research gaps involving KG question answering, ontology generation, KG validation, enhancement of KG accuracy and consistency through LLMs, and other topics. The ISKG is a textual version of a large set of sentences that could be represented as a knowledge graph.

3 Work System Theory and the Work System Perspective

The current research builds on a stream of research that developed the work system method (WSM) [1], a systems analysis method for business professionals. WSM guides high-level analysis of a work system, starting with the main problems or opportunities, analyzing the situation as deeply as is appropriate, and explaining why a possible improvement is or is not worth pursuing. Ideas underlying the WSM were formalized as work system theory (WST), which applies to work systems in general and to information systems, projects, and other special cases of work system. WST and its extensions form a much broader work system perspective (WSP) [14, p. 363] that includes a service value chain framework, a theory of workarounds, a theory of system interactions, facets of work, and other concepts, models, and frameworks not cited here.

Figure 1 shows WST's three components. The ISKG used here builds directly on the work system framework (Fig. 1), which identifies nine elements of a basic understanding of a work system. The work system life cycle model is not used here.

WST and WSP apply to information systems because they are work systems whose essential activities are devoted to processing information. Most information systems exist to produce product/services for other work systems, which may be information systems on their own right. Projects are work systems designed to produce specific results and then go out of existence. IS development is a work system because it is a project that creates or significantly modifies an information system. As with other types of work systems, information systems may be sociotechnical (where human participants perform at least some of the work) or may be totally automated.

Definition of work system. A system in which human participants and/or machines perform work (processes and activities) using information, technology, and other resources to produce specific product/services for internal and/or external customers and/or for themselves.

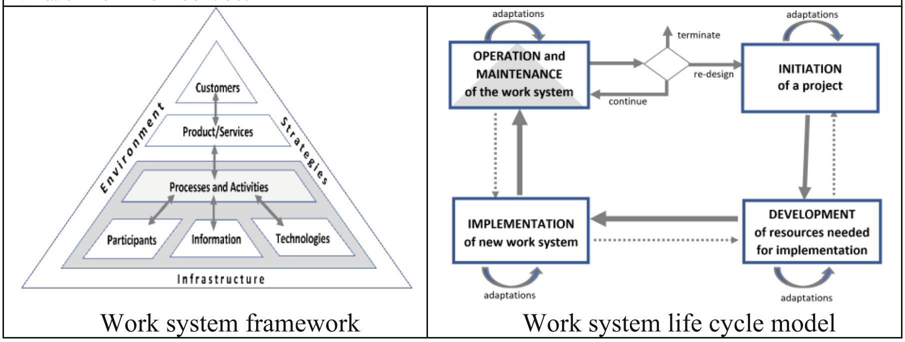


Fig. 1. Three Components of Work System Theory

Facets of Work. The work system framework says nothing about aspects of activities such as making decisions, communicating, processing information, coordinating, controlling execution, and so on. The idea of "facet" is analogous to a facet of a cut gem. It is not a separate component, but rather a face or aspect that can be observed or analyzed.

"Facet" has been defined and used differently in personality studies, information science, and facet modeling in computer science (e.g., [15]).

[16] introduces the idea of "facets of work" and explains the iterative process that identified 18 facets of work based on criteria for deciding whether an aspect of a process or activity might qualify. A facet of work must be easily understood, widely applicable, and associated with concepts and other knowledge, evaluation criteria, and typical design trade-offs that are useful for analyzing work systems and information systems; it must have sub-facets that can be discussed; it must bring open-ended questions for starting conversations [16, pp. 323–331]. Other facets might have been identified. The various facets of work often are not independent in operational systems, as in situations where making decisions involves communicating, learning, and thinking.

Facets of Work System Elements. The idea of facet is a path for identifying important aspects of every element of the work system framework, not just processes and activities. Figure 2 links the original 18 "facets of work" (making decisions, communicating, etc.) to the work system element *processes and activities* while also showing other facets in that broader view. For example, facets of a work system's *environment* include organizational culture, national culture, organizational politics, organizational history, and so on. Most of the facets in Fig. 2 are relevant to many analysis and design situations and satisfy most of the criteria for facets of work. Overall, the idea of facets provides part of a path for identifying requirements and issues that otherwise might be overlooked when analyzing and designing systems in organizations.

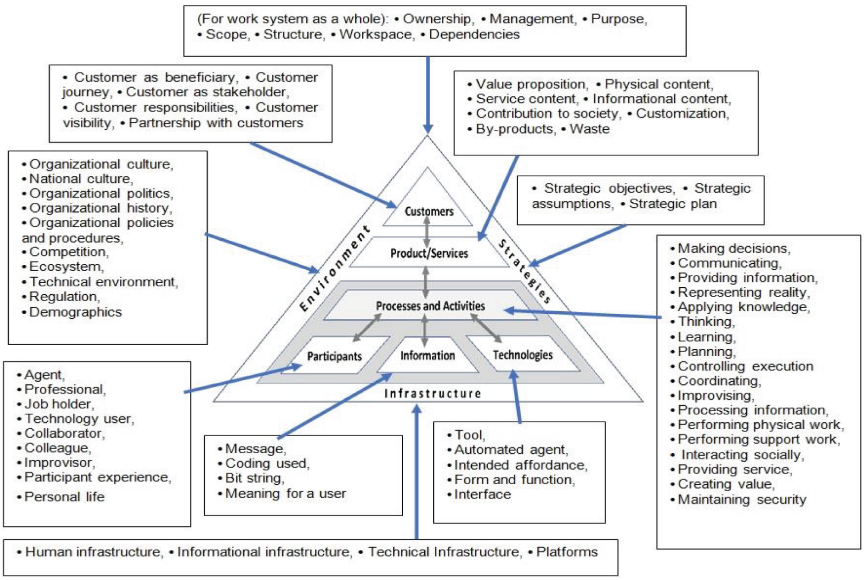


Fig. 2. Facets of work systems and work system elements [16, p. 335]

Knowledge Objects. A knowledge object (KO) is an identifiable item of knowledge. The taxonomy of KOs in Fig. 3 stems from the assumption that science is the creation, evaluation, accumulation, dissemination, synthesis, and prioritization of KOs, including the re-evaluation, improvement, or replacement of existing KOs by other KOs that are more effective within the relevant domain [17]. The taxonomy says that five types of concepts are applied in identifying data and in expressing interpretations, generalizations, and methods, techniques, and tools of various types. KOs of each of those types can be applied directly in requirements analysis.

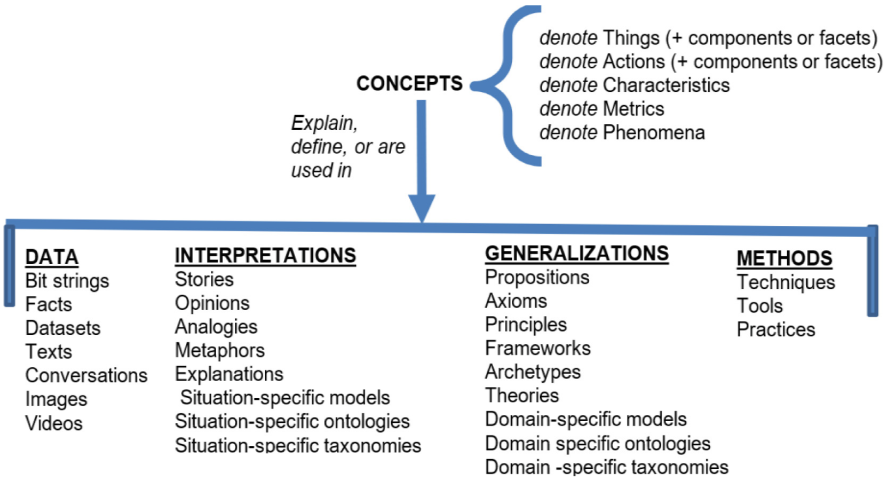


Fig. 3: Taxonomy of knowledge objects [17, pp. 9–10]

4 An AI-Based Requirements Analysis Assistant that Applies Explicit Knowledge and Includes Humans-in-the-Loop

The current version of the AI-based RA assistant (Fig. 4) produces a semi-structured overview (SSO) of the situation based on an ISKG, situation-specific information, and corrections and additional prompts from an interactive user. The core of the ISKG is a textual representation of the work system framework (Fig. 1) and related facets (Fig. 2). The ISKG extends that core by including frequently relevant KOs of the various types shown in Fig. 3. The multi-stage prompt for producing the SSO uses the ISKG in conjunction with case-specific information. The analyst is encouraged to correct LLM responses and to provide additional prompts at each of seven stages. That process generates an SSO that is created and organized using the ISKG. Neither the ISKG nor the SSO are viewed as complete because it is possible to extend the ISKG with new facets and related KOs and because the analysis continually adds and corrects case- specific information that might be added to the SSO where necessary. (cf., [18]).

The ISKG. The ISKG's goal is to facilitate organized access to KOs that LLMs and people might use for describing and analyzing work systems. The design of the ISKG does not aspire to the level of accuracy or specificity that a rigorous conceptual model might try to achieve. That level of precision is not necessary when human analysts work with an LLM (see Fig. 4) in early stages of requirements analysis.

The size of the ISKG used in this research precludes displaying it in diagrammatic form. The KOs in the ISKG were produced in past and ongoing research related to the WSP. Stated as short sentences, the current textual version of the ISKG (available on request) contains 1,594 words related to work system elements, facets of work system elements, and characteristics, evaluation criteria, phenomena, principles, and typical obstacles and risk factors for work systems in general and for individual work system elements. Those types of KOs are not included for facets of work system elements since that would have been overkill for current purposes. A more comprehensive version of an ISKG could link to many other KOs, such as modeling, analysis, and design templates in a proposed toolkit for systems analysis and design [19]. Convenient linkage to templates would require the use of a well-constructed user interface.

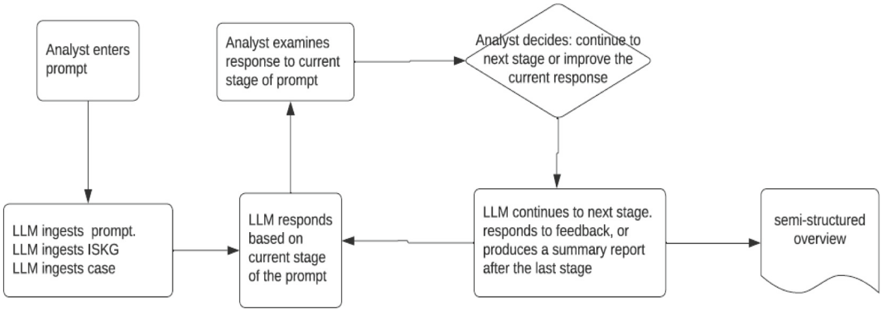


Fig. 4. The process of using the AI-based requirements analysis assistant

The seven sections of the ISKG can be summarized as follows:

Section 1 identifies elements of a work system, starting as follows: A *work system* produces (one or more) *product/services.* A *work system's product/services* are directed toward (one or more) *customers.* A work system's *customers* receive, use, or otherwise benefit directly from its *product/services.* A *work system* operates through (one or more) *activities.* And so on.

Section 2 lists facets of work system in general and of work system elements. As shown in Fig. 3, facets of *product/services* include: *Value proposition, Physical content, Service content, Informational content, Contribution to society, Customization, By-products, Waste*. Figure 3 also shows facets of other work system elements.

Sections 3, 4, and 5 list characteristics, evaluation criteria, and phenomena related to elements of work systems. For example, characteristics of *participants* include *Knowledge, Skills, Goals, Ambitions, Attitudes, Certifications, Age*. Evaluation criteria for *product/services* include *Effectiveness, Cost to customer, Production cost, Usability, Value, Value in Use, Reliability*.

Section 6 lists principles related to work system elements and work systems as a whole. These were developed based partly on sociotechnical theory and partly on feedback from employed EMBA students who evaluated interim versions [20]. For example, principles related to customers include: *Please the customers, Balance priorities of different customers*. Principles related to work system as a whole include: *Maintain compatibility and coordination with other work systems. Incorporate goals, measurement, evaluation, and feedback. Minimize unnecessary risks,* etc.

Section 7 lists typical obstacles and risk factors. For *processes and activities* these include: *Inadequate resources, Inadequate quality controls, Uncertainty about how work should be done, Excessive variability in work practices, Over-structured work practices, Excessive interruptions, Excessive complexity, Inadequate security,* etc.

Nuanced interpretation of many parts of the ISKG calls for a detailed discussion of WST that is beyond the current scope. For example, that discussion would note that *customers* sometimes serve as *participants* (as in medical care, consulting, and education) and would explain why the concept *product/service* is used instead of the separate concepts *product* and *service*. Experience with [1] showed that LLMs tend to confuse or ignore nuances such as those. The ISKG used here was extensive enough to help in generating understandable semi-structured overviews (SSOs) of six case situations including the three disparate situations shown here. Those SSOs seemed to be good enough to contribute to early stage of requirements analysis, but with the caveat that a human analyst would need to identify and correct errors. An example shown later involves ChatGPT-4o's initial misinterpretation of one of the cases.

The Prompt. An adjustable standard prompt was developed by trial and error to enable the following capabilities: 1) ChatGPT-4o should ingest PDFs of both the case itself and the ISKG. That was possible due to ChatGPT-4o improvements regarding expanded context windows and use of PDFs. 2) The prompt should be adjustable to facilitate consistent application to many different cases. That was done by including "[NAME OF CASE]" numerous times in a standard prompt and then replacing that term with a coded name for each case such as RDH403 to customize it for specific cases. 3) The prompt should subdivide the analysis into manageable parts for the benefit of both the analyst and the LLM itself. This was done by dividing the prompt into seven sections corresponding directly to the seven sections of the ISKG. The standard prompt starts as shown below and then goes into separate sections that correspond with the sections in the ISKG. ("Temperature = 0.1" on a scale from 0.0 to 1.0 states that response should be factual and formal rather than creative. Other ChatGPT parameters such as "set seed" and "top P" could have been used but seemed too restrictive for this exploratory study).

> *Temperature = 0.1. I am a manager developing specifications for a new or improved work system. You are my assistant. I need you to answer queries in a formal tone that emphasizes facts and is justified directly by the content of a description of a real world situation in a PDF called [NAME OF CASE]. Your answers to queries shown below in seven sections should try to help me. Each section starts and ends with the symbol ##. After producing the answer to each section, display that answer to me and wait for my permission to continue to the next section. I may ask you to regenerate an answer, may correct errors you made, and may provide additional prompts that are not included in this original prompt. At the end of all seven sections, I will give you permission to produce a final answer consisting of the final answers to each of the seven sections. Those answers should include any improvements or corrections that I provide before*

giving permission to continue to another section. Start your efforts by reading a PDF called ISKG. It provides the ideas to be used in answering the queries about the work system that is described in the PDF called [NAME OF CASE]. ISKG is separated into seven parts that are numbered 1 to 7 and that correspond with queries listed below in Sections 1 through 7. Answers to any of those questions should be consistent with answers to queries answered in previous parts. Please be sure to derive your answers only from the PDF called [NAME OF CASE]. Except in Section 7, avoid using statements that express possibilities using modal verbs such as could, should, or might. In Section 7, comments about risks are about possibilities and therefore can be stated using could, should, or might. Details or suggestions based on knowledge available from the internet or other sources beyond the PDF called [NAME OF CASE] are not relevant and should not be used. Also, do not repeat definitions or generalizations that may appear in ISKG or in any section of this prompt.

The Cases. The approach illustrated in Fig. 4 was applied to six case examples whose noninformative six-character identifiers were used to simplify communication with ChatGPT-4o. The case examples were selected based on personal familiarity or were found using searches of Google Scholar. None of the cases was designed to illustrate topics covered by the standard prompt, which was designed to cover a wide range of potential issues that sometimes require attention in real world situations. Three of the six cases were selected for discussion in this paper because they illustrate the range of possibilities that deserve further study. The other three cases and the related interactions with ChatGPT can be accessed through the Dropbox link provided earlier.

- SBT206 (454 words) is a straightforward software engineering teaching case used in an experiment in [21] about helping students identify use cases. It illustrates that ISKG could be used to organize and expand on the clarity of a situation that was worked out in advance so that an instructor could explain right and wrong answers.
- RDH403 (3,385 words) was extracted from an *Organization Science* article [22] that explored resistance versus control in ride hailing and gig work. The excerpt used here eliminated all references to gig work and sociological theory.
- EMR558 (9,248 words) is a PDF of a famous surgeon's non-research article "Why Doctors Hate Their Computers." [23]. It was selected to see whether the ISKG could be applied to a narrative account that covered different views of an important situation. That situation involved frustrations related to an electronic medical records system whose use occurred while providing medical care. It was interesting to see what ChatGPT interpreted as the relevant system.

The three cases that are not discussed here are roughly of the same length and scope as RDH403. They are not discussed because results for those cases were comparable to results for RDH403. One of those cases had been used in [1].

5 Results

The discussion of results will focus only on exploratory observations that extend beyond results from the earlier EMMSAD 2024 paper [1].

Word Counts. Table 1 shows word counts for the three cases, the full ChatGPT responses to those cases, and the summaries generated from those full responses.

Table 1. Word counts of cases and related ChatGPT-4o responses and summaries

Case > > > >	SBT206	RDH403	EMR558
Word count for case	454	3385	9,248
Word count for response	5,809	6,630	15,823
Word count for summary	3,089	2,376	2,362

All of the ChatGPT-4o responses to the case-specific versions of the standard prompt were longer than the original cases. The response to the prompt for the simple teaching case SBT206 was more than 10 times longer than the case itself because the prompt required separation of the answers into 7 subdivided sections. That subdivision was inconsistent with case's purpose of helping undergraduates learn about use cases. The case focused on simple IT concerns and avoided mentioning personal, interpersonal, and organizational issues that were present in more sophisticated portrayals of real situations in the other cases. The full response for RDH403 was more than twice as long as the case, but the summary was slightly shorter than the case. EMR588 was much longer than the other cases. It was not abbreviated at all to explore whether the ISKG could be applied to system descriptions intertwined with other issues.

Response to Guidance from the Analyst. The response to EMR588 was much longer than the other responses largely due to the following interchange after ChatGPT provided initial answers to the prompt's seven sections but before producing the SSO.

> ***ChatGPT***: *Please review this section [Section 7]. Once you confirm that everything is satisfactory or provide any additional corrections, I will compile the final report with all the sections.*
>
> ***User***: *This report has a serious flaw. You did not understand that the EMR is an information system that supports a work system of providing medical service. Please start with the first section and re-do the entire analysis, section by section, based on the understanding that the relevant system is the work system of providing medical care rather than the electronic medical record system.*
>
> ***ChatGPT***: *Thank you for the clarification. I will restart the analysis, focusing on the work system of providing medical care supported by the EMR, rather than analyzing the EMR as a standalone system. I will proceed with Section 1 first, ensuring that the focus is on the broader work system of medical care delivery.*

That misunderstanding had been noticed in the original response to Sect. 1, but permission to continue was granted at that point because it seemed worthwhile to see how that would affect responses in the other Sections. The misunderstanding was not surprising because many people are also confused when *system* sometimes refers to a set of related business capabilities and sometimes refers to IT capabilities. In this case, overlaps between EMR systems and medical care systems frustrated doctors who needed to switch back and forth between providing medical care and performing data entry. The "analyst" requested a different re-do earlier in the same case when ChatGPT made up its own facets instead of using the facets specified in the prompt.

Responses Concerning Work System Elements. The following work system summary was generated for the IS engineering teaching case SBT206, which was written as a narrative and was not organized by topic. The understandability of the summary below shows that an AI-based RA assistant might help students and business users understand use cases and user stories. While less structured than a BPMN model, this summary understandably identifies major topics and concerns.

1. Customers: *Subscribers: Individuals or organizations who subscribe to the journal and receive issues regularly. Authors: Researchers and professionals in software engineering who submit articles for publication. Advisors: Members of the editorial board who review and provide feedback on submissions. Editor and Assistant: Internal stakeholders who manage the publication process and day-to-day operations.*

2. Product/Services: *Monthly Journal Issues: Comprising 5–10 articles per issue focused on software engineering. Subscription Service: Management of subscription plans, including renewals and adjustments based on customer needs.*

3. Processes and Activities: *Article Submission and Review Process: Submissions are reviewed initially by the editor and then by assigned advisors for detailed feedback. Subscription Management: The assistant handles new and renewal subscriptions, processes payments, and maintains records. Publication Planning: Issues are planned months in advance, with a focus on maintaining a steady flow of content.*

4. Participants: *Editor: Oversees the overall publication process and content quality. Assistant: Manages subscriptions, submissions, and administrative tasks. Advisors: Provide peer reviews and contribute to the editorial decision-making process. Authors: Submit research articles for consideration in the journal.*

5. Information: *Article Metadata and Content: Includes details about submissions, such as author information, article title, and abstract. Subscription Data: Maintains records of subscriber information, payment history, and subscription status. Review Feedback: Comments and suggestions from advisors, stored and tracked within the system.*

6. Technologies: *In-house Computerized System: Used to manage subscriptions, article submissions, and review feedback. Communication Tools: Email and fax for submissions, correspondence, and handling credit card payments.*

7. Environment: *Publishing Industry Standards: Adheres to trends and regulations in scientific and technical publishing. Technological Environment: Involves the use of digital tools for managing subscriptions, submissions, and reviews.*

8. Infrastructure: *Technical Infrastructure: Includes the in-house computerized system, email servers, and other IT resources. Financial Infrastructure: Banking* *systems and processes for handling subscription payments, including credit cards and checks.*

9. Strategies: *Content Strategy**:** Focus on high-quality, peer-reviewed content in the field of software engineering. Subscription Strategy: Offers flexibility in subscription plans to cater to varying customer needs.*

The following excerpts show how ChatGPT applied analyst feedback to correct its initial understanding of the relevant system in EMR588 and produce a revised view.

***Initial** for Customers**:** The **primary customers** of the EMR system described in EMR588 are the healthcare professionals, including doctors, nurses, pharmacists, and administrative staff. Patients are indirect customers, as the system impacts their care quality and experience.*

***Revised** for Customers: **Primary Customers**: Patients who receive medical care are the primary customers of the work system. Their health and well-being depend on the effectiveness and efficiency of the care provided. **Secondary Customers:** Healthcare providers, including doctors, nurses, and administrative staff, who rely on the system to deliver care effectively. They use the information and resources provided by the EMR to make informed decisions and coordinate care.*

***Initial** for Product/Services**:** The **service** provided by the EMR system is the digital management of patient records, including tracking patient history, managing prescriptions, ordering tests, and facilitating communication between different parts of the healthcare system. It aims to replace paper records with a more integrated, efficient, and accessible digital system.*

***Revised** for Product/Services**:** **Core Service:** The core service provided by the work system is high-quality medical care, which includes diagnosis, treatment, and ongoing management of patient health. The EMR system supports this by providing access to accurate and timely medical records. **Supporting Services:** These include patient education, preventative care, and coordination of care among multiple healthcare providers. The EMR system facilitates communication and record-keeping to support these services.*

***Initial** for Processes and Activities**:** **Processes and Activities:** The key processes include patient data entry, updating medical records, managing appointments, handling prescriptions, and ensuring compliance with healthcare regulations. The system also involves training sessions for staff, and technical support for troubleshooting issues.*

***Revised** for Processes and Activities**:** **Clinical Processes:** These include patient assessment, diagnosis, treatment planning, and administration of care. The EMR system supports these processes by providing access to patient histories, test results, and treatment plans. **Administrative Processes:** Scheduling appointments, billing, and compliance with regulatory requirements are also critical activities. The EMR system helps streamline these processes, ensuring that they are handled efficiently.*

Responses Concerning Compliance with Work System Principles. Section 6 of the prompt asked for evaluation of compliance with 25 principles [20] that were stated as short sentences. Table 2 shows the 12 principles with highest and lowest average compliance ratings (5.0 and 4.67 versus 3.67 and 3.33). Giving too much credence to ChatGPT's compliance ratings would be inappropriate because it had no reliable way to make those assessments. On the other hand, the high-level professional organization (EMR588) seemed to show higher degrees of compliance (all 4s and 5) than SBT206, the exercise for SE undergraduates that did not mention human issues and RDH403, the account of ride hailing involving contested incentives. The main implication for a possible AI assistant is that a serious attempt to identify compliance and noncompliance would require separate analysis routines for each principle. Producing those routines would require substantial effort, but ideally would contribute to IS-related knowledge.

Table 2. Degree of compliance with work system principles (1 = very low, 5 = very high)

Principle Number and Statement	SBT206	RDH403	EMR588	Avg
8. Maintain Quality of Product/Services	5	5	5	5.00
2. Incorporate Goals, Measurement, Evaluation, Feedback	4	5	5	4.67
6. Please the Customers	5	5	4	4.67
13. Control Problems at Their Source	4	5	5	4.67
18. Operate with Clear Roles and Responsibilities	4	5	5	4.67
19. Provide Information Where It Will Affect Action	4	5	5	4.67
--	-----	-----	-----	-----
3. Minimize Unnecessary Risks	3	3	5	3.67
7. Balance Priorities of Different Customers	4	3	4	3.67
9. Ensure Products/Services are Appropriate for Use	3	3	5	3.67
12. Encourage Appropriate Use of Judgment	3	4	4	3.67
17. Align Participant Incentives with System Goals	3	4	4	3.67
16. Take Care of Participant Needs	3	3	4	3.33

Responses Concerning Facets, Characteristics, Evaluation Criteria, Phenomena, and Obstacles and Risk Factors. Section 2 of the prompt (facets of work) illustrates the need for a more elaborate ISKG if an LLM is to assess variables and issues in a narrative description of a case. That section asked ChatGPT to identify six of the 18 facets that required management attention. The three 5's for *making decisions* (Table 2) and the three 4's for *coordinating* made sense, but other responses seemed questionable. For example, neither *controlling execution* nor *processing information* were mentioned for EMR588 even though those facets mattered to key stakeholders. Similar issues were evident in responses for characteristics, evaluation criteria, phenomena, and obstacles and risk factors. Once again, the main implication is that identifying and dealing with related issues would require separate analysis routines for each topic in each category.

6 Conclusion

This paper explained and applied a new approach (Fig. 4) for using LLMs and organized knowledge (the ISKG) in early stages of requirements analysis. The results in Sect. 5 support the notion that an LLM guided by an ISKG and an analyst could contribute by summarizing the target work system and related issues in a useful way. The analyst would need to identify and correct any misinterpretations or errors introduced by the LLM, but the overall process might be less time-consuming than producing RA documentation from scratch, a point supported by recent research mentioned earlier.

Limitations and Directions for Future Research. The limitations of this research start with its exploratory nature. Many aspects of the prompt, the ISKG, and the interaction with an analyst could be modified or evaluated in different ways. For example, a prompt pattern catalog provided by [24] identifies many techniques for conversing with LLMs that were not used here. Likewise, [25] identifies many challenges in using LLMs for requirements engineering (RE) and suggests a systematic approach to selecting and adapting LLMs for different RE tasks. It notes that "RE demands a nuanced understanding of domain-specific jargon, context, and semantic nuances. Thus, the fine-tuning process necessitates meticulous calibration to preserve the fidelity of the model's pre-trained knowledge while tailoring it to the idiosyncrasies of RE requirements. (p. 2). This paper only hinted at those issues because it focused on establishing an initial understanding that precedes detailed design and documentation.

Elaborate evaluation of the results presented here seems inappropriate because capabilities used here were enabled by GenAI advances that became available only recently. There is no well-justified agreement about which technical capabilities will become available next year to extend this research. With that caveat, the following are possible directions for extending this research:

- Use new cases to compare the current version of the AI-based RA assistant versus alternatives including extended versions of the control and seven treatments in [1].
- Develop and test better versions of the prompt in order to use chain-of-thought more effectively. Improvements should emphasize more reliable ways for the LLM to justify its answers based on facts, snippets, separate KGs for different topics and issues, or use of interpreted excerpts from documents. There should be less reliance on generic statements that often make sense but apply equally to many other cases.
- Develop better versions of the ISKG. The current version was more effective than the treatments in [1] but surely could be extended, as noted in Sect. 5.
- Compare results from applying different LLMs that are now readily available (e.g., ChatGPT-4o, Claude Sonnet 3.7, Grok 3, Gemini, Llama 3.1, and Perplexity) to the same cases, the same or improved prompts, and the same or improved ISKGs.

- Test an RA assistant in instructional situations using an experiment similar to [21]. At minimum, reflections on uses of an RA assistant by business and/or IT students would reveal areas where they lack important understandings.
- Test an RA assistant in field experiments similar to [6] but focused on early stages of RA. Challenges related to the form and content of available documents in real situations likely would suggest ways to improve both the prompt and the ISKG.
- Embed a version of the RA assistant in an interactive analysis and design system that would store texts and interim results, would make that information accessible, and would facilitate the organized use of an analysis and design toolkit described in [19].
- Extend the ISKG and other parts of the RA assistant even further as a testable step toward addressing a still unsolved grand challenge for IS research identified in *BISE* in 2015: "rethink the theoretical foundations of the IS discipline." [26] (cf. [27]).

Disclosure of Interests. The author has no competing interests.

References

1. Alter, S.: Could a large language model contribute significantly to requirements analysis? In: Proceedings of EMMSAD (2024)
2. Dutta, S., Chakraborty, T.: Thus spake ChatGPT. Commun. ACM **66**(12), 16–19 (2023)
3. Teubner, T., et al.: Welcome to the era of Chatgpt et al. The prospects of large language models. Bus. Inf. Syst. Eng. **65**(2), 95–101 (2023)
4. Truex et al.: Systems analysis for everyone else: empowering business professionals through a systems analysis method that fits their needs. In: Proceedings of ECIS (2010)
5. Alter, S.: Work system theory: overview of core concepts, extensions, and challenges for the future. J. Assoc. Inf. Syst. **14**(2), 72–121 (2013)
6. Dell'Acqua, F. et al.: Field experimental evidence of the effects of AI on knowledge worker productivity and quality. In: Harvard Business School Technology & Operations Management Unit Working Paper, pp. 24–013, (2023)
7. Nam, D., et al.: Using an LLM to help with code understanding. In: Proceedings of the IEEE/ACM 46th International Conference on Software Engineering, pp. 1–13. (2024)
8. Liu, J., et al.: Is your code generated by ChatGPT really correct? Rigorous evaluation of large language models for code generation. In: Advances in Neural Information Processing Systems, vol. 36 (2024)
9. Liu, Z., et al.: No need to lift a finger anymore? Assessing the quality of code generation by ChatGPT. IEEE Trans. Softw. Eng. **50**, 1548–1584 (2024)
10. Lin, F., Kim, D. J.: When LLM-based code generation meets the software development process. arXiv:2403.15852 (2024)
11. Wei, B.: Requirements are all you need: from requirements to code with LLMs. arXiv preprint arXiv:2406.10101 (2024)
12. Chis, A., et. al.: Informing enterprise knowledge graphs with a work system perspective. Enterprise Modell. Inf. Syst. Archit. **19**(7), 1–21 (2024)
13. Khorashadizadeh, H., et al.: Research trends for the interplay between large language models and knowledge graphs. arXiv:2406.08223 (2024)
14. Alter, S.: Steps toward articulating a work system perspective that addresses a grand challenge for the IS discipline. In: S. Strecker, J. Jung (eds.), Informing Possible Future Worlds: Essays in Honour of Ulrich Frank, Logos Verlag, pp. 343–367. Berlin (2024)

15. Opdahl, A.L., Sindre, G.: Facet modelling: an approach to flexible and integrated conceptual modelling. Inf. Syst. **22**(5), 291–323 (1997)
16. Alter, S.: Facets of work: enriching the description, analysis, design, and evaluation of systems in organizations. Commun. Assoc. Inf. Syst. **49**(13), 321–354 (2021)
17. Alter, S.: Making cyber-human systems smarter. Inf. Syst. **127**, 102428 (2025)
18. Buchmann, R., et al.: Large language models: expectations for semantics-driven systems engineering. Data Knowl. Eng. **152**, 102324 (2024)
19. Alter, S., Bork, D.: Systems analysis and design toolkit based on work system theory and its extensions. J. Database Manag. **31**(3), 1–13 (2020)
20. Alter, S., Wright, R.: Validating work system principles for use in systems analysis and design. In: Proceedings of ICIS (2010)
21. Bolloju, N., et al.: Improving scrum user stories and product backlog using work system snapshots. In: Proceedings of AMCIS (20facets)
22. Cameron, L.D., Rahman, H.: Expanding the locus of resistance: understanding the co-constitution of control and resistance in the gig economy. Organ. Sci. **33**(1), 38–58 (2022)
23. Gawande, A.: Why do doctors hate their computers? The New Yorker (2018)
24. White, J., et al.: A prompt pattern catalog to enhance prompt engineering with ChatGPT. arXiv preprint arXiv:2302.11382 (2023)
25. Vogelsang, A, Fischbach, J.: Using large language models for natural language processing tasks in requirements engineering: a systematic guideline. arXiv preprint arXiv:2402.13823 (2024)
26. Becker, J., et al.: In search of information systems (grand) challenges: a community of inquirers perspective. Bus. Inf. Syst. Eng. **57**, 377–390 (2015). https://doi.org/10.1007/s12599-015-0394-0
27. Alter, S.: Produce a useful and teachable theoretical foundation for IS engineering. In: CAISE Research Projects Exhibition, pp. 1–9 (2024)

Benchmarking LLMs for Business Architecture Modelling with Hierarchical Capability Maps

Iromie Samarasekara[1](✉), Madhushi Bandara[2], Fethi Rabhi[1], and Boualem Benatallah[3]

[1] School of Computer Science and Engineering, University of New South Wales, Sydney, Australia
{i.samarasekara,f.rabhi}@unsw.edu.au

[2] School of Computer Science, University of Technology Sydney, Sydney, Australia
Madhushi.Bandara@uts.edu.au

[3] Insight SFI Research Centre for Data Analytics, Dublin City University, Dublin, Ireland
boualem.benatallah@dcu.ie

Abstract. Business Capability Map is one of the core instruments of Business Architecture (BA) modeling and analysis and an essential tool for driving business/IT alignment. However, the process of crafting a structured and hierarchical overview of organisational capabilities as a business capability map is a manual, knowledge-intensive process that consumes a significant amount of effort and time. Large Language Models (LLMs) have demonstrated their ability to automate knowledge-intensive tasks such as business process modeling through their internalised knowledge. However, they tend to perform poorly when the task requires specific domain knowledge as opposed to handling general knowledge. This is a hurdle in adapting LLMs for BA modeling, as domain expertise is crucial for generating business architecture models. To further our understanding in this challenge, this paper presents a benchmark experiment that systematically and comprehensively evaluates the utility of LLMs in BA modeling. We propose BCM-Eval, a novel business capability map benchmark, and use it to evaluate key state-of-the-art LLMs in different prompt settings. We report on the potential and limitations of LLMs for business capability modeling, concluding that LLMs still have a limited grasp on industry expertise and do not precisely capture the semantics related to capability models. Our results also indicate the need for advanced prompting and domain knowledge augmentation techniques that can probe the knowledge of LLMs towards capability maps and other hierarchical BA models.

Keywords: Business Capability Modelling · Capability Maps · Large Language Models · Business Architecture

J. Krogstie et al. (Eds.): CAiSE 2025, LNCS 15701, pp. 37–53, 2025.
https://doi.org/10.1007/978-3-031-94569-4_3

1 Introduction

A business capability map can be defined as the starting point of business architecture, a hierarchical arrangement of organisational capabilities [29] and a core artefact that facilitates alignment of business and IT and drives organisational change [17]. However, creating a capability map is an exercise that involves introspective analysis [29] and decomposition that requires a significant level of sociotechnical skills coupled with substantial knowledge on a particular organisation or domain. This has created a high entry point for many organisations to adopt business capability modeling in practice [14].

An industry template of a capability map is a design artefact [30] that provides a good starting point for this process, as businesses within the same industry vertical or sector share similar value propositions and goals, thus requiring a similar set of capabilities to operate. Such industry-specific capability map templates, usually shared by consulting agents, direct the process and ensure completeness of resulting maps. However, such templates are often proprietary [30] or too abstract [19] compared to those produced by domain experts.

Rapid advancement of LLMs' ability to answer natural language questions using their internalised knowledge has created potential for automation of various knowledge-intensive tasks. However, the feasibility of employing them in the context of business architecture modeling remains an open question, as the current literature has not evaluated the potential of LLMs to comprehend and produce hierarchical BA models. This study aims to bridge this gap by providing a detailed understanding of the limits of LLMs through a systematic evaluation that answers the critical question: **Can LLMs effectively assist in the generation of business capability maps?**. Using four sub-questions, we study three dimensions of the capability map development process (comprehending hierarchical structures, identifying the key areas of a particular industry vertical, introspective analysis and decomposition [29]), and the effect of prompting techniques.

– **RQ1**: How reliable are LLMs for comprehending hierarchical structures in business capability maps?
– **RQ2**: How reliably can LLMs identify the top-level capabilities (key areas) required to operate within a particular industry?
– **RQ3**: How reliably can LLMs perform capability decompositions?
– **RQ4**: Do different prompt settings influence the question answering accuracy?

To answer these questions, there are no comprehensive benchmark datasets that encompass a wide variety of capability maps that cover their diverse characteristics. Formulating an evaluation strategy is also a challenge due to the unique hierarchical structure of capability maps created in line with BA best practices and guidelines. Therefore, our contributions are as follows.

– **BCM-Eval (Business Capability Map-Evaluation) dataset**: BCM-Eval dataset is a comprehensive benchmark dataset that curates capability maps from eight different industry sectors with a range of unique capabilities and a varying number of hierarchical levels that represent both proprietary and publicly available templates.

- **BCM-Eval strategy**: We propose a unique evaluation strategy that suits the characteristics of business architecture models such as capability maps.
- **A comprehensive evaluation**: We apply the BCM-Eval dataset and strategy across four LLMs that are of reasonable size (GPT, Llama, Mixtral, and Vicuna) and rigorously assess the knowledge of LLMs on different hierarchical aspects of the business capability map. We also employ different prompting techniques (zero-shot, few-shot, and chain-of-thought) to systematically evaluate the question answering ability of LLMs under different configurations.

The paper is structured as follows. Section 2 provides an overview of the background and literature. Sections 3 and 4 describe the methodology details for constructing the benchmark and the design of the evaluation strategy. Section 5 presents the evaluation setup and the results of the experiment. The discussion and future directions are presented in Sect. 6 and the paper concludes in Sect. 7.

2 Background and Related Work

The creation of a business capability map is a challenging task and requires numerous design cycles with the involvement of multiple experts [7,30]. The process begins by identifying a good set of Level 1 capabilities (key areas). As a business capability itself is a business-focused abstraction of singular functions, processes, and information elements within a business [29], deriving an appropriate set of Level 1 capabilities requires thinking at a significantly high level of abstraction. This is often considered a challenge even for experts with sufficient knowledge of business processes and workflows. The rest of the process involves capability decompositions (identify Level 2, 3 etc. sub-capabilities of a Level 1 capability) and requires thorough analysis and reflection across different lines of business, usually conducted through workshops, one for each Level 1 capability with professionals from the associated line of business [29]. Capability map templates can be utilised to guide this process. As an example, the BIZBOK guide developed by the Business Architecture Guild[1] offers a set of proprietary business capability maps collectively created by a group of practitioners who have extensive knowledge of each vertical industry. These models are continuously updated by guild members from around the world and strictly adhere to BA standards and are closely aligned with domain-specific key value propositions. BA tool vendors such as LeanIX[2] also provide free capability map templates in several industries. However, they do not offer the same level of depth or details as the proprietary ones. For example, while a template produced by LeanIX for the finance sector has top-level capabilities such as retail banking and digital banking, the BIZBOK proprietary template has extra

[1] https://www.businessarchitectureguild.org/store/ListProducts.aspx?catid=677483.

[2] https://www.leanix.net/en/wiki/ea/business-capability-map-examples-and-templates.

capabilities such as collateral management and financial instrument management which are not captured in former templates.

LLMs are a promising new paradigm of automation in several domains and application areas such as question answering, text summarisation, and named entity recognition [20]. In relation to BA modeling, several studies provide insight into the potential of LLMs for mining business processes [4–6,13,15,16,25] and acting as proxy domain experts in BA modelling [3,12]. These studies show that LLMs are far from replacing the role of a domain expert [12] and are more capable of helping with general questions than with tasks that are unique or specific to a particular context [3]. None of the studies has looked at other complex hierarchical BA abstractions such as capability maps, and the evaluation of each study is limited to a single industry/LLM and based on input from few domain experts. So, conducting a quantitative study across multiple industries and LLMs is a timely need.

A few studies have also been undertaken to capture the intersection of conceptual modelling with LLMs. One of the pioneering works in this area is the exploratory study conducted by Fill et al. [10] where they present some impressions on the initial experiments with ChatGPT. In this study they explore how well LLMs can generate and interpret ER, Business Process, and UML class diagrams as well as Heraklit models. Their results revealed that LLMs possess the potential to assist in modelling tasks when provided with a textual description of the problem domain. A study undertaken by Ali et al. [2] investigates a different aspect of LLM driven conceptual modelling by looking into the interaction behaviour and user perceptions. They identified, the specific LLM, application domain and the task to be the three independent variables that mainly affect the modelling process and user experience. They also highlight the importance of developing prompt templates and the importance of LLM selection thus, emphasising the need for exploring the potential of alternative LLMs. Buchmann et al. [8] present some important insights on the interplay between LLMs and conceptual models in the context of semantics-driven systems engineering. In this work they stress on the importance of conducting experiments to explore which LLM properties are most helpful in various scenarios pertaining to automated modelling, which is a key focus of our study. They also highlight the importance of considering inherent shortcomings of LLMs when leveraging them for information systems engineering. Fill et al. presents CMAG which is a framework for augmenting GenAI outputs with conceptual models [11]. The framework treats conceptual models as the central medium of interaction with LLMs and it allows users to easily validate these outputs, and they correspond to a particular conceptual schema.

LLM evaluation benchmarks are critical assets to facilitate research and application in this area. Multiple studies (Lc-quad [9], CRAG [31], Natural Questions [18] and EntityQuestions [24] present question-response benchmarks to evaluate the scope of knowledge embedded in language models. TaxoGlimpse [28] is one of the first attempts to create a benchmark that incorporates hierarchical concepts. However, to our knowledge, no attempts have been made to produce benchmarks related to hierarchical BA abstractions.

The need for automated tool support for BA modeling and the potential LLMs have shown in the area indicates the value of an appropriate benchmark dataset and a comprehensive evaluation. Hence, the goal of this study is to address this gap by curating a benchmark dataset and performing a systematic evaluation to determine the knowledge limits of LLMs to be used in complex BA modeling tasks by curating the required benchmark dataset.

3 BCM-Eval Construction

3.1 BCM-Eval Data Curation

Evaluating LLM performance to address our research questions requires a dataset composed of capability maps with sufficient variety. We formulated following criteria to determine the eligibility of such maps based on the input gathered from expert consultation.

1. **Diversity of domains**: The scope should cover a wide variety of industry sectors with varying levels of domain specialty.
2. **Number of hierarchical levels**: This captures different levels of complexity in decomposition and its utility within the BA practice.
3. **Availability of descriptions and stratification**: Denotes different levels of detail, ranging from just a label to detailed capability descriptions.
4. **Source of Origin**: The dataset should include both publicly available and proprietary models.

3.2 Benchmark Data

We created the BCM-Eval benchmark dataset by curating 10 business capability maps belonging to eight different industry verticals considering the above criteria. Industries include manufacturing, finance, healthcare, insurance, transportation, telecommunications, energy, and pharmaceuticals. Consequently, six proprietary templates were extracted from the BIZBOK practitioner guide[3] and four publicly available templates were taken from LeanIX[4] based on their availability. Templates that are too generic and not associated with a specific industry were excluded [21,26]. Table 1 provides summary statistics associated with each template included in the benchmark.

4 BCM-Eval Strategy Design

This section discusses how we formulated the evaluation strategy to address the first three subquestions presented in Sect. 1. We limited our study to Level 1 to 3 capabilities following common industry practice [29].

[3] https://www.businessarchitectureguild.org/store/ListProducts.aspx?catid=677483.
[4] https://www.leanix.net/en/wiki/ea/business-capability-map-examples-and-templates.

Table 1. Summary of BCM-Eval Dataset (Total:total Number of capabilities, Level 1: Number of level 1 capabilities)

Industry	Source	Levels	Total	Level 1	Descriptions	Stratification
Manufacturing	BIZBOK	4	1794	42	Available	Available
	LeanIX	2	77	10	Unavailable	Unavailable
Finance	BIZBOK	4	1551	37	Available	Available
	LeanIX	2	111	13	Unavailable	Unavailable
Healthcare	BIZBOK	4	1668	42	Available	Available
Insurance	BIZBOK	4	1681	37	Available	Available
Transportation	BIZBOK	4	1888	43	Available	Available
Telecommunications	BIZBOK	4	1960	44	Available	Available
Pharmaceutical	LeanIX	2	90	11	Unavailable	Unavailable
Energy	LeanIX	2	112	12	Unavailable	Unavailable

4.1 RQ1: Comprehending Hierarchical Structures in Capability Maps

To understand the ability of LLMs to understand hierarchical relationships in capability maps, we used a Multiple Choice Question (MCQ) template inspired by the work of Sun et al. on taxonomy generation [28]. This allows us to understand how well LLMs can determine the right parent capability for a given lower-level capability.

We formulated these questions around Level 3 capabilities to prevent LLM from providing answers based on the naming similarity of the parent and child capabilities, which is common between the Level 1 and Level 2 capabilities. The MCQs were made more challenging by providing sibling capabilities (other Level 2 Capabilities that come under the same Level 1 Parent Capability as the ground truth Level 2 Capability) as negative options along with the associated ground truth Level 2 capability. All prompt templates are provided in our Github repository.[5]

4.2 RQ2: Identifying Level 1 Capabilities

To understand how well LLMs can generate key capability areas associated with an industry, we designed the prompt asking to provide a set of Level 1 capabilities suitable to be included in a business capability map of a given industry. This evaluation reveals how well LLMs can play the role of an industry expert and perform extreme high level of abstractions in the capability map creation process discussed in Sect. 2.

[5] https://github.com/Iromie-Samarasekara/Iromie-Samarasekara-LLM-Driven-Capability-Modelling.

4.3 RQ3: Performing Capability Decompositions

Here we evaluate how well LLMs can perform capability decomposition by prompting the LLMs to decompose a given Level 1 capability of a particular industry into appropriate Level 2 capabilities. We evaluated only Level 1 to Level 2 decomposition, as the Level 3 capability decomposition requires inputs and validation from professionals within a specific organisation who possess in depth knowledge on the workflows as Level 3 capabilities should directly map to the organisation value stream stages [29].

5 Evaluation

This section provides details on all our experiment configurations followed by the results and discussion.

5.1 Large Language Model Setup

With the intention of providing a comprehensive and well-balanced evaluation of the knowledge limits of LLMs, we selected four widely used state-of-the-art language models representing both open-source and closed-source categories and are sufficiently large. It should be noted that the goal of this study is not to compare the performance of different LLMs, but to report on common patterns among representative LLMs by examining their outputs. LLM details are as follows:

- **GPT:** GPT (Generative Pre-Trained Transformers) series by OpenAI is closed-source and considered the most advanced set of language models currently in use. We selected GPT-4 from their series for our experiments.
- **Llama:** Llama is a set of open source language models by Meta. We selected the 70B model of Llama-3.1 series with instruct settings, which is specifically designed to strike a fine balance between high performance and resource efficiency and is more suitable for question-answering applications.
- **Vicuna:** Vicuna series includes open-source language models that are trained by fine-tuning Llamas. We selected the 33B model due to its ability to handle extensive context information [32].
- **Mistral:** This is a series of open-source LLMs produced by MistralAI and recorded to outperform GPT 3.5 and the Llama-2 series on standard benchmarks [1,28]. We adopted Mixtral-8*7B under instruct settings for our experiments.

We deployed all the models on an NVIDIA A100 GPU. All GPT interactions were based on OpenAI's official API and we used Ollama [22] to host the rest of the open-source models.

5.2 Evaluation Process and Metrics

We evaluated all LLMs based on the same set of questions. Zero-shot prompt setting was used for experiments associated with RQ1, RQ2, and RQ3. RQ1, RQ2 and RQ3 were answered using Zero-shot, Few-shot and Chain-of-Thought (COT) prompting to provide answers to **RQ4: Do different prompt settings influence the question answering accuracy?**. Under the few-shot technique, we conducted five-shot experiments and included examples from each layer (strategic, support, and industry specific) along with examples that are both domain-specific and domain-agnostic to avoid introducing bias into examples. CoT experiments allowed us to tap into the reasoning ability embedded in language models to enhance the performance in the context of capability modeling. We did this by adding the phrase 'Let's think step by step' to each prompt to take the model through more steps of reasoning.

We use 2 evaluation metrics to analyse results supported by the literature [27,28].

- **Accuracy (A):** Number of questions accurately answered by LLMs/ Number of questions
- **Rate of hallucination (H):** Number of capabilities that do not match semantically with the ground truth/ Number of capabilities included in the LLM answer.

Except in the case of MCQs where there exists a direct mapping between the LLM's answer and the ground truth, the answer accuracy and semantic match were manually assessed to ensure semantic validity by calculating the descriptions of the capabilities provided, thus accounting for the similarities and disparities in naming.

5.3 Experimental Results

This section presents the results obtained for each experiment in the four research questions.

RQ1: How Reliable Are LLMs for Determining Hierarchical Categories in Business Capability Maps? As shown in Table 2, we evaluated the accuracy of the MCQ question answers of each model in six industries, as well as support and strategic capabilities, using 3246 questions.

Table 2. Accuracy: Determining Hierarchical Categories.

Industry	GPT	Llama	Mixtral	Vicuna
Support	**0.8506**	0.8278	**0.7822**	**0.1468**
Manufacturing	0.8333	0.8427	0.7322	0.1124
Insurance	0.8329	0.8383	0.7616	0.1370
Finance	0.8306	0.8439	0.7740	0.1096
Telecommunications	0.8224	0.8408	0.7796	0.1286
Transportation	0.8158	**0.8515**	0.7669	0.1147
Strategic	0.8127	0.8161	0.7358	0.1170
Healthcare	**0.7394**	**0.7818**	**0.6939**	**0.0879**

We limited this experiment to BIZBOK templates, as these templates provide a comprehensive hierarchical arrangement with each Level 1 capability decomposing at least up to 3 levels which is a prerequisite for generating the MCQs as discussed in Sect. 4.1. LeanIX templates offer only upto 2 levels depth (Table 1).

We observed that all models perform best when identifying the hierarchical relationships of support capabilities where three out of four models achieved the highest overall accuracy. Also, all models show comparatively low performance for healthcare. We observed GPT-4 showing an overall decreasing trend in accuracy when moving from more common industries such as manufacturing and insurance to more specialised domains such as telecommunications and healthcare. Llama, Mixtral and Vicuna show no specific pattern across the industries.

The accuracy reported by Vicuna is significantly low compared to the rest of the models in all industry sectors, as illustrated in Fig. 1. The results reported by other open source models indicate competitive performance with the proprietary GPT model, with Llama reporting the highest accuracy of 85.15% in Transportation, which is slightly higher than the best reported by GPT and Mixtral for any sector.

RQ2: How Reliably Can LLMs Identify the Top-Level Capabilities Required to Operate Within a Particular Industry? We prompted LLMs to identify the Level 1 capabilities for eight industry sectors using the method proposed in Sect. 4.2. The results are shown in Table 3 and Fig. 2.

Table 3. Accuracy and Hallucination: Identifying Level 1 Capabilities

Template	GPT-A	GPT-H	Llama-A	Llama-H	Mixtral-A	Mixtral -H	Vicuna-A	Vicuna-H
BIZBOK								
Finance	0.1892	0.4167	**0.0270**	**0.9000**	0.1351	0.5000	**0.1351**	0.6429
Insurance	**0.1463**	**0.5385**	0.0732	0.5714	0.0976	0.6000	0.1220	**0.5000**
Healthcare	**0.2143**	**0.3077**	0.0476	0.6000	0.0952	0.7333	**0.0000**	**1.0000**
Transportation	0.1628	0.4167	0.0465	0.7143	**0.0465**	**0.8000**	0.1163	**0.5000**
Telecommunications	0.1818	0.3333	**0.1136**	**0.2857**	**0.1364**	**0.4000**	0.1136	0.5833
Manufacturing	0.1667	0.4615	0.1190	0.5000	0.0952	0.6000	0.1190	0.6667
LeanIX								
Finance	0.5556	**0.8380**	0.2222	**0.8000**	0.3333	0.7000	**0.3333**	**0.7857**
Manufacturing	**0.6000**	0.8333	**0.6000**	**0.4000**	**0.4000**	**0.6000**	**0.5000**	0.6667
Energy	**0.4167**	**0.7120**	0.2500	0.5000	**0.1667**	**0.8000**	**0.3333**	**0.6000**
Pharmaceutical	**0.6000**	0.8333	**0.2000**	0.6000	0.3000	0.7000	**0.5000**	0.6667

When we consider BIZBOK (Top half of Table 3) the overall performance for different industry sectors varies substantially between the different LLMs. In contrast to the RQ1 experiment, the accuracy is very low for all LLMs and the performance of Llama, Mixtral and Vicuna is noticeably low compared to GPT as well.

The accuracy is significantly better for the same experiment done on LeanIX templates (bottom half of Table 3). The highest overall precision in all models is 60%, reported by GPT as well as Llama for the manufacturing and pharmaceutical industries. All models perform comparatively low for the energy sector and comparatively higher for manufacturing. Some of the open-source models such as Vicuna and Llama report accuracies that are equal to or higher than GPT for industries such as Finance and Manufacturing for LeanIX templates. GPT's rate of hallucination is also observed to be higher when compared to the Llama, Mixtral and Vicuna for LeanIX templates.

RQ3: How Reliably Can LLMs Perform Business Capability Decompositions? We conducted the experiment designed in Sect. 4.3 for 76 unique Level 1 capabilities from both BIZBOK and LeanIX templates. We then compare the resulting Level 2 capabilities against the template. The results are presented in Table 4, Fig. 3 and Fig. 4. We observe that the accuracy is higher than in the RQ2 experiment for BIZBOK templates. This indicates that LLMs perform better when it comes to decomposing capabilities of a given industry than abstracting. However, the decomposition accuracy for LeanIX templates is lower compared to the abstraction accuracy in RQ2. The highest overall accuracy remains low at 35% and is reported by GPT and Llama. The open source models show competitive performance with GPT in several instances for both the BIZBOK and LeanIX models.

Table 4. Accuracy and Hallucination: Decomposing of Level 1 Capabilities

Template	GPT-A	GPT-H	Llama-A	Llama-H	Mixtral-A	Mixtral -H	Vicuna-A	Vicuna-H
BIZBOK								
Healthcare	0.1861	0.6800	0.1395	0.7000	0.1395	0.7600	**0.0233**	**0.9600**
Transportation	0.1429	0.7333	0.0909	0.8333	0.1364	0.8235	0.0910	0.8947
Insurance	**0.3500**	0.6667	**0.3500**	0.5333	0.1500	**0.8750**	**0.1500**	0.8000
Finance	0.2857	**0.6429**	0.3429	**0.4000**	**0.2857**	**0.5833**	0.1429	**0.7500**
Support	**0.0959**	0.8667	**0.0656**	**0.8667**	0.1475	0.7857	0.0574	0.9067
Strategic	**0.0959**	**0.8679**	0.1370	0.7222	0.1781	0.7593	0.0822	0.8333
Other Core	0.1062	0.8182	0.1081	0.7272	**0.1081**	0.8033	0.0901	0.7727
LeanIX								
Energy	**0.0667**	**0.9200**	**0.1333**	**0.8000**	0.2000	**0.8000**	0.1000	0.8800
Manufacturing	0.2857	0.6000	0.2500	0.5625	0.2143	**0.7000**	**0.3214**	0.7188
Pharmaceutical	0.2000	0.7500	0.1600	**0.8000**	**0.1600**	0.7500	**0.0800**	**0.8824**
Finance	0.2610	0.6757	**0.2826**	0.6886	**0.2500**	0.7143	0.2750	0.8358
Other	**0.3750**	**0.4000**	0.2292	**0.5417**	0.1875	**0.7000**	0.2292	**0.6333**

RQ4: Do Different Prompt Settings Influence the Question Answering Accuracy? The results of all experiments performed for Zero-shot, Few-shot and Chain-of-Thought (COT) prompting are presented here.

As illustrated in Fig. 2, compared to zero-shot performance, few-shot prompting showcases a slight improvement in accuracy for GPT and Llama when determining hierarchical structures in capability maps. In the same context, CoT prompting negatively affects the performance of the models. However, these performance differences are not very significant for both GPT and Llama. In contrast, for Mixtral and Vicuna, both few-shot and CoT prompting has caused a significant reduction in accuracy for a majority of industries, and the few-shot setting reports the lowest accuracies for these models. This can possibly occur due to the significant increase in the length of context in the prompts caused by the inclusion of examples that can hinder the performance of smaller models.

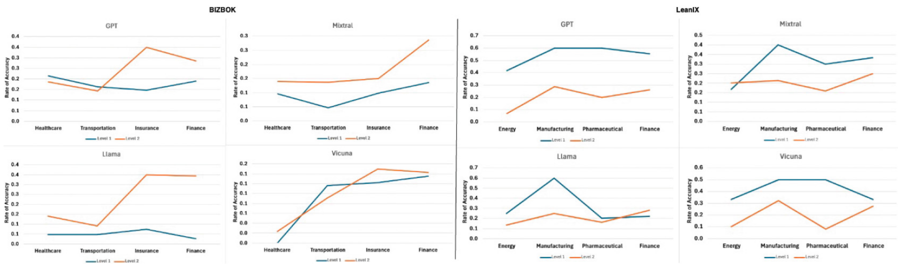


Fig. 1. Accuracy: Decomposing Level 1 Capabilities.

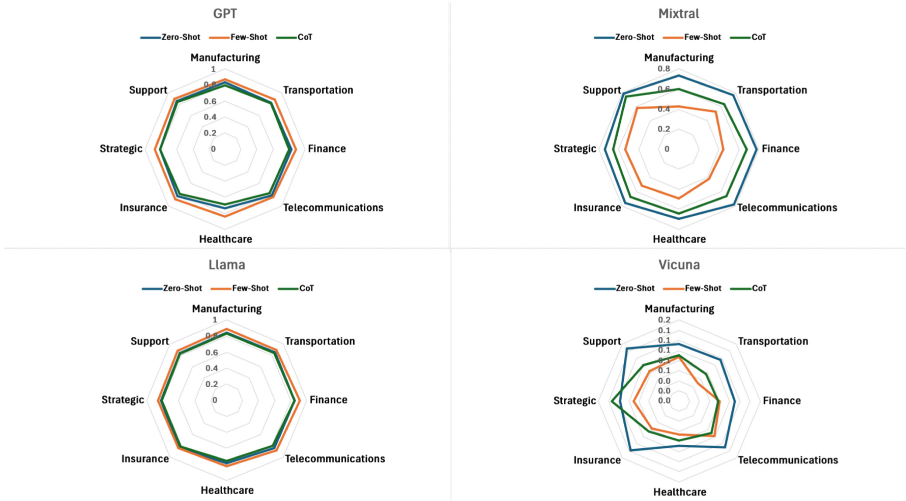


Fig. 2. Accuracy Across Prompt Settings: Determining Hierarchical Categories.

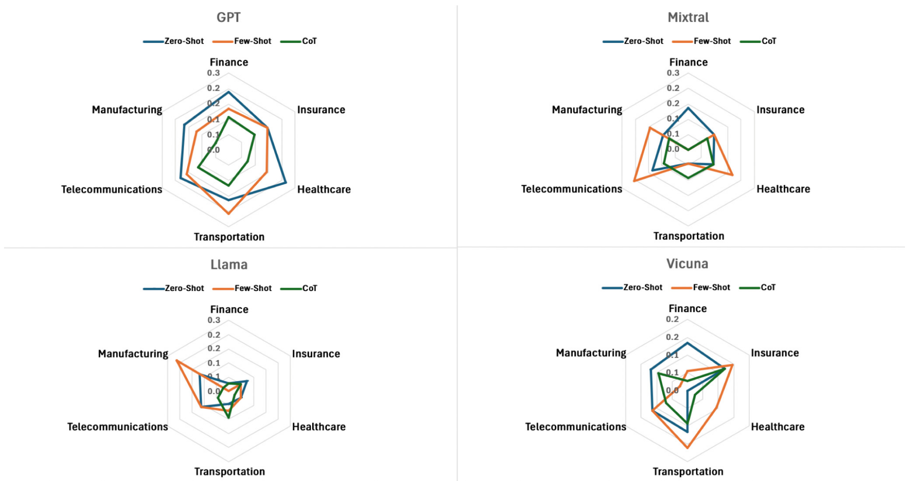


Fig. 3. Accuracy Across Prompt Settings: Identifying Level 1 Capabilities.

Unlike the performance variations observed in the previous experiment, when identifying Level 1 capabilities with Mistral and Vicuna, the few-shot prompt outperforms the zero-shot setting for several industries (Fig. 3).

The same pattern was observed for Llama, where a few shots resulting in a performance peak for the manufacturing industry. For GPT, the zero-shot setting reports the best performance across all industries except transport, where the few-shot prompting reports the highest accuracy. This may have occurred

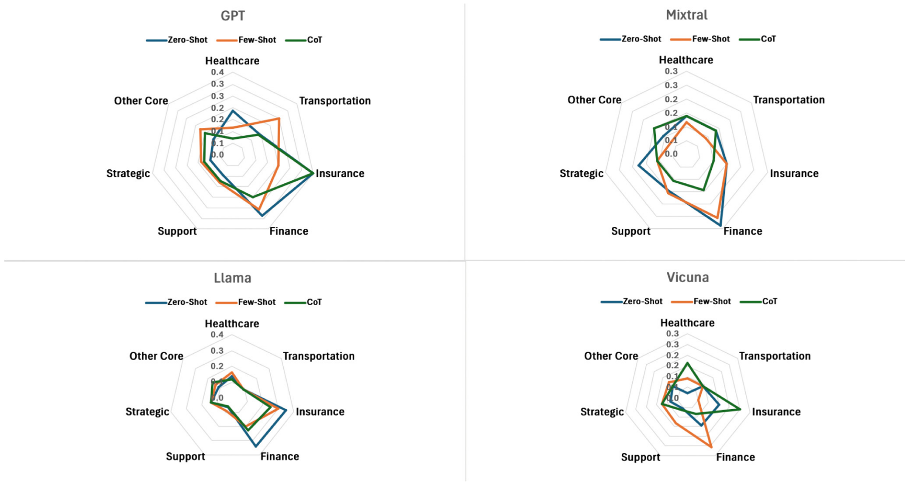


Fig. 4. Accuracy Across Prompt Settings: Decomposing Level 1 Capabilities.

through the models gaining more confidence in producing a larger number of guesses by imitating concrete examples in the prompt.

Overall, the performance of CoT prompting has been poor across all models, except for one or two instances in identifying Level 1 capabilities. However, the increased rates of hallucination indicate that few-shot examples mainly contribute to increase the number of responses and not much to improve the quality of responses and reduce hallucination.

Interestingly, when it comes to decomposing Level 1 capabilities, the few-shot and CoT settings show competitive performance against the zero-shot setting in all models, including GPT for many industries with no significant pattern in the variations (Fig. 4). The higher accuracies reported for the zero-shot setting for industries such as finance, insurance, and healthcare for most of the models indicate that neither providing examples nor improving the reasoning ability has helped the models to decompose high-level capabilities of such industries where the capabilities are more domain intensive.

5.4 Discussion

We discuss the results of our experiments under four broad themes that are aligned with our research questions as below.

Comprehending Hierarchical Structures in Capability Maps: despite having less coverage of knowledge on more specialised industry verticals, LLMs are capable of understanding hierarchical relationships among capabilities with satisfactory confidence, indicating that they can potentially assist business architects in designing models that are hierarchical in nature. We attribute the variation of performance across industries to the fact that the knowledge on more common areas such as support capabilities which include capabilities such as Human Resource Management are possibly covered by original training data of these LLMs to some extent. However, knowledge on more specialised sectors such as Healthcare, where business architecture modeling may not be widely used and reported, and are less likely to be covered in training data. The operations of such businesses are also scarcely documented in the public domain. We believe that the error rates (30%) that LLMs display in this endeavor can be addressed through specialised prompting techniques and domain knowledge augmentation.

Identifying Higher-Level Capabilities: even the state-of-the-art LLMs are observed to lag in their performance, relative to an industry expert in identifying an overarching set of Level 1 capabilities. The performance disparity for the BIZBOK and LeanIX templates is an indication of their knowledge limits in producing a capability map for a particular industry or sector that meets the standard of proprietary templates. Yet, the level of industry expertise internalised in LLMs seems sufficient to produce templates comparable to the freely offered resources.

Decomposing Higher-Level Capabilities: results indicate that despite not having much knowledge on an overarching set of capabilities that are essential for a particular industry vertical, LLMs are more familiar with what is done around a particular business or information object. We attribute the disparity of results between the BIZBOK and LeanIX models to the lack of capability descriptions in the LeanIX templates. They are also more abstract and do not provide clarity on the business or information objects associated with the capabilities. Moving down the hierarchical levels shows that the ability of an LLM to decompose a given capability is tightly coupled to the quality of parent capability descriptions. Urich and Rosen [30] also state that the process of decomposition begins with understanding how a capability is defined and viewed.

Performance Variation Across Different Prompting Techniques: experimenting with different prompting techniques shows that the performance enhancements brought about by the few shot and CoT are minimal, especially for the best performing LLMs such as GPT. We attribute this observation to the simple-formed nature of these questions in the case of CoT as it is mostly known to be helpful in tasks that involve complex reasoning. Even the accuracy improvements observed for few-shot and CoT in some instances came at the cost of reduced reliability due to the higher rates of hallucination. As BA modelling is a precise discipline where aspects such as reliability and completeness play a critical role, it is unlikely that these conventional methods of prompting would have a meaningful impact in this context.

6 Study Limitations and Future Directions

In this paper, the proposed BCM-Eval benchmark helped us understand the variations between industries and determine their utility from the practitioner's perspective. However, we acknowledge its size limitations and the need to collect more data in the future for further expansion. The results indicate the possibility of achieving an improvement in performance using an appropriate prompting method, possibly investigating beyond traditional prompting techniques to achieve the right balance between the length of the context and the amount of emphasis placed on the semantic relationships between concepts [23].

It is also worth exploring the robustness of the model to paraphrasing and other ways in which models can be queried, as this study only evaluates the extent to which an LLM knows a fact with high confidence.

Overall, the results of our evaluation show that despite the mastery demonstrated by state-of-the-art LLMs in determining hierarchical structures in a capability map, their knowledge coverage is insufficient to meet the need for industry expertise in capability modeling. From a practitioner perspective, we recommend that businesses that require high-quality capability maps continue to invest in proprietary industry templates, or consider LLM generated maps as a basic template that need further investments for refining. Business architecture tool vendors can consider leveraging the power of LLMs to improve their industry template offerings. However, such templates should only be viewed as starting points and not as prescriptive solutions.

7 Conclusion

This paper presents a comprehensive evaluation on the utility of LLMs on hierarchical business architecture abstractions and generation, capability maps, in particular. Alongside this evaluation, we introduce BCM-Eval, the first benchmark designed to assess the factuality of LLMs in the context of generic business capability modeling. We conduct this evaluation over a wide spectrum of business capability map templates of varying levels of complexity across multiple industry sectors using four latest LLMs and three prompting techniques.

The practical relevance of this work lies in investigating the possibility of automating hierarchical business architecture modelling and our results provide valuable insight to researchers and industry practitioners on the challenges and opportunities associated with an LLM-driven approach. We demonstrate that, despite their ability to comprehend hierarchical relationships within capability maps, even the best performing LLMs do not meet the quality of outcome generated by the domain experts in the capability modeling process, specifically for more specialized industry sectors.

Possible future research directions to mitigate these challenges could be studying the effect of LLM pre-training and fine-tuning for BA modeling and designing new methods to incorporate further domain-specific knowledge through prompts and other augmentation techniques.

Acknowledgments. We are grateful to Capsifi and its founder, Dr. Terry Roach, and the research director, Dr. Rouzbeh Meymandpour, for supporting and sponsoring the research that led to this article.

References

1. AI, M.: Mixtral: a mixture of experts model (2024). https://mistral.ai/news/mixtral-of-experts/. Accessed 16 Nov 2024
2. Ali, S.J., Reinhartz-Berger, I., Bork, D.: How are LLMs used for conceptual modeling? An exploratory study on interaction behavior and user perception. In: International Conference on Conceptual Modeling, pp. 257–275. Springer (2024). https://doi.org/10.1007/978-3-031-75872-0_14
3. Barn, B., Sandkuhl, K., Barat, S.: Neural text generators in enterprise modeling: Can ChatGPT be used as proxy domain expert? In: 31st International Conference on Information Systems Development, pp. 87090
4. Bellan, P., Dragoni, M., Ghidini, C.: Leveraging pre-trained language models for conversational information seeking from text (2022). arXiv preprint arXiv:2204.03542
5. Bellan, P., Dragoni, M., Ghidini, C.: Extracting business process entities and relations from text using pre-trained language models and in-context learning. In: International Conference on Enterprise Design, Operations, and Computing, pp. 182–199. Springer (2022). https://doi.org/10.1007/978-3-031-17604-3_11
6. Berti, A., Schuster, D., van der Aalst, W.M.: Abstractions, scenarios, and prompt definitions for process mining with LLMs: a case study. In: International Conference on Business Process Management. pp. 427–439. Springer (2023). https://doi.org/10.1007/978-3-031-50974-2_32
7. Bondel, G., Faber, A., Matthes, F.: Reporting from the implementation of a business capability map as business-it alignment tool. In: 2018 IEEE 22nd International Enterprise Distributed Object Computing Workshop (EDOCW), pp. 125–134. IEEE (2018)
8. Buchmann, R., et al.: Large language models: expectations for semantics-driven systems engineering. Data Knowl. Eng. **152**, 102324 (2024)
9. Dubey, M., Banerjee, D., Abdelkawi, A., Lehmann, J.: LC-QuAD 2.0: a large dataset for complex question answering over Wikidata and DBpedia. In: Ghidini, C., et al. (eds.) ISWC 2019. LNCS, vol. 11779, pp. 69–78. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-30796-7_5
10. Fill, H.G., Fettke, P., Köpke, J.: Conceptual modeling and large language models: impressions from first experiments with ChatGPT. Enterprise Model. Inf. Syst. Archit. (EMISAJ) **18**, 1–15 (2023)
11. Fill, H.G., et al.: CMAG: a framework for conceptual model augmented generative artificial intelligence (2024)
12. Görgen, L., Müller, E., Triller, M., Nast, B., Sandkuhl, K.: Large language models in enterprise modeling: Case study and experiences. In: MODELSWARD, pp. 74–85 (2024)
13. Grohs, M., Abb, L., Elsayed, N., Rehse, J.R.: Large language models can accomplish business process management tasks. In: International Conference on Business Process Management, pp. 453–465. Springer (2023). https://doi.org/10.1007/978-3-031-50974-2_34

14. Kampik, T., et al.: Large process models: a vision for business process management in the age of generative AI. KI-Künstliche Intelligenz, pp. 1–15 (2024)
15. Kourani, H., Berti, A., Schuster, D., van der Aalst, W.M.: Process modeling with large language models. In: van der Aa, H., Bork, D., Schmidt, R., Sturm, A. (eds.) International Conference on Business Process Modeling, Development and Support, pp. 229–244. Springer, Cham (2024). https://doi.org/10.1007/978-3-031-61007-3_18
16. Kourani, H., Berti, A., Schuster, D., van der Aalst, W.M.: ProMoAI: process modeling with generative AI. arXiv preprint arXiv:2403.04327 (2024)
17. Koutsopoulos, G., Henkel, M., Stirna, J.: Improvements on capability modeling by implementing expert knowledge about organizational change. In: Grabis, J., Bork, D. (eds.) PoEM 2020. LNBIP, vol. 400, pp. 171–185. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-63479-7_12
18. Kwiatkowski, T., et al.: Natural questions: a benchmark for question answering research. Trans. Assoc. Comput. Linguist. **7**, 453–466 (2019)
19. LeanIX: LeanIX Official Website. https://www.leanix.net/en/ (2024). Accessed 27 July 2024
20. Liu, P., Yuan, W., Fu, J., Jiang, Z., Hayashi, H., Neubig, G.: Pre-train, prompt, and predict: a systematic survey of prompting methods in natural language processing. ACM Comput. Surv. **55**(9), 1–35 (2023)
21. Miro: Business capability map template. https://www.smartsheet.com/content/business-capability-templates (2024). Accessed 18 Nov 2024
22. Ollama: Ollama (2024). https://ollama.com. Accessed 16 Nov 2024
23. Samarasekara, B., Rabhi, B., Meymandpour: LLM driven approach for capability modelling with context-enriched prompt engineering. In: Australasian Conference on Information Systems. Manuscript accepted for publication (2025)
24. Sciavolino, C., Zhong, Z., Lee, J., Chen, D.: Simple entity-centric questions challenge dense retrievers. arXiv preprint arXiv:2109.08535 (2021)
25. Simon, C., Haag, S., Zakfeld, L.: Experiments on GPT-3 assisted process model development. ECMS **2023**, 37 (2023)
26. Software, L.: Business capability map template. https://www.lucidchart.com/pages/templates/business-capability-map (2024). Accessed 18 Nov 2024
27. Sun, K., Xu, Y.E., Zha, H., Liu, Y., Dong, X.L.: Head-to-tail: how knowledgeable are large language models (LLM)? AKA will LLMs replace knowledge graphs? arXiv preprint arXiv:2308.10168 (2023)
28. Sun, Y., et al.: Are large language models a good replacement of taxonomies? arXiv preprint arXiv:2406.11131 (2024)
29. Ulrich, W., Rosen, M.: The business capability map: the" rosetta stone" of business/it alignment. Cutter Consortium, Enterprise Architecture **24**(4) (2011)
30. Van Riel, J., Poels, G.: A method for developing generic capability maps: a design science study in the professional sport industry. Bus. Inf. Syst. Eng. **65**(4), 403–424 (2023)
31. Yang, X., et al.: Crag–Comprehensive rag benchmark. arXiv preprint arXiv:2406.04744 (2024)
32. Zhang, Z., et al.: Balancing specialized and general skills in LLMs: the impact of modern tuning and data strategy. arXiv preprint arXiv:2310.04945 (2023)

Security

Evaluating Organization Security: User Stories of European Union NIS2 Directive

Mari Seeba[1,2](✉), Magnus Valgre[1], and Raimundas Matulevičius[1]

[1] Institute of Computer Science, University of Tartu, Tartu, Estonia
{mari.seeba,magnus.valgre,raimundas.matulevicius}@ut.ee
[2] Estonian Information System Authority, Tallinn, Estonia

Abstract. The NIS2 directive requires EU Member States to ensure a consistently high level of cybersecurity by setting risk-management measures for essential and important entities. Evaluations are necessary to assess whether the required security level is met. This involves understanding the needs and goals of different personas defined by NIS2, who benefit from evaluation results. In this paper, we consider how NIS2 user stories support the evaluation of the level of information security in organizations. Using requirements elicitation principles, we extracted the legal requirements from NIS2 from our narrowed scope, identified six key personas and their goals, formulated user stories based on the gathered information, and validated the usability and relevance of the user stories with security evaluation instruments or methods we found from the literature. The defined user stories help to adjust existing instruments and methods of assessing the security level to comply with NIS2. On the other hand, user stories enable us to see the patterns related to security evaluation when developing new NIS2-compliant security evaluation methods to optimize the administrative burden of entities.

Keywords: NIS2 Directive · Security Evaluation · User Stories · Organizations Security level

1 Introduction

In 2015, the enactment of the European (EU) GDPR data protection regulation changed the attitude toward data privacy and raised awareness of the issue [40]. The impact of the implementation of the GDPR has been global. Similar data protection regulations have now been established all over the world [6]. With the NIS2 Directive, the aim of the EU Commission [8] is to change the information security management postures of organizations in the EU to effectively protect the digital single market, and reduce the damaging impacts of security incidents on the economy and society [8]. Similarly to the enactment of GDPR, a widespread increase in security awareness and implementation of the directive's requirements outside the EU is expected.

J. Krogstie et al. (Eds.): CAiSE 2025, LNCS 15701, pp. 57–74, 2025.
https://doi.org/10.1007/978-3-031-94569-4_4

From a policymaker's perspective, NIS2 creates explicit measures for entities required to implement the directive's requirements [13,48]. From the perspective of implementers and engineers, the complexity of interpreting and implementing the regulations is recognized [11,18,23]. For engineers, security is relative, depending on many factors, and the *all-hazards approach* used in NIS2 [8] is an unattainable situation. Therefore, it is appropriate to reformulate the requirements of the regulations into a format understandable to engineers. This allows policymakers and implementers to break out of their silos and get involved in a dialogue, and exchange feedback on the practical effectiveness of the directive [1,32]. Additionally, implementing the regulations should not involve reinventing the wheel but rather building on existing standards and solutions for harmonization [41].

In this paper, we focus on one of the NIS2 directive's objectives - achieving a common high level of security across the EU [8]. We narrowed our research area to the evaluation of the implementation level of risk-management measures, which are presented in NIS2 to organizations. We analyzed the regulation to identify who must meet the specified requirements and for what purpose, based on data from the assessment of organizations' information security levels. Ultimately, we elicit user stories related to the results of organizations' information security level evaluation support that meet NIS2 requirements. Using the requirement engineering method, we discovered 6 personas and 10 user stories that are directly related to an organization's security level evaluation results. Identified user stories are the prerequisite for identifying ways for NIS2 implementation.

2 Backgound

2.1 NIS2

The NIS2 Directive [8] was published in the Official Journal of the EU and entered into force on January 16, 2023. It aims to establish a high level of cybersecurity across the Union to protect the European single market from security incidents that could disrupt the economy and society [8]. NIS2 provides risk-management measures for entities, addresses communication channels and reporting, defines contact points during security incidents, and guides supervisory activities and penalties. The organizations who are required to comply with NIS2 are public or private entities in the high- criticality sectors, such as energy, transport, drinking and wastewater, public administration, digital infrastructure, and others listed in Annexes I and II of NIS2 [8]. Compared to NIS1 (published 2016) the NIS2 added more than 10 obliged sectors. Therefore, the requirements must be explicit to the implementing entities and achievable with reasonable investment and administrative costs. EU Member States (MS) had to transpose NIS2 into local law by the 17th of October, 2024 [8].

Eliciting requirements from a legal text is complicated because (i) the text is fragmented and uses concepts and terminology that are different from software engineering, (ii) requirements can arise from different levels of law (e.g., EU level or the member state level or regulative standard), (iii) imperfection and vagueness of the law and its wording allows multiple interpretations and (iv) dynamics of the law over time [18,23]. To mitigate risks (ii) and (iv), we only considered the EU-level NIS2 directive, which all Member States must adopt based on the principle of minimum harmonization stated in Article 5 [8]. That means the Member State must adopt NIS2 as the minimum baseline. We do not address the level of Member State's requirements. We only used the version of NIS2 [8] from 2022. To mitigate risks (i) and (iii), we used methods described in Sect. 4.1.

2.2 Security Evaluation

Various methods and instruments [22,24,35] can be used to assess an entity's security level, which can be done through direct measurements of risk management measures, self-assessments (e.g., security maturity models), or second or third-party evaluations (e.g., audits, penetration testing). More indirect measurements can also be used to assess the level of security, such as counting the number of organizations that hold some kind of security management certificate (e.g., ISO/IEC27001 Information security management system compliance certificates). This study explores how the security level evaluation results of NIS2-obliged entities can be interpreted and applied to evaluate risk-management measures implementation independently of any specific evaluation method or instrument.

3 Related Work

Only six of the 27 Member States (Belgium, Croatia, Hungary, Italy, Latvia, and Lithuania)[1] succeeded in transposing the NIS2 into local law by October 17, 2024. One of the reasons for the delay in this transposition is the different views of lawyers and implementers on the applicability of the legal text.

Juridical publications (e.g., [3,48]) see the NIS2 Directive primarily as an enabler of raising the security levels of the Member States and emphasize sanctions for non-compliance, and describes NIS2 as a regulation with explicit requirements. In the view of the engineers, the intricate structure and complex legal language of the texts cause questions and ambiguities [13,14]. From the legal point of view, the primary concern is about the excessive administrative burden [48]. However, this originates from the separate implementation of every single clause rather than a comprehensive information security management system, which would align with best practices (e.g., the international standard ISO/IEC 27001 controls [19]) and support the entities' whole management system. If the lawyers recommend balancing requirements in implementing regulations [48], legal text analysis by engineers would instead find optimization patterns and holistic models [13,14,21].

There are few references to the analysis of evaluating the security level concerning NIS2. Wanecki et al. [49] developed a cybersecurity model based on NIS2

[1] https://dnsrf.org/nis2-transition/ NIS2 transition tracker status by 2024-11-01.

but did not cover the evaluation of the achieved security level. The only option mentioned is conducting audits, which only cover essential entities. Grigaliunas et al. [13] created a GDPR, NIS2, and ISO/IEC 27001-based framework that categorizes controls into preventive, detective, and corrective; allowing entities to align their security maturity levels but without considering other stakeholders' expectations on security level evaluations.

Fatema et al. [11] first extracted the relevant legal clauses and eliminated the irrelevant to determine the relevant scope. Hassani et al. [14], using LLM-s for legal compliance analysis, turned attention to legal text sentences that are not separate units. It is essential to follow the sequence, definitions, and cross-references as a whole. It is not an option to treat individual sentences out of context. Here the personas and their relationship models can be helpful. Therefore, the legal text analysis cannot start with extracting relevant information, the entire text of the regulation must first processed.

Legal text ambiguity patterns (lexical, analytical, vagueness, and generality) are described by Alsaadi et al. [1], who studied the EU Medical Device Regulation (MDR) and the Health Insurance Portability and Accountability Act (HIPAA). After analyzing the text, they set a goal to remove the ambiguity by rephrasing the legal requirements into user stories. They used more relevant terms to make user stories unambiguous for engineers and enable discussions about the personas' activities and their purposes.

Based on the previously described references, we performed our requirements elicitation. The method is described in more detail in the next section.

4 Creating User Stories

4.1 Method

We elicited the personas, their goals and dependencies, and user stories. As our study is based on the NIS2 Directive's [8] legal text, we also followed the suggestions on legal text analysis described by [11,18,21].

Following the example of [11], after reading the entire NIS2 directive text, we only selected the clauses relevant to our study. Then we analyzed the sentences individually, marking actors, actions, and resources as suggested by Islam et al. [18]. Following the steps outlined in [18,21], we created the strategic dependency goal model of the personas in i* modelling language [51].

Next, we defined the user stories (see in Sect. 4.4) related to the organization's security level evaluation following the template format proposed by Cohn [4,5]:

```
As a <type of user>, I can <some goal> so that <some reason>.
```

The simple template-based structure is understandable to stakeholders and software engineers by helping to reach a common understanding of the requirements and define the quality guideline [25]. To validate the user stories, we aligned them with existing methods and instruments to demonstrate that the use cases they covered already exist in practice.

4.2 NIS2 Requirements Elicitation

Our scope is to find from NIS2 the clauses related to organizations' (in NIS2 vocabulary - essential and important entities) security level evaluation. At first, we got acquainted with the whole NIS2 text, and we highlighted the relevant clauses that can be interpreted in the context of security evaluation of entities. We also used searches for keywords such as *ensure*, *level*, *assess**, *oversee*, and *measures* for crosschecking. Keywords were chosen based on interpretation options: at the Member State level, the term *ensure* can be interpreted as requiring the Member State to evaluate and measure entities' security levels. This evaluation process ensures that entities comply with regulations and maintain an expected level of cybersecurity. To get the *oversee*, an activity related to evaluation is needed. A term *level* describes an association with something, which refers to measurement, assessment, or evaluation. We selected string *measure** to find the relationship between risk-management measures, as well as the relationship between measurement. In the Appendix A are shown the filtered clauses, which relate to entities and their security evaluation.

To identify the personas and find their dependencies and goals, we analyzed all selected clauses of NIS2 using [18] legal text analysis model steps. In the legal text, we marked personas or subjects of action as <u>underlined</u>, normative phrases and modal verbs are marked in **bold**, and actions in *italics* like:

Art20(1) "<u>Member States</u> **shall ensure** that the <u>**management bodies of essential and important entities**</u> *approve the cybersecurity risk-management measures taken* by those <u>entities</u> in order *to comply with Article 21*, **oversee its implementation** and can be held liable for infringements by the entities of that Article." [8]

This allowed us to pick out the preliminary mentioned personas (actors): ENISA, European Parliament, peer reviewers, Member State, small and medium-sized enterprises, management bodies of essential and important entities, important entities, essential entities, service providers & suppliers, and competent authority for supervisory.

We excluded the European Parliament from this list as it is outside the scope of organization-level security evaluation. Also, we excluded the organization's internal structure and processes and focused only on the organization as a general entity with its management body and employees. We engaged peer reviews under the Member State persona, as the process of peer reviews is organized at the Member State level in cooperation with cybersecurity experts from at least two Member States. In our security evaluation scope, the essential and important entities differ only in the supervisory context, where essential entities should be subject to a comprehensive supervisory regime (preventative and after security incidents). In contrast, important entities should be subject to a simplified supervisory regime after a security event or someone's hint of an entity security violation. Security level evaluation is similar in both cases. We also included small and medium-sized enterprises under the persona Entity and Suppliers or Service Providers because all Entities can simultaneously be someone's Service Provider & Supplier and essential and important entities.

Additionally to already mentioned personas, from NIS2 recitals No 56, we found the persona called *Member State point of contact* for small and medium-sized enterprises, who should guide and assist small and medium-sized enterprises regarding cybersecurity-related issues. Impersonally, but the same guidance and assistance issue is mentioned as an expected clause of Member State National cybersecurity policy (Art 7(2)(f) and (i) [8]). To avoid confusion with another single point of contact used for different processes described in NIS2, we named this guidance and assistance provider as a Security Consultant.

So, we limited the personas of NIS2, who are relevant in the context of organization security level evaluation with the list: *Member State*, *Supervisory Authority*, *ENISA*, *Entity*, *Security Consultant* and *Service Provider & Supplier*. In the next section, based on legal text analysis, we describe the personas mutual relations goal model.

4.3 Personas' Dependency Model

The six personas rely on organizations' security data or generalized results to achieve their objectives. The goal of the **Member State** is to receive secure services from Entities, obtain the service's security statuses, assign Consultants to support Entities with cyber-security issues, assign a Supervisory Authority and provide guidance and training on cybersecurity to Entities ([8] Articles: Art1(1); Art7(2); Art19(1.a); Art20(1),(2); Art21(1),(2),(3),(4); Art31(2), Art32(2),(4); Art33(2)). **Supervisory Authority** is assigned by Member State. It should provide feedback on Entities' security status ([8] Articles: Art21(1),(2),(4); Art31(2); Art32(2),(4); Art33(2)). **ENISAs'** goal is to evaluate the security status of Member States and Entities and provide results to the EU Parliament so that it could assess the EU security level ([8] Articles: Art1(1), Art18(1)). **Security Consultants** assist and guide Entities on risk-management measures implementation and could be assigned by Member State ([8] Articles: Art 7(2); Art20(2); Art21(2)). **Entity** provides secure services to Member State and follows Member State regulations (implements risk-management measures, passes training). It also gets secure services and products from Service Provider & Supplier ([8] Articles: Art7(2); Art20(1),(2); Art21(1),(2),(3),(4); Art32(2),(4); Art33(2)). **Service Provider & Supplier provides** provide secure services or products to Entity, ([8] Articles: Art21(2); Art21(3)).

We illustrate the above dependencies in Fig. 1. The model emphasizes the personas' dependencies and supports the user stories. For simplification, ENISA is not included. However, as described above, ENISA obtains the security status of the Member State and Entities and shares the best practices with other Member States. The prioritization and optimization of activities is the task of the Supervisory Authority. It should be noted that a specific organization can take different roles. For instance, an Entity can simultaneously be a Service Provider, Supplier, and Security Consultant. In some cases, the Entity can be a Member State or Supervisory Authority (e.g., Computer Security Incident Response Team, CSIRT).

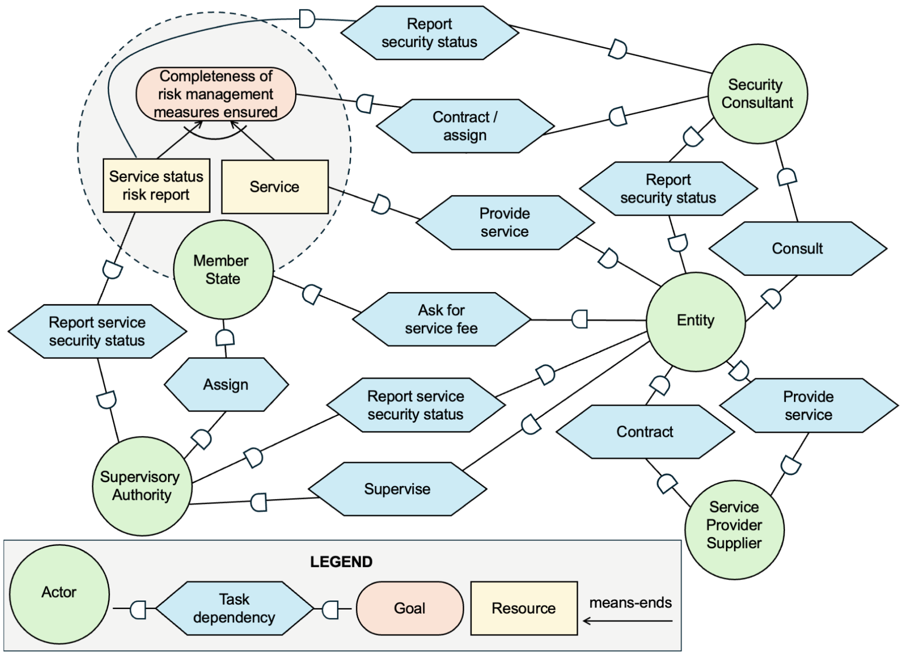


Fig. 1. Personas' Dependency Model

4.4 Security Evaluation User Stories

Next, we describe the User Stories. They are formulated based on the legal requirements quoted in the Appendix A) and the dependencies illustrated in Fig. 1. User Stories are divided into groups based on Personas. The result is presented in Table 1, where the goals regarding security level evaluations are described for each Persona. We also included references to NIS2 clauses.

Table 1. User Stories of NIS2 [8] Related to Security Level Evaluation of Entities

Role:	**Member State**
Goal:	Factual proof of achieving a high common level of cybersecurity in all sectors and entities to avoid cyber incidents causing major damage to economics and society.
Reference:	Art1(1); Art7(2); Art19(1.a); Art20(1),(2); Art21(1),(2),(3),(4) of NIS2 [8]
US1.1: As a Member State, I can oversee the security posture of Entities through structured security level evaluation results, so that I achieve awareness of compliance with regulations.	
US1.2: As a Member State, I can evaluate an entity's cybersecurity level using an all-hazards approach, so that I can allocate resources to address directly on identified vulnerabilities.	

(*continued*)

Table 1. (*continued*)

Role:	**Supervisory Authority**
Goal:	The Supervisory Authority should base on risk assessments when planning their supervisory tasks, but they should optimize the workflow and not unnecessarily hamper the business activities of the entity concerned.
Reference:	Art21(1),(2),(4); Art31(2); Art32(2); Art32(4); Art33(2)of NIS2 [8]
US2.1 As a Supervisory Authority, I can prioritize supervisory tasks by using all hazard-covering security evaluation results so that I can focus supervisory tasks on high-risk entities or areas.	
US2.2 As a Supervisory Authority, I can ensure (with a security evaluation instrument) that entities that did not comply with regulatory requirements implement corrective risk-management measures within reasonable deadlines so that supervisory resources are used effectively and unnecessarily hamper the business activities of the entity is avoided.	
Role:	**ENISA**
Goal:	Collect data to evaluate EU security level and report the result to EU Parliament
Reference:	Art1(1), Art18(1) of NIS2 [8]
US3.1 As ENISA, I can collaborate with Member States to assess collected evaluation data on cybersecurity capabilities and awareness, so that I can share cybersecurity best practices and gaps across the European Union.	
Role:	**Security Consultant**
Goal:	Consultants should help improve the entity's security level by finding and focusing on vulnerable areas of the entity.
Reference:	Art20(2); Art21(2) of NIS2 [8]
US4.1 As a security consultant, I can get an overview of the entity's security maturity evaluation results so that the most vulnerable areas can be prioritized in a timely manner for an improvement plan.	
US4.2 As a security consultant, I can re-evaluate the entity's risk-management measures implementation so that tracking characterizes risk-management measures implementation status progress.	
Role:	**Entity**
Goal:	To obtain an overview of the entity's cybersecurity risk-management measures all-hazard approach to confirm security status and improve vulnerable areas
Reference:	Art20(2); Art21(2) of NIS2 [8]
US5.1 As an entity, I can ensure the entity adopts an all-hazards approach[4] to cybersecurity so that the evaluation results show strengths and direct to plan improvements to our security shortcomings.	
Role:	**Service Provider & Supplier**
Goal:	Get an evaluation of Service Provider & Supplier security to share with partners and assess compliance with partner requirements
Reference:	Art21(2); Art21(3) of NIS2 [8]
US6.1 As a Service Provider & Supplier, I can provide the risk-management measures in all-hazard evaluation approach results to the partner Entity so that the Entity can choose us as the most suitable secure suppliers.	
US6.2 As a Service Provider & Supplier, I can regularly evaluate my cybersecurity practices so that I can present my evaluation results to my partner Entity to demonstrate our security.	

[4]Some recognized standard like ISO27001 [19] can limit the uncertainty of the all-hazard approach.

To explain the user stories provided in the Table 1, we will take a closer look at how to use user stories and also show how user stories are connected and need additional analysis by the Member State (MS).

US2.1 calls for the Supervisory Authority to prioritize activities in the context of all risks and plan its activities based on this. In essence, this prepares a sample of organizations to focus on. MSs can have several security-related supervisory authorities with different focuses in terms of sectors and functions (e.g., in Estonia, there are three: NCSC-EE, whose supervision deals with cybersecurity in general; the Financial Supervision Authority focuses on the financial sector, and the Data Protection Authority monitors data protection. In Finland, oversight of cybersecurity issues is divided between 8 institutions by area.) Each supervisory unit must prepare its plans based on its objectives and, in addition, national risk assessments.

Therefore, a Supervisory Authority needs aggregated data collected from organizations (reusing the data collected during US5.1) that distinguishes between relevant sectors and their vulnerabilities. This focuses the monitoring process on identifying causes and finding inputs for improvement according to sectors (e.g., finance, energy, research) or security functions (e.g., data protection, incident management, cyber hygiene, and awareness).

However, the MS must specify which metadata must be collected during the US5.1 process for filling in the US2.1 (but also US1.1, US1.2, US3.1). This could depend on MS supervisory authorities, their sectoral affiliation, and how detailed the aggregation needs to be. The questionnaire or instrument detail level should match the supervisory focus needs and data protection requirements, balancing aggregation and listing specific technical measures.

4.5 Validity of User Stories

We validated the user stories by aligning them to existing or proposed security evaluation instruments. The instruments were chosen to cover a wide spectrum of applications and to show that the user stories described in this paper and in NIS2 reflect the situation in the real world. The reviewed instruments can be divided into the following categories: publications [2,20,26,27,33,34,37,50], ENISA or state-sponsored tools [9,12,15,31,42–44], cybersecurity indices [7,10,17,28,47], maturity models [46], and official audits [29,30,45].

An overview of the instruments and their coverage of user stories is illustrated in Table 2. Each covers at least one of the user stories. The instruments have been created for different purposes and levels of abstraction. Some instruments compare and describe the cybersecurity postures of countries on a global scale (e.g. [7,10,28,47]), while other instruments are meant for individual organizations (e.g. [15,43,44]) and comply with the Entity, Security Consultant or Service Provider & Supplier user stories.

Different abstraction levels require different approaches, which can lead to loss of detail. Cybersecurity indices [7,10,28,47] compare the security postures of entire countries. However, they do not consider the differences in the levels of digitalization, that determines the actual required level of security. Indices

Table 2. Instruments that implement user stories: '+' instrument covers the given user story; '−' instrument does not cover the user story; 'v' instrument can be used to cover the user story, but it is not explicitly meant for that purpose; '*' instrument is not usable *as-is* and needs significant effort to be workable.

Instruments	US1	US2	US3	US4	US5	US6
F4SLE [37], Kybermittari [12], Jazri et al. [20]*	+	+	v	+	+	+
Bernik et al. [2]*, Prislan et al. [33]*, Maleh et al. [27]*, Rae and Patel [34]*, Malaivongs et al. [26]*, You et al. [50]*, self-assessment tools by Ireland [43] and Greece [15], C2M2 Maturity Model [46]	-	-	v	v	+	v
NÚKIB Report [31]*	+	+	-	-	-	-
EU CSI [10]*	+	-	+	-	-	-
Cybersecurity Indices NCSI [7]*, GCI [17]*, NCPI [47]*, CDI [28]*,	+	-	-	-	-	-
Official Audits by Estonia [30]*, [29]* and Latvia [45]*	+	v	-	-	-	-
Self-Assessment tools by ENISA [9], IASME [42] and Spain [44]	-	-	-	v	+	v

rely on high-level data, such as the existence of appropriate legislation and cybersecurity-related institutions, but these facts will not be helpful for individual entities in determining their security level. Still, indices can contain some data on entities (e.g., how many organizations have attained some specific security certifications [10]). The EU Cybersecurity Index [10] created by ENISA uses data gathered from EU member states and is also the subject of US3.

From the bottom-up perspective, most of the methods available are created to help individual organizations perform a self-assessment to find areas of improvement (e.g. [42,44]) and they lack functionality in aggregating data and presenting it at a higher level of abstraction. Still, they fulfill the goals of US5 but can also cover US4 and US6.

The least covered is US2, which means the needs of governmental overseeing bodies are the least considered by the currently available instruments. Two notable instruments here are the audits performed by national audit offices [30,45] and the annual report [31] composed by the National Cyber and Information Security Agency of the Czech Republic (NÚKIB). Due to their thorough nature, audits bring a lot of insight into a given topic and provide concrete recommendations for improvement, but they are not periodically done on the same topic and this limits their usefulness in verifying that improvements were implemented. The NÚKIB report [31] gathers its data surveys of entities without any individual feedback to the Entity.

There is a real lack of instruments and methods that help collect security evaluation data on individual entities and bring it together for central decision-makers but also, at the same time, provide the given organization with feedback on their current capabilities and areas that need enhancement. In other words, there are only a few tools [12,37] that simultaneously cover the User Stories related to Entities and MSs.

Few security evaluation instruments directly corresponding to NIS2 requirements have yet been created [12,37]. However, an ENISA security assessment pilot report based on NIS2 Article 18 [8] has already been completed, but only for internal use and not publicly available.

The analysis showed that the tools tend to be mono-functional, but some are multifunctional (e.g. [2,43]), simultaneously covering several user stories. Instruments are divided into Member State (US1-US3) and Entity goals (US4-US6). From Table 2, we can see that all our user stories are covered at least by one instrument, showing that user stories are realistic. However, not all instruments are usable for each user story or suitable for periodic/repeatable evaluation.

5 Discussion

A security evaluation aims to provide situational awareness and present the dynamics of security. During the validation of the user stories, we identified different security evaluation methods and instruments which fulfill the needs of either the Member States or the Entities.

Security risk management is an iterative process. If an adversary is able to identify a single vulnerability, they are able to damage the system. However, the Entity (i.e., defender) must implement multiple security countermeasures to mitigate various security risks. This task requires considerable resources. Any activity (including an evaluation of the security level) that does not directly contribute to risk mitigation can significantly burden the organization's administration. Therefore, evaluating the security level should not be a goal in itself; rather, it must be an integrated part of security management and create value for the Entity. We observed four personas (Member State, Supervisory Authority, Consultant and ENISA) who require security evaluation input from entities to support their tasks. The needs of other stakeholders should be integrated into the Entity's security evaluation process. The user stories could help achieve this goal of finding ways to reuse data, optimize, automate, and manage security evaluation, especially from the Entity's point of view.

Finding patterns within legal requirements is the job of experts and engineers. So is optimizing or balancing burdensome activities, as described in Sect. 3 by [13,14,21]. Our research showed that the NIS2 analysis allowed us to find optimization points that could simplify the implementation of NIS2 in a less burdensome way.

Limitation: Our scope is narrowed to NIS2 and does not expand to other EU regulations. User stories follow the NIS2 Directive [8] and are written at a high level of abstraction. This approach allows us to avoid conflicts with the local laws of the Member States when the Member States have transposed NIS2. It also ensures the instantiation of user stories in Member States by adding contextual details. The flexibility of the user stories may lead to challenges in implementation, as they are not explicitly aligned with specific standards or tools.

NIS2 does not refer to any reference standards, the user stories must remain flexible in their application. However, this flexibility may limit the coverage of user stories and their scope in the security assessment. For example, information security management standards (e.g., ISO/IEC27002 [16]) cover the risk-management measures detailed in Article 21(2) of NIS2, but these standards do not specify NIS2-compliant reporting during incidents. Similar concerns are observed for the security awareness training and awareness evaluation of the management boards. However, the Estonian Information Security Standard (E-ITS) [36], a detailed catalog of measures and guidelines, provides corresponding instructions.

The user stories do not explicitly address the continuous compliance required by NIS2, as this is ensured by default through repeated security level evaluations.

6 Concluding Remarks

In this paper, we identified six stakeholders (personas) from the NIS2 directive who depend on the results of an organization's (entity's) security level evaluations to fulfill their tasks. We created user stories that reflect each persona's relation to the evaluation results. The user stories are not dependent on any standards or instruments used for security evaluations. Instead, the user stories are described at the general level.

When adopting NIS2 and planning the security evaluation activities, a Member State should consider how to avoid overloading organizations. Different personas might behave in ways that are based only on their individual needs. The defined user stories could support the planning process by reusing security evaluation results without overburdening the organizations.

Further Work. As we are developing the FASLE instrument [37–39], we work with stakeholders to test all the described user stories with F4SLE in real-world situations to achieve their operational objectives. This way, we can detail the user stories at the national level to show how they can be implemented with the instrument that collects data only once and uses it for different stakeholders.

Acknowledgments. This work is part of the Cyber-security Excellence Hub in Estonia and South Moravia (CHESS) project funded by the European Union under Grant Agreement No. 101087529. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or European Research Executive Agency. Neither the European Union nor the granting authority can be held responsible for them.

Disclosure of Interests. The authors have no competing interests to declare relevant to this article's content.

A Appendix: NIS2 Extracted and Marked Clauses

Art1(1) "This Directive lays down *measure*s that aim to achieve a high common *level* of cybersecurity across the Union, with a view to improving the functioning of the internal market" [8].

Art7(2) As part of the national cybersecurity strategy, Member States **shall** in particular **adopt policies**: (f) promoting and developing education and training on cybersecurity, cybersecurity skills, awareness raising and research and development initiatives, as well as guidance on good cyber hygiene practices and controls, aimed at citizens, stakeholders and entities; (i) strengthening the cyber resilience and the cyber hygiene baseline of small and medium-sized enterprises, in particular those excluded from the scope of this Directive, by providing easily accessible *guidance and assistance for their specific needs*" [8].

Art18(1) requires ENISA **to compile the Report on the state of cybersecurity** in the Union ENISA **shall adopt**, in cooperation with the Commission and the Cooperation Group, a biennial *report on the state of cybersecurity* in the Union and **shall submit and present** that report to the European Parliament. The report **shall**, inter alia, be made available in machine-readable data and **include the following**: (b) an **assessment of the development of** cybersecurity capabilities in the public and private sectors across the Union; (c) **an assessment of the general level of cybersecurity awareness and cyber hygiene** among citizens and entities, including small and medium-sized enterprises; (e) an aggregated **assessment of the level of maturity** of cybersecurity capabilities and resources across the Union, including those at sector level, as well as of the extent to which the Member States' national cybersecurity strategies are aligned" [8].

Art19(1) The peer reviews **shall cover** at least one of the following: a) the **level of implementation of the cybersecurity risk-management measures** /.../ laid down in Articles 21 " [8].

Art20(1)"Member States **shall ensure** that the **management bodies of essential and important entities** *approve the cybersecurity risk-management measures taken* by those entities in order *to comply with Article 21*, **oversee its implementation** and can be held liable for infringements by the entities of that Article." [8].

Art20(2)Member States **shall ensure** that the members of the management bodies of essential and important entities are required to follow training, and shall encourage essential and important entities to offer similar training to their employees on a regular basis, in order that they gain sufficient knowledge and skills to enable them to identify risks and *assess cybersecurity risk-management practices* and their impact on the services provided by the entity" [8].

Art21(1) Member States **shall ensure** that **essential and important entities** take **appropriate and proportionate technical, operational and organizational measures to manage the risks** posed to the security of network and information systems which those entities use for their operations or for the provision of their services, and to prevent or minimise the impact of incidents on recipients of their services and on other services" [8].

Art21(2) The *measures* referred to in paragraph 1 **shall be based on an all-hazards approach** that aims to protect network and information systems and the physical environment of those systems from incidents, and shall include at least the following: a) policies on risk analysis and information system security; b) incident handling; c) business continuity, such as backup management and disaster recovery, and crisis management; d) supply chain security, including security-related aspects concerning the relationships between each entity and its direct suppliers or service providers; e) security in network and information systems acquisition, development and maintenance, including vulnerability handling and disclosure; f) policies and procedures to assess the effectiveness of cybersecurity risk-management measures; g) basic cyber hygiene practices and cybersecurity training; h) policies and procedures regarding the use of cryptography and, where appropriate, encryption; i) human resources security, access control policies and asset management; j) the use of multi-factor authentication or continuous authentication solutions, secured voice, video and text communications and secured emergency communication systems within the entity, where appropriate" [8].

Art21(3) Member States **shall ensure** that, when considering which measures referred to in paragraph 2, point (d), of this Article are appropriate, entities *take into account the vulnerabilities* specific to each direct supplier and service provider and the overall quality of products and *cybersecurity practices of their suppliers and service providers*, including their secure development procedures" [8].

Art21(4) Member States **shall** *ensure* that an entity that finds that it does not comply with the measures provided for in paragraph 2 *takes*, without undue delay, *all necessary, appropriate and proportionate corrective measures*" [8].

Art31(2) Member States**may allow their competent authorities to prioritise supervisory tasks**. Such prioritisation s**hall be based on a risk-based approach**. To that end, when exercising their supervisory tasks provided for in Articles 32 and 33, the competent authorities may establish supervisory methodologies allowing for a prioritisation of such tasks following a risk-based approach" [8].

Art32(2) Member States **shall ensure** that the competent authorities, when exercising their supervisory tasks in relation to essential entities, have the power to subject those entities at least to: a) on-site inspections and *off-site supervision*, including random checks conducted by trained professionals; e) r*equests for information necessary to assess the cybersecurity risk-management measures* adopted by the entity concerned, including documented cybersecurity policies, as well as compliance with the obligation to submit information to the competent authorities pursuant to Article 27" [8].

Art32(4) Member States **shall ensure** that their competent authorities, when exercising their enforcement powers in relation to essential entities, have the power at least to: d) order the entities concerned to *ensure* that their cybersecurity risk-management *measure*s comply with Article 21 /.../ , in a specified manner and within a specified period; f) order the entities concerned to implement the recommendations provided as a result of a security audit within a reasonable deadline; g) designate a monitoring officer with well-defined tasks **for**

a determined period of time to oversee the compliance of the entities concerned with Articles 21 /.../" [8].

Art33(2) Member States **shall ensure** that the competent authorities, when exercising their supervisory tasks in relation to important entities, have the power to subject those entities at least to: a) *on-site inspections and off-site ex post supervision* conducted by trained professionals" [8].

Recital (56) Member States **should have** a point of contact for small and medium-sized enterprises at national or regional level, which either *provides guidance and assistance* to small and medium-sized enterprises **or** *directs them to the appropriate bodies* for guidance and assistance with regard to cybersecurity related issues" [8].

References

1. Alsaadi, M., Lisitsa, A., Qasaimeh, M.: Minimizing the ambiguities in medical devices regulations based on software requirement engineering techniques (2019). https://doi.org/10.1145/3368691.3368709
2. Bernik, I., Prislan, K.: Measuring information security performance with 10 by 10 model for holistic state evaluation (2016). https://doi.org/10.1371/journal.pone.0163050
3. Chiara, P.G.: Towards a right to cybersecurity in EU law? Challenges Ahead (2024). https://doi.org/10.1016/j.clsr.2024.105961
4. Cohn, M.: User Stories Applied: For Agile Software Development (2004)
5. Cohn, M.: The Two Ways to Add Detail to User Stories (2017). https://www.mountaingoatsoftware.com/blog/preview/1691. Accessed 20 April 2024
6. Dixon, P., Emerson, J.: Global Visualization of Countries with Data Privacy Laws, Treaties, or Conventions. https://www.worldprivacyforum.org/2024/06/countries-with-data-privacy-laws/. Accessed 25 Nov 2024
7. e-Governance Academy: National cyber security index. https://ncsi.ega.ee/. Accessed 12 April 2024
8. European Parlament: Directive (EU) 2022/2555 of the European Parliament and of the Council of 14 December 2022 on measures for a high common level of cybersecurity across the Union, amending Regulation (EU) No 910/2014 and Directive (EU) 2018/1972, and repealing Directive (EU) 2016/1148 (NIS 2 Directive) (2022). https://eur-lex.europa.eu/legal-content/en/TXT/?uri=CELEX%3A32022L2555
9. European Union Agency for Cybersecurity: Cybersecurity Maturity Assessment for Small and Medium Enterprises. https://www.enisa.europa.eu/cybersecurity-maturity-assessment-for-small-and-medium-enterprises#/. Accessed 13 June 2024
10. European Union Agency for Cybersecurity: EU Cybersecurity Index. https://www.enisa.europa.eu/topics/cybersecurity-policy/nis-directive-new/eu-cybersecurity-index. Accessed 20 May 2024
11. Fatema, K., Debruyne, C., Lewis, D., OSullivan, D., Morrison, J.P., Mazed, A.A.: A Semi-Automated Methodology for Extracting Access Control Rules from the European Data Protection Directive (2016). https://doi.org/10.1109/SPW.2016.16
12. Finnish Transport and Communication Agency National Cyber Security Centre: Cybermeter. https://www.kyberturvallisuuskeskus.fi/fi/palvelumme/tilannekuva-ja-verkostojohtaminen/kybermittari. Accessed 13 May 2024

13. Grigaliūnas, S., Schmidt, M., Brūzgienė, R., Smyrli, P., Andreou, S., Lopata, A.: Holistic Information Security Management and Compliance Framework (2024). https://doi.org/10.3390/electronics13193955
14. Hassani, S., Sabetzadeh, M., Amyot, D., Liao, J.: Rethinking Legal Compliance Automation: Opportunities with Large Language Models (2024). https://doi.org/10.1109/RE59067.2024.00051
15. Hellenic Ministry of Digital Governance Government department: Cybersecurity Self Assessment Tool (2021). https://mindigital.gr/wp-content/uploads/2022/03/cybersecurity-self-assessment.xlsm. Accessed 27 April 2024
16. International Organization for Standardization: ISO/IEC 27002:2022 Information security, cybersecurity and privacy protection - Information security controls (2022)
17. International Telecommunications Union: Global Cybersecurity Index. https://www.itu.int/en/ITU-D/Cybersecurity/Pages/global-cybersecurity-index.aspx. Accessed 20 May 2024
18. Islam, S., Mouratidis, H., Wagner, S.: Towards a framework to elicit and manage security and privacy requirements from laws and regulations. In: Wieringa, R., Persson, A. (eds.) REFSQ 2010. LNCS, vol. 6182, pp. 255–261. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-14192-8_23
19. ISO/IEC 27001:2022(en) Information security, cybersecurity and privacy protection - Information security management systems - Requirements. Standard, International Organization for Standardization (2022)
20. Jazri, H., Zakaria, O., Chikohora, E.: Measuring cybersecurity wellness index of critical organisations (2018)
21. Jorshari, F.Z., Mouratidis, H., Islam, S.: Extracting security requirements from relevant laws and regulations (2012). https://doi.org/10.1109/RCIS.2012.6240443
22. Khaleghi, M., Aref, M.R., Rasti, M.: Comprehensive Comparison of Security Measurement Models (2022). https://doi.org/10.1080/19361610.2021.1981089
23. Kiyavitskaya, N., Krausová, A., Zannone, N.: Why eliciting and managing legal requirements is hard (2008). https://doi.org/10.1109/RELAW.2008.10
24. Leszczyna, R.: Review of cybersecurity assessment methods: applicability perspective (2021). https://doi.org/10.1016/j.cose.2021.102376
25. Lucassen, G., Dalpiaz, F., Werf, J., Brinkkemper, S.: The Use and Effectiveness of User Stories in Practice. In: Daneva, M., Pastor, O. (eds.) REFSQ 2016. LNCS, vol. 9619, pp. 205–222. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-30282-9_14
26. Malaivongs, S., Kiattisin, S., Chatjuthamard, P.: Cyber trust index: a framework for rating and improving cybersecurity performance (2022). https://doi.org/10.3390/app122111174
27. Maleh, Y., Ezzati, A., Sahid, A., Belaissaoui, M.: Towards A Capability Assessment Framework for Information Security Governance in Organization (2017)
28. MIT Technology Review Insights: Cyber Defense Index. https://www.technologyreview.com/2022/11/15/1063189/the-cyber-defense-index-2022-23/. Accessed 12 May 2024
29. National Audit Office of Estonia: Administration and reliability of X-road. https://www.riigikontroll.ee/DesktopModules/DigiDetail/FileDownloader.aspx?FileId=14778&AuditId=2520. Accessed 17 Nov 2024
30. National Audit Office of Estonia: Implementation of system of IT security measures in local governments. https://www.riigikontroll.ee/DesktopModules/DigiDetail/FileDownloader.aspx?FileId=14270&AuditId=2466. Accessed 17 Nov 2024

31. National Cyber and Information Security Agency of the Czech Republic: 2023 Report on the State of Cybersecurity in the Czech Republic. https://nukib.gov.cz/download/publications_en/2023_Report_on_the_State_of_Cybersecurity_in_the_Czech_Republic.pdf. Accessed 8 Nov 2023
32. Pisa, M., Dixon, P., Ndulu, B., Nwankwo, U.: Governing data for development: trends, challenges, and opportunities (2020). https://www.cgdev.org/sites/default/files/governing-data-development-trends-challenges-and-opportunities.pdf
33. Prislan, K., Mihelič, A., Bernik, I.: A real-world information security performance assessment using a multidimensional socio-technical approach (2020). https://doi.org/10.1371/journal.pone.0238739
34. Rae, A., Patel, A.: Defining a new composite cybersecurity rating scheme for SMEs in the U.K. In: Heng, S.-H., Lopez, J. (eds.) ISPEC 2019. LNCS, vol. 11879, pp. 362–380. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-34339-2_20
35. Rea-Guaman, A.M., Sánchez-García, I.D., Feliu, T.S., Calvo-Manzano, J.A.: Maturity models in cybersecurity: a systematic review (2017). https://doi.org/10.23919/CISTI.2017.7975865
36. RIA (Estonian Information System Authority): E-ITS. Portal of Estonian Information Security Standard (2022). https://eits.ria.ee/
37. Seeba, M., Mäses, S., Matulevičius, R.: Method for Evaluating Information Security Level in Organisations (2022). https://doi.org/10.1007/978-3-031-05760-1_39
38. Seeba, M., amefon Obot Affia, A., Mäses, S., Matulevičius, R.: Create your own MUSE: a method for updating security level evaluation instruments. Comput. Standards Interfaces **87**, 103776 (2024). https://doi.org/10.1016/j.csi.2023.103776
39. Seeba, M., Oja, T., Murumaa, M.P., Stupka, V.: Security Level Evaluation with F4SLE. In: Proceedings of the 18th International Conference on Availability, Reliability and Security. ARES 2023, Association for Computing Machinery, New York, NY, USA (2023). https://doi.org/10.1145/3600160.3605045
40. Szczepaniuk, E.K., Szczepaniuk, H., Rokicki, T., Klepacki, B.: Information security assessment in public administration (2020). https://doi.org/10.1016/j.cose.2019.101709
41. Tasheva, I., Kunkel, I.: In a hyperconnected world, is the EU cybersecurity framework connected? (2022). https://doi.org/10.1177/17816858221136106
42. The IASME Consortium: IASME Cyber Essentials. https://getreadyforcyberessentials.iasme.co.uk/. Accessed 13 June 2024
43. The National Cyber Security Centre of Ireland: Cyber Security Baseline Standards Self-Assessment Form (2023). https://www.ncsc.gov.ie/pdfs/Cyber_Resilience_Self-Assessment_Framework_Version_1.4_Jan_23.xlsx. Accessed 27 May 2024
44. The Spanish National Cybersecurity Institute: Herramienta de Autodiagnóstico, https://adl.incibe.es/#. Accessed 27 May 2024
45. The State Audit Office of the Republic of Latvia: Can we rely on the access to information systems and the receipt of e-services? (2022). https://www.lrvk.gov.lv/en/getrevisionfile/29525-5Aio6j7MwYsuSG4nKlzFVmCMG0JZircA.pdf
46. United States Department of Energy: C2M2 C2M2 V2.1 HTML-Based Tool. https://c2m2.doe.gov/c2m2-assessment. Accessed 13 June 2024
47. University of Harvard Belfer Center: National cyber power index 2020 (2022). https://www.belfercenter.org/publication/national-cyber-power-index-2022
48. Vandezande, N.: Cybersecurity in the EU: how the NIS2-directive stacks up against its predecessor (2024). https://doi.org/10.1016/j.clsr.2023.105890

49. Wanecki, P., Jasek, R., Drofova, I.: The Contribution of the European NIS2 directive to the design of the cyber security model (2023). https://doi.org/10.1109/IDT59031.2023.10194454
50. You, Y., Cho, I., Lee, K.: An advanced approach to security measurement system. J. Supercomput. **72**(9), 3443–3454 (2015). https://doi.org/10.1007/s11227-015-1585-7
51. Yu, E., Giorgini, P., Maiden, N., Mylopoulos, J.: Social Modeling for Requirements Engineering: An Introduction (2010). https://doi.org/10.7551/mitpress/7549.001.0001

LitroACP: A Lightweight and Robust Framework for Extracting Access Control Policies from Specifications

Yanqiu Zhang[1,2], Zhen Xu[1,2], DongDong Huo[1,2(✉)], Xiaokun Guo[1,2], Qihui Zhou[1], and Yu Wang[1]

[1] Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China
{zhangyanqiu,xuzhen,huodongdong,guoxiaokun,zhouqihui,wangyu}@iie.ac.cn
[2] School of Cyber Security, University of Chinese Academy of Sciences, Beijing, China

Abstract. Access control is essential for safeguarding data in various enterprise systems. However, creating initial access control policies from high-level specifications is time-consuming and error-prone, and it can introduce security risks. Automated Access Control Policy (ACP) generation is crucial to simplify this process. Despite advancements, challenges such as the lack of standardized datasets, coarse-grained policy extraction, and high resource demands remain. We propose **LitroACP**, a lightweight and robust method for fine-grained policy extraction. By leveraging Named Entity Recognition (NER) and integrating language models, LitroACP introduces three key modules: **ACPUIE** for semi-automated data annotation, **DisAdver** for identifying policy decisions, and **GLiACP** for extracting critical policy components. The extracted components can be further synthesized into structured access control policies based on the identified decision type. LitroACP enhances the use of limited real-world datasets through Projected Gradient Descent (PGD) adversarial training on DistilBERT. Experimental data comes from diverse industries, including healthcare and education, and LitroACP bridges the gap between general knowledge and access control policy domain knowledge through pre-trained models. Compared with three existing frameworks, LitroACP achieves an average F1-score of 93.77% for Natural Language Access Control Policy (NLACP) identification, 77.69% for policy component extraction, and 85.2% for policy decision identification. These results underscore the effectiveness of our framework and provide a solid basis for releasing an open dataset to facilitate further research (Dataset and Code are available at https://github.com/AmberQZ/LitroACP.)

Keywords: Access control · Automated policy extraction · Named entity recognition · Transformer

J. Krogstie et al. (Eds.): CAiSE 2025, LNCS 15701, pp. 75–90, 2025.
https://doi.org/10.1007/978-3-031-94569-4_5

1 Introduction

Access control is crucial for data security, yet creating initial policies remains challenging. Organizations typically provide high-level requirements, drafted by security experts, to define access control policy rules [14]. Access control is important not only in security-sensitive industries but also in fields like healthcare, education, finance, and government. However, specification documents are often unstructured, making manual policies extraction both error-prone and repetitive, leading to risks such as data breaches [15]. To mitigate such risks, implementing automated policy generation is essential. Specification documents typically contain two types of texts: Natural Language Access Control Policy sentences (NLACPs) and unrelated sentences (non-NLACPs) [14]. The primary tasks of ACP extraction are policy decision identification and policy component extraction [6,9,21,22]. For example, given the NLACP sentence *"Doctors can view patient medical records after obtaining consent."*, the system must first classify it as an *Allow* decision, then extract components: *Subject-Doctors, Action-view, Resource-patient medical records, Condition-after obtaining consent*. These components collectively define a structured policy: *Allow (Subject=Doctors, Action=view, Resource=medical records) if Condition=consent obtained*.

High-quality annotated data is key to training models, yet datasets like those from Slankas et al.[1] [20] and Text2Policy[2] are no longer available. Current annotated data, such as that from RAGent [9], suffer from issues like inconsistent annotation and non-compliance with the Principle of Least Privilege (PoLP) of access control. Traditional methods for NLACP identification include rule-based [23], Machine Learning (ML), and Deep Learning (DL) approaches. However, these methods face challenges, such as the lack of sufficient high-quality data [8], and the impracticality of BERT-based models in low-resource settings. For policy component extraction, syntactic parsing methods like dependency parsing [1] and DL-based approaches like Named Entity Recognition (NER) [16] and Semantic Role Labeling (SRL) [3] have limitations, including difficulty in handling complex relationships between subjects and actions. SRL, in particular, struggles with extracting multiple subjects or resources from a single NLACP.

Key challenges in NLACP identification and policy component extraction include (1) domain-specific annotated datasets are both scarce and overly coarse-grained, (2) LLM-based methods struggle in low-resource scenarios where domain-specific fine-tuning data is scarce, and their high computational demands hinder deployment on edge devices; (3) extracted components often lack accuracy. To overcome these issues, we propose **LitroACP**, a fine-grained annotated dataset and a lightweight and robust ACP generation framework. Our contributions are as follows.

1. We propose a semi-automatic annotation method, **ACPUIE**, based on Unified Information Extraction (UIE), which is used to develop a fine-grained, annotated dataset of real-world access control policies.

[1] https://sites.google.com/site/AccessControlRuleExtraction.
[2] http://research.csc.ncsu.edu/ase/projects/text2policy.

2. LitroACP is the first lightweight LLM-based access policy generation method, integrating **DisAdver** and **GLiACP** for policy decision identification and component extraction. DisAdver utilizes PGD-based adversarial training to enhance the identification of policy decisions, while GLiACP integrates general and domain-specific knowledge to extract policy components.
3. We demonstrate the effectiveness of our framework through comparisons with three representative approaches. LitroACP achieves outstanding results, with an F1-score of 93.77% for NLACP identification, 77.69% for policy component extraction, and 85.2% for policy decision identification.

2 Related Work

Automated ACP generation identifies policies from high-level requirement specifications, minimizing manual errors. The typical process based on natural language processing (NLP) involves three stages: document preprocessing, NLACP identification, and extraction of policy components [8], as shown in Fig. 1. Subsequently, the extracted policies are converted into standard formats such as XACML [4] and PERMIS [7].

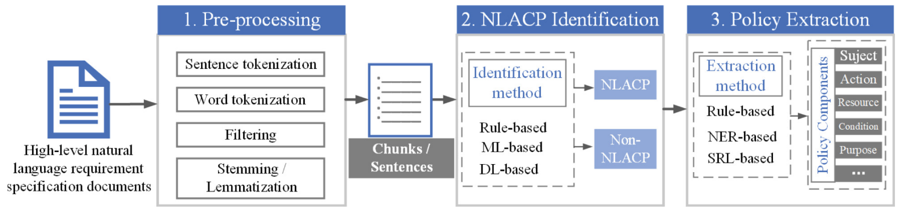


Fig. 1. Pipeline of automated policy generation approaches.

Datasets and Preprocessing. The widely used dataset for ACP extraction, introduced by Slankas et al. [20], contains 2,477 real-world system sentences from five sources: iTrust (ACRE and Text2Policy) [12], IBM course registration system, CyberChair, and the Collected ACP data [22]. However, by 2025, this dataset has become inaccessible. RAGent [9] offers policy component data, but it falls short in enabling fine-grained policy generation. Preprocessing techniques such as tokenization [1] and stemming [19] are crucial for preparing clean textual data.

NLACP Identification. Distinguishing NLACPs from non-NLACPs involves classification techniques. Rule-based methods, matching syntactic and semantic patterns against pre-defined patterns, are commonly used. Xiao et al. [22] defined four common grammatical patterns to identify NLACPs. However, Slankas et al. [20] found that only 34.4% of the sentences matched these patterns. To overcome the limitations of pre-defined patterns, ML and DL methods are employed to learn common patterns. The k-Nearest Neighbors (K-NN) algorithm identifies the closest k samples to a given sentence, but selecting the optimal k value remains challenging [13]. Recently, Transformer has been applied to identify NLACPs. Heaps et al. [6] used BERT to classify user stories as NLACPs, non-NLACPs, or ambiguous. RAGent [9] and Xia et al. [21] also employ Transformer to realize text identification, but their effectiveness also depends heavily on domain-specific fine-tuning data. In practice, such data is scarce and costly to annotate manually [9]. Moreover, fine-tuning large models (e.g., LLaMA3-8B) requires significant computational resources, making them impractical for organizations lacking cloud-scale infrastructure.

Policy Component Extraction. Current ACP extraction methods include rule-based and DL-based algorithms. Some prior studies have applied dependency parsing to capture policy components and their relationships. Slankas et al. [20] utilized bootstrapping to identify similar dependency patterns, while Alohaly et al. [2] manually identified the five most common relations that encoded subject-attribute and object-attribute associations. However, due to the ambiguity and complexity inherent in NLACPs, accurately parsing components remains challenging. NER and SRL are two widely adopted DL-based approaches. Heaps et al. [6] fine-tuned BERT for entity extraction in access control policies, focusing on NER tasks. Shi et al. [18] applied BERT-based SRL to identify users and resources associated with specific actions. Additionally, RAGent [9] employed an Information Extraction(IE) method, using the LLaMa3 8B-Instruct model to extract key components. However, the large number of parameters in most LLM-based models makes them computationally expensive.

Existing research on ACP extraction is limited by insufficient annotated data, the complexity of fine-grained policy extraction, and high computational costs. To address these issues, we introduce a semi-automatic annotation method to improve annotated efficiency and use it to build the fine-grained dataset. We further propose a lightweight framework that extracts policy components directly from specifications without relying on deep semantic understanding or large models.

3 Methodology

Figure 2 illustrates the workflow of LitroACP, which automates ACP analysis through three modules: (1) ACPUIE for annotating raw policy text, (2) DisAdver leverages a DistilBERT model enhanced by PGD training to perform decision identification, and (3) GLiACP for extracting policy components. A key feature is that ACPUIE and GLiACP integrate general and domain knowledge to improve the accuracy of ACP extraction. We also introduce five annotated datasets to validate the effectiveness of LitroACP.

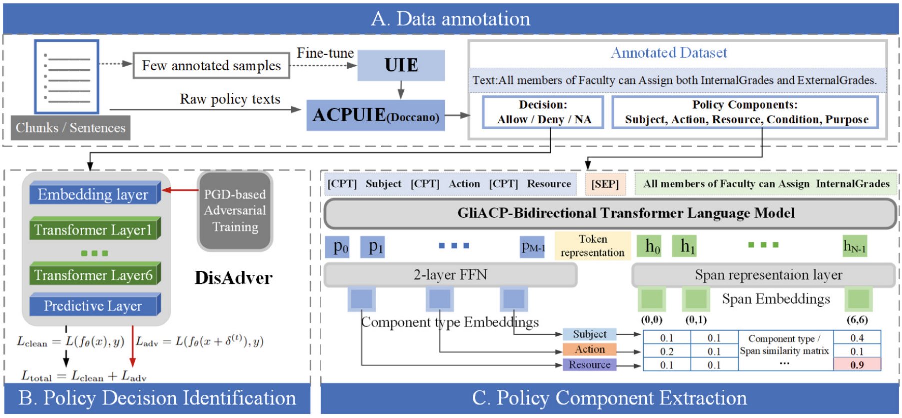


Fig. 2. Access control policy extraction pipeline of LitroACP.

3.1 Data Preprocessing and Annotation

Current ACP datasets rely heavily on the annotations of Slankas et al. [20], but their accessibility issues highlight gaps in public datasets. Recently, LLM-based NER methods have shown great effectiveness in IE by leveraging both pre-trained knowledge and strong generalization capabilities [10]. Referencing RAGent's data, we developed NER-based ACP annotation datasets from original requirements. By employing semi-automatic annotation, we reduce manual errors and ensure adherence to the PoLP, resulting in accurate and consistent policy components that facilitate fine-grained policy extraction.

Preliminaries. We assume that the high-level requirement specifications are written by security experts, with the semi-structured text in the document being semantically clear. Each sentence contains only one policy decision and allow the presence of multiple instances of the same type of *Subject*, *Action* and *Resource*. For example, in the NLACP [*"A UAP or patient representative can input and read the blood pressure, glucose levels or pedometer readings for a patient."*], the policy extraction result is: {*Decision: allow; Subject: UAP, patient representative; Action: input, read; Resource: blood pressure, glucose levels, pedometer readings; Purpose: none; Condition: none*}.

In data preprocessing, we first apply coreference resolution to replace pronouns and ambiguous expressions [13], then segment the text into sentences. For cases that do not meet our assumptions, we first use conjunctions (e.g., *but*, *except for*) to split the sentences, then apply character matching or substitution to complete the sentences. In the data annotation, we first annotate the policy decision type for each sentence as *Not Applicable (NA)*, *Allow*, or *Deny*. Next, we annotate the access control policy components (*Subject*, *Action*, *Resource*, *Condition*, and *Purpose*) using NER, offering a more fine-grained approach than RAGent's IE-based method. Detailed procedures are described below:

Procedure 1: Our annotation adheres to the PoLP in access control, while the data provided by RAGent violates this principle. For example, in the sentence, [*"After the program committee meeting has taken place, and papers have been selected, the (anonymous) comments can be automatically sent to the authors."*], the RAGent's annotation result is: {*decision: allow; subject: none; action: send; resource: comment; purpose: none; condition: program committee meeting has taken place | decision: allow; subject: none; action: send; resource: comment; purpose: none; condition: papers have been selected*}. However, in reality, both conditions must be met simultaneously for the policy to be executed. Therefore, we annotate the sentence as a single policy. Moreover, to enhance the accuracy, our access control policy components are directly annotated from the sentences, without relying on inferred or composite components derived from sentence understanding.

Procedure 2: We correct annotation errors, such as in the NLACP [*"Not all reviewers like to fill in review forms online."*]. In RAGent, the result is: {*decision: deny; subject: reviewer; action: fill; resource: review form; condition: none; purpose: none*}, but the NLACP does not express a denial; it actually implies that reviewers are allowed to choose to fill in review forms. Besides, the resource should likely refer to the "*review forms*" rather than just "*review*".

Procedure 3: We ensure finer granularity and a more uniform labeling format. For example, RAGent inconsistently annotated some parts as *Condition* while leaving others unlabeled, which we rectify by standardizing the annotation. In the case [*"In both cases, the system presents the name of the patient"*], RAGent labels "*name*" as the *Resource*, but it should indeed be considered "*name of the patient*" as the *Resource*. Besides, RAGent's annotation includes inferred and cross-text components, limiting the dataset's applicability to resource-intensive generative models. Our approach directly annotates components from texts, supporting efficient ACP extraction via NER tasks.

Annotation Assistance Model. To improve annotation efficiency and reduce errors, we utilize semi-automated methods to perform annotation. By fine-tuning the UIE [11] model with minimal annotated data, we accelerate the annotation process. UIE encodes extraction structures using a Structured Extraction Language (SEL) and adaptively generates target extractions via a Structural Schema Instructor (SSI). Leveraging a large-scale pre-trained text-to-structure model, UIE effectively captures general IE capabilities. We formulate policy component extraction task as the entity identification, using spotting transformation operations to identify spans associated with specified semantic types [5]. As shown in Fig. 3, the ACPUIE model takes the SSI s and a policy sentence x as an input, generating a SEL output y that contains extracted policy components. For this task, we define the SSI schema as: *[spot] Subject [spot] Action [spot] Resource [spot] Condition [spot] Purpose*. The objective is outlined as follows:

$$y = ACPUIE(s \oplus x) = ACPUIE[[spot], \ldots, [spot], \ldots, [text], x_1, \ldots, x_{|x|}] \quad (1)$$

We formulate the policy sentence-to-SEL generation process using an encoder-decoder architecture. Given the raw policy sentence x and the policy SSI s, the Transformer encoder computes the hidden representation H for each token:

$$H = TransEncoder(s_1, s_2, \ldots, s_{|s|}, x_1, x_2, \ldots, x_{|x|}) \tag{2}$$

At decoding step i, ACPUIE generates the i-th token y_i in the SEL sequence with the decoder state h_i computed as follows:

$$y_i, h_d^i = TransDecoder([H; h_1, h_2, \ldots, h_d^{i-1}]) \tag{3}$$

Based on the UIE pre-trained knowledge and our annotated corpus $D_{acp} = (s, x, y)$, we fine-tune the original UIE model with a teacher-forcing cross-entropy loss.

$$l_{ACP} = \sum_{(s,x,y \in D_{acp})} -\log p(y|x, s; \theta_e, \theta_d) \tag{4}$$

where θ_e and θ_d are the parameter of encoder and decoder, respectively. After obtaining the ACPUIE model, we deploy it within the annotation framework, Doccano[3]. ACPUIE enables the automated annotation of policy components, requiring minimal human intervention to achieve more accurate annotations than those produced by purely manual annotation.

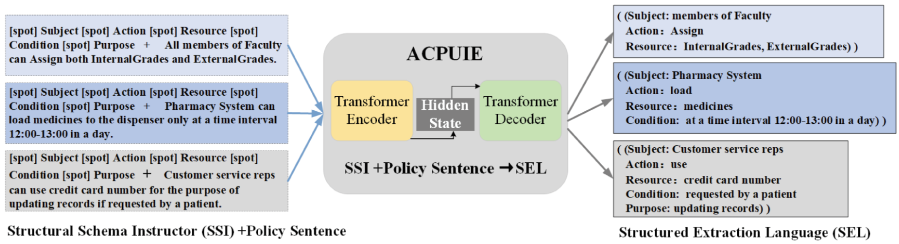


Fig. 3. The overall framework of ACPUIE.

3.2 Policy Decision Identification

Many studies formulate NLACP identification as a binary classification task, classifying each sentence as either NLACP or non-NLACP. Transformer models like BERT excel by using self-attention to capture semantic dependencies. Some prior research has successfully applied BERT to NLACP identification. However, BERT-based models are costly and inefficient due to high resource demands and

[3] https://doccano.github.io/doccano/.

slow inference. In contrast, small models are lightweight, efficient, and easier to deploy, making them ideal for practical automation tasks. Thus, we selected DistilBERT [17] as the model architecture, which reduces BERT's size by 40%, retains 97% of its language understanding capacity, and improves speed by 60%. Given the scarcity of real-world datasets, we propose a method called Disadver, which combines DistilBERT with adversarial training to enhance model robustness and generalization. Compared to traditional data augmentation, adversarial training strengthens the model's resilience by generating targeted perturbations near decision boundaries, providing robustness against subtle semantic variations while preserving the underlying semantic structure [25].

Algorithm 1. PGD-based Adversarial Training for DistilBERT

Require: Model f_θ, training data $D = \{(x_i, y_i)\}$, attack steps K, max perturbation ϵ, step size α
Ensure: Optimized model parameters θ
Initialize: PGD parameters K, ϵ, α
for each epoch **do**
    for each batch (x, y) in D **do**
        Step 1: Clean Loss
        Compute $L_{\text{clean}} = L(f_\theta(x), y)$
        Back-propagate to get $\nabla_\theta L_{\text{clean}}$; backup gradients
        Step 2: PGD Attack on Embeddings
        Initialize $\delta = 0$, backup embeddings
        for $t = 1$ to K **do**
            Update $\delta^{(t)} = \Pi_\epsilon \left(\delta^{(t-1)} + \alpha \frac{\nabla_\delta L(f_\theta(x+\delta^{(t-1)}), y)}{\|\nabla_\delta L(f_\theta(x+\delta^{(t-1)}), y)\|} \right)$
            Compute adversarial loss $L_{\text{adv}} = L(f_\theta(x + \delta^{(t)}), y)$
            Back-propagate L_{adv}
        end for
        Restore original embeddings and gradients
        Step 3: Model Update
        Compute total loss $L_{\text{total}} = L_{\text{clean}} + L_{\text{adv}}$
        Update model parameters $\theta \leftarrow \theta - \eta \nabla_\theta L_{\text{total}}$
    end for
end for

We first developed a binary classification Disadver to assess its effectiveness in identifying NLACPs. Additionally, we reframed the identification process as a three-class classification problem for policy decision: non-NLACPs are categorized as *NA*, while NLACPs are further classified as *Allow* or *Deny* based on their semantic content. PGD-based adversarial training treats adversarial robustness as a saddle point problem, involving an inner maximization problem and an outer minimization problem. The mathematical expression is formulated as follows:

$$\min_\theta \mathbb{E}_{(x,y)\sim D} \left[L(f_\theta(x), y) + \max_{\|\delta\| \le \epsilon} L(f_\theta(x + \delta), y) \right] \tag{5}$$

In the training process, PGD creates adversarial examples by iteratively adjusting perturbations based on loss gradients, and the details are described in Algorithm 1. Lines 10 and 11 update the adversarial perturbation using:

$$\delta^{(t)} = \Pi_\epsilon \left(\delta^{(t-1)} + \alpha \frac{\nabla_\delta L(f_\theta(x + \delta^{(t-1)}), y)}{|\nabla_\delta L(f_\theta(x + \delta^{(t-1)}), y)|} \right) \tag{6}$$

where δL is the gradient direction for maximum loss increase, α is the step size, and projection Π_ϵ ensures δ stays within the defined bound.

Unlike FGSM's single-step attack, PGD iteratively refines adversarial perturbations. It enhances robustness by finding worst-case perturbations for better adversarial defense, improving generalization across models, and maintaining fine-grained control over perturbations. By integrating PGD-based training into DistilBERT, Disadver achieves higher robustness in policy decision identification. It also enables precise control over the magnitude of perturbations, leading to improved model performance and enhanced resilience to a variety of policy sentences.

3.3 Policy Component Extraction

After identifying the policy decision for each sentence, we proceed to extract the ACP components of NLACPs, focusing specifically on those whose policy decisions are categorized as *Allow* or *Deny*. The extraction components include *Subject*, *Action*, *Resource*, *Purpose*, and *Condition*. The recent RAGent [9] model, implemented with LLaMA-8B, has achieved high accuracy in component extraction. However, LLaMA-8B's 8-billion parameters typically require 8 GB to 16 GB of GPU memory for inference, making it unsuitable for dynamic or distributed environments requiring real-time policy updates, such as IoT-enabled healthcare systems and smart factories with legacy PLCs that need on-device policy extraction. In such scenarios, lightweight models (e.g., <100 MB memory footprint) are critical for low-latency processing. Thus, we fine-tune GLiNER-Small [24] for parallel extraction of policy components. GLiNER employs a bidirectional language model structure with three main components: a pre-trained encoder, span representation, and an entity representation module. GLiNER has been trained on the Pile-NER[4] dataset and can extract a diverse range of entity types. We leverage GLiNER's pre-existing knowledge and fine-tune it on our ACP dataset to obtain the GLiACP model, as shown in Fig. 2.

For training, we input text data related to original access control components, using *[CPT]* to separate the access control components and *[SEP]* to distinguish them from the primary text input. After token representation, access control components are encoded into $p = \{p_i\}_0^{M-1} \in \mathbb{R}^{M \times D}$, while the input text is encoded into $h = \{h_i\}_0^{N-1} \in \mathbb{R}^{N \times D}$ through specific token representations. In the component and span representation, the goal is to encode component types and span embeddings into a unified latent space. By using a

[4] https://huggingface.co/datasets/Universal-NER/Pile-NER-type.

two-layer feedforward network, the initial representation p are transform into $q = \{q_i\}_0^{M-1} \in \mathbb{R}^{M \times D}$. The representation of a span in the input text can be computed by:

$$S_{ij} = FFN(h_i \otimes h_i) \tag{7}$$

In the component type and span matching process, the matching score is computed using the sigmoid activation function as follows:

$$\phi(i,j,t) = \sigma(S_T^{ij} q_t) \in \mathbb{R} \tag{8}$$

In the fine-tuning process, the objective is to maximize the matching score for span-type pairs (positive pairs) and minimize the matching score for incorrect pairs (negative pairs). The loss function can be defined as:

$$\mathcal{L}_{Finetune} = -\sum_{s \in \mathcal{S} \times \mathcal{T}} f_{s \in \mathcal{P}} \log \phi(s) + f_{s \in \mathcal{N}} \log(1 - \phi(s)) \tag{9}$$

where $\mathcal{S}, \mathcal{T}, \mathcal{P}, \mathcal{N}$ are comprising spans, component types, positive pairs and negative pairs, respectively, and f is an indicator function. After fine-tuning, GLiACP can effectively extract policy components from policy sentences. This approach combines robust feature representation with efficient span-type matching, ensuring high precision in component extraction. The use of a carefully designed loss function helps the model differentiate between correct and incorrect span-type matches, leading to improved performance and generalization.

4 Evaluation

In this section, we evaluate LitroACP through a series of experiments conducted on five real-world annotated datasets. First, we use ACPUIE to assist with the data annotation process. Next, we assess the performance of DisAdver in policy decision identification. Finally, we evaluate the reliability of GLiACP in ACP component extraction.

4.1 Baselines and Dataset

Baselines. We compare LitroACP with three existing ACP generation methods. The first method, proposed by Narouei et al. [13], utilizes an attribute-based ACP generation framework, where a Recurrent Neural Network (RNN) model is employed for NLACP identification, and SRL is used to extract policy attributes. The second method, developed by Xia et al. [21], leverages BERT and SRL for both ACP sentence identification and rule extraction. These two methods are limited to extracting only three policy components: *Subject*, *Action* and *Resource*. The most recent approach, RAGent [9], uses BERT and LLaMA3-8B Instruct to extract five policy components: *Subject*, *Action*, *Resource*, *Purpose*, and *Condition*. However, despite being among the latest research in the access control policy domain, these models do not fully provide publicly available datasets or source code. Therefore, we attempt to follow the recommendations provided in the original papers.

Dataset. During the data annotation process, we initially manually annotated 10 random NLACP sentences from each dataset, resulting in a 50-shot samples for fine-tuning the UIE. After training, ACPUIE achieved an F1-score of 46.88%. We implemented it within the Doccano framework, a widely used annotated tool for NER. Although only 50 samples were used for fine-tuning, we observed that the model significantly reduced the manual effort required for annotation and effectively labeled the components, even those which are difficult for humans to identify.

Table 1. Access control policy dataset.

Dataset	Allow (NLACP1)	Deny (NLACP2)	NA (non-NLACP)	NLACP
Collected ACP(CACP)	99	31	28	130
CyberChair(CC)	119	1	106	120
IBM(IBM)	123	6	86	129
iTrust T2P (T2P)	354	13	48	367
iTrust ACRE (ACRE)	487	12	151	499
Overall	1,182	63	419	1,245

In the policy decision annotation process, each sentence is first classified as either *NA (0)*, *Allow (1)*, or *Deny (2)*, where *Allow* and *Deny* are considered as NLACPs, and *NA* represents non-NLACPs. For sentences containing both *Allow* and *Deny* policies, we split them by conjunctions during the pre-processing. In the component annotation process, each sentence is labeled with its five components, *Subject*, *Action*, *Resource*, *Purpose*, and *Condition*, if they are present. The final annotated dataset comprises five fine-grained datasets with a total of 1,664 sentences, as detailed in Table 1. These datasets were collected from real-world systems across diverse domains, including business, education, and healthcare, reflecting the complex access control requirements in these industries.

Details. We use the F1-score as the evaluation metric, which is the harmonic mean of precision and recall for a single class, reflecting performance on one category. We also use the Macro F1-score because it is more robust to imbalanced data. It is the average of F1-scores across all classes, treating each class equally. The experiments are implemented using PyTorch 1.10.2 and executed on a single NVIDIA GeForce RTX 3070 GPU.

4.2 Policy Decision Identification

We first evaluate the performance of DisAdver in traditional binary classification, identifying NLACPs and non-NLACPs. The results are compared with three baseline methods for policy identification. Next, we reformulate the task into

Table 2. Performance Comparison on Binary NLACP Identification.

Framework	T2P	ACRE	IBM	CACP	CC	Average	Overall
Narouei et al. [13]	75.45	83.78	85.16	73.56	76.12	78.81	76.45
Xia et al. [21]	97.93	84.62	79.54	90.43	76.21	85.77	88.17
RAGent [9]	98.61	**94.34**	91.67	92.31	78.73	91.13	92.28
DistilBERT	97.93	93.90	88.00	89.29	79.17	89.69	92.44
DisAdver	**98.61**	94.29	**95.83**	**92.31**	**87.80**	**93.77**	**94.56**

a three-class policy decision problem based on our annotated datasets. This approach enables a more fine-grained classification model. To further assess the PGD adversarial training performance, we conduct the ablation experiments.

NLACP Identification. As presented in Table 2, *Average* is Macro-F1 scores computed separately for each dataset, then averaged across all datasets, *Overall* is the Micro-F1 score calculated after merging all datasets into a single dataset. DisAdver achieves an average F1-score of 93.77%, outperforming nearly all existing methods. When compared to RAGent, we observe that the BERT model (*RAGent*) outperforms the original DistilBERT in NLACP identification due to its larger pre-trained knowledge base and more complex architecture. However, when applying the PGD adversarial training method, DisAdver demonstrates superior robustness and effectiveness compared to BERT (*RAGent*).

Policy Decision Identification. After demonstrating the effectiveness of DisAdver in NLACP identification, we extend the task to fine-grained policy decision identification. In this formulation, each sentence is classified into one of three categories: *NA*, *Allow*, or *Deny*. To systematically evaluate model robustness against data imbalance and quantify the impact of training data volume on architectural effectiveness, we implement stratified sampling across policy decision categories. This approach specifically addresses the acute scarcity of *Deny* policy instances (representing only 3.78% of total samples in Table 1), which risks introducing classification bias in security-critical scenarios. We design two experimental groups based on dataset splitting, where the training data accounts for 50% (CACP-5, T2P-5 and OVERALL-5) and 80%(CACP-8, T2P-8 and OVERALL-8) of the total dataset, respectively.

As shown in Fig. 4, our results demonstrate that DisAdver outperforms DistilBERT, and for a detailed comparison of each decision type, the results are listed in Table 3. Notably, for sentences with *Allow* decisions, the overall F1-score reaches up to 93% after adversarial training. Although the datasets for *NA* and *Deny* policies are relatively small, DisAdver consistently outperforms the original DistilBERT across nearly all decision types.

4.3 Policy Component Extraction

We evaluate the performance of GLiACP with the goal of developing a lightweight yet powerful policy component extraction model. To achieve this, we begin by comparing the sub-models and parameters of existing LLM-based methods. As shown in Table 4, the total parameter size of LitroACP is approximately half that of Xia et al. [21] and about 1/80 of Ragent [9].

During the experiment process, we encountered several challenges when replicating RAGent's [9] results: 1) It did not provide the pre-training dataset mentioned, and the pre-training data that matches its settings is not available in the Huggingface database; 2) When directly using its trained checkpoints for reference, even without utilizing its Retrieval and Update functions, it took at least

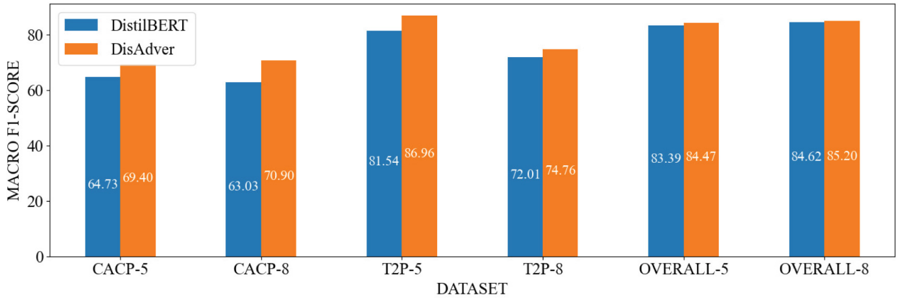


Fig. 4. The Macro F1-score comparison of DisAdver in policy decision identification, with "X-8" for 80% and "X-5" for 50% of the total data used for training.

Table 3. Comparison of DistilBERT and Disadver models for decision identification.

Data	NA		Allow		Deny	
	DistilBERT	Disadver	DistilBERT	Disadver	DistilBERT	Disadver
CACP-5	48.00	**50.00**	81.9	**84.11**	64.29	**74.07**
CACP-8	44.44	**66.67**	87.50	**88.89**	57.14	57.14
T2P-5	73.17	**75.00**	96.46	**97.00**	75.00	**88.89**
T2P-8	55.60	**63.16**	93.79	**94.44**	66.67	66.67
OVERALL-5	78.41	**79.26**	91.07	**91.99**	80.70	**82.14**
OVERALL-8	79.50	**82.49**	92.88	**93.10**	**81.48**	80.00

Table 4. Comparison of model architectures and parameters.

Method	Xia et al. [21]	RAGent [9]	LitroACP
Identification	BERT(110M)	BERT(110M)	DistilBERT(66M)
Extraction	BERT(110M)	LLaMa3(8B)	GLiNER-Small(50M)
Total Parameters	220M	8,110M	**116M**

6 min to obtain a prediction for a single sample on our device. This indicates that it is impractical for real-world, resource-limited environments; 3) When applying RAGent's prompt to ChatGPT-4o, we found that even without any training, ChatGPT's policy extraction results outperformed RAGent. Therefore, since our goal is to design a practical method, we do not consider resource-intensive, chat-based generative LLMs for policy extraction in our comparison. In this section, our direct comparison model is Narouei et al. [13] and Xia et al. [21], with RAGent's experimental results serving as an indirect comparison.

Table 5. Component extraction performance of LitroACP compared with others.

Framework	T2P	ACRE	IBM	CACP	CC	Average	Overall
Narouei et al. [13]	14.78	18.38	19.03	28.43	15.62	19.24	18.35
Xia et al. [21]	48.46	38.12	44.65	48.70	30.34	42.05	41.94
LitroACP(SAR)	76.64	74.89	70.18	79.41	**72.14**	74.65	77.39
LitroACP(SARCP)	**77.12**	**79.16**	**78.16**	**82.12**	71.93	**77.69**	**78.92**

As shown in Table 5, since Narouei et al. [13] and Xia et al. [21] can only extract the *Subject*, *Action*, and *Resource* (SAR), we set up two different configurations. SRACP, on the other hand, refers to extracting *Subject*, *Action*, and *Resource*, along with *Condition* and *Purpose*. The results show that LitroACP achieves an average F1-score of 77.69%, outperforming both Narouei et al. [13] and Xia et al. [21], and is close to the 80.7% F1-score reported by RAGent [9].

5 Discussions and Future Work

Our approach demonstrates good applicability across various scenarios. Compared to existing methods, which rely on resource-intensive models, LitroACP achieves high accuracy with a more efficient architecture. Our semi-automated annotation method, ACPUIE, improves dataset quality, addressing the scarcity of high-quality labeled data. Furthermore, Disadver and GLiACP outperform previous approaches in NLACP decision identification and policy component extraction. In real-world applications, new data constantly emerge, and our framework can adapt to these changes, ensuring its robustness and relevance over time.

However, data scarcity remains a significant challenge. Despite efforts to annotate limited datasets accurately, inherent ambiguities introduce noise, leading to annotation inaccuracies. To address this, we plan to collaborate with multiple enterprises to obtain non-sensitive specification documents and annotate them using our approach, creating a larger and cleaner dataset. In policy extraction, while our method uses NER for component extraction, the semantic relationships identified by the GLiACP module are constrained by the model's prior knowledge and lack interpretability. Future work will focus on integrating relation extraction methods to improve accuracy in component extraction.

6 Conclusion

We introduced LitroACP, a lightweight and efficient framework for fine-grained access control policy extraction. Unlike resource-intensive LLM-based approaches, LitroACP combines NER with language models to achieve high accuracy with lower computational cost. We also propose a semi-automated data annotation method, ACPUIE, which significantly improves annotation efficiency. By applying PGD-based adversarial training on DistilBERT and fine-tuning GLiACP, our approach enhances the efficiency of automated policy generation.

Experimental results show that LitroACP excels in policy decision identification and component extraction. LitroACP can be applied to various enterprise scenarios, including healthcare, education, and finance, leveraging pre-trained model knowledge to support access control policy extraction. To encourage further research, we have made the LitroACP dataset and code publicly available.

Acknowledgments. This work was supported by the Youth Innovation Promotion Association, Chinese Academy of Sciences (No. 2023166).

References

1. Abdelgawad, M., Ray, I., Alqurashi, S., Venkatesha, V., Shirazi, H.: Synthesizing and analyzing attribute-based access control model generated from natural language policy statements. In: Proceedings of the 28th ACM Symposium on Access Control Models and Technologies, pp. 91–98 (2023)
2. Alohaly, M., Takabi, H., Blanco, E.: Automated extraction of attributes from natural language attribute-based access control (abac) policies. Cybersecurity **2**(1), 2 (2019)
3. Ariyanto, A.D.P., Purwitasari, D., Fatichah, C.: A systematic review on semantic role labeling for information extraction in low-resource data. IEEE Access (2024)
4. Brodie, C., Karat, C.M., Karat, J., Feng, J.: Usable security and privacy: a case study of developing privacy management tools. In: Proceedings of the 2005 Symposium on Usable Privacy and Security, pp. 35–43 (2005)
5. Chen, X., Yuille, A.L.: Detecting and reading text in natural scenes. In: Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2004. CVPR 2004, vol. 2, pp. II–II. IEEE (2004)
6. Heaps, J., Krishnan, R., Huang, Y., Niu, J., Sandhu, R.: Access control policy generation from user stories using machine learning. In: Data and Applications Security and Privacy XXXV: 35th Annual IFIP WG 11.3 Conference, DBSec 2021, Calgary, Canada, July 19–20, 2021, Proceedings 35, pp. 171–188. Springer (2021)
7. Inglesant, P., Sasse, M.A., Chadwick, D., Shi, L.L.: Expressions of expertness: the virtuous circle of natural language for access control policy specification. In: Proceedings of the 4th Symposium on Usable Privacy and Security. pp. 77–88 (2008)
8. Jayasundara, S.H., Arachchilage, N.A.G., Russello, G.: Sok: Access control policy generation from high-level natural language requirements. arXiv preprint arXiv:2310.03292 (2023)

9. Jayasundara, S.H., Arachchilage, N.A.G., Russello, G.: Ragent: Retrieval-based access control policy generation. arXiv preprint arXiv:2409.07489 (2024)
10. Keraghel, I., Morbieu, S., Nadif, M.: A survey on recent advances in named entity recognition. arXiv preprint arXiv:2401.10825 (2024)
11. Lu, Y., et al.: Unified structure generation for universal information extraction. arXiv preprint arXiv:2203.12277 (2022)
12. Meneely, A., Smith, B., Williams, L.: Appendix b: itrust electronic health care system case study. Softw. Syst. Traceability **425** (2012)
13. Narouei, M., Takabi, H.: Towards an automatic top-down role engineering approach using natural language processing techniques. In: Proceedings of the 20th ACM Symposium on Access Control Models and Technologies, pp. 157–160 (2015)
14. Narouei, M., Takabi, H., Nielsen, R.: Automatic extraction of access control policies from natural language documents. IEEE Trans. Dependable Secure Comput. **17**(3), 506–517 (2018)
15. Nobi, M.N., Gupta, M., Praharaj, L., Abdelsalam, M., Krishnan, R., Sandhu, R.: Machine learning in access control: a taxonomy and survey. arXiv preprint arXiv:2207.01739 (2022)
16. Qu, X., Gu, Y., Xia, Q., Li, Z., Wang, Z., Huai, B.: A survey on Arabic named entity recognition: Past, recent advances, and future trends. IEEE Transactions on Knowledge and Data Engineering (2023)
17. Sanh, V.: Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. arXiv preprint arXiv:1910.01108 (2019)
18. Shi, P., Lin, J.: Simple bert models for relation extraction and semantic role labeling. arXiv preprint arXiv:1904.05255 (2019)
19. Singru, R., Bhandari, P., Patel, K., Mane, P., Gulhane, C.: Efficient electronic document access control management using natural language processing. In: 2020 Fourth International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud)(I-SMAC), pp. 714–719. IEEE (2020)
20. Slankas, J., Xiao, X., Williams, L., Xie, T.: Relation extraction for inferring access control rules from natural language artifacts. In: Proceedings of the 30th Annual Computer Security Applications Conference, pp. 366–375 (2014)
21. Xia, Y., Zhai, S., Wang, Q., Hou, H., Wu, Z., Shen, Q.: Automated extraction of abac policies from natural-language documents in healthcare systems. In: 2022 IEEE International Conference on Bioinformatics and Biomedicine (BIBM), pp. 1289–1296. IEEE (2022)
22. Xiao, X., Paradkar, A., Thummalapenta, S., Xie, T.: Automated extraction of security policies from natural-language software documents. In: Proceedings of the ACM SIGSOFT 20th International Symposium on the Foundations of Software Engineering, pp. 1–11 (2012)
23. Yang, L., Chen, X., Luo, Y., Lan, X., Chen, L.: Purext: automated extraction of the purpose-aware rule from the natural language privacy policy in Iot. Secur. Commun. Netw. **2021**(1), 5552501 (2021)
24. Zaratiana, U., Tomeh, N., Holat, P., Charnois, T.: Gliner: Generalist model for named entity recognition using bidirectional transformer. arXiv preprint arXiv:2311.08526 (2023)
25. Zhu, C., Cheng, Y., Gan, Z., Sun, S., Goldstein, T., Liu, J.: Freelb: Enhanced adversarial training for natural language understanding. arXiv preprint arXiv:1909.11764 (2019)

Sustainability

Energy Profiling of Data-Sharing Pipelines: Modeling, Estimation, and Reuse Strategies

Sepideh Masoudi[1](✉), Sebastian Werner[1], Pierluigi Plebani[2], and Stefan Tai[1]

[1] Information Systems Engineering, Technische Universität Berlin, Berlin, Germany
{smi,sw,st}@ise.tu-berlin.de
[2] Politecnico di Milano, Milan, Italy
pierluigi.plebani@polimi.it
https://www.tu.berlin/en/ise

Abstract. Data-sharing pipelines involve a series of stages that apply policy-based data transformations to enable secure and effective data exchange among organizations. Although numerous tools and platforms exist to manage governance and enforcement in these pipelines, energy efficiency in data exchange has received limited attention. This paper introduces a novel method to model and estimate the energy consumption of different execution configurations in data-sharing pipelines. Additionally, this method identifies reuse potential in shared stages across pipelines that hold the key to reducing energy in large data-sharing federations. We validate this method through simulation experiments, revealing promising potential for cross-organizational pipeline optimization and laying a foundation for energy-conscious execution strategies.

Keywords: Energy Profiling · Data-Sharing Pipeline · Federated Data Product · Pipeline Configuration

1 Introduction

Companies have long recognized data as a critical business asset, and sharing these assets between organizations can unlock revenue and offer vast opportunities [21]. However, facilitating data exchange across organizations requires both organizational and technical support. Data-sharing pipelines represent an established method for facilitating data exchange among independent organizations [20]. With such pipelines, source data residing within one organization is collected and transformed through sequential processing stages for utilization by a consuming organization. These transformations are executed, leveraging pipelines as data-sharing method, according to the requirements and policies mutually established by the collaborating organizations, which may encompass privacy, analytical, or other relevant transformations.

In practice, using cross-organizational data-sharing pipelines is often hindered by technical and organizational barriers [7], along with the lack of well-defined agreements between parties, unclear costs and resource requirements,

J. Krogstie et al. (Eds.): CAiSE 2025, LNCS 15701, pp. 93–108, 2025.
https://doi.org/10.1007/978-3-031-94569-4_6

complexity in configuration and implementation, and the need for assurances that the shared data will not be misused. With the continued desire for open data marketplaces, several emerging governance and exchange platforms started to emerge that aim to enable organizations to manage shared data, with governance and enforcement strategies [3]. Specifically, data spaces [11] and federated data meshes [5] are gaining recognition as environments that offer technical and occasional solutions. Moreover, the emergence of guiding principles for data providers aims to enhance the reusability of their data and the ability of machines to automatically find and use the data [28], highlighting the complexities involved in the configuration of data-sharing.

A growing area of concern within such one-to-many (1-to-n) data-sharing pipelines is the need to assess the costs associated with data sharing [5]. Particularly concerning energy expenditures, the aspiration for an energy-efficient data-sharing pipeline is paramount, as highlighted in recent studies [22]. To achieve this objective, it is essential to develop an understanding of energy consumption at each stage of the pipeline, considering factors such as computational expense, data volume, and other pertinent criteria. Moreover, by modeling the energy consumption of stages in a data-sharing pipeline, it becomes possible to explore opportunities to reduce effort and energy consumption, as pipelines and their results can be reused when sharing the same or similar data repeatedly. However, the possibilities and complexities in configuring these data-sharing pipelines raise the need for tools to support conscious design choices to maximize reusability and energy savings.

Hence, we pose the following research question: *How can we model energy consumption to promote energy estimation and reuse strategies in cross-organizational data-sharing pipelines?*

To address this question, in this paper, we introduce a novel energy profiling model and method. We propose a comprehensive set of metrics designed to characterize energy utilization at individual stages of a 1-to-n pipeline. Each specific one-to-one (1-to-1) pipeline can be conceptualized as a configuration of steps derived from the 1-to-n pipeline model, whereby an increased number of reused steps can facilitate the evaluation and enhancement of individual and overall energy efficiency. Our model and method provide a framework for informed assessments of energy requirements within data-sharing pipelines, enabling the identification of common stages that can be shared and reused across different configurations, as well as the estimation and customization of execution configurations, which can serve to optimize energy consumption.

The remainder of this paper is organized as follows: In the next section, we review related work on energy consumption management, focusing on data-sharing pipelines and data provider services. In Sect. 3, we introduce the proposed energy consumption model and profiling method. In Sect. 4, we present a case study to evaluate the application of the model. In Sect. 5, we discuss the limitations of our method. Finally, in Sect. 6, we conclude the paper with a discussion of future work.

2 Related Works

The challenges in data sharing, such as reusability, discoverablity and programmatic actionability are among the key concerns for science and industry. For example, the FAIR principles [28] are one of the earliest proposals for enhancing the reusability of data holdings, with an emphasis on programmatic actionability. These principles apply not only to data in the conventional sense but also to the algorithms, tools, and workflows that led to that data [28]. Naturally, these principles have also inspired emerging infrastructure and platform support. Grossman et al., for example, introduced the term 'data commons' to describe a cloud-based data platform with a governance structure that allows a community to manage, analyze, and share its data [9]. Moreover, Hofman [13] discussed the necessity of a federated infrastructure as the solution required to construct data pipelines, allowing small and medium-sized enterprises to collaborate and share their data. Consequently, the next challange is to increase the interoperability among different data platforms through a set of platform services and protocols for data sharing. Here, Hofman et al. [13] already hinted at the use of data sharing pipelines as one approach to archive this interoperability, however, not yet with en emphasis on energy efficiency and reuse.

Data pipelines and their efficiency improvements are not exclusive for data sharing. For example, Chen et al. present the real-time data processing pipeline used at Facebook. They identify five important design decisions that affect ease of use, performance, fault tolerance, scalability, and correctness [2]. Additionally, Goodhope et al. discuss the design and engineering problems related to leveraging a real-time publish-subscribe system for building data pipelines [8]. Raman et al. explore big data pipelines as a means to break down complex analyses of large datasets into a sequence of simpler tasks. Each task features independently tuned components to enhance performance [23]. In these studies, the authors attempt to establish pipelines as a means to model and implement data transformation while clarifying the engineering problems and challenges in designing pipelines.

Regarding the efficiency challenges of the data management systems that pipelines utilize, Kunjir et al. present a power model based on the evaluation of different pipeline plans for processing queries [15]. Roukh [24] suggested using polynomial regression techniques for building an energy model based on pipelines. The authors in [17,27] investigated specific scenarios and benchmark studies to propose a non-linear relationship between energy efficiency and performance in distributed DBMS in distributed environment for optimizing the trade-offs between energy consumption and performance. Kurpicz et al. provided a profiling and estimation model to attribute the overall energy costs per virtual machine (VM) in heterogeneous environments [16]. This cost model considered both the dynamic energy consumption of VMs and the proportional static cost of using cloud infrastructure. In [10], the authors provided an energy consumption model for running queries and sub-queries in distributed database systems and used that model for energy consumption estimation. Most of the existing works also concentrate on predicting and optimizing energy consumption at the query execution level in data management systems and do not provide a comprehensive insight into energy consumption at different levels in data-sharing pipelines, such as the platform level and application level, for running data transformation pipelines.

Regarding energy profiling, Tomasoni et al. investigate the energy and network performance of different data collection frameworks (DCFs) in a mobile crowd-sensing scenario. They propose a methodology to profile energy consumption using an Android application, which can be used to make harvesting data from the crowd more energy-efficient [26]. In [29], the authors suggest using profiling in commercial buildings to identify energy consumption patterns and leverage those patterns along with energy predictions for anomaly detection. Marinakis et al. propose a methodological approach for adopting big data platforms with smart energy services, gathering energy-related information from multiple sources [18].

Many prior studies do not consider the importance of pipeline execution and configuration in the energy consumption of the data-sharing process. Additionally, they do not investigate the potential for reusing pipelines for energy efficiency due to their multi-stage nature among different parties.

Hence, our energy profiling and estimation approach for a federated data-sharing pipeline can be seen as a more holistic method for reducing and optimizing pipelines at an organizational level, prior to the implementation of query optimization or database optimization. This contributes to the growing body of work aimed at making data-sharing systems more energy-efficient.

3 Comprehensive Energy Profiling

Data-sharing pipelines are generally a set of sequential stages, each called in order during execution. These pipelines manage and usually transform the data between owners and consumers into a shareable and consumable form, often enacting a set of predetermined qualities and policies. Each Data consumer can be unique regarding its requirements, such as intervals of querying data from the owners, the level of access to the data, and data selection logic, and the rights the consumer has to that data. Thus, sharing pipelines combine both necessary steps such as filtering, anonymization, and encryption, with quality-improving steps such as compression, formatting, and serialization into a set of stages that must be executed in the continuum between data owner and consumer to enable the sharing.

3.1 Problem Statement

When independent organizations with heterogeneous data policies and formats agree to share their data using pipelines, they establish a cross-organizational data-sharing pipeline, often as part of general data-sharing agreements. In these one-to-many (1-to-n) data-sharing pipelines, data is collected and offered to specific consumer groups through a data provider. However, this data must still be transformed by a specific one-to-one (1-to-1) pipeline based on the unique requirements of each data-sharing agreement between the data consumer and the data provider. Implementing these data-sharing pipelines should be done by the data provider organization to guarantee the correctness of transformations and transmission of data. Nevertheless, the execution of them could be done either on the data provider side or the data consumer side.

Even though each one-to-one pipeline is specifically designed to meet a consumer's requirements, some stages may be common and reusable across different one-to-one pipelines for consumers of the same type. Hence, when designing the execution plan for a data-sharing pipeline, we need to choose from a wide range of possible configurations with different efficiencies. In cases where each stage of the pipeline can run on different parties' infrastructure across organizations, selecting the optimal execution configuration becomes a non-trivial decision. As shown in Fig. 1, a data consumer (C1) can also function as a data provider for another consumer (C2) of the same type. This occurs when C1 has a higher level of access and fewer restrictions on data filtering and anonymization compared to C2, and both consumers expect similar data quality and frequency. In these cases, a data owner can reduce and potentially save resources, cost and energy. For example, as shown in Fig. 1, we can create a new pipeline by reusing the common stages shared by two pipelines (op_1, op_2, op_3) or a subset of these stages (e.g., op_1, op_2) on the data provider side, while executing the unique stages for each consumer on the consumer side. Additionally, a data consumer can become a new data provider for other consumers. It is worth mentioning that all new pipeline configurations must adhere to governance policies and agreements between the parties involved.

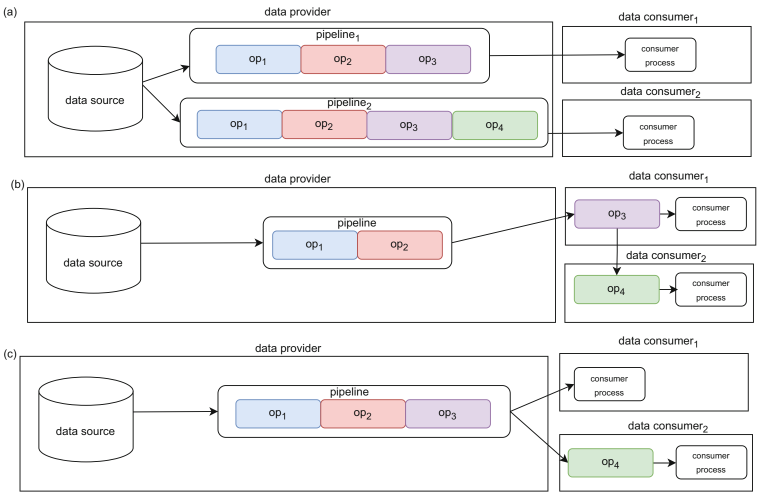


Fig. 1. Different execution configurations of two data-sharing pipelines.

However, while these data-sharing pipelines are loosely coupled and composed of independent stages, any effort to achieve energy efficiency reduction requires a comprehensive understanding of different pipeline execution configurations and the ability to estimate their energy consumption to find and propose these reuse strategies. To address this challenge, we investigate ways to model energy consumption that support accurate energy estimation and encourage reuse strategies in cross-organizational data-sharing pipelines.

3.2 Modeling the Energy of Data-Sharing Pipelines

The variety of infrastructure and hardware available to different parties in a federated data-sharing network can impact the energy consumption pipeline configurations. Therefore, when selecting the execution configuration plan for a pipeline concerning the execution of stages across different parties and reusing common stages, we need a model that contains detailed information about the energy consumption of all executed pipeline stages at the infrastructure and hardware layers. As a first step to address this problem, we define an energy consumption model for data-sharing pipelines.

Moreover, considering the five characteristics of big data -volume, value, variety, velocity, and veracity- we assume that the volume of data impacts the energy consumption of data-sharing pipelines. This relationship is reflected in our energy model through the energy consumption associated with data transmissions at each stage. Furthermore, data-sharing pipelines address the challenge of data variety through data transformation, while the velocity aspect is reflected in their performance.

Table 1. Reference table for the formula notation. The units of measurement are Joules (J) for energy and Gigabytes (GB) for data volume.

Abbreviation	Definition
E_{cpu}	Energy consumption of CPU for running the stage
E_{mem}	Energy consumption of storage for running the stage
E_{dt}	Energy consumption for data transmission in pipeline
E_{dt_op}	Energy consumption for data transmission of the stage
E_{ob}	Energy consumption for monitoring data transmission
E_{op}	Energy consumption of deploying and running the stage (operational energy consumption)
E_p	Energy consumption of pipeline
IS_{op}	Source input size
OS_{op}	Output size at stage
TF_{op}	Size transformation factor at stage

Each one-to-one pipeline can be conceptualized as a configuration of transformation stages derived from a broader set of stages used in one-to-many pipelines for the same data provider. Thus, reusing a greater number of stages can enhance both individual and overall energy efficiency. Energy consumption in a data-sharing pipeline can be divided into three main categories. The energy consumed by deploying and running stages falls under the operational energy consumption category(E_{op}). For operational consumption we always collect the energy consumption of all stages normalized by 1GB of input data, thus, enabling later estimation of energy consumption proportional to expected data consumption. We use TF_{op} to indicate the expected mean data outcome of a stage given IS_{op} for a pipeline that got 1 GB at the source. Operational energy consumption is, thus, the sum of the energy consumption of the CPU (E_{cpu}) and memory (E_{mem}) used by each stage while running. These stages can range from complex preprocessing to simple conversions (e.g., changing the output data format). Data transmission within the multiparty environment also contributes to energy consumption. When different stages are deployed across the infrastructures of different parties, the energy consumption for data transmission may be affected due to changes in the volume of data exchanged between owners and consumers. Therefore, the total energy consumed for data transmission is one of the factors to consider (E_{dt}), which is a sum of the energy consumed for data transmission between every two consequential stages(E_{dt_op}). Monitoring the data-sharing pipeline is essential not only to ensure quality but also to record various metrics, including the energy consumption of data-sharing pipelines. Therefore, the transmission and collection of monitoring data, with a focus on energy consumption, will also incur energy consumption (E_{ob}). As shown in Eq. (1), the total energy consumption of a data-sharing pipeline is the sum of these three categories. Therefore, the total energy consumed by a pipeline is the sum of the energy needed for running all the stages ($\sum_{k=1}^{n} E_{op(k)}$) and the energy that would be consumed for data transmission and monitoring the data-sharing process. The abbreviations used in the formula are defined in Table 1.

$$\begin{cases} E_{op} = E_{cpu} + E_{mem} \\ E_{dt} = \sum_{k=1}^{n} E_{dt\_op(k)} \\ E_p = (\sum_{k=1}^{n} E_{op(k)}) + E_{dt} + E_{ob} \end{cases} \tag{1}$$

Equation Eq. (2) defines the normalized output size OS_{op} at pipeline stage *op*, where IS_{op} is the source input size (e.g., 1 GB) and TF_{op} is the stage-specific scaling factor. This formulation allows us to quantify how each stage transforms the data volume: a) Compression ($TF_{op} < 1$) – The stage reduces the data size (e.g., filtering or aggregation), b) Expansion ($TF_{op} > 1$) – The stage increases the data size (e.g., feature extraction or replication), c) Preservation ($TF_{op} \approx 1$) – The stage maintains the data size (e.g., format conversion).

$$OSop = TFop * ISop \rightarrow \begin{cases} TF_{op} < 1 \text{indicates compression/reduction} \\ TF_{op} > 1 \text{indicates expansion} \\ TF_{op} \approx 1 \text{indicates size preservation} \end{cases} \tag{2}$$

To achieve energy efficiency in data-sharing pipelines, it is necessary to measure energy consumption at the different stages of the data-sharing life cycle. Gaining insight into the energy consumption of various stages provides the means to estimate energy consumption of different configurations and scenarios. As a result of these energy consumption estimates, configurations can be selected with energy awareness, helping to improve energy efficiency.

3.3 Energy Profiling

We refer to the process of energy consumption modeling and measurement as *energy profiling*, and we will explain how energy profiling can lead to improved energy consumption in data-sharing pipelines.

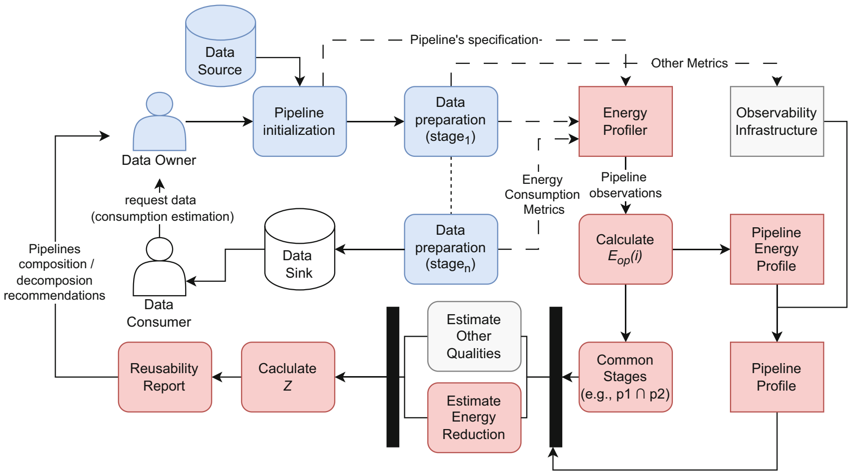


Fig. 2. Data flow diagram of energy profiling in data-share pipelines.

As shown in Fig. 2, after the data provider defines, implements, and starts the pipeline, data preparation begins by transforming the data through multiple stages. In this step, all stages of the pipeline are initially executed on the data provider's side, and only the final data is forwarded to the consumer side. The energy profiler, as a third-party between data provider and data consumer, will collect the pipeline specifications, including the sequence and type of stages that are running in the pipeline and the owner of the infrastructure on which the stages are running. While these stages are running, the energy profiler will collect and store energy consumption metrics for each stage to create an energy profile over time for every stage executed on the data provider's side.

The data provider and data consumer administrators are responsible for deploying tools that can measure energy metrics within their infrastructure. At first, the energy profiler calculates and stores the total energy consumption of the pipeline and each stage separately using the energy model in Sect. 3.2 and energy consumption metrics received from the data provider. Afterwards, the energy profiler will begin searching for pipelines originating from the same data provider that have common stages.

In a data-sharing pipeline, stages that typically overlap between consumers fall into four categories: filtering, anonymization, aggregation, and converting (formatting). Among these well-known types of stages, filtering and anonymization are most likely to be common across different pipelines and reusable due to the symmetry in requirements and policies. In contrast, the other types of stages like aggregation, converting, and any other custom stages are often unique to each data consumer. Therefore, in search for common stages, the energy profiler should only consider stages that belong to the appropriate types. Subsequently, based on the energy consumption of the detected common stages, we can estimate the energy efficiency of reusing those stages in different pipelines. Thus, the final report on the energy estimation of alternative execution configuration plans for pipelines with common stages can help the system administrator optimize decisions regarding the configurations of existing pipelines to run the common stages only once. The data provider administrator can also modify the execution plan so that the unique stages of a consumer are offloaded to be executed on the consumer's side.

However, it is important to consider that reusing stages between pipelines and deciding where to run them can affect the data transmission volume, processing, as well as the cost and energy consumption associated with running stages in different infrastructure (e.g., running them on the consumer's side). Although these parameters are not completely predictable before trying the new configuration, as shown in Eqs. (3) and (4), the new configuration is most likely to be efficient if the energy consumption of the common stages is noticeably higher compared to the energy consumption of the rest of the pipeline. Moreover, by using the new configuration and continuously profiling energy consumption, the administrator of the data-sharing system can assess the efficiency of the new configuration for future use. Additionally, it is worth mentioning that the new execution configuration should align with the initially agreed-upon policies and rules between the data provider and the consumer. Administrators on both the provider and consumer sides can check for any possible policy violations during the negotiation phase concerning the new execution configuration and running certain consumer-specific stages on the consumer's side. In a real-world scenario, cross-organizational data-sharing pipelines using federated environments as infrastructure [13,22] represent a suitable underlying structure for applying the proposed method to increase reusability and to save energy.

$$(p_1 \cap p_2) \rightarrow \begin{cases} p_1 = op_A \rightarrow op_B \rightarrow op_C \\ p_2 = op_A \rightarrow op_B \rightarrow op_D \rightarrow op_F \end{cases} \Rightarrow op_A \rightarrow op_B \tag{3}$$

$$\begin{cases} E_{op(A)} + E_{op(B)} \gg E_{dt\_op(B)} + E_{op(C)} \\ E_{op(A)} + E_{op(B)} \gg E_{dt\_op(B)} + E_{op(D)} + E_{op(F)} \end{cases} \tag{4}$$

In addition to energy, other factors such as performance, response time, and the volume of transmitted data may influence the selection of a new execution configuration. While the primary focus of our method is on energy efficiency, it is essential to incorporate a mechanism that allows the administrator to evaluate the feasibility of reusing a subset of stages in the new execution configuration or altering their execution location. This evaluation should be based on insights provided by other methods or monitoring tools, such as the performance [12], data quality [25], deployment planning of stages [1], the average response time of a stage, and the pipeline's sensitivity to delays. As illustrated in Eq. (5), the administrator can assign an impact weight to each factor, enabling the consideration of other qualities (e.g., performance) alongside energy in the decision-making process. This means that, as shown in Eq. (2), if reusing common stages is expected to have a significantly negative impact on other qualities-such that it outweighs the estimated energy savings-the reuse of those stages will be excluded. In Eq. (5), **W** represents the impact weight assigned to energy and other qualities, while **Q** denotes the estimated value of metrics other than energy. The value of **Q** can be positive (indicating an expected improvement) or negative (indicating an expected degradation). If the result of Eq. (5) is negative, the administrator may decide to exclude the reuse of related common stages due to constraints imposed by other qualities.

$$\begin{cases} W > 0, E_r > 0, Q \in R \\ \\ Z = (W_E * E_r) + \sum_{k=1}^{n} (W_k * Q_k) \\ \\ Z > 0 \rightarrow \text{consider reusing common stages} \\ Z < 0 \rightarrow \text{skip} \end{cases} \tag{5}$$

4 Evaluation

For the evaluation of the proposed method regarding energy model and reuse strategies, we implemented a preliminary simulation of federated data-sharing pipelines. The goal of this running example is to examine the feasibility of the presented energy model and demonstrate the impact of energy profiling on improving energy awareness and estimating energy consumption for different pipeline execution plans. To the best of our knowledge, our approach is the first to propose a systematic way to measure and estimate energy consumption in data-sharing pipelines, making comparisons with other approaches impossible at this time. An energy profile for each pipeline consists of the total energy consumption of each stage of that pipeline on its current infrastructure. Moreover, the energy profile also demonstrates a list of common stages shared between that specific pipeline and other pipelines (including the identifiers of those pipelines), along with the percentage that shows the portion of common stages' energy consumption relative to the overall energy consumption of the pipeline.

The validation is being performed on a synthetic dataset of pipeline stages stored as a `*.csv` file. The stages fall into four categories: filtering, anonymization, aggregation, and converting. Each row in the dataset (each stage) contains information about the energy consumption of that stage at different levels, as defined in this study (shown in Table 3). The amount of energy consumption per unit (e.g., CPU unit, data transmission unit, etc.) is specified as static metrics at the beginning of the experiments. Therefore, the total energy consumption is estimated by multiplying the resource usage amount by the energy consumption of that resource. This simulation is open source and available on GitHub[1] The implementation and execution of this model are not restricted to the use of a specific software. In a real-world scenario, these metrics can be measured using well-known monitoring and observation tools. Kepler[2] is a well-known energy consumption exporter that exposes statistics from applications running in Kubernetes clusters at the container and node levels. The metrics measured by Kepler cover a wide range, including CPU, GPU, and storage energy consumption [14]. Also, Tapo P115[3] is a smart monitoring socket. Moreover, the tool introduced in [19] can also be utilized to implement the method proposed in this paper, demonstrating its feasibility for potential real-world experiments in the future (Table 2).

Table 2. Information stored for each stages in dataset.

Name	Definition
stages_id	stages's unique identifier
stages_type	stages's type(filtering, anonymization, aggregation, converting)
cpu_usage_unit	cpu consumption for running the stage
memory_usage_unit	storage consumption for running the stage
output_data_volume	the volume of output data of the stage
input_data_volume	the volume of input data of the stage
observation_unit	resource usage for monitoring the stage

The running example simulates energy profiling, and the dataset of stages (.csv file) could be the output of any energy metrics collector. In this validation, the capabilities are implemented using Java. All experiments are performed on a MacBook Pro M3 with 36 GB of RAM and an Apple M3 Pro chip. The creation of data-sharing pipelines is automated, with pipelines being randomly populated based on the stages in the dataset (.csv file). The total number of stages in each pipeline, and which stages should be executed, are selected randomly. In the next step, for every pair of pipelines, common stages in the same order are identified,

[1] GitHub address: https://github.com/Sepide-Masoudi/Energy-profiling-in-federated-data-sharing-pipelines/tree/master.

[2] https://sustainable-computing.io/.

[3] https://www.tp-link.com/de/home-networking/smart-plug/tapo-p115/.

and the energy consumption of these common stages is calculated using the energy model explained in Sect. 3.2. In Fig. 3, the plot shows the common stages between different pipelines in one iteration of the validation.

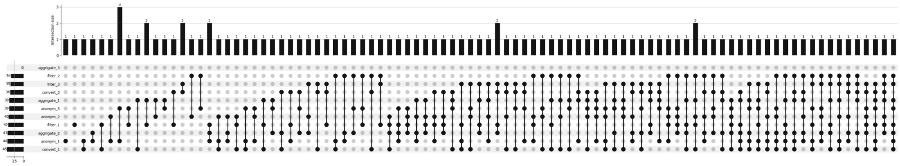


Fig. 3. Aggregation of stages across different pipelines in a single simulation round, with 100 pipeline populations created from random combinations of 11 stages from 4 types(filter, anonymization, aggregation, convert).

In the final report of the validation, as shown in Table 3, the administrator can view a list of pipelines with common stages between them, the energy consumption of those common stages, the total energy consumption of each pipeline in its current configuration, and the percentage showing the portion of energy consumption attributed to common stages relative to all stages in the pipeline. This report helps the data-sharing system administrator gain a comprehensive understanding of potential alternative execution configurations that could be shared among multiple data consumers, as well as the amount of energy that could be saved by creating a new pipeline for the common stags and reusing them across different consumers. Therefore, in a real-world scenario where the number of pipelines and operational complexities are greater, leveraging energy profiling enables the system administrator to identify opportunities for improving energy efficiency by running common stages only once and estimating the percentage of energy consumption for the shared components relative to the total. This estimation takes into account various factors, such as the specific resource usage of each stage on the current infrastructure, the energy associated with data transmission between stages, and the energy consumption from monitoring those stages (shown in Table 3).

Table 3. Part of energy profiler report about the common stages and energy consumption of them in one iteration of the validation (E_{CO}: common stages total energy consumption,$E_{P_{A,B}}$: total energy consumption of pipeline).

P_A	P_B	CO	E_{CO}	E_{CO}/E_{P_A}	E_{CO}/E_{P_B}
3,4,5	2,3,4	1,5	2911	30%	84%
2,3,5,6,9,10	2,3,5,6,7,9	2,3,5,6	27542	13%	24%
4,5,6,8,10	3,4,5	4,5	8638	1,7%	91,5%

Through this experiment, we confirmed the feasibility and benefits of using the proposed energy model and energy profiling in identifying and highlighting reuse opportunities. The model successfully calculated the energy consumption of a pipeline, broken down by its stages, detected common stages across different pipelines, and estimated the potential energy savings from reusing those stages. This demonstration validates that the model can enable data provider and data consumer administrators to make informed decisions on optimizing execution configurations, thereby minimizing overall energy consumption through the reuse of common stages or offloading specific stages to the consumer's side.

5 Limitations

The proposed model represents an early-stage effort to reduce waste in data-sharing environments. Consequently, it relies on several assumptions that require validation in real-world settings. Nonetheless, it is anticipated that ongoing developments in academia and industry – particularly in data mesh architectures [4], data spaces [11], and federated data sharing [5] – are progressing toward sufficient maturity to enable such validation in the near future.

The primary objective of this work was to model the reuse and energy-saving potential in data-sharing pipelines. To assess the feasibility of the approach, pipelines were modeled as sequences of fine-grained, easily distinguishable operations, inspired by prior work [20]. While this abstraction enables the intended optimization, it also limits the immediate applicability of the model, as current pipeline frameworks typically do not expose such granular and clearly defined operation boundaries.

Furthermore, the model assumes that the energy consumption associated with data operations and transfers can be profiled precisely. In practice, such measurements are subject to environmental variability [6], interference from co-located processes (e.g., noisy neighbors), and instrumentation limitations, which often result in only approximate values. Accordingly, future extensions should incorporate probabilistic representations of energy consumption to more accurately account for such variances.

Finally, while the simulation conducted in this work demonstrates the potential of the proposed method, its generalization remains to be validated. On the one hand, federated data-sharing environments are still in early stages of adoption, and the supporting pipelines are not yet sufficiently mature for full-scale application. On the other hand, existing platforms do not currently accommodate both the requirements of federated data sharing and the specific capabilities assumed in the presented model. However, it is expected that this gap will diminish as such platforms evolve. For example, the TEADAL approach [22] envisions a compatible environment and is currently undergoing evaluation. Application of the proposed method within such a setting is planned for future work.

6 Conclusions

In this study, we propose a method for modeling the energy consumption of data-sharing pipelines when independent parties share their data. Additionally, we introduce a method to improve the energy efficiency of data-sharing pipeline execution through energy profiling and the reuse of common stages between different pipelines. We evaluate our approach by running an example of energy profiling on synthetic pipelines and demonstrate how this method can assist with energy estimation in various configurations and utilized by other tools for energy saving.

In future studies, we aim to extend the energy profiling method by using machine learning algorithms for the estimation process, enabling more accurate predictions of total energy consumption for pipelines in different configurations and reuse strategies. This will allow us to account for changes in parameters such as the infrastructure on which the pipelines run and provide estimations for elements that are difficult to measure, such as energy consumption related to data transmission over the network. We also plan to consider the cost of different configurations, in addition to energy consumption, in future studies. Finally, we will apply the energy profiling method in a real-world data-sharing context to investigate the impact of the proposed approach in a practical use case by developing a tool that leverages this method to optimize pipeline execution planning.

Acknowledgments. Funded by the European Union (TEADAL, 101070186). Views and opinions expressed are however those of the authors only and do not necessarily reflect those of the European Union. Neither the European Union nor the granting authority can be held responsible for them.

References

1. Amado, L.R., Vogel, A., Griebler, D., Licks, G.P., Simon, E., Meneguzzi, F.: Automated planning for optimal data pipeline instantiation. arXiv preprint arXiv:2503.12626 (2025)
2. Chen, G.J., et al.: Realtime data processing at Facebook. In: Proceedings of the 2016 International Conference on Management of Data, pp. 1087–1098 (2016)
3. Dalmolen, S., Bastiaansen, H., Kollenstart, M., Punter, M.: Infrastructural sovereignty over agreement and transaction data ('metadata') in an open network-model for multilateral sharing of sensitive data. In: 40th International Conference on Information Systems, ICIS 2019. Association for Information Systems (2020)
4. Dehghani, Z.: Data Mesh: Delivering Data-Driven Value at Scale. Inc, O'Reilly Media (2022)
5. Falconi, M., Lombardo, G., Plebani, P., Werner, S.: Data friction: physics-inspired metaphor to evaluate the technical difficulties in trustworthy data sharing. In: International Conference on Advanced Information Systems Engineering, pp. 300–315. Springer (2024)
6. Fieni, G., Rouvoy, R., Seinturier, L.: Smartwatts: Self-calibrating software-defined power meter for containers. In: 2020 20th IEEE/ACM International Symposium on Cluster, Cloud and Internet Computing (CCGRID), pp. 479–488 (2020). https://doi.org/10.1109/CCGrid49817.2020.00-45
7. Gelhaar, J., Gürpinar, T., Henke, M., Otto, B.: Towards a taxonomy of incentive mechanisms for data sharing in data ecosystems. In: PACIS p. 121 (2021)

8. Goodhope, K., et al.: Building Linkedin's real-time activity data pipeline. IEEE Data Eng. Bull. **35**(2), 33–45 (2012)
9. Grossman, R.L.: Ten lessons for data sharing with a data commons. Sci. Data **10**(1), 120 (2023)
10. Guo, B., Wu, J., Pu, Y., Zhang, J., Yu, J.: Energy consumption estimation and profiling for queries in distributed database systems based on a bottom-up comprehensive energy model. Future Gener. Comput. Syst. **159**, 379–394 (2024). https://doi.org/10.1016/j.future.2024.04.059, https://www.sciencedirect.com/science/article/pii/S0167739X24001973
11. Halevy, A., Franklin, M., Maier, D.: Principles of dataspace systems. In: Proceedings of the Twenty-fifth ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems, pp. 1–9 (2006)
12. Hassan, N.: Managing data dependencies in cloud-based big data pipelines: challenges, solutions, and performance optimization strategies. Orient J. Emerg. Paradigms Artif. Intell. Autonom. Syst. **15**(2), 20–28 (2025)
13. Hofman, W.: Towards a federated infrastructure for the global data pipeline. In: Open and Big Data Management and Innovation: 14th IFIP WG 6.11 Conference on e-Business, e-Services, and e-Society, I3E 2015, Delft, The Netherlands, October 13-15, 2015, Proceedings 14, pp. 479–490. Springer (2015)
14. Kepler, I.: Kubernetes efficient power level exporter (kepler). https://sustainable-computing.io. Accessed 10 Sept 2024
15. Kunjir, M., Birwa, P.K., Haritsa, J.R.: Peak power plays in database engines. In: Proceedings of the 15th International Conference on Extending Database Technology, pp. 444–455. EDBT '12, Association for Computing Machinery, New York, NY, USA (2012). https://doi.org/10.1145/2247596.2247648, https://doi.org/10.1145/2247596.2247648
16. Kurpicz, M., Orgerie, A.C., Sobe, A., Felber, P.: Energy-proportional profiling and accounting in heterogeneous virtualized environments. Sustain. Comput.: Inform. Syst. **18**, 175–185 (2018). https://doi.org/10.1016/j.suscom.2017.11.002, https://www.sciencedirect.com/science/article/pii/S221053791630107X
17. Mansouri, Y., Prokhorenko, V., Ullah, F., Babar, M.A.: Resource utilization of distributed databases in edge-cloud environment. IEEE Internet Things J. **10**(11), 9423–9437 (2023). https://doi.org/10.1109/JIOT.2023.3235360
18. Marinakis, V., et al.: From big data to smart energy services: an application for intelligent energy management. Future Gener. Comput. Syst. **110**, 572–586 (2020). https://doi.org/10.1016/j.future.2018.04.062, https://www.sciencedirect.com/science/article/pii/S0167739X17318769
19. Masoudi, S.: Pre-share data: assistance tool for resource-aware designing of data-sharing pipelines. In: 22nd IEEE International Conference on Software Architecture Companion 2025(ICSA-C) (2025)
20. Munappy, A.R., Bosch, J., Olsson, H.H.: Data pipeline management in practice: challenges and opportunities. In: Morisio, M., Torchiano, M., Jedlitschka, A. (eds.) PROFES 2020. LNCS, vol. 12562, pp. 168–184. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-64148-1_11
21. Otto, B.: Quality and value of the data resource in large enterprises. Inf. Syst. Manag. **32**(3), 234–251 (2015). https://doi.org/10.1080/10580530.2015.1044344
22. Plebani, P., et al.: Teadal: Trustworthy, energy-aware federated data lakes along the computing continuum. In: CEUR Workshop Proceedings, vol. 3413, pp. 28–35. CEUR-WS. org (2023)

23. Raman, K., Swaminathan, A., Gehrke, J., Joachims, T.: Beyond myopic inference in big data pipelines. In: Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 86–94 (2013)
24. Roukh, A.: Estimating power consumption of batch query workloads. In: Model and Data Engineering: 5th International Conference, MEDI 2015, Rhodes, Greece, September 26-28, 2015, Proceedings, pp. 198–212. Springer (2015)
25. Smith, J., Brown, E., Rodriguez, S., Johnson, M., Deshmukh, A., Williams, D.: Integrating data quality management into data pipeline optimization strategies
26. Tomasoni, M., Capponi, A., Fiandrino, C., Kliazovich, D., Granelli, F., Bouvry, P.: Why energy matters? profiling energy consumption of mobile crowdsensing data collection frameworks. Pervasive Mobile Comput. **51**, 193–208 (2018). https://doi.org/10.1016/j.pmcj.2018.10.002, https://www.sciencedirect.com/science/article/pii/S1574119217305965
27. Ullah, F., Mohammed, I., Babar, M.A.: A framework for energy-aware evaluation of distributed data processing platforms in edge-cloud environment. arXiv preprint arXiv:2201.01972 (2022)
28. Wilkinson, M.D., et al.: The fair guiding principles for scientific data management and stewardship. Sci. Data **3**(1), 1–9 (2016)
29. Zhu, J., Shen, Y., Song, Z., Zhou, D., Zhang, Z., Kusiak, A.: Data-driven building load profiling and energy management. Sustain. Cities Society **49**, 101587 (2019). https://doi.org/10.1016/j.scs.2019.101587, https://www.sciencedirect.com/science/article/pii/S221067071832362X

Determining Window Sizes Using Species Estimation for Accurate Process Mining over Streams

Christian Imenkamp[1(✉)], Martin Kabierski[2], Hendrik Reiter[3], Matthias Weidlich[2], Wilhelm Hasselbring[3], and Agnes Koschmider[1,2]

[1] Business Informatics and Process Analytics, University of Bayreuth, Bayreuth, Germany
{christian.imenkamp,agnes.koschmider}@uni-bayreuth.de

[2] Department of Computer Science, Humboldt-Universität zu Berlin, Berlin, Germany
{martin.kabierski,matthias.weidlich}@hu-berlin.de

[3] Department of Computer Science, Christian-Albrechts-Universität zu Kiel, Kiel, Germany
{hendrik.reiter,hasselbring}@email.uni-kiel.de

https://www.pa.uni-bayreuth.de/en/

Abstract. Streaming process mining deals with the real-time analysis of event streams. A common approach for it is to adopt windowing mechanisms that select event data from a stream for subsequent analysis. However, the size of these windows denotes a crucial parameter, as it influences the representativeness of the window content and, by extension, of the analysis results. Given that process dynamics are subject to changes and potential concept drift, a static, fixed window size leads to inaccurate representations that introduce bias in the analysis. In this work, we present a novel approach for streaming process mining that addresses these limitations by adjusting window sizes. Specifically, we dynamically determine suitable window sizes based on estimators for the representativeness of samples as developed for species estimation in biodiversity research. Evaluation results on real-world data sets show improvements over existing approaches that adopt static window sizes in terms of accuracy and robustness to concept drifts.

Keywords: streaming process mining · Data Representativeness · Log completeness · Window size

1 Introduction

Generally, process mining techniques may be employed in *offline* and *online* use cases. Techniques for offline process mining evaluate historical event data, while streaming process mining aims at providing immediate insights based on a continuous stream of event data. As such, streaming techniques handle event data as soon as it becomes available, thereby enabling a timely reaction to unexpected process dynamics. Streaming techniques for process mining may generally adopt

J. Krogstie et al. (Eds.): CAiSE 2025, LNCS 15701, pp. 109–124, 2025.
https://doi.org/10.1007/978-3-031-94569-4_7

three different paradigms [4]: (i) Window-based approaches select a particular part of the event data at specific time points [4], which then facilitates the analysis using traditional process mining techniques. (ii) Other approaches implement a problem reduction, i.e., they trace back the computations required for streaming process mining to more generic streaming problems [4], such as counting or pattern detection over data streams. (iii) Finally, streaming process mining may rely on the pre-computation of possible analysis results, which are then merely selected based on the event data encountered in a stream [4].

Considering these paradigms, window-based approaches have the major advantage that they do not impose any assumption on the analysis algorithms to use. Unlike the approaches that rely on traditional streaming algorithms or pre-computation, they provide a generic solution for a wide range of specific analysis algorithms. However, when adopting a window-based approach, an important challenge is the selection of suitable parameters for the applied windows. While the question of the *points in time* for the evaluation of a window (commonly referred to as the window *slide*) is typically answered directly based on the latency requirements of the analysis, the question of a suitable *size* of the window imposes important challenges. While the window size should be selected so that the window is *representative* of the underlying process dynamics (i.e., how well does a sample K (the content of the window) captures or preserve important characteristics of a population P (a generative system or the origin of the event stream)). In practice, processes and hence the event data generated by them tend to show certain variability, stemming from exceptional process executions or concept drift. A fixed, static size of a window can, therefore, not be expected to capture a process accurately, which biases the downstream analysis tasks.

Let's consider the following example to motivate the importance of the optimal window size: A hospital monitors thousands of patient treatment processes daily, utilizing an information system. The hospital can therefore derive information on process conformance, staff utilization and treatment protocols in real-time. However, the treatment processes naturally evolve due to factors like seasonal changes, staff rotation or new medical protocols. It's likely that a static window size will result in: (1) It will miss important changes during high-variation periods if the window size is too small. (2) The window include outdated information and therefore decrease the accuracy if the window is too big. Moreover, the higher window size will result in higher resource consumption.

In this paper, we address the challenge of selecting representative windows for streaming process mining. Our approach builds upon recent results on assessing the representativeness of event data for process mining using notions and measures from biodiversity research, particularly relying on species estimation. Specifically, we contribute an approach for the dynamic adaptation of the size of a window, based on its estimated representativeness of the underlying process, in terms of the completeness and coverage of the species induced by the event data. Additionally, we provide an algorithm to automatically adjust the coverage threshold of the window. We evaluate our approach in three steps. First, we demonstrate robustness to concept drift by using synthetic event logs with different drift types. We show how the window automatically adapts to the new stream characteristics. Next, we assess the accuracy of the windows in terms of the F1-score of the discovered process model utilizing real-world event logs and benchmark our approach against existing windowing methods. Finally, we show the real-world applicability by analyzing the runtime efficiency and latency.

In the remainder, Sect. 2 provides the background for our work. Section 3 presents our approach for the selection of a window size. Section 4 discusses evaluation results, Sect. 5 reviews related work, and Sect. 6 presents conclusions.

2 Background

This section introduces preliminaries for our work, in terms of representativeness, an event stream model (Section 2.1), sliding windows (Section 2.2), concept drifts (Section 2.3) and estimation techniques for the completeness and coverage of event data (Section 2.4).

2.1 Event Stream Model

Following [6], we define an event as a triple $e = (c, a, t)$, where c is a case identifier, a is an activity name, and t is a timestamp. We write $e.c$, $e.a$, and $e.t$ to refer to the components of event e. The set of all possible events is called the event universe, denoted as $\mathcal{E} = \mathcal{C} \times \mathcal{A} \times \mathbb{N}$, where $\mathcal{C}$ is the set of all case identifiers and $\mathcal{A}$ is the set of all activity names. A potentially infinite sequence of events $S : \mathbb{N} \rightarrow \mathcal{E}$ is called an event stream. Here, we assume that the order of events in the stream complies with their temporal order, i.e., for all $i, j \in \mathbb{N}$, it holds that $i < j$ implies that $S(i).t < S(j).t$.

2.2 Windowing Methods

To process a stream of event data, it is common to employ sliding windows that characterize the elements that are currently active. We shortly review prominent notions of sliding windows that are used in the context of streaming process mining [3]. In a time-based window, events are gathered within a predefined time interval. For count-based windows, both the size and slide parameters are given in terms of a fixed number of events. In a landmark window, a designated event serves as a "landmark" that indicates the end of a window.

2.3 Concept Drifts

Concept drift occurs when the behavior of a process changes over time in unforeseen ways [26]. Drifts can occur in many different ways (i) **Sudden Drift**: The behavior of the stream changes abruptly. (ii) **Recurring Drift**: Recurring changes appear seasonally over time. (iii) **Gradual Drift**: Gradual changes occur through slow degradation, starting in specific contexts and eventually affecting the entire stream. (iv) **Incremental Drift**: Incremental changes involve many small-scale modifications, leading to changed behavior over time.

2.4 Completeness and Coverage Estimation

To assess the representativeness of event data, we adopt measures for the completeness and coverage of event logs [18]. They relate the number of observed distinct values of interest in an event log, called *log species*, to the estimated total number of species, the *species richness*, in the system generating the event log.

We here define auxiliary concepts, as follows. Let P be a potentially infinite population of individuals, each belonging to exactly one species, whose occurrence probabilities are fixed, unknown and assumed to be independent from each other. Let $\Omega_P = 1, 2, \ldots, S_P$ be the finite, enumerated set of species in P, i.e. *the species richness* of P with each species having a *species occurrence probability* of $p_1, p_2, \ldots, p_{S_P}$, respectively. Furthermore, let $\zeta : P \rightarrow \Omega_P$ be a *species retrieval function*, which assigns to an individual, the species it belongs to.

Now, consider a sample $N \subseteq P$ of size $n = |N|$ that is drawn randomly from the possibly infinite population P. By $S_N \leq S_P$, we denote the number of species observed in sample N. Let X_i be the *sample species frequency* of species i, i.e. the total number of occurrences of species i in sample N, so that any species with $X_i = 0$ is unobserved in sample N.

Based thereon, $f_k = \sum_{i=1}^{S_P} I(X_i = k)$ is the *abundance frequency count*, i.e., the number of species that are represented by exactly k observations in N. We denote as f_1 the number of species represented by exactly one individual (called a *singleton*) and as f_2 the number of species denoted by exactly two individuals (called a *doubleton*). Under this multinomial sampling model of abundance-based species data [13], the probability of observing species counts $X_1 = x_1, X_2 = x_2, \ldots, X_{S_P} = x_{s_P}$ is given as:

$$P(X_1 = x_1, X_2 = x_2, \ldots, X_{S_P} = x_{s_P}) = \frac{n!}{x_1! x_2! \ldots x_{s_P}!} p_1^{x_1} p_2^{x_2} \ldots p_{S_P}^{x_{s_P}}. \tag{1}$$

For this data model, we obtain an estimate of the species richness S_P using the Chao1-estimator [9] as:

$$\hat{S}_{Chao1} \approx \begin{cases} S_N + f_1^2 / (2f_2) & \text{if } f_2 > 0, \\ S_N + f_1 (f_1 - 1)/2 & \text{if } f_2 = 0. \end{cases} \tag{2}$$

Using the estimated species richness $\hat{S}_{Chao1}$, we then derive the *completeness* of sample N, i.e. the fraction of observed species from P in N, denoted $\hat{Com}_N$, as:

$$\hat{Com}_N = \frac{S_N}{\hat{S}_{Chao1}} \tag{3}$$

Furthermore, we obtain the *coverage* of sample N, i.e. the probability space covered by by the observed species in N, as:

$$\hat{Cov}_N = 1 - \frac{f_1}{n} \left(1 - \frac{2f_2}{(n-1)f_1 + 2f_2} \right) \tag{4}$$

The Chao1-estimator yields a lower bound on S_P, and for the optimal case, in which the occurrence probabilities of all undetected species are equal, it is an unbiased point estimator of S_P [10].

For samples, for which $f_1 = 0$, the right-hand side of Equation (2) evaluates to 0, which indicates the sample to have full completeness and coverage.

We note, that in general, the above metrics are subject to various assumptions and are thus suspectible to certain scenarios in which these do not hold. First, the estimators assume that each species will eventually be reobserved under infinite sampling. Clearly, should this not be the case, for instance in the case of erroneous events, that are recorded only once, in the case of noisy event streams, which may also generate spurious events, or in the case of process drifts, that remove previously existent behavior, then perfect completeness may never be reached. Coverage on the other hand will still converge to full coverage, due to the diminishing impact of these spurious species for increasingly growing samples. Furthermore, the species domain needs to be finite, i.e., the number of species in the system needs to be bounded, as estimates will otherwise approach infinity.

For a small sample, in turn, it may occur that no singleton species has been sampled, which inadvertently indicates full completeness and coverage. To avoid this effect, the sample size shall be increased, as to ensure a proper estimate of the species richness, see also [18].

In contrast to [18], we here utilize the Chao1-estimator, instead of the Chao2-estimator. While similar in structure, Chao1 assumes each observation to consist of exactly one species and retains exact occurrence counts per species, whereas Chao2 assumes observations to contain multiple species and relies on a different sampling model that does not account for occurrences of the same species in one observation [13]. Since event streams consist of individual observations containing atomic information, we thus opt for estimation using the Chao1-estimator.

Lastly, we note that, while the multinomial sampling model assumes independence of species, estimates are still accurate in the case of dependence between species, as long as the occurrence counts of each species are approximately proportional to the global occurrence probabilities in the population.

2.5 Species Definitions

In process analysis, we categorize process characteristics as distinct species, for which the above metrics are computed. Intuitively, each observed species represents an abstraction of process behavior observed in the event stream, with the type of abstraction defining the species view of interest. We define $\zeta : P \rightarrow \Omega_P$ as a species retrieval function that assigns to each individual process element its species. Following [18], we utilize three categories of species definitions: Simple Species - Activities (ζ_{act}), Complex Species - Trace Variants (ζ_{tv}) and Intermediate Species - Directly Follows Relations (ζ_{df}). (1) ζ_{act} defines each activity of a trace as an observed species, i.e. Ω_P is the set of activities in the system. (2) ζ_{tv} treats each trace as the occurrence of a species, i.e. Ω_P is the set of trace variants. (3) ζ_{df} represents the directly follows relations between activities as species, i.e. Ω_P is the set of directly-follows relations in the system.

3 Dynamic Window Size Selection

We now turn to the discussion of the representativeness of streaming windows. First, in Sect. 3.1, we motivate, why completeness and coverage should be considered as a measure of representativeness, when selecting a window size in streaming process mining. Then, we propose a dynamic threshold heuristic to determine the respective threshold for the coverage of a window Sect. 3.2, before explaining, how completeness and coverage may guide the selection of an appropriate windows size in Sect. 3.3.

3.1 Motivation for Window Representativeness

When aiming at process discovery based on a stream of events, the selection of a window size needs to consider the variability of the data, in addition to its volume and velocity. Specifically, data variability is manifested in the number of distinct activities for which execution is indicated by the events in the stream, and by extension, the behavioral dependencies between activities, which usually build the foundation for state-of-the-art process mining algorithms. Here, missing a crucial activity or dependency may induce significant and unpredictable changes in analysis results. Hence, the window size shall be sufficiently large to provide a *representative* view of the activities and their dependencies regarding the process underlying the stream.

Intuitively, if a process is rather homogenous, i.e. the number of activities and their dependencies are relatively small and stretched out over a short interval, a time-based or count-based windows of small size is sufficient for capturing the process' dynamics. Conversely, for a rather heterogeneous process, with many activities and dependencies between them that are observed over a long time, we expect larger window sizes for capturing all possible behaviors. Yet, process dynamics can be expected to change between these extreme poles, being subject to concept drift. With the event stream becoming more homogenous or heterogeneous as a result of these drifts, the window size shall, therefore, decrease or increase accordingly, to accurately capture the process.

We illustrate the above intuition with two simple event streams S_1 consisting of events indicating the continuous repeated execution of activities $\langle A, B, C \rangle$ in this order, and S_2, consisting of events indicating the continuous and repeated execution of activities $\langle A, B, C, D, E \rangle$, in the same order as well. Events are introduced to both streams with the same frequency. Assume that we want to select a window size, aiming at a representative view of the process' behavior, in terms of completeness.

In the light of the repeating patterns in the streams, we could conclude the view on the stream to be complete sooner for S_1 than for S_2, since activities start reappearing earlier, at the fourth event, compared to S_2, where repetition materializes with the sixth event. Thus, independently of when exactly we are confident that we have seen enough evidence for a complete collection of events, we conclude that the window size for S_2 needs to be larger than the window size for S_1, based only on the number of observed activities and their occurrence counts in the stream. Likewise, if S_1 drifts into S_2 or the other way around, then the window size could be increased or decreased according to the observed change and repetition of behavior.

3.2 Selecting a Window Size

Any open window should be closed, once the species observed in the windows lifetime exceed a dynamically chosen completeness threshold. For this selection, the coverage (see Eq. 4) is preferred over completeness (see Eq. 3) in scenarios with drifts or errors, as the former still converges to a measure of completeness in the presence of spurious singleton species. Here, we introduce a dynamic threshold heuristic to determine this threshold (see Algorithm 1). The algorithm is inspired by the elbow method [24] also used for threshold heuristics in [25]. First, the algorithm approximates the second derivative, i.e., second-order differences (lines 3–6). In particular, it identifies the point at which additional increases in window size provide diminishing returns. To ensure that open windows are eventually closed, the algorithm furthermore checks for stagnation in completeness (lines 8–15). If stagnation is detected (line 8), it increases a smoothing factor and consequently decreases the window size (lines 9 and 10). Otherwise, this smoothing factor is reduced (lines 12 and 13), allowing for slower changes (i.e., to avoid rapid adjustments) in the window size. This ensures that the window size does not fluctuate too quickly in response to short-term changes in coverage. The algorithm assumes responsibility for establishing the threshold, which consequently affects the window size. However, new parameters (e.g., smoothing factors, decay rate) are introduced. Nevertheless, these do not require alteration or adaptation to accommodate new data sets, and a (start) smoothing factor of 0.2 and decay rate of 0.1 can be employed universally.

3.3 Estimations for Streaming Process Discovery

To assess the representativeness of a window in the context of streaming process discovery, we operationalize the above estimators as follows. Given a window containing the events $e_1, e_2, \ldots, e_n$, we treat each of the n events as an observed individual. Based on the ordered list of events $e_1, e_2, \ldots, e_n$, we obtain an ordered list of associated activities as $e_1.act, e_2.act, \ldots, e_n.act$. Using this list,we may instantiate any of the species definitions discussed in Sect. 2.5.

Consider a sliding window containing events that yield the following sequence of activities $\langle A, B, A, C, B, D, A, C, E\rangle$. Considering ζ_{act}, species A occurs three times, species B and C two times, and species D and E once. Thus, there are five species observed in the sample, $S_N = 5$, and the abundance frequency counts are $f_3 = 1$, $f_2 = 2$, and $f_1 = 2$. Here, an estimate of the species richness gives $\hat{S}Chao1 = 5 + \frac{2^2}{2 \cdot 2} = 6$. That is, based on the seen species, we expect six distinct activities to be present in the process, which yields estimates of $\hat{ComN} = \frac{5}{6} \approx 0.83$ and $\hat{Cov}_N = 1 - \frac{2}{9}\left(1 - \frac{2 \cdot 2}{(9-1)2+2 \cdot 2}\right) \approx 0.82$. By considering,

Algorithm 1. Dynamic Threshold Heuristic

Input: coverage history C_i,$|C_i|$ n, smoothing factor sf, current threshold ct, decay rate dr, minimum threshold mt, stagnation threshold δ, stagnation window w,
Output: Updated threshold ct_{new}
▷ Compute second-order differences (a discrete approximation of the second derivative)
for $i = 1$ **to** $n - 2$ **do**
  $r''(i) \leftarrow C_{i-1} - 2C_i + C_{i+1}$
end for
$i^* \leftarrow \arg\max_i (r''(i))$
$C_{\text{optimal}} \leftarrow C_{i^*+1}$ ▷ Optimal threshold value (i.e., the "elbow" point)
▷ Check for coverage stagnation
if $n \geq w$ **and** $|C_k - C_{k+1}| < \delta \; \forall k = n - w$ **to** $n - 2$ **then**
  $\text{sf} \leftarrow \min(1.2 \times \text{sf},\ 0.99)$ ▷ Increase smoothing
  $\text{ct}_{\text{temp}} \leftarrow \max(\text{ct} - \text{dr},\ \text{mt})$ ▷ Decrease threshold
else
  $\text{sf} \leftarrow \max(0.8 \times \text{sf},\ 0.01)$ ▷ Decrease smoothing for faster reaction
  $\text{ct}_{\text{temp}} \leftarrow \text{ct}$ ▷ Threshold remains unchanged
end if
▷ Smoothly update the threshold
$\text{ct}_{\text{new}} \leftarrow \text{sf} \times C_{\text{optimal}} + (1 - \text{sf}) \times \text{ct}_{\text{temp}}$
return ct_{new}

ζ_{df} we obtain the species $AB, BA, AC, CB, BD, DA, CE$ with corresponding occurrence counts $1, 1, 2, 1, 1, 1, 1$, respectively.

In a streaming setting, the estimators need to be evaluated continuously and efficiently upon the arrival of new events. Assuming that all relevant data structures are initialized before a new event e is introduced into the stream, the algorithmic complexity of the estimation depends on three steps: (1) Retrieving the species from e, (2) Updating the observed species counts, (3) Updating the completeness and coverage measures.

The species retrieval function ζ can be computed in $O(1)$ for a given event e. Since this yields a single activity associated with e, updating the species counts can be done in $O(S_N)$, assuming a hash-based implementation of the species counts. Lastly, updating the completeness and coverage measures requires the retrieval of f_1, f_2, n, and S_n from the updated species counts. The value of n is obtained in $O(1)$ by incrementing a counter upon arrival of a new event. S_n can be obtained in $O(1)$ from the species counts.

f_1 and f_2 can be computed without iterating over all observed species for each event by updating a counter for f_1 and f_2 as follows. Should the species count after updating be 1, i.e., the current species created by event e is a newly observed singleton, increment f_1. Should the updated count be 2, i.e., a doubleton species, decrement the counter for f_1 and increment the counter for f_2. Lastly, should the updated count be 3, decrement the counter for f_2. Since the updates are tied to the update of the species counts, they can be computed in $O(1)$. Based on f_1,

f_2, n, and S_N, we obtain $\hat{S}Chao1$, $\hat{ComN}$, and $\hat{Cov}_N$ in $O(1)$. Thus, the overall time complexity of updating the estimators is $O(S_N)$.

4 Evaluation Results

This section summarizes results of evaluations of different aspects of the windowing method. First, we describe the experimental setup (Sect. 4.1). We then report on our results regarding the impact of process drifts (Sect. 4.2), the role of species definitions on the accuracy (Sect. 4.3) and the performance of the method (Sect. 4.4).

4.1 Experimental Setup

We implemented a prototype in Python[1]. The implementation is based on processing events that are emitted via the distributed stream processing platform Apache Kafka[2]. In general, no parameters between event logs were altered in the evaluation unless otherwise specified (such as the species definition). Our program interacts with the Kafka cluster via the Faust[3] library. We used logs from a public repository containing synthetic event logs [16]. Additionally, we used CDLG, a tool for generating event logs with concept drifts [17]. The selected event logs vary in complexity in terms of variability, uniqueness of events and length of traces. The event logs are described in more detail in Table 1. To evaluate the completeness- and coverage-based windowing and the used species definition, we set up an event streaming environment.

All event logs in Table 1 were used in the drift robustness analysis (Sect. 4.2) with 1-gram for the species (ζ_{act}), while the real-world logs (RTFMP [21], Sepsis Cases [23], BPI Challenge logs [14,15,27], hospital-billing [22]) were used for accuracy evaluation (Sect. 4.3) relying on all species described in Sect. 2.5. The performance evaluation (Sect. 4.4) was conducted using the Sepsis Cases log due to its moderate size and complexity.

In particular, we intend to answer the following questions which are derived from the weaknesses of classic windowing methods: **(R01)** Can the window dynamically adjust to concept drifts without being rigid? **(R02)** Can we find an accurate window with only minor process knowledge? **(R03)** How does the approach perform in terms of computational efficiency?

First, we evaluate our approach on datasets with concept drifts (Sect. 4.2). Second, we benchmark our approach against different species definitions (Sect. 4.3). Finally, we evaluate the runtime efficiency (Sect. 4.4).

4.2 Robustness to Concept Drifts

Fig. 1 shows the results of our experiments on concept drifts and window rigidity when closing a window, considering ζ_{act} as species definition. With each closing

[1] https://github.com/chimenkamp/adaptive-window-for-online-process-mining.
[2] https://kafka.apache.org.
[3] https://github.com/robinhood/faust.

Table 1. Simulated Concept Drift Logs

Log Name	Mean Trace Length	Number of Events	Drift Position
Sudden Drift			
cm-10000 [16]	25.08	125413	Always after 10%
cd-10000 [16]	25.52	127621	Always after 10%
cb-10000 [16]	25.09	125447	Always after 10%
Recurring Drift			
Recurring [17]	12.31	61536	Three seasonal changes
Gradual Drift			
Gradual 5000 traces [17]	9.64	48203	Between 40% and 60%
Incremental Drift			
Incremental [17]	11.37	79608	Five increments

window, we save the size of the window and plot it against the total number of all windows. Here, for all logs (described in Table 1), we can see the dynamic adjustment of the window size.

In fact, the size of the window correlates loosely to the mean trace length, i.e. the window size adapts to the complexity of the process. For instance, a simple process requires less events to be considered complete, while the window for a more complex process increases, capturing the process behavior. This behavior can be observed in the gradual drift log. It starts with window sizes close two 10. When the process gradual drifts into a more complex process, the window size increases to capture the relevant behavior.

In the case of the sudden drifts, the window detects the change and adjusts the window size accordingly. The sudden drifts are marked with a vertical line.

The recurring drift demonstrates the adaption to seasonal changes. The windows capture the expected number of events. Gradual and incremental drifts pose a more challenging scenario, where the process changes progressively over a long time. First, the gradual drift shows that if the events vary too much, the window sizes increase to reach coverage. Additionally, if the event variability is too high, a larger window size supports the inclusion of more events, capturing the broader range of activities. Finally, the incremental drift emphasizes this behavior. Due to the complexity of the processes, the window size increases and adjusts with the next incremental shift.

To answer **R01**, we analyzed window size changes around known drift points. To accomplish this, we calculate the Mean Relative Change, the Standard Deviation of Relative Change, and the Coefficient of Variation. We analyze 10 windows before the drift and 20 windows after the drift for the Sudden Drift: CM

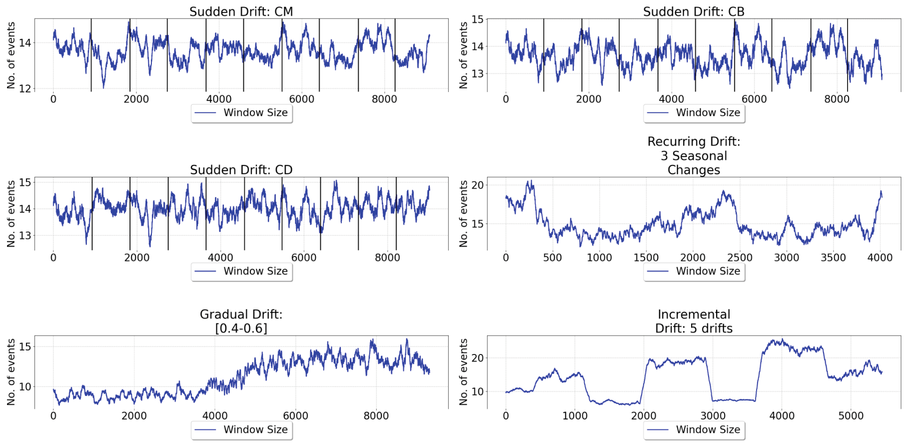


Fig. 1. The x-axis shows all windows found for the dataset, while the y-axis shows the number of events captured in the specific window. This demonstrates the adaption of the window to drifts.

and Gradual Drift event logs. **Sudden Drift: CM**: The results show the window size typically adapts by 8.5% (±4.3%) during drift periods. A coefficient of variation of 0.269 (0 indicates rigidness and 1 indicates volatile) demonstrates that the approach avoids excessive fluctuations. **Gradual Drift**: The window size gradually increased from an average of 9.3 events pre-drift to 11.3 events during drift, and finally to 13.3 events post-drift. This represents a 21.6% relative increase during the drift period. A low coefficient of variation during drift (0.113) and a small average trend (0.003 ± 0.111) demonstrated the controlled adoption. These metrics confirm that the approach can dynamically adjust to concept drifts without being overly rigid or volatile.

4.3 Species Evaluation and Accuracy

Next, we assess the impact of the species definitions on coverage estimation. We conducted experiments by sequentially applying different species retrieval functions to the event stream. Specifically, we processed the first 2,000 events of each event log. Then, we applied the species retrieval functions to the coverage estimation, and recorded conformance metrics (i.e., including Fitness, Precision, F1-score) once a window was identified. Table 2 summarizes our findings.

It presents the species retrieval function that achieves the highest and lowest average F1-score across all windows and various event logs. The ζ_{act} performs particularly well for logs with structured and repetitive sequences, as it captures local activity patterns (e.g., bpi-c-2013). In contrast, ζ_{df} species definitions excel in logs where the timing between events is crucial (e.g., Sepsis Cases). ζ_{tv} species are more suited for highly variable, complex logs, where the diversity in process behavior is more pronounced (e.g., hospital-billing). Additionally, Fig. 2 illustrates the evolution of the conformance metrics over all windows. For instance, in the case of *bpi-c-2013*, the F1-score remains consistently high, supported by an average window size of 69.44 (Min: 25, Max: 187). By contrast, the *Sepsis Cases* data exhibits significant variability, with an average window size of 14.53 (Min: 4, Max: 118). This volatility may be attributed to the differences in process execution patterns, possibly due to varying practices during night and day shifts. Furthermore, the *Road-Traffic-Fine-Management-Process* reveals three notable precision drops. Here, the average window size is 18.96 (Min: 10, Max: 72). In most cases, windows begin with the activity *Create Fine* and conclude with either *Send for Credit Collection* or *Payment*, which aligns with the expected process flow. For those instances where precision falls, the window instead starts with *Payment*, potentially indicating deviations in process execution.

To answer **R02**, we compare the average F1-score of the best species with a fitting windowing approach. In particular, we utilize the Sepsis Cases Event Log with a Landmark Window and choose the activity label *ER Registration*. Please note that, this assumes that the process has a clearly defined starting activity. As the sepsis cases fulfil this, the landmark window will therefore achieve the best results, even if it is susceptible to concept drifts. The average F1-Score over all discovered windows is 0.62 (min: 0.58, max: 0.88). Further, a count-based tumbling window with a fixed size of 20 achieves an average F1-Score of 0.61 (min: 0.34 m max: 0.81). Both archive significant lower F1-Scores compared to our approach (0.76). A benchmarking of the other windows and event logs is available in the repository.

4.4 Performance Evaluation

Now, we report on evaluation results in terms of runtime efficiency. We evaluate latency and throughput at the system level, which are considered conventional metrics for assessing the efficiency of stream data processing systems [19]. First, we define latency as processing time latency, i.e., the time interval between the ingestion of the first event within a window and the emission of the result. To evaluate latency, we use the python library perfplot.[4] As shown in Fig. 3, the results confirm our assessment of time complexity (see Sect. 3.3), i.e. we observe a linear ($O(S_N)$) increase in latency.

To answer **R03**, we use the Throughput. It is defined as the number of events that are processed in a given time interval. That is, the unit of throughput is processed events per second. We calculate an average throughput of 8962.13 events/s, with a standard deviation of 1166.18 events/s. However, external factors such as the Kafka Cluster and the Faust implementation influence this result. Notably, since Faust is implemented in Python, there might be a decrease of performance.

The results show that our approach can find accurate windows only with minor process knowledge. Furthermore, It can dynamically adjust to concept drifts without being rigid, while, being efficient in terms of utilizing computational resources.

[4] https://github.com/nschloe/perfplot.

Table 2. Comparison of best/worst species with F1-scores

Log Name	ζ (Highest) - F1-score	ζ (Lowest) - F1-score
Road-Traffic-Fine Mngmt Process	ζ_{act}(5-gram) (0.9257)	ζ_{act}(3-gram) (0.6212)
Sepsis Cases	ζ_{df} (0.7686)	ζ_{tv} (0.6282)
bpi-c-2012	ζ_{tv} (0.5850)	ζ_{act}(5-gram) (0.3525)
bpi-c-2013	ζ_{act}(5-gram) (0.9718)	ζ_{df} (0)
bpi-c-2015	ζ_{act}(1-gram) (0.6596)	ζ_{tv} (0)
bpi-c-2017	ζ_{act}(1-gram) (0.6105)	ζ_{act}(5-gram) (0.4321)
hospital-billing	ζ_{tv} (0.9237)	ζ_{act}(4-gram) (0.7170)
synthetic, online order	ζ_{tv} (0.9756)	ζ_{df} (exponential) (0.6709)

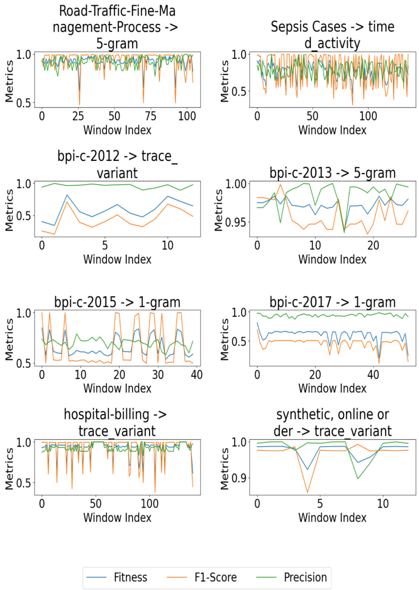


Fig. 2. Conformance metrics for all event logs over all windows

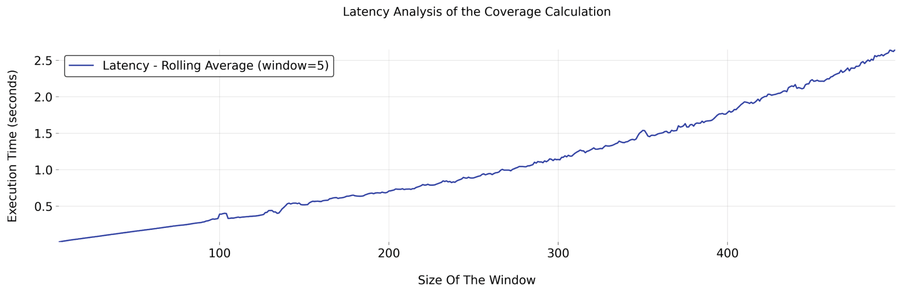


Fig. 3. Analysis of the latency time for incrementally larger windows ($n_{\max} = 500$)

5 Related Work

Recently, several proposals for process mining over event streams have been made [3]. Typically, streaming algorithms maintain some computational state based on the properties of past events, which may employ approximations, such as lossy counting [7]. The computational state may be managed separately, which then enables the integration of offline discovery algorithms [29]. Streaming algorithms have also been presented to assess the conformance of event streams with a process model, e.g., by evaluating model constraints over a stream [8], by replaying the events from the stream in the model [5], by aligning trace prefixes [28], or by learning a state predictor [20].

The above techniques ignore windows over streams (e.g., by assuming that events are partitioned in traces and traces will end eventually) or implement simple heuristics to select the window size, see Sect. 2.2. In this work, we propose a systematic approach to dynamically select an appropriate window size.

The question of how to determine a suitable window size is closely linked to sample-based process mining. However, these measures and existing sampling strategies are typically defined for a fixed population of events, which does not materialize in a streaming setting [1,2].

Our selection of a window size is guided by completeness estimators from biodiversity research. In addition to the mentioned Chao2-estimator, species richness may also be assessed using other estimators, such as abundance-based coverage [11]. Those estimators, as well as estimators for further population properties [12], provide an avenue for further research on window selection mechanism in process mining.

6 Conclusion and Future Work

This paper presents a novel approach for streaming process mining relying on species discovery and dynamically adjusted sliding windows aiming to determine the optimal window size to preserve the accuracy and completeness of the discovered process models. The comparison of our approach with sliding window approaches utilizing time-based, count-based and landmark windows show superiority in terms of accuracy and robustness to concept drifts. Future work shall address the transferability of our approach to additional process mining scenarios involving both highly fluctuating time intervals and high variability of process changes. Furthermore, the application of the proposed approach in distributed scenarios (i.e., dealing with volume, velocity and variability) is planned, where scalability as well resource constraints are essential. Additionally, we plan to consider behavioral aspects like resource allocation and workload distribution.

Acknowledgements. This work was partly supported by the German Federal Ministry of Education and Research (BMBF), grant number 16DII133 (Weizenbaum-Institute). This work received funding by the Deutsche Forschungsgemeinschaft (DFG), grant 496119880. The responsibility for the content of this publication remains with the authors.

References

1. Bauer, M., van der Aa, H., Weidlich, M.: Sampling and approximation techniques for efficient process conformance checking. Inf. Syst. **104**, 101666 (2022)
2. Bauer, M., et al.: How much event data is enough? A statistical framework for process discovery. In: Advanced Information Systems Engineering. Ed. by John Krogstie and Hajo A. Reijers. Cham: Springer International Publishing, pp. 239–256 (2018)

3. Burattin, A.: Streaming process mining. In: Process Mining Handbook. van der Aalst, W.M., Carmona, J. (eds.) Cham: Springer International Publishing, pp. 349–372 (2022). URL: https://doi.org/10.1007/978-3-031-08848-3_11
4. Burattin, A.: Streaming Process Mining. In: Process Mining Handbook. van der Aalst, W.M., Carmona, J., (eds.), Vol. 448, pp. 349–372 (2022)
5. Burattin, A., Carmona, J.: A framework for online conformance checking. In: Teniente, E., Weidlich., M., (eds.), vol. 308. LNBIP, pp. 165–177 (2017)
6. Burattin, A., Cimitile, M., Maggi, F.M., Sperduti, A.: Online discovery of declarative process models from event streams. IEEE Trans. Serv. Comput. **8**(6), 833–846 (2015)
7. Burattin, A., Sperduti, A., van der Aalst, W.M.P .: Control-flow discovery from event streams. In: CEC, pp. 2420–2427 (2014) . DOIurl-https://doi.org/10.1109/CEC32438.2014
8. Burattin, A., van Zelst, S.J., Armas-Cervantes, A., van Dongen, B.F., Carmona, J.: Online conformance checking using behavioural patterns. In: Weske, M., Montali, M., Weber, I., vom Brocke, J. (eds.) BPM 2018. LNCS, vol. 11080, pp. 250–267. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-98648-7_15
9. Chao, A.: Nonparametric estimation of the number of classes in a population. IScandinavian J. Stat. **11**(4), 265–270 (1984). (Visited on 06/04/2024)
10. Chao, A., Colwell, R.K.: Thirty years of progeny from Chao's inequality: estimating and comparing richness with incidence data and incomplete sampling. SORT Stat. Oper. Res. Trans. **41**(1) (2017)
11. Chao, A., Lee, S.M.: Estimating the number of classes via sample coverage. J. Am. stat. Assoc. **87**(417), 210–217 (1992)
12. Chao, A., et al.: Rarefaction and extrapolation with hill numbers: a framework for sampling and estimation in species diversity studies **84**(1), 45–67 (2014)
13. Robert, K., Colwell et al.: Models and estimators linking individual-based and sample-based rarefaction, extrapolation and comparison of assemblages. J. Plant Ecol. **5**(1), 3–21(2012)
14. Boudewijn van Dongen. BPI Challenge 2015 Municipality 1. 2015. URL: https://data.4tu.nl/articles/dataset/BPI_Challenge_2015_Municipality_1/12709154/1
15. Boudewijn van Dongen. BPI Challenge 2017. 2017. URL: https://data.4tu.nl/articles/dataset/BPI_Challenge_2017/12696884/1
16. Fontenla, G.:. A conformance checking-based approach for sudden drift detection in business processes. IEEE Trans. Serv. Comput. **16**(1), 13–26 (2023)
17. Grimm, J., Kraus, A. and van der Aa, H.: CDLG: a tool for the generation of event logs with concept drifts. In: International Conference on Business Process Management (2022). https://api.semanticscholar.org/CorpusID:251436795
18. Kabierski, M., Richter, M., Weidlich, M.: addressing the log representativeness problem using species discovery. In: 2023 5th International Conference on Process Mining (ICPM), pp. 65–72 (2023)
19. Karimov, J., et al.: Benchmarking distributed stream data processing systems. In: 2018 IEEE 34th International Conference on Data Engineering (ICDE), IEEE (2018)
20. Lee, W.L.J., Burattin, A., Munoz-Gama, J., Sepúlveda, M.: Orientation and conformance: a HMM-based approach to online conformance checking. Inf. Syst. **102**, 101674 (2021). https://doi.org/10.1016/j.is.2020.101674
21. M. (Massimiliano) Leoni, D., Mannhardt, F.: Road traffic fine management process (2015). https://data.4tu.nl/articles/dataset/Road_Traffic_Fine_Management_Process/12683249/1

22. Mannhardt, F.: . Hospital Billing - Event Log (2017). https://data.4tu.nl/articles/dataset/Hospital_Billing_-_Event_Log/12705113/1
23. Mannhardt, F.: Sepsis Cases - Event Log (2016). https://data.4tu.nl/articles/dataset/Sepsis_Cases_-_Event_Log/12707639/1
24. Marutho, D., Handaka, S.H., Wijaya, E.,Muljono.: The determination of cluster number at K-mean using elbow method and purity evaluation on headline news. In: 2018 International Seminar on Application for Technology of Information and Communication, pp. 533–538 (2018)
25. Nolle, T., Seeliger, A., Mühlhäuser, M.: BINet: multivariate business process anomaly detection using deep learning. In: Weske, M., Montali, M., Weber, I., vom Brocke, J. (eds.) BPM 2018. LNCS, vol. 11080, pp. 271–287. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-98648-7_16
26. Schlimmer, J.C., Granger Jr, R.H.: Beyond incremental processing: tracking concept drift. In: Proceedings of the Fifth AAAI National Conference on Artificial Intelligence. AAAI'86. Philadelphia, Pennsylvania: AAAI Press, pp. 502–507 (1986). (Visited on 06/06/2024)
27. Steeman, W.: BPI Challenge 2013, incidents (2013). https://data.4tu.nl/articles/dataset/BPI_Challenge_2013_incidents/12693914/1
28. van Zelst, S.J., Zelst et. al.: Online conformance checking: relating event streams to process models using prefix-alignments. Int. J. Data Sci. Anal. **8**(3), 269–284 (2019)
29. van Zelst, S.J., van Dongen, B.F., van der Aalst, W.M.: Event stream-based process discovery using abstract representations. Knowl. Inf. Syst. **54**(2) , 407–435 (2018)

Engineering Early Warning Systems: an Industrial Experience

Alessandro Burastero[1], Giuseppina Cappelluti[1], Martina De Sanctis[2](✉), Amleto Di Salle[2], Ludovico Iovino[2], Claudio Pompilio[2], Cosimo Versace[1], and Luca Ferraris[1]

[1] CIMA Foundation, Savona, Italy
{alessandro.burastero,giusy.cappelluti,cosimo.versace, luca.ferraris}@cimafoundation.org

[2] Gran Sasso Science Institute, L'Aquila, Italy
{martina.desanctis,amleto.salle,ludovico.iovino,claudio.pompilio}@gssi.it

Abstract. With climate change underway, the need for early warning systems (EWS) has become crucial for real-time monitoring and forecasting of natural disasters like storms, droughts, and floods. The United Nations (UN) considers EWS cost-effective tools that save lives and has launched the Early Warnings for All initiative to ensure global protection by 2027. This paper stems from a collaboration with the CIMA Foundation (*CIMA*), which re-engineered its EWS, MyDewetra, to address new requirements and growing data and user demands. We describe the system's evolution, focusing on its updated architecture and technical aspects. The re-engineered MyDewetra was evaluated through qualitative and comparative analysis, showing improvements in development effectiveness, user experience, maintainability, reliability, and security.

Keywords: Early Warning Systems · MyDewetra

1 Introduction

As the planet continuously breaks climate records, the need for Early Warning Systems (EWS) becomes increasingly evident. The 2023 has been the warmest year on record, further underlining the urgency for effective climate action [22]. However, as reported by the United Nations (UN) Secretary-General at COP28, current national plans indicate a 9% increase in emissions [36], contrary to the Paris Agreement's goal of limiting global warming. This underscores a significant gap between the necessary actions and actual commitments. In this context, the *Early Warnings for All* initiative [39] launched by the UN aims to fill this gap by providing global protection from hazardous climate-related events by 2027. Achieving this ambitious goal demands collaboration and coordination among all stakeholders. Building resilient communities and advancing climate justice depends heavily on the development and enhancement of EWS.

J. Krogstie et al. (Eds.): CAiSE 2025, LNCS 15701, pp. 125–142, 2025.
https://doi.org/10.1007/978-3-031-94569-4_8

In this context, both governmental [25,40,44] and research [4,26,27] initiatives have been launched. Within our academic institution, the Gran Sasso Science Institute (*GSSI*), we started researching EWS in the context of a project with the Regional Civil Protection Agency[1]. This led us to engage the *CIMA* Foundation[2], an international center for environmental monitoring and the primary provider of such systems for Italy's National Civil Protection Department.

This paper reports on our experience collaborating on re-engineering the *CIMA*'s EWS, namely MyDewetra, to meet new requirements and systematically address the increasing complexity of the previous version. The re-engineering process is driven not only by architectural, functional, and technological improvements but also by the redefinition of functionalities' application contexts. An application context refers to the definition of the purpose, scope, and integration of a functionality within the system.

As a result, we present the cloud-based high-level architecture of the re-engineered MyDewetra, along with the new technology employed. Additionally, we evaluated the re-engineered system through qualitative and comparative analysis, highlighting how it addresses the re-engineering requirements. The evaluation shows significant improvements in development effectiveness, user experience, maintainability, reliability, and security. Lastly, we believe that the reported experience provides valuable insights to support the goals of the *Early Warnings for All* initiative.

The rest of the paper is organized as follows. Section 2 gives the background on EWS and significant related work. Section 3 describes the industrial context and the research methodology we followed. Section 4 gives an overview of MyDewetra together with the needs that triggered its re-engineering. Section 5 describes the re-engineered MyDewetra and presents an operational scenario. Evaluation results are reported in Sect. 6, while Sect. 7 concludes the paper with lessons learned and future research directions.

2 Background and Related Work

EWS are crucial for climate change adaptation and disaster risk reduction [40]. They enable communities and authorities to take timely action by issuing warnings for hazards such as hurricanes, floods, and droughts. Their effectiveness lies in protecting lives, reducing economic damage, and offering a return on investment [45]. Investing in EWS aids disaster prevention and mitigation [16], yet significant gaps remain in global implementation [52]. Half of the world's countries lack adequate multi-hazard EWS, particularly in small islands, developing states, and least-developed countries, where extreme climate events cause high mortality. Many countries lack regulatory frameworks linking warnings to emergency plans and have insufficient global monitoring systems. As a result, there is growing focus on developing EWS through government and research efforts.

[1] https://allarmeteo.regione.abruzzo.it/.
[2] https://www.cimafoundation.org/en/.

The UN launched the *Early Warnings for All* initiative [39] as part of its Acceleration Agenda [35]. It aims to boost collaboration among UN agencies, governments, civil society, and partners to deliver people-centered, multi-hazard EWS. The initiative supports the Sendai Framework's focus on the availability and accessibility of such systems [44]. The UN has also partnered with various agencies to introduce new EWS in vulnerable areas around the world [40]. United Nations Development Programme projects in Africa, across Asia and the Pacific improve disaster preparedness and response [41–43]. Initiatives like the UN Environment Programme Climate Warning project and the CREWS program [51] aim to improve emergency response in typhoon- and flood-prone areas. Some EWS provide services and products for multiple climate-related risks, as for instance Meteoalarm [25]. Other EWS focus on specific climate-related risks. For example, The European Forest Fire Information System (EFFIS) [11] provides support for the protection of forests against fires in the EU countries. EWS addressing specific climate-related risks have been developed at various levels. For example, Austria has created an EWS for railway transport to tackle alpine hazards like debris flows, rock falls, avalanches, and landslides [9].

Regarding research initiatives, cloud computing is considered an enabler for global climate-related EWS [2]. Despite offering scalable, reliable access for real-time data processing and timely warnings, it is used by few EWS. It also enhances efficiency and flexibility with easy deployment, scalability, and broad accessibility [2]. Cloud computing supports technologies like the Internet of Things (IoT), enabling the collection, prediction, detection, monitoring, and analysis of climate-related data [8]. The joint use of IoT and cloud technologies also allows for improved data analysis, storage, processing, and device monitoring [26]. Additionally, cloud computing can be used in conjunction with artificial intelligence allowing algorithms to collect and analyze data in a logical and structured way, enabling the development of effective strategies for natural disaster response [19]. Pierleoni et al. propose an IoT-based EWS using low-cost, compact seismic monitoring devices to create a dense seismic network [26]. Pillai et al. integrate real-time sensor data into a service-oriented EWS using machine learning on a cloud server [27]. Bai et al. present an EWS for landslides that employs microservices to decouple and distribute system functions according to the usage load to facilitate real-time warning [4]. Akanbi proposes a formal model for decomposing monolithic EWS components as containerized microservices managed by Kubernetes [14] for realizing a multi-hazard EWS [3]. The SWITCH framework, part of the EU SWITCH project, proposes a flexible co-programming architecture to specify and support the life cycle of time-critical microservice-based cloud-native applications, such as EWS [34]. In this context, there is a clear need for open, distributed, cloud-based, and flexible EWS.

3 Industrial Context and Research Methodology

Within *GSSI*, we began researching EWS in collaboration with the Regional Civil Protection Agency. To strengthen our expertise in the EWS domain, we partnered with the *CIMA* Foundation, an international environmental monitoring center with over twenty years of experience in developing and managing multi-hazard EWS. Among its various stakeholders, *CIMA* is the primary provider of these systems for Italy's National Civil Protection Department, serving national, regional, and local government agencies, as well as international organizations and developing countries worldwide. With approximately 90 projects, around 150 employees, an IT department with experienced developers, researchers, and technicians, and a wide customer base using their applications and services, *CIMA* can boast comprehensive knowledge in the domain of EWS.

At the start of our collaboration, *CIMA* was undertaking a re-engineering of its main EWS, MyDewetra, to address new requirements and handle the growing volume of data, functionalities, and users worldwide. During this process, we had the opportunity to work closely with the development team, gaining deeper insights into the system, its diverse stakeholders, the operational processes, the limitations encountered, and the re-engineering requirements. After studying MyDewetra and the EWS context, we collaborated to figure out the new technologies and development stack for the re-engineering process, and to shape the final architecture of MyDewetra.

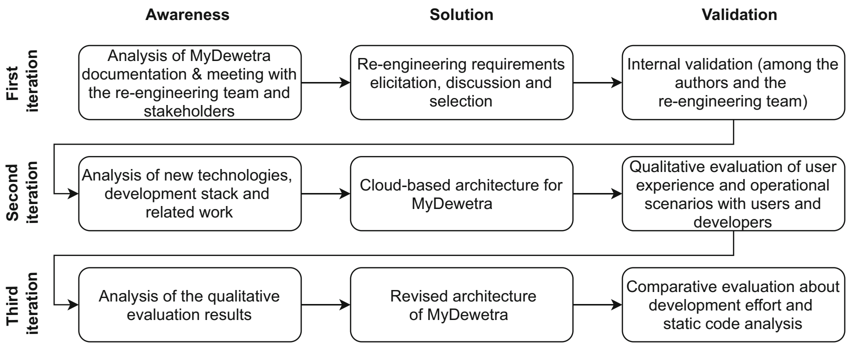


Fig. 1. The design science research methodology

To achieve this, we followed a *design science* [5,20,47] research methodology. Self-explanatory Fig. 1 provides a graphical description of the iterative process. We performed three iterations. The whole process was triggered by the re-engineering needs identified by the *CIMA* Foundation. The elicited re-engineering requirements and a review of new technologies and existing works triggered the second iteration. It resulted in the first version of the cloud-based architecture for MyDewetra, validated through a qualitative evaluation by users and developers. The third iteration allowed for the architecture refinement and a comparative evaluation with the previous version of the system.

4 My Dewetra Case Study

This section provides an overview of the MyDewetra system (Sect. 4.1) and outlines the re-engineering requirements (Sect. 4.2).

4.1 System Overview

The MyDewetra system[3] is an integrated EWS used by Italy's Civil Protection Department for real-time monitoring, forecasting, and prevention of natural hazards like floods, landslides, and wildfires. It synthesizes and integrates data from various sources, including satellite information and in-situ monitoring, to assess and predict risk scenarios and their evolution. MyDewetra is an information system and decision support tool for creating the national criticality/alert bulletin, summarizing regional assessments of weather phenomena and their impacts. It is accessible to all regional Civil Protection Service components and international bodies, based on assessment and compliance with its purpose.

A global version of the system[4], namely MyDewetra **world**, is available to national meteorological and hydrological services, disaster risk management authorities, research organizations, and other international entities involved in risk reduction and management. This version supports the Sendai Framework for Disaster Risk Reduction [44] by enabling real-time analysis and risk information exchange among practitioners. To address different operational and institutional needs, the system is available in four versions: **Standard**, **Customized**, **Pro**, and **Ad-hoc**. The **Standard** version provides access to global datasets and represents the base configuration of the system. The **Customized** version extends the Standard version by including additional static and dynamic datasets, the integration of the MyDewetra additional application (e.g., Warning Bulletins), a customized dashboard, support for displaying institutional logos after authentication, and the implementation of new functions for data exploration. The **Pro** version includes all features of the Customized version and it enables the integration of customized web interfaces and the addition of new services within MyDewetra. The **Ad-hoc** version provides an independent installation designed to meet specific operational requirements and includes the same integration and customization options available in the PRO version.

Table 1 provides an overview of the geographical domains and international projects associated with each version of the MyDewetra *world* system. The Standard version has been adopted in several countries in Europe, Africa, Asia and Oceania, including the PPRD S3 (Algeria, Egypt, Israel, Jordan, Lebanon, Morocco, Palestine, Tunisia) and WMO Volta (Benin, Burkina Faso, Ivory Coast, Ghana, Mali, Togo) domains. The Customized version has been implemented in various countries in Africa, Asia and South America in initiatives promoted by organizations including UNDRR and WMO. The Ad-hoc version has been implemented in several countries in America, within the context of projects managed by institutions such as CIMH and the World Bank.

[3] https://www.mydewetra.org/.
[4] https://www.infomydewetra.world/.

Table 1. myDEWETRA.world: domains and projects

Version	Domains	Projects
Standard	Albania, Angola, Armenia, Azerbaijan, Bosnia Herzegovina, Georgia, PPRD S3, Kosovo, Libya, Macedonia, WMO Volta, Moldova, Montenegro, Rwanda, Serbia, Solomon Islands, Tunisia, Turkey, Ukraine, Zambia	ProNEWS [38], PPRD-EAST [37], Volta [49]
Customized	Africa, Burundi, Cambodia, Ethiopia, Kenya, Laos, Lebanon, Mozambique, Namibia, Peru, Somalia, Sudan, Tanzania, Uganda	UNDRR-IGAD [48], ACMAD [1], WMO [50], UNDRR [24], CNRS-L [30], Ready2Act [13], Wirwina [46], AICS [12], UNDRR-NOE [23]
Pro	Malawi	UNDRR [24]
Ad-hoc	Barbados, Belize, Guyana, Paraguay, Saint-Lucia	CIMH [6], Wirwina [46], WB [18]

The national and global platforms integrate modular applications to provide data and services tailored to each end-user. For instance, one application monitors meteo-hydrogeological risks, while another supports forest fire warning. Both platforms offer continuously updated, high-resolution information to monitor weather events, create risk scenarios, and assess the impact on communities and infrastructure.

Spatial and geospatial data can be visualized as geo-referenced layers in both static and dynamic modes. The applications allow users to view station values, advanced observations, and analyze current or past events using interactive tools. Both platforms continuously update relational and geospatial databases, collecting data from diverse sources like station measurements, weather models, and satellite images. The database holds around 22 billion station measurements and 45,000 archived documents for real-time and historical access.

Since its inception in 2010, MyDewetra has prioritized user needs in its evolution. Initially built on a classic *client-server* architecture, it transitioned to a *three-tier* system, isolating the data layer and moving the client interface to the web for remote access. Over time, MyDewetra has expanded in data volume, functionality, and user base, both nationally and globally. To meet growing demands and evolving technology, the platform has continuously evolved and adapted with new data, applications, features, and programming advancements.

4.2 Re-Engineering Requirements

In early 2023, *CIMA* conducted a thorough analysis to assess the growing complexity of the data and system, and to evaluate the limitations of the existing architecture. This analysis aimed to strategically plan improvements to enhance the system's efficiency and functionality, going beyond the limitations of the previous version. Specifically, it was decided to organize the system re-engineering in multiple iterations, prioritizing the most challenging requirements. Indeed, as highlighted in the research methodology shown in Fig. 1, multiple requirements were elicited and discussed, with only a subset selected for the first re-engineering iteration. In particular, priority was given to development effectiveness, user experience, maintainability, reliability and security, to lay a solid foundation for subsequent iterations. This paper presents the first re-engineering iteration, which was carried out to address the following requirements:

R1. The system should be evolved to improve *development effectiveness* by enhancing the developer experience, easing the implementation of new features in response to new requirements, and facilitating configuration for new customers. This includes providing streamlined tools and workflows that enable faster and more efficient ways to address customer needs. These enhancements aim to reduce operational costs, improve operational usability, and streamline development tasks.

R2. The system should enhance the *user experience* by providing a more intuitive and seamless interface. This will make it easier to perform common tasks, such as uploading time-varying and static data in the relevant applications.

R3. The system should increase its *maintainability* by reducing code smells and technical debt; improve its *reliability* by reducing the number of bugs; and enhance its *security* by decreasing security-sensitive code and minimizing vulnerabilities.

5 The Re-Engineered MyDewetra Architecture

In this section, we illustrate the revised software architecture of MyDewetra, in Fig. 2, resulting from the re-engineering process. The re-engineering process, driven by the need to manage the growing complexity of the previous version, involved re-evaluating the purpose, scope, and integration of functionalities. For instance, one old system application is now part of another, while a feature from an existing application has been promoted to a standalone one. This redefinition enhances workflow efficiency, system's modularity, and overall user experience.

Users access the system using an identity provider, such as Keycloak [21], to authenticate and gain access to the applications for which they are authorized. Each application can store specific authorization information in the system via the `Portal service`, which can be retrieved through dedicated APIs. An application consists of a front-end (App_i FE in Fig. 2), developed using Angular [17], and the related back-end (App_i BE in Fig. 2) service. Front-end applications communicate with the related back-end services using an `API Gateway`. The `DDS service` (MyDewetra Data System) is implemented through a plugin of the Open-Source software Geoserver [15] that extends its rest API by adding the management of time-varying layer publication and by providing native data.

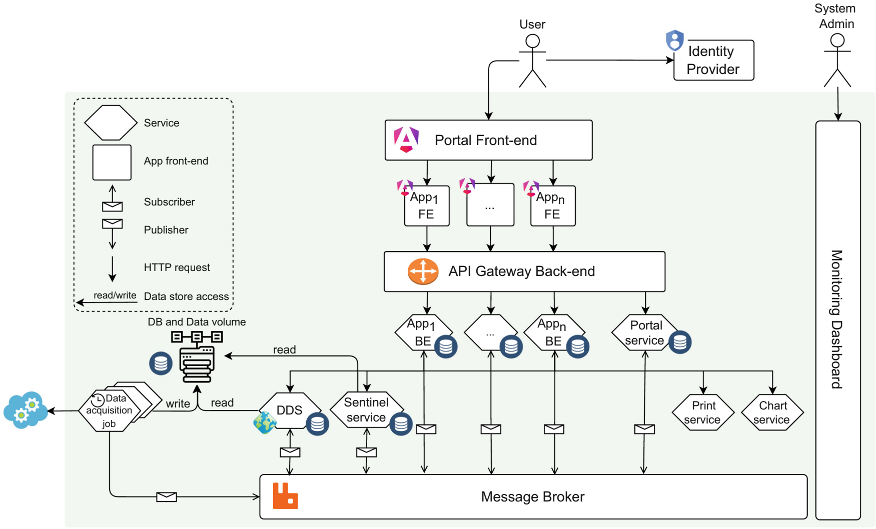


Fig. 2. MyDewetra high-level architecture

Each layer displayed in the map-controls of all applications is a Web Map Service (OGC standard) layer provided by the DDS, which is responsible for transforming the specific format of each data into the tiles displayed on the map. The `Sentinel service` monitors threshold exceedances defined on the values observed by the weather station sensors. It is implemented using Java and the Spring ecosystem [33]. The `Print service`, implemented using Node.js, and `Chart service` are used by other back-end services to generate a PDF report or a chart. The other back-end services are implemented using Python and the Django framework [7]. A `Data acquisition job` is a scheduled service that collects data from external sources and sensors, inserts this information into the appropriate database, and notifies all subscribed services through the `Message Broker`. The message broker, implemented using RabbitMQ [29], is also used to notify back-end services of updated data. All services generate operating metrics in Prometheus [28] format, collected and monitored using the `Monitoring dashboard`. The source code of MyDewetra is open-source under the EUPL license[5] and available upon request, either as source code or as an executable.

In the following, we present one of the most significant operational scenarios of MyDewetra. Figure 3 reports a data acquisition job scenario from a rain gauge sensor using a UML sequence diagram. The acquisition job process starts by receiving data from a rain gauge sensor. Then, the process inserts the raw data into a PostgreSQL database and publishes a message to the related topic in the message broker. The subscribed Sentinel service receives the new rain gauge data and updates the last six hours rain gauge's observations extracted from the database. If the sum exceeds the threshold, the Sentinel service sends a message

[5] https://eupl.eu/1.2/en.

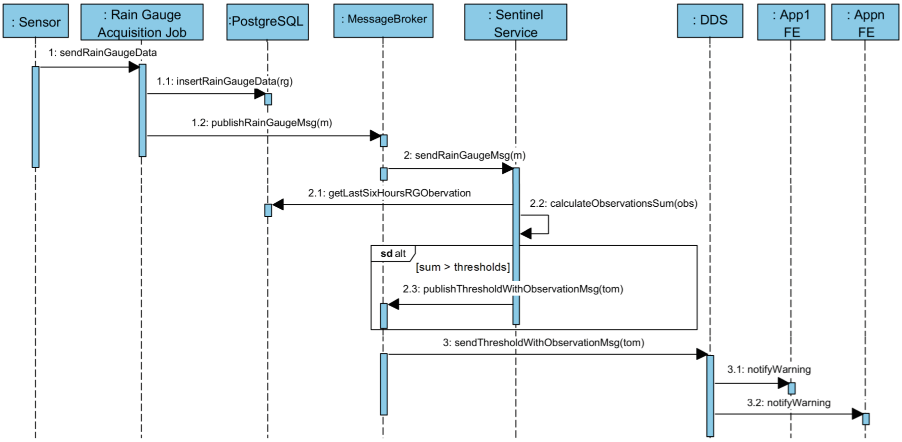


Fig. 3. Rain Gauge Acquisition Job Scenario

to the *exceeding thresholds on sensors* topic. All subscribed services on the topic can read the message and act accordingly. For example, the DDS service reacts notifying all connected users the status update of the rainfall warning layer.

6 Evaluation and Results

In this section, we present the results of a qualitative and comparative evaluation of MyDewetra, demonstrating how the system meets the re-engineering requirements [**Ri**] outlined in Sect. 4.

6.1 Qualitative Evaluation

Methodology. We performed the evaluation using a questionnaire addressed to stakeholders of MyDewetra, with different roles, expertise, and experience. To avoid conflicts of interest with the research outcomes, we selected only people who did not participate in the re-engineering activity of the MyDewetra system, but who are familiar with the system, including its previous and new versions. The questionnaire is structured as follows: (*i*) Introductory questions to collect roles, years of experience and expertise in projects for developing/using EWS (**Q1**–**Q4**). (*ii*) Questions to assess the user experience of the MyDewetra system, w.r.t. the previous version of the system (**Q5**–**Q7**), according to some of the factors influencing user experience [10], namely utility, usability, and findability. (*iii*) Questions to assess the operational scenarios of MyDewetra and the developer experience, w.r.t. the previous version of the system (**Q9**–**Q17**). (*iv*) Questions about the need of re-engineering MyDewetra and for collecting final feedback (**Q18**–**Q21**). In addition, **Q8** assessed the participants' level of knowledge about MyDewetra and was used to direct them only to questions they could answer. The questionnaire and the anonymized responses are available online[6].

Table 2 shows anonymized information about the participants in the questionnaire, their role and experience, and information on their expertise in projects for developing/using EWS. Overall, 10 out of the 20 participants are developers and technicians. Half of the respondents reported having more than 10 years of experience in their role, and the majority of the remaining respondents have more than 2 years of experience.

Table 2. Overview of the experts who evaluated MyDewetra

Job Title	Work Experience	Development of EWS	Use of EWS
Researcher	>10 years	>8 years	>8 years
Technician	>10 years	>8 years	>8 years
Graphic design and communication	8 – 10 years	2 – 5 years	2 – 5 years
Developer	5 – 8 years	>8 years	>8 years
Researcher	2 – 5 years	2 – 5 years	5 – 8 years
Researcher	8 – 10 years	<years	<2 years
Communication staff	<2 years	<2 years	<2 years
Researcher	5 – 8 years	<2 years	2 – 5 years
Researcher	2 – 5 years	2 – 5 years	5 – 8 years
Developer	2 – 5 years	<2 years	<2 years
Researcher	>10 years	2 – 5 years	>8 years
Technician	>10 years	5 – 8 years	<2 years
Developer	>10 years	2 – 5 years	2 – 5 years
Developer	>10 years	>8 years	>8 years
Developer	>10 years	>8 years	>8 years
Developer	2 – 5 years	2 – 5 years	2 – 5 years
Communication staff	2 – 5 years	<2 years	<2 years
Developer	>10 years	>8 years	2 – 5 years
Researcher	>10 years	>8 years	>8 years
Technician	>10 years	>8 years	>8 years

Results of the Evaluation. The summary of participants' responses to questions Q5-Q7, assessing the user experience of the MyDewetra system compared to its previous version, is shown in Fig. 4. Overall, concerning the *utility* of the system, intended as its effectiveness and efficiency in helping users achieve their goals through visual interactions (**Q5**), 15 out of 20 participants think that it is better/strongly better than before. At the same time, 5 out of 20 remain neutral, and none report a negative evaluation. The *usability* of the system,

6 https://github.com/gssi/caise2025-cima.git.

intended as its ease of use, allowing users to navigate and operate the system with minimal difficulty (**Q6**), has been overall positively evaluated. However, 3 out of 20 participants find it more difficult to use. Similarly, the *findability*, defined as the ease with which the information and features available on the new system can be found (**Q7**), has been positively assessed. Again, 3 out of 20 participants find it more difficult to find the needed features. Based on this assessment, we can conclude that the user experience has been significantly improved, and the new system successfully meets the requirement [**R2**].

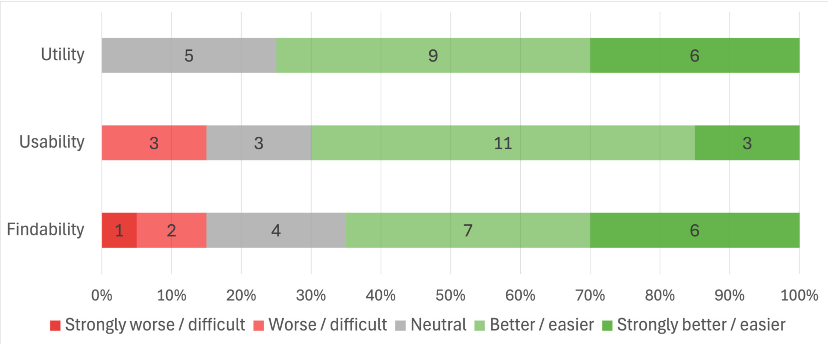


Fig. 4. Assessment of the user experience of the MyDewetra system

With respect to the operational scenarios of MyDewetra, we asked specific questions concerning the complexity of performing three key scenarios expected to benefit from the system's re-engineering. Participants were also asked to estimate the time required to complete each scenario and to explain the reasons behind their assessments. The complexity of adding a new function to an existing application was positively assessed (easier/strongly easier) by the four participants eligible to answer questions **Q9**–**Q11**. They attributed this to the system's modular architecture and the availability of new functionalities and libraries. The estimated implementation time is 1–3 days.

The complexity of initializing a front-end for a new application within the system upon stakeholder request was positively assessed (neutral/easier/strongly easier) by the four participants eligible to answer questions **Q12**–**Q14**. They attributed their assessment to the continuous deployment process and the availability of numerous ready-to-use modules. No agreement emerged regarding the time required to perform the task.

The complexity of loading new time-varying and static data into one of the new system's applications was overall positively assessed (neutral/easier/strongly easier) by the ten participants eligible to answer questions **Q15**–**Q17**, although two of them found it more difficult. Participants attributed their assessment to improvements in screen layout, icons, and menus. Some also noted minor graphical issues (e.g., lack of error messages, icon size), which are typical of a staging phase and can be easily fixed. On average, they estimated less than three days to perform the task. This assessment indicates a significant improvement in development effectiveness, showing that the new system meets requirement [**R1**].

Lastly, we asked participants whether re-engineering the system was necessary and why (**Q18**–**Q21**). The majority, 13 out of 20, considered it necessary for several reasons: improving security, improving response time (especially when loading complex data), ensuring compliance with new cloud technologies, improving dynamic data visualization, addressing the complexity of the previous version, and increasing maintainability. Only two participants felt that re-engineering was unnecessary: one found the previous version more intuitive, while the other noted that, despite the system's widespread use and the redesign of both front-end and back-end, a training course is now required, with minimal gains in usability and usefulness. Additional comments are available online.

6.2 Comparative Evaluation

We conducted a comparative evaluation based on (*i*) the development effort required to implement new functionalities and (*ii*) static code analysis. This evaluation further assesses how effectively the re-engineered system meets **R1**, and also addresses **R3**.

Development Effort. The development effort was measured using data collected from the project management system, tracking development activities, e.g., implementation timelines and task completion. Using this data, we conducted a comparative analysis to assess the impact of the re-engineering process on the development. We selected three functionalities-*bulletin modification after midnight*, *print*, and *myCLOUD*-and compared the development effort (in man-hours) between the previous and re-engineered versions. The analysis involved the same team of developers, ensuring consistent expertise and familiarity with the technologies in both versions. Figure 5 provides an overview of the comparison results, followed by detailed information on the three functionalities.

The system generates criticality bulletins, official documents from the Civil Protection Department with hydrogeological, hydraulic, and meteorological forecasts for the current day (D0) and the next day (D1). Compiled by regional centers and validated by the national Central Functional Center, these bulletins are essential for risk management and public safety. In some cases, sudden and unexpected changes in meteorological conditions require updates to the criticality levels for the following day (D1). To address this need, the system includes the **modification after midnight** functionality, which allows users to modify the D1 forecast of an issued bulletin. For example, if a change is needed at 2 a.m. on D1, the user can update that day's forecast while keeping the original content for D0, ensuring accurate and responsive forecasts for evolving weather. The development time for this functionality in the previous system version required 7 d (56 man–hour), whereas, in the re-engineered version, it required 4 d (32 man–hour), with a reduction of approximately 43%.

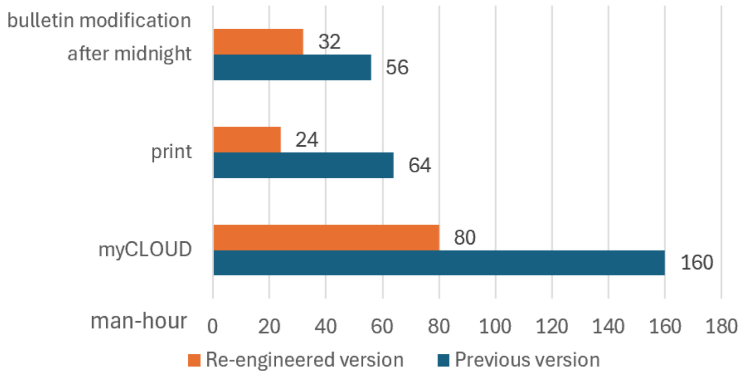


Fig. 5. Development effort measurement

The **print** functionality is used to generate criticality maps that are included in PDF documents, such as the "National Criticality Bulletin" and "Regional Criticality Bulletins". Users define the maps interactively on the client side using the web application's map control, while the backend processes image generation and PDF creation via a REST API call. The implementation of the print functionality in the previous system version relied on PhantomJS, a headless browser affected by several critical limitations, such as unreliable asynchronous resource loading, frequent timeout errors, poor support for DOM events, reliance on an outdated WebKit engine, and the absence of active maintenance or updates. In the new version of the system, the print functionality was re-engineered as a dedicated service that uses REST APIs to generate images of criticality maps from JSON data. The service returns a reference to the image, enabling easy download and integration into PDF bulletins. The development time for the print functionality in the previous version of the system was 8 d (64 man–hour), while in the re-engineered version, it was reduced to less than 3 d (24 man–hour), resulting in a reduction of 62.5%.

myCLOUD is a functionality for archiving and sharing images as attachments. In the previous system version, it was an internal functionality of the DATALOG application, limiting integration with other applications. In the new version, myCLOUD has been re-engineered as a standalone app with REST APIs, enabling document and image uploads accessible across all applications and significantly improving integration. In terms of development time, the implementation of myCLOUD in the previous system version took one month (160 man–hour), whereas, in the new version of the system, the development time was significantly reduced, with implementation completed in two weeks (80 man–hour), with a reduction of 50%.

Overall, this comparative evaluation confirms the improvement in development effectiveness, as already highlighted by the qualitative evaluation, demonstrating that the new system successfully meets requirement [**R1**].

Static-Code Analysis. The MyDewetra code was evaluated using metrics [32] obtained through a static code analysis of both system versions conducted with SonarQube [31]. This analysis focused on assessing improvements in maintainability by reducing code smells and technical debt, enhancements in reliability by minimizing the number of bugs, and advancements in security by decreasing security-sensitive code and minimizing vulnerabilities.

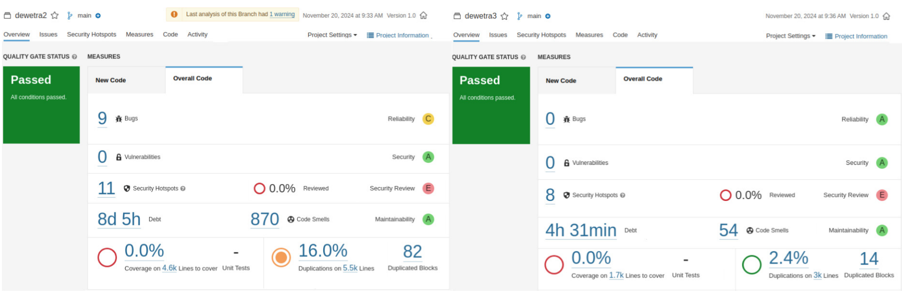


Fig. 6. SonarQube analysis

Figure 6 presents the results of the code analysis. On the left side are the measures obtained from the previous system, whereas on the right are the measures obtained from the re-engineered version. Regarding reliability, the re-engineered version achieved a significant improvement, moving from a C rating to an A rating, together with a reduction in the number of bugs from 9 to 0. Concerning maintainability, although both versions have an A rating, the re-engineered system showed a substantial decrease in code smells, from 870 to 54 (a reduction of approximately 94%), and a corresponding reduction in technical debt, from 8 d and 5 h to 4 h and 31 min (a reduction of around 98%). Relating to security, both versions achieved an A rating with 0 vulnerabilities. However, for the security review, both versions maintained an E rating, although the number of security hotspots (i.e., security-sensitive of code) in the re-engineered system was reduced from 11 to 8.

Overall, although the re-engineered version still has some security hotspots and continues to experience technical debt, it demonstrates significant improvements in terms of reliability, maintainability, and security. Thus, we can conclude the new system meets the requirement [**R3**].

7 Lessons Learned and Future Work

This section presents the lessons learned from the MyDewetra system re-engineering. They capture key insights from the comparative and qualitative evaluations, addressing a complementary aspect of the re-engineering process and reflecting how specific needs led to design decisions and observable outcomes.

Architectural Decisions Driven by Development Needs. Architectural decisions were based on development needs such as the implementation of new features and the simplification of system configuration for new users. This approach led us to prioritize pragmatic solutions: modularizing the architecture to separate concerns and responsibilities, integrating open-source components familiar to the team, and designing service boundaries to reflect actual development workflows. These choices aligned with the practices and resources of the team, making the architecture not only technically sound but also sustainable in the long term. These benefits became evident in the comparative evaluation that showed reduced development effort and improved maintainability, and in the qualitative evaluation that highlighted better developer (Q9-Q17) and user experience (Q5-Q7). This confirmed that aligning architectural decisions with development needs resulted in measurable improvements and minimized rework.

Modularization Laid the Foundation for Future Evolution. The modular structure introduced in the re-engineered system was designed to address development effectiveness (R1), user experience (R2), and aspects of maintainability, reliability, and security (R3). Components were developed with a clear separation of responsibilities and consistent interfaces, allowing for streamlined development and improved internal structure. The comparative evaluation confirmed reduced development effort and improved code quality, while the qualitative evaluation highlighted improvements in user experience (Q5-Q7). This approach has laid a solid foundation for future reuse, customization, and system evolution, including the planned migration of MyDewetra world to the new architecture.

These lessons will serve as a foundation for the next steps of the re-engineering process and the development of a more general reference architecture for EWS. As future work, we plan to: (i) define such a reference architecture; (ii) identify any further needs requiring additional re-engineering iterations; and (iii) incorporate AI, e.g., machine learning techniques for data analysis, leveraging the extensive data available, e.g., to build dynamic predictors for hazards.

Acknowledgements. This work has been partially funded by (a) by the MUR (Italy) Department of Excellence 2023 - 2027 for GSSI, (b) by the European Union - NextGenerationEU under the Italian Ministry of University and Research (MUR) National Innovation Ecosystem grant ECS00000041 - VITALITY - CUP: D13C21000430001, (c) by the European Union - NextGenerationEU under the Italian Ministry of University and Research (MUR) National Innovation Ecosystem, grant PE0000020 - CHANGES - CUP: D53C22002560006, (d) by the European HORIZON-KDT-JU research project MATISSE "Model-based engineering of Digital Twins for early verification and validation of Industrial Systems", HORIZON-KDT-JU-2023-2-RIA, Proposal number: 101140216-2, KDT232RIA_00017, (e) by the ALLARMETEO research project with the Agenzia di Protezione Civile Regione Abruzzo (prot. GSSI n. 6351/2022).

References

1. ACMAD: ACMAD - African Centre of Meteorological Applications for Development. https://acmad.org/
2. Agbehadji, I.E., Mabhaudhi, T., Botai, J., Masinde, M.: A systematic review of existing early warning systems' challenges and opportunities in cloud computing early warning systems. Climate **11**(9) (2023). https://doi.org/10.3390/cli11090188, https://www.mdpi.com/2225-1154/11/9/188
3. Akanbi, A.: Towards a microservice-based middleware for a multi-hazard early warning system. In: Masinde, M., Bagula, A. (eds.) Emerging Technologies for Developing Countries, pp. 179–191. Springer Nature Switzerland (2023). https://doi.org/10.1007/978-3-031-35883-8_12
4. Bai, D., Tang, J., Lu, G., Zhu, Z., Liu, T., Fang, J.: The design and application of landslide monitoring and early warning system based on microservice architecture. Geomat. Nat. Haz. Risk **11**(1), 928–948 (2020)
5. Baskerville, R.L., Baiyere, A., Gregor, S., Hevner, A.R., Rossi, M.: Design science research contributions: finding a balance between artifact and theory. J. Assoc. Inf. Syst. **19**(5) (2018).https://doi.org/10.17705/1JAIS.00495
6. CIMH: CIMH - Caribbean Institute for Meteorology and Hydrology. https://www.cimh.edu.bb
7. Django: The Web Framework for Perfectionists with Deadlines. https://www.djangoproject.com/
8. Esposito, M., Palma, L., Belli, A., Sabbatini, L., Pierleoni, P.: Recent advances in internet of things solutions for early warning systems: a review. Sensors **22**(6) (2022). https://doi.org/10.3390/s22062124
9. European Climate Adaptation Platform Climate-ADAPT: Building railway transport resilience to alpine hazards in Austria. https://climate-adapt.eea.europa.eu/en/metadata/case-studies/building-railway-transport-resilience-to-alpine-hazards-in-austria
10. Eva Youngsin Han: Develop an inclusive UX evaluation framework. B2C e-commerce heuristic metrics (EHM). https://hdl.handle.net/10589/149236 (2019)
11. Expert Group on Forest Fires: European forest fire information system. https://forest-fire.emergency.copernicus.eu/
12. Foundation, C.R.: Programme for a continental coordination, early warning and action system in africa. https://www.cimafoundation.org/en/project/programme-for-a-continental-coordination-early-warning-and-action-system-in-africa/
13. Foundation, C.R.: Ready2Act: enhancing coordination mechanisms and disaster preparedness capacities. https://www.cimafoundation.org/en/project/ready2act/
14. Foundation, T.L.: Kubernetes. https://kubernetes.io/
15. GeoServer: Server for sharing geospatial data. https://geoserver.org/
16. Global Center on Adaptation: The global commission on adaptation. https://gca.org/about-us/the-global-commission-on-adaptation/
17. Google: Angular. https://angular.dev/
18. Group, W.W.B.: WB - World Bank Group. https://www.worldbank.org/ext/en/home
19. Gupta, S., Modgil, S., Kumar, A., Sivarajah, U., Irani, Z.: Artificial intelligence and cloud-based collaborative platforms for managing disaster, extreme weather and emergency operations. Int. J. Prod. Econ. **254**, 108642 (2022). https://www.sciencedirect.com/science/article/pii/S0925527322002249?via%3Dihub

20. Hevner, A.R., Storey, V.C.: Externalities of design science research: preparation for project success. In: Chandra Kruse, L., Seidel, S., Hausvik, G.I. (eds.) DESRIST 2021. LNCS, vol. 12807, pp. 118–130. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-82405-1_14
21. Keycloak: Open Source Identity and Access Management. https://www.keycloak.org/
22. NASA: Nasa analysis confirms 2023 as warmest year on record. https://www.nasa.gov/news-release/nasa-analysis-confirms-2023-as-warmest-year-on-record/
23. Nations, U.: Noe - Africa network of centres of excellence for disaster risk reduction. https://www.undrr.org/news/launch-drr-network-centers-excellence-africa
24. Nations, U.: UNDRR - united nations office for disaster risk reduction. https://www.undrr.org/
25. Network of European Meteorological Services: MeteoAlarm. https://meteoalarm.org/en/live/
26. Pierleoni, P., Belli, A., Esposito, M., Concetti, R., Palma, L.: Earthquake early warning services based on very low-cost internet of things devices. In: 2022 61st FITCE International Congress Future Telecommunications: Infrastructure and Sustainability (FITCE), pp. 1–5 (2022). https://doi.org/10.23919/FITCE56290.2022.9934792
27. Pillai, A.S., Chandraprasad, G.S., Khwaja, A.S., Anpalagan, A.: A service oriented IoT architecture for disaster preparedness and forecasting system. Internet Things **14**, 100076 (2021). https://doi.org/10.1016/J.IOT.2019.100076
28. Prometheus: From metrics to insight. https://prometheus.io/
29. RabbitMQ: One broker to queue them all. https://www.rabbitmq.com/
30. for Scientific Research, N.C.: CNRS l - national council for scientific research - Lebanon. https://www.cnrs.edu.lb/english/home
31. SonarQube: Code Quality and Security, Under your Control. https://www.sonarsource.com/products/sonarqube/
32. SonarSource: SonaQube metrics. https://docs.sonarsource.com/sonarqube-server/latest/user-guide/code-metrics/metrics-definition/
33. Spring: VMware Tanzu. https://spring.io/
34. Stefanic, P., et al.: SWITCH WorkBench: a novel approach for the development and deployment of time-critical microservice-based cloud-native applications. Future Gener. Comput. Syst. **99**, 197–212 (2019). https://doi.org/10.1016/J.FUTURE.2019.04.008
35. The United Nations Secretary-General's: Climate action acceleration agenda. https://www.un.org/sites/un2.un.org/files/un_sgs_acceleration_agenda.pdf
36. UN Secretary-General António Guterres: Fossil fuel industry's promises fall short, secretary-general tells 'early warnings for all' presidency event, urging clarity on reaching net-zero emissions by 2050. https://press.un.org/en/2023/sgsm22069.doc.htm
37. Union, E.: PPRD east 3 - preparedness and response to natural and man-made disasters in the eastern partnership countries. https://www.pprdeast3.eu/
38. Union, E.: PRONEWS - Programme for improving national early warning system and flood prevention in Albania. https://euprojects.al/euprojects/pro-news-programme-for-improving-national-early-warning-system-and-flood-prevention-in-albania/
39. United Nations: Early warnings for all. https://earlywarningsforall.org/site/early-warnings-all
40. United Nations: United nations early warning systems. https://www.un.org/en/climatechange/climate-solutions/early-warning-systems

41. United Nations Development Programme: Strengthening climate information and early warning systems for climate resilient development and adaptation to climate change in guinea. https://tinyurl.com/ykka29y5
42. United Nations Development Programme: Strengthening climate information and early warning systems to support climate-resilient development in cambodia. https://tinyurl.com/bdd56wpt
43. United Nations Environment Programme: Climwarn project. https://www.unep.org/fr/node/1260
44. United Nations Office for Disaster Risk Reduction: https://www.undrr.org/publication/sendai-framework-disaster-risk-reduction-2015-2030
45. United Nations Office for Disaster Risk Reduction: Global status of multi-hazard early warning systems 2022. https://www.undrr.org/publication/global-status-multi-hazard-early-warning-systems-2022
46. WeWorld: Wirwina: building resilience to face hydro-metereological hazards in south america. https://www.weworld.it/en/what-we-do/global-projects/wirwina-building-resilience-to-face-hydro-metereological-hazards-in-south-america
47. Wieringa, R.: Design science as nested problem solving. In: Proceedings of the 4th International Conference on Design Science Research in Information Systems and Technology, pp. 1–12 (2009)
48. WMO: IGAD - disaster risk management programme. https://www.icpac.net/our-projects/igads-disaster-risk-management-programme/
49. WMO: VFDM - Volta flood and drought management. https://www.floodmanagement.info/volta-basin/
50. WMO: WMO - world meteorological organization. https://wmo.int/
51. World Meteorological Organization: Climate risk and early warning systems. https://wmo.int/activities/climate-risk-and-early-warning-systems-crews
52. World Meteorological Organization: Early warnings for all. https://earlywarningsforall.org/site/early-warnings-all

Chatbots and Social Networks

LLMs to Replace Crowdsourcing in Generating Syntactically Diverse Paraphrases for Task-Oriented Chatbots

Auday Berro[1(✉)], Vitor Gaboardi dos Santos[2], Boualem Benatallah[3], and Khalid Benabdeslem[1]

[1] Université Claude Bernard Lyon 1, LIRIS UMR 5205, Lyon, France
{auday.berro,khalid.benabdeslem}@univ-lyon1.fr
[2] School of Computing, Dublin City University (DCU), Dublin, Ireland
vitor.gaboardidossantos2@mail.dcu.ie
[3] Insight SFI Research Center on Data Analytics, Dublin City University, Dublin, Ireland
boualem.benatallah@dcu.ie

Abstract. Task-oriented chatbots rely on extensive and diverse dataset of user utterances to recognize tasks and intents effectively. Traditionally, these datasets are created through crowdsourcing, where crowd workers expand an initial set of seed utterances through paraphrasing. Although effective, crowdsourcing techniques have disadvantages, including high cost, time-consuming, low output quality, and often focusing only on lexical diversity. The emergence of Large Language Models (LLMs) presents a promising alternative for generating high-quality and diverse paraphrases more efficiently. In this paper, we investigate whether LLM can replace the crowd in generating syntactically diverse paraphrases to train chatbots. We replicate an existing crowdsourcing workflow using GPT, maintaining similar scale, prompts, and data. We evaluate the data across three dimensions: (1) comparing paraphrases generated by crowd workers and GPT; (2) examining the impact of changing the number of paraphrases per request; and (3) assessing performance across different prompt strategies. Our findings reveal that GPT generated paraphrases with greater syntactical diversity and semantic relevance while resulting in a 98% cost reduction compared with crowdsourcing.

Keywords: Paraphrasing · LLMs · GPT · Syntax diversity · Task-oriented chatbots · Crowdsourcing

1 Introduction

Task-oriented chatbots are designed to engage with users in natural language to complete specific tasks [24]. Understanding a user utterance involves steps like intent recognition and slot filling. For example, in the utterance "*Find restaurants serving Lebanese food in Paris*", the chatbot must identify the intent (e.g., "*find-Restaurant*") and recognise the associated slots (e.g., "*cuisine*" = "*Lebanese*",

J. Krogstie et al. (Eds.): CAiSE 2025, LNCS 15701, pp. 145–162, 2025.
https://doi.org/10.1007/978-3-031-94569-4_9

"*location*" ="*Paris*"). Since human language is diverse, the same request can be phrased in multiple ways. Therefore, it is crucial to use a variety of utterances to train chatbots that handle different expressions of the same request, improving their robustness [28].

Leveraging a high-quality dataset of utterances to train chatbots can be achieved through *paraphrasing*, which reformulates an initial set of seed utterances into multiple variations while preserving their meaning [23]. Paraphrase techniques mostly focus on incorporating specific *lexical* variations (such as synonyms replacements) or *syntactic* variations (structural changes) into the seed utterances, while still maintaining *semantic similarity* to the original utterance [29]. For instance, given the previous example as a seed utterance, "*Search restaurants offering Lebanese cuisine in Paris*" introduces lexical variations, and "*List the restaurants in Paris that serve Lebanese food*" gives syntactical alternative. Both variations retain the same intent of *finding a restaurant*, making them semantically relevant to the seed utterance. Failing to include a diverse and semantically relevant set of utterances can reduce the chatbot's effectiveness and negatively affect the user experience [32].

Crowdsourcing is a well-established approach for generating paraphrases [32] [30]. In this process, a seed utterance, provided by an expert, is presented as the starting point and workers paraphrase it into new variations. By controlling the types of variations, the process yields useful paraphrases for training and testing models [10]. Advancements include iterative multistage workflows involving seed utterances from previous rounds, random sampling [12], introducing semantic outliers [14], and countering priming effects [32]. Efforts to enhance lexical diversity include strategies such as swapping words with images [25], employing taboo words to limit common choices [15], and word recommendations [32]. Despite their effectiveness, crowdsourcing solutions have drawbacks, such as cost, time consumption, and output quality issues [6], besides often focusing mostly on lexical diversity [24].

The recent emergence of Large Language Models (LLMs) (e.g., GPT, Mistral and Falcon) [36] offers significant improvements across many NLP applications, including text classification [35], generation [27], and summarization [7]. In addition, LLMs provide a more scalable alternative to traditional crowdsourcing methods while reducing the manual effort involved in generating benchmarks and paraphrases [6,16]. In paraphrasing tasks, LLM-based approaches demonstrated the ability to generate lexical diverse and semantically relevant paraphrases [5]. These results highlight the potential of using LLMs in augmenting data to improve the performance of downstream NLP tasks.

In this paper, we investigate whether GPT-3.5-turbo[1,2] can replace crowd workers to enhance syntactic diversity when creating datasets used for training and evaluating task-oriented bots. We replicate the syntax-aware multi-stage crowdsourcing workflow of Ramirez et al. [24], but with GPT instead. The workflow guide workers to generate paraphrases that should adhere to target syntax

[1] https://platform.openai.com/docs/models/gpt-3.5-turbo.
[2] Henceforth GPT refers to GPT-3.5-turbo for simplicity.

patterns (*patterns-by-examples*) or avoid frequent patterns (*taboo-patterns*) by analysing syntax patterns of paraphrases from previous rounds. Our study replicates their syntax-aware paraphrasing workflow, including scales, prompts, and seed data. Additionally, we examine the impact of changing the number of paraphrases per request while maintaining the same total number of paraphrases. For instance, we can create 24 paraphrases by either requesting 3 paraphrases per request using 8 requests or requesting 12 paraphrases per request using 2 requests. We investigate the quality of GPT-generated paraphrases in terms of semantic relevance and syntactic diversity through the following research questions:

– *RQ1. What is the impact of varying the number of paraphrases per GPT request?*
– *RQ2. How do paraphrases generated through syntactic-diversity-aware GPT prompt workflows compare to those generated by crowd workers?*
– *RQ3. How does the performance of syntactic-diversity-aware GPT prompting workflows change across different prompt strategies?*

In summary, we found that: (1) increasing the number of paraphrases per request enhances semantic relevance and diversity; (2) GPT outperforms crowdsourcing in semantic relevance and syntactic diversity, reducing costs by 98%; and (3) *patterns-by-examples* and *taboo-patterns* prompt strategies have similar results, with the former showing slightly better performance. We released the dataset, code, and reproduction details of our study[3].

The remainder of this paper is organised as follows: Sect. 2 reviews related work on leveraging LLMs to replace crowd workers in paraphrasing tasks. Section 3 outlines the workflow for generating syntactically diverse paraphrases from seed utterances. Section 4 presents the experimental setup, including the data collection process, evaluation methods and metrics. Section 5 presents the results and discusses the findings. Finally, Sect. 6 discusses the limitations and potential directions for future work and Sect. 7 summarises the conclusion.

2 Related Work

In this section, we review studies that investigate the effectiveness of LLMs in performing tasks traditionally managed by human crowd workers, with a focus on paraphrasing methods or generating utterances for training intent classifiers.

Wu et al. [31] explored whether LLMs can replace human workers in crowdsourcing tasks. They designed an experiment where students replicated a crowdsourcing pipeline using LLMs. They found that LLMs and humans react to instructions in different ways. While LLMs are responsive to adjectives and comparison-based instructions, humans understand instructions with trade-off criteria better.

[3] https://github.com/AudayBerro/CrowdFreeParaphraseLLM.

Cox et al. [9] investigated whether using crowdsourcing instructions as prompts in GPT-4 can generate motivational messages that encourage physical activity. They found that GPT-4 with the crowdsourcing pipeline generated more diverse messages when compared with two baseline GPT prompts, though still less diverse than the messages written by human crowd workers.

Li et al. [17] used ChatGPT to generate utterances that do not belong to any supported intents in task-oriented dialog systems, also known as out-of-scope (OOS) utterances. They used these LLM-generated utterances for training intent classifiers. Their findings show improved classifier robustness in detecting OOS utterances, while being less costly than traditional crowdsourcing methods.

Santos et al. [26] proposed a prompting pipeline using GPT-3.5 to generate lexically diverse paraphrases from utterances created using OpenAPI Specifications. They use the *logit bias* parameter of LLMs to penalise frequently used tokens and encourage word diversity in the generated paraphrases. They found that incorporating these paraphrases into the pool of valid utterances enhanced lexical diversity and increased the performance of mapping utterances to APIs.

Cegin et al. [6] investigated whether LLMs, specifically ChatGPT and Falcon-40B, could replace crowd workers for generating paraphrases in intent classification datasets. They conducted a quasi-replication of the approach from Larson et al. [15], which uses taboo words to improve diversity. Their findings showed that ChatGPT produced more diverse paraphrases than human workers, and models trained on ChatGPT-generated data had comparable robustness with those trained on crowd-generated data. More recently, Cegin et al. [5] examined the lexical diversity and downstream model performance of text generated by LLMs using three established crowdsourcing incentive methods: *taboo words*, where frequent words are identified and workers are instructed to avoid them; *hints*, where previous outlier paraphrases are included as examples in the instructions; and *chaining*, where outliers paraphrases from earlier iterations are selected and used as seed sentences in the subsequent data collection cycle. They found that *taboo words* incentive method had higher lexical diversity while *hints* was the best method for improving model performance.

Most related work focuses on improving lexical diversity when generating paraphrases. Cegin et al. [6] evaluate syntactical diversity, but they do not use methods that explicitly steer the LLM toward sentence-level syntactic variation. Instead, they rely on inserting *taboo words* in the prompt to encourage paraphrase variations. In this work, we explore two prompting strategies specifically designed to steer the model toward generating syntactically diverse sentence variations using GPT. Furthermore, we investigate the impact of changing the number of paraphrases generated per GPT request and compare the performance of the two prompt strategies using multiple diversity metrics.

3 Syntactically Diverse Paraphrase Generation Workflow

In this section, we outline the approach to generate syntactically diverse paraphrases from seed utterances. As summarised in Fig. 1, this approach involves two phases: paraphrase generation and paraphrase verification, repeated over multiple rounds. We replicate the data collection process from Ramirez et al. [24], but instead of crowd workers, we use GPT for paraphrase generation and BERTScore to assess their semantic relevance. Furthermore, we adopt the same definition of syntax pattern considering the top two levels of a constituency parse tree [11].

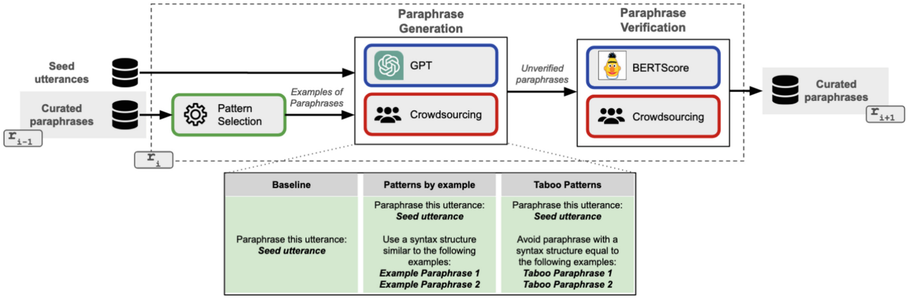


Fig. 1. Approach overview for generating syntactically diverse paraphrases. Boxes highlighted in blue and red represents strategies proposed in this paper and by Ramirez et al. [24], respectively. The prompt descriptions highlight only the main concepts. (Color figure online)

Paraphrase Generation. In each data collection round r, the process begins with a dataset of seed utterances X and a curated collection of paraphrases Y (with Y being empty in the first round). Paraphrases are generated using three specific prompts: **baseline**, **pattern-by-examples**, and **taboo-patterns**. The latter two are innovative prompts proposed by Ramirez et al. [24]. While Ramirez et al. [24] relied on crowdsourcing to produce these paraphrases, we use GPT [4]. The prompts instruct workers to generate a set of m paraphrases for each seed utterance $x \in X$, with guidelines that clarify the concept of paraphrasing.

The novel prompts are designed to guide the generation of new paraphrases by incorporating examples that either *emphasize* or *discourage* specific syntax patterns. These examples are selected by analysing the syntax patterns within the curated dataset Y. The **pattern-by-examples** prompt guides workers in creating paraphrases with patterns similar to examples selected from the least frequent patterns identified in Y from a previous round. The aim is to create more examples with underrepresented patterns. On the other hand, the **taboo-patterns** prompt provides examples of patterns that workers must avoid when creating paraphrases. These taboo patterns are chosen from the most frequent patterns identified in Y from a previous round. The goal is to prevent the generation of paraphrases with overused patterns, thus encouraging syntactical diversity in the dataset.

Paraphrase Verification. The generated paraphrases are then validated to ensure their semantic relevance with the original seed utterance. The goal is to enhance the consistency and quality of paraphrases for more accurate representation. In this step, Ramirez et al. [24] employed a crowdsourcing evaluation task. On the other hand, we leverage BERTScore [34], an automatic metric to measure semantic similarity based on contextual embeddings. In our implementation, we compare the BERTScore (BS) between each generated paraphrase with its corresponding seed utterance. Paraphrases with a BS below 0.5 are considered semantically irrelevant (and therefore discarded), while those with a BS above 0.98 are regarded as duplicates (and also discarded). These threshold values were chosen based on findings from previous research [2,22], which demonstrated their effectiveness. The valid paraphrases are subsequently added to the curated collection Y.

4 Experiments

The experiments are designed to evaluate: (1) the impact of changing the number of paraphrases requested per prompt; (2) whether GPT can effectively replace crowd workers in generating syntactically diverse paraphrases; and (3) the comparative performance of the *patterns-by-example* and *taboo-patterns* prompts in creating such paraphrases. In what follows, we provide details about the data collection process, the evaluation configurations, and metrics used to assess the results.

4.1 Data Collection

We replicated the workflow introduced by Ramirez et al. [24] to generate paraphrases. The process consists of two rounds.

Round 1: Baseline Paraphrase Collection. In the first round, Ramirez et al. [24] generated paraphrases for 51 seed utterances using a baseline prompt that simply requested workers to generate paraphrases. Each seed utterance was given to 8 crowd workers and each worker was asked to provide 3 paraphrases. This resulted in 24 paraphrases per seed utterance and a total of 1,224 paraphrases, which were manually validated for semantic relevance.

In this paper, we replicated the process using GPT with the same 51 seed utterances, generating 24 paraphrases per seed. To mimic the use of 8 crowd workers, we made 8 model calls, each requesting 3 paraphrases. We refer to this first round of obtaining paraphrases using crowdsourcing and GPT as *Crowd-bootstrap* and *GPT-bootstrap*, respectively.

Round 2: Enhanced Paraphrase Collection. In the second round, Ramirez et al. [24] again employed 8 crowd workers, each tasked with creating 3 paraphrases per seed, using the same seed utterances as in the first round. However, in this round, the workers were guided by the pattern-by-examples (*Crowd-P*) and taboo-patterns (*Crowd-T*) prompts with examples taken from Y obtained in the *Crowd-bootstrap* dataset.

In this paper, we performed requests to GPT to create a total of 24 paraphrases per seed utterances using the *pattern-by-example* (GPT-P) and *taboo-patterns* (GPT-T) prompts. Minor adjustments were made to the prompts to prevent unpredictable outputs and hallucinations from LLMs. Following previous studies [3,18,19], we added instructions for LLMs to label unclear inputs as "uncertain" and defined a JSON output format, ensuring consistent data formatting during generation.

Additionally, while maintaining 24 paraphrases per seed utterance, we varied the number of paraphrases requested in each model request (OpenAI API call). For instance, if we asked GPT to generate 3 paraphrases per model request, we would need to send 8 API requests (3 paraphrases per request x 8 requests = 24 paraphrases). Conversely, if we instructed GPT to generate 12 paraphrases per request, we would only need to send 2 API requests (12 paraphrases per request x 2 requests = 24 paraphrases). This approach allowed us to assess the impact of different batch sizes on the quality and diversity of the generated paraphrases. These configurations resulted in six paraphrase datasets for the second around using GPT:

- GPT-P-3, GPT-P-8, GPT-P-12: *pattern-by-examples* prompt, generating 3, 8, and 12 paraphrases per request, respectively.
- GPT-T-3, GPT-T-8, GPT-T-12: *taboo-patterns* prompt, generating 3, 8, and 12 paraphrases per request, respectively.

Finally, we generated paraphrases using the GPT-3.5-turbo model with a temperature of 1.0 to encourage more diverse outputs, and a presence penalty of 1.5 to reduce repetition of tokens based on their appearance in the text. These values are consistent with those used by Cegin et al. [6] when replicating a crowdsourcing study using ChatGPT and Falcon. The remaining parameters used are default values, as specified in the official OpenAI Chat Completions API documentation[4].

4.2 Evaluation

We evaluate semantic relevance and syntactic diversity across ten configurations for generating paraphrases. These include two configurations from the first round: *GPT-bootstrap* and *Crowd-bootstrap*, both using a baseline prompt that requests paraphrases for a given seed utterance. The remaining eight configurations relate to the second round and use curated bootstrap paraphrases from the first round. This includes configurations for both the *patterns-by-example* and *taboo-patterns* prompts. Specifically, there are two configurations based on crowdsourcing - one for each prompt - and six GPT-based configurations, with three variations for each prompt, differing in the number of paraphrases requested per prompt.

[4] https://platform.openai.com/docs/api-reference/chat/create.

4.3 Metrics

To evaluate **semantic relevance**, we use BERTScore [34]. We compute the BERTScore (**BS**) for each generated paraphrase by comparing it to its corresponding seed utterance. We then average these results across all paraphrases to obtain a single value. Higher BS values reflect higher semantic relevance. Furthermore, we compute the number of semantically valid paraphrases (**VP**) after applying the paraphrase verification detailed in Sect. 3, where paraphrases with a BS higher than 0.5 are considered valid. A higher VP indicates a greater number of paraphrases that accurately capture the intended meaning of the original sentence. We also count the number of duplicates (**DP**) within the pool of valid paraphrases, where we compare the BS between all pairwise paraphrases and define as duplicates the ones with BS higher than 0.98. This allows us to assess which configuration tends to create the same paraphrases, influencing diversity in the final training corpus of paraphrases. Corpus with fewer pairwise duplicates imply greater diversity and better performance for chatbots.

We use several metrics to evaluate **diversity**. The Syntax Similarity Mean (**S-mean**) metric, introduced by Chen et al. [8], applies the tree edit distance (**TED**) algorithm between two syntax parse trees, excluding word tokens. We calculate this metric as the mean TED for pairs of paraphrases with the same intent. A higher S-mean value indicates greater syntactic diversity, as it reflects a higher average number of edits required to align the syntax trees of different paraphrases. The Unique Syntax (**US**) metric represents the count of distinct syntax patterns found across all valid paraphrases generated for all seed utterances. A higher US value indicates greater syntactic diversity within the paraphrase dataset. The Syntax Novelty (**SN**) metric measures the number of unique syntax patterns introduced in the current round of paraphrase generation that were not present in the dataset from the previous round. The SN is presented as the median number of new syntax patterns per seed utterance, providing an indication of how many novel syntactic structures were created in the paraphrasing process. Finally, **DIV** [13] analyses n-gram variations among pairs of paraphrases within the same intent.

5 Results

Table 1 presents the metrics used to evaluate semantic relevance and diversity across all ten paraphrase generation configurations. We will refer to this table to discuss the results corresponding to each research question.

5.1 RQ1: What Is the Impact of Varying the Number of Paraphrases per GPT Request?

Semantic Relevance. We found that **increasing the number of paraphrases per request reduced duplicates while keeping the number of valid paraphrases consistent, though it slightly lowered semantic relevance**. Similar trends are observed across both prompt strategies (GPT-P

Table 1. Summary of performance metrics for all experiments conditions.

Dataset	BS	VP	DP	S-Mean	US	SN	DIV
GPT-bootstrap	0.921	1188	146	10.58	230	-	0.745
Crowd-bootstrap	0.635	790	34	11.27	343	-	0.598
Crowd-P	0.648	726	3	11.89	431	5	0.686
Crowd-T	0.640	658	8	12.68	425	5	0.680
GPT-P-3	0.906	1211	81	10.94	246	8	0.810
GPT-P-8	0.895	1220	22	16.92	354	12	0.891
GPT-P-12	0.879	1220	4	18.96	458	15	0.932
GPT-T-3	0.915	1208	119	11.37	232	5	0.782
GPT-T-8	0.899	1213	18	16.21	365	12	0.885
GPT-T-12	0.885	1218	9	18.29	476	14	0.926

and GPT-T). The BS decreases as the number of paraphrases requested in a single request increases. For instance, in the GPT-P series, the BS drops from 0.906 (3 paraphrases) to 0.879 (12 paraphrases). This likely happened because the model creates utterances with higher semantic deviations to meet the quantity demands when prompting for more paraphrases per request. Despite that, VP remains stable across different request sizes, indicating that the model generates semantically valid paraphrases regardless of request size. Furthermore, DP decreases as more paraphrases are requested per prompt. For instance, in the GPT-T configurations, the DP decreases from 119 (3 paraphrases) to 9 (12 paraphrases).

Diversity. We found that **prompting for more paraphrases per request significantly improved diversity and introduced a wider range of novel syntax patterns**. The S-Mean rises notably when prompting for more paraphrases per request, indicating greater syntactic variation across both prompt strategies. For example, in the GPT-P configurations, S-Mean increases from 10.94 (3 paraphrases) to 18.96 (12 paraphrases). Similarly, US and DIV show an upward trend, suggesting that larger batch requests encourage the model to produce more diverse syntax within the same request. This occurred because generating more paraphrases at once prompts the model to explore more syntactic structures and different word combinations. SN also increases with higher number of paraphrases, meaning a greater introduction of novel syntax patterns.

In summary, when a higher number of paraphrases are requested in a single prompt, GPT tends to produce more diverse outputs within the same request. Therefore, it is more effective to request multiple paraphrases within a single prompt, rather than repeating the same prompt multiple times to generate paraphrases.

5.2 RQ2. How Do Paraphrases Generated Through Syntactic-Diversity-Aware GPT Prompt Workflows Compare to Those Generated by Crowd Workers?

Semantic Relevance. In general, we found that **GPT generated more semantically relevant and valid paraphrases, though crowdsourcing resulted in fewer duplicates**. In the first round, GPT-bootstrap (0.921) outperforms Crowd-bootstrap (0.635) in semantic relevance, as measured by BS, suggesting that GPT-generated paraphrases better retain the original meaning of the seed utterances to those generated by crowd workers. Additionally, GPT-Bootstrap produced 1,188 valid paraphrases, whereas Crowd-Bootstrap generated 790. However, GPT-Bootstrap also produced more duplicates (146) compared to Crowd-Bootstrap (34), suggesting that GPT may struggle with introducing sufficient variation between paraphrases. This issue occurred given the GPT-Bootstrap setup, where 8 requests were made, each generating 3 paraphrases. As noted earlier, using multiple requests requesting for fewer paraphrases yield more duplicates.

In the second round, GPT outperforms crowdsourcing in semantic relevance for all configurations. For instance, GPT-P-3 (0.906) and GPT-T-3 (0.915) had higher BS scores compared to Crowd-P (0.648) and Crowd-T (0.640), respectively. In addition, GPT generates more VP than crowdsourcing. These results highlight the superior capacity of GPT to generate paraphrases that match the original meaning of the seed utterances regardless of the prompt strategy. Crowdsourced paraphrases has lower DP compared to GPT-generated ones. While the duplication gap narrows with more paraphrases per request - Crowd-P generated 3 duplicates compared to GPT-P-12's 4, and Crowd-T produced 8 duplicates versus GPT-T-12's 9 - crowdsourcing maintains a slight advantage in reducing duplicates. However, the lower duplication rate in the crowdsourced data is partly given to the smaller number of VPs used for comparison. In contrast, GPT generates a larger number of VPs, leading to more potential duplicates.

Table 2 provides examples of paraphrases generated by crowdsourcing and GPT. Examples 1 and 2 present semantically accurate paraphrases, while Examples 3 and 4 illustrate unusual or odd outputs from both sources.

In Example 1, the crowd adds "*I want to*", emphasizing the speaker's desire while preserving the original meaning. This demonstrates human intuition in paraphrasing, with minimal syntactic variation that remains close to the original utterance. In Example 2, GPT introduces politeness markers and uses a wider range of words. In Example 3, the paraphrase is grammatically incorrect and incomplete since it is missing the verb "*is*". In Example 4, GPT generates a verbose paraphrase that introduces unnecessary details, which changes the original meaning of the seed utterance. Finally, some crowdsourced paraphrases were single words, like "*this*", "*is*", and "*not*". This behaviour is called *cheating* in the literature [33] and does not occur in GPT, since the model always aims to provide a complete and contextually relevant response.

Table 2. Examples of seed utterances (S) and their respective paraphrases (P) generated using Crowd or GPT configurations.

Example	Dataset	Text
1	Crowd	**S**: Book a table for eight people, please. **P**: *I want to book a table for eight people, please.*
2	GPT	**S**: Request a taxi from the airport to home. **P**: *Could you book a taxi to pick me up at the airport and drop me off at home?*
3	Crowd	**S**: How hot is it in Pesotum? **P**: *How hot it Pesotum.*
4	GPT	**S**: I would like to go to a pub that has Italian dressing for a party of 5. **P**: *Hopefully, I could consult with your knowledge base concerning tavern venues serving an accompaniment exquisite sauce native to a nation shaped...*
5	Crowd	**S**: How hot is it in Pesotum? **P**: *In Pesotum, how hot is it?*
6	GPT	**S**: Please, play me some Satire music **P**: *I'm in the mood for Satire music, can you play it for me?*

Diversity. We found that **GPT introduced more novel syntactic structures and unique patterns while also generating more novel syntactic structures compared to previous rounds.**

In the first round, the crowd-bootstrap generates paraphrases with higher syntactic diversity, as indicated by higher S-Mean (11.27 vs. 10.58) and US values (343 vs. 230). The primary reason is that when generating the GPT-bootstrap, we made multiple requests using the same prompt, with each request asking for three paraphrases. This approach led to GPT generating similar syntax structures across these independent calls, resulting in reduced syntactic diversity. In contrast, human workers may naturally introduce more varied syntax through individual differences in interpretation, creativity and background. Nevertheless, GPT-bootstrap demonstrates higher general diversity with a higher DIV score (0.745 vs. 0.598), indicating higher variation in n-gram compositions despite having fewer unique syntactic patterns.

In the second round, GPT outperforms crowdsourcing in all metrics for both strategies where GPT was requested to generate 12 paraphrases per prompt. GPT-P-12 achieves an S-Mean of 18.96, while GPT-T-12 has an S-Mean of 18.29, both higher than Crowd-P's 11.89 and Crowd-T's 12.68. These results indicate that the paraphrases generated by GPT require more syntactic modifications than the paraphrases generated by the crowd. The US metric reinforces this observation. The values were 458 for GPT-P-12 and 476 for GPT-T-12, outperforming Crowd-P's 431 and Crowd-T's 425. This suggests that GPT-generated paraphrases include a higher number of different syntactic patterns than those produced by the crowd. GPT-P-12 achieves an SN of 15, and GPT-T-12 has an SN of 14, compared to just 5 for both Crowd-P and Crowd-T. These results indicate that GPT is more effective at introducing novel syntactic patterns in the second round when compared to crowdsourcing. In addition, GPT showed higher diversity (DIV) across both prompt strategies, indicating that generated paraphrases have higher n-gram variation.

Furthermore, different syntactic patterns occurred exclusively in the GPT or crowdsourced configurations. For instance, Example 5 in Table 2 shows a pattern unique to crowdsourcing not found in GPT. Here, the crowd worker reordered the words of the seed utterance to generate a paraphrase. This reordering is a common technique in crowdsourced paraphrasing tasks. Example 6 shows a pattern exclusive to GPT not found in the crowdsourced paraphrases. Here, GPT generates a paraphrase by incorporating personal context and a polite tone, which results in a novel syntactical pattern.

In summary, GPT demonstrated superior performance across most metrics in the second round, where diversity-aware prompts were employed to generate paraphrases. This leads to paraphrases with a greater syntactic diversity, a higher number of unique patterns, higher semantical relevance, and more valid paraphrases when compared to paraphrases generated using crowdsourcing.

5.3 RQ3. How Does the Performance of Syntactic-Diversity-Aware GPT Prompting Workflows Change Across Different Prompt Strategies?

Semantic Relevance. We found that **performance across both prompt strategies were similar overall, with *patterns-by-example* producing more valid paraphrases, fewer duplicates and *taboo-patterns* achieving higher semantic relevance**. GPT-P-12 achieves a BS of 0.879, while GPT-T-12 has a slightly higher BS of 0.885. Although the difference is modest, it indicates that GPT-T-12 has a marginally better semantic relevance in its paraphrases compared to GPT-P-12. Both GPT-P-12 and GPT-T-12 generated a high number of valid paraphrases, with GPT-P-12 generating 1220 valid paraphrases and GPT-T-12 generating 1218 valid paraphrases. The DP score is lower for GPT-P-12, which generated 4 duplicates, compared to GPT-T-12, which generated 9 duplicates.

Diversity. We found that **both prompts performed similarly in terms of diversity. However, *patterns-by-example* offered greater syntactic diversity and novel syntax patterns, while *taboo-patterns* produced a higher number of unique patterns.** The S-Mean for GPT-P-12 is 18.96, while GPT-T-12 is slightly lower at 18.29. This indicates that GPT-P-12 exhibits marginally higher syntactic diversity compared to GPT-T-12. In terms of US score, GPT-P-12 scores 458, surpassing GPT-T-12's score of 476. This difference indicates that GPT-T-12 introduces a slightly higher syntax variation compared to GPT-P-12. Furthermore, SN and DIV metrics are similar in both configurations, with *patterns-by-examples* achieving slightly higher values than *taboo-patterns* prompts.

In summary, both prompts performed similarly when using GPT, though *patterns-by-example* offered slightly higher syntactic diversity and semantic relevance compared to the *taboo-patterns* prompt.

5.4 Cost Considerations

We generated 8,568 paraphrases for the 51 seed utterances across 7 configurations using GPT. Each configuration comprises 1,224 paraphrases with a total cost of $7.15 using the OpenAI API[5]. This represents a significant cost reduction compared to the $428.40 for crowdsourcing, based on the rate of $0.15 per 3 paraphrases (as reported by Ramirez et al. [24]). The 98.3% cost reduction highlights the significant cost-savings of employing GPT over crowdsourcing.

6 Limitations and Future Work

In this section, we present future directions to address some limitations of our study. We found four main limitations: relying only on prompts originally tailored for crowdsourcing tasks; generating paraphrases only with GPT-3.5-turbo model; creating paraphrases that may not adhere to constraints imposed by chatbots; and a lack of simulation of multiple crowd workers with diverse backgrounds.

6.1 Designing Syntax-Aware Paraphrase Generation Prompts

A primary limitation of our study was the reuse of prompts and instructions originally tailored for crowdsourcing tasks. To fully harness the capabilities of LLMs, one step involves designing prompts specifically suited for LLMs, rather than adapting existing prompts from disparate crowdsourcing tasks. It is well established that the performance and efficacy of LLMs are profoundly influenced by prompts design [37]. Small variations in the prompt can lead to substantial differences in model output [20,21]. We plan to introduce new prompts tailored specifically to generate syntactically guided paraphrases using LLMs rather than adapting and improving existing prompts initially optimized for crowdsourcing.

6.2 Paraphrasing Using Different LLMs

Another limitation of this study is that it evaluated only GPT-3.5-turbo, and did not consider other LLMs. We plan to evaluate syntax diversity-aware workflows using other LLMs such as GPT-4, Falcon-LLM, Mistral, and Llama [36]. By

[5] The cost was subtracted from the API usage summary on our personal OpenAI account page at https://platform.openai.com/account/usage.

applying the same workflow and methodology to these models, we aim to provide a comparative analysis that highlights performance variations in syntactically controlled paraphrase generation tasks. This provides valuable information on the strengths and weaknesses of different LLMs in handling syntactic diversity.

6.3 Constraint-Aware Paraphrasing

A third limitation of this study is that the generated paraphrases may not adhere to the constraints imposed by the interfaces used by chatbots to fulfil user tasks (e.g., required or optional API parameters). For instance, when paraphrasing a seed utterance that requests to convert currencies, the paraphrase must still mention the source and target currencies and the amount to be converted. While the values can change, all required parameters must be included to guarantee the utterance can be fulfilled. We believe that constraint-aware prompting can improve the quality of the generated paraphrases. Incorporating such constraints into the prompt can help guide the LLM to produce paraphrases that are not only diverse but also semantically relevant to the seed utterance.

6.4 Simulating Worker Diversity with Multiple LLMs

In this study, we used a single ChatGPT-3.5-turbo instance to generate paraphrases. However, this setup does not reflect the diversity typically introduced by multiple crowd workers. Relying on a single LLM instance is equivalent to assigning the entire task to one human worker. To address this, we plan to use multiple LLM instances to simulate contributions from multiple workers. We aim to integrate alternative LLMs into the workflow to simulate a heterogeneous pool of models. This strategy mirrors the diversity found among crowd workers, where participants from different backgrounds may contribute varied perspectives [1].

7 Conclusion

In this paper, we investigated whether GPT can replace crowd workers to generate syntactically diverse paraphrases. We replicate the crowdsourcing workflow developed by Ramirez et al. [24] using the same scale, prompts, and seed data. The prompts focus on generating paraphrases that comply to syntax patterns (*patterns-by-example*) or avoid frequent patterns (*taboo-patterns*). Furthermore, we examine the impact of changing the number of paraphrases per request while maintaining the same total number of paraphrases.

Our findings show that (1) requesting multiple paraphrases in a single request is more effective than repeating the same request multiple times; (2) GPT-generated paraphrases exhibit greater syntactic diversity while maintaining stronger semantic relevance to the seed utterance, demonstrating its potential to effectively replace crowd workers in paraphrasing tasks; and (3) both prompts performed similarly when applied using GPT, with the *patterns-by-example* prompt yielding slightly better results than the *taboo-patterns* prompt.

Acknowledgement. We acknowledge the financial support of Taighde Éireann - Research Ireland under Grants Nos. 18/CRT/6223 and SFI/12/RC/2289_P2.

References

1. Bapat, R., Kucherbaev, P., Bozzon, A.: Effective crowdsourced generation of training data for chatbots natural language understanding. In: Mikkonen, T., Klamma, R., Hernández, J. (eds.) ICWE 2018. LNCS, vol. 10845, pp. 114–128. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-91662-0_8
2. Berro, A., Baez, M., Benatallah, B., Benabdeslem, K., Fard, M.: Automated paraphrase generation with over-generation and pruning services. In: Hacid, H., Kao, O., Mecella, M., Moha, N., Paik, H. (eds.) ICSOC 2021. LNCS, vol. 13121, pp. 400–414. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-91431-8_25
3. Beurer-Kellner, L., Fischer, M., Vechev, M.: Prompting is programming: a query language for large language models. Proc. ACM Program. Lang. **7**(PLDI) (2023). https://doi.org/10.1145/3591300
4. Brown, T.B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P.e.a.: Language models are few-shot learners. In: Proceedings of the 34th International Conference on Neural Information Processing Systems. NIPS '20, Curran Associates Inc., Red Hook, NY, USA (2020). https://dl.acm.org/doi/abs/10.5555/3495724.3495883
5. Cegin, J., Pecher, B., Simko, J., Srba, I., Bielikova, M., Brusilovsky, P.: Effects of diversity incentives on sample diversity and downstream model performance in LLM-based text augmentation. In: Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pp. 13148–13171. Association for Computational Linguistics, Bangkok, Thailand (2024). https://aclanthology.org/2024.acl-long.710/
6. Cegin, J., Simko, J., Brusilovsky, P.: ChatGPT to replace crowdsourcing of paraphrases for intent classification: higher diversity and comparable model robustness. In: Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, pp. 1889–1905. Association for Computational Linguistics, Singapore (2023). https://aclanthology.org/2023.emnlp-main.117
7. Chang, Y., Lo, K., Goyal, T., Iyyer, M.: BooookScore: a systematic exploration of book-length summarization in the era of LLMs (2024). https://arxiv.org/abs/2310.00785
8. Chen, M., Tang, Q., Wiseman, S., Gimpel, K.: Controllable paraphrase generation with a syntactic exemplar. In: Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, pp. 5972–5984. Association for Computational Linguistics, Florence, Italy (2019). https://aclanthology.org/P19-1599
9. Cox, S.R., Abdul, A., Ooi, W.T.: Prompting a large language model to generate diverse motivational messages: a comparison with human-written messages. In: Proceedings of the 11th International Conference on Human-Agent Interaction, pp. 378–380. HAI '23, Association for Computing Machinery, New York, NY, USA (2023). https://doi.org/10.1145/3623809.3623931

10. Goyal, T., Durrett, G.: Neural syntactic preordering for controlled paraphrase generation. In: Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pp. 238–252. Association for Computational Linguistics (2020). https://aclanthology.org/2020.acl-main.22
11. Iyyer, M., Wieting, J., Gimpel, K., Zettlemoyer, L.: Adversarial example generation with syntactically controlled paraphrase networks. In: Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, vol. 1 (Long Papers), pp. 1875–1885. Association for Computational Linguistics, New Orleans, Louisiana (2018). https://aclanthology.org/N18-1170
12. Jiang, Y., Kummerfeld, J.K., Lasecki, W.S.: Understanding task design trade-offs in crowdsourced paraphrase collection. In: Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Vol. 2: Short Papers), pp. 103–109. Association for Computational Linguistics, Vancouver, Canada (2017). https://aclanthology.org/P17-2017
13. Kang, Y., Zhang, Y., Kummerfeld, J.K., Tang, L., Mars, J.: Data collection for dialogue system: a startup perspective. In: Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, vol. 3 (Industry Papers), pp. 33–40. Association for Computational Linguistics, New Orleans - Louisiana (2018). https://aclanthology.org/N18-3005
14. Larson, S., et al.: Outlier detection for improved data quality and diversity in dialog systems. In: Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, vol. 1 (Long and Short Papers), pp. 517–527. Association for Computational Linguistics, Minneapolis, Minnesota (2019). https://aclanthology.org/N19-1051
15. Larson, S., et al.: Iterative feature mining for constraint-based data collection to increase data diversity and model robustness. In: Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), pp. 8097–8106. Association for Computational Linguistics (2020). https://aclanthology.org/2020.emnlp-main.650
16. Li, T., et al.: From crowdsourced data to high-quality benchmarks: arena-hard and benchbuilder pipeline (2024). https://arxiv.org/abs/2406.11939
17. Li, Z., Larson, S., Leach, K.: Generating hard-negative out-of-scope data with ChatGPT for intent classification. In: Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024), pp. 7634–7646. ELRA and ICCL, Torino, Italia (2024). https://aclanthology.org/2024.lrec-main.674/
18. Liu, M.X., Liu, F., Fiannaca, A.J., Koo, T., Dixon, L., Terry, M., Cai, C.J.: we need structured output: towards user-centered constraints on large language model output. In: Extended Abstracts of the 2024 CHI Conference on Human Factors in Computing Systems. CHI EA '24, Association for Computing Machinery, New York, NY, USA (2024). https://doi.org/10.1145/3613905.3650756
19. Lu, A., Zhang, H., Zhang, Y., Wang, X., Yang, D.: Bounding the capabilities of large language models in open text generation with prompt constraints. In: Findings of the Association for Computational Linguistics: EACL 2023, pp. 1982–2008. Association for Computational Linguistics, Dubrovnik, Croatia (2023). https://aclanthology.org/2023.findings-eacl.148/
20. Madaan, A., Yazdanbakhsh, A.: Text and patterns: for effective chain of thought, it takes two to tango (2022). https://arxiv.org/abs/2209.07686

21. Maus, N., Chao, P., Wong, E., Gardner, J.: Black box adversarial prompting for foundation models (2023). https://arxiv.org/abs/2302.04237
22. Parikh, S., Vohra, Q., Tiwari, M.: Automated utterance generation (2020). https://arxiv.org/abs/2004.03484
23. Park, S., Hwang, S.W., Chen, F., Choo, J., Ha, J.W., Kim, S., Yim, J.: Paraphrase diversification using counterfactual debiasing. Proc. AAAI Conf. Artif. Intell. **33**(01), 6883–6891 (2019). https://doi.org/10.1609/aaai.v33i01.33016883
24. Ramírez, J., Baez, M., Berro, A., Benatallah, B., Casati, F.: Crowdsourcing syntactically diverse paraphrases with diversity-aware prompts and workflows. In: Advanced Information Systems Engineering: 34th International Conference, CAiSE 2022, Leuven, Belgium, June 6-10, 2022, Proceedings. p. 253-269. Springer (2022), https://doi.org/10.1007/978-3-031-07472-1_15
25. Ravichander, A., Manzini, T., Grabmair, M., Neubig, G., Francis, J., Nyberg, E.: How would you say it? Eliciting lexically diverse dialogue for supervised semantic parsing. In: 18th Annual SIGdial Meeting on Discourse and Dialogue, pp. 374–383. Association for Computational Linguistics, Saarbrücken, Germany (2017). https://aclanthology.org/W17-5545
26. dos Santos, V.G., Benatallah, B., Berro, A., MacMahon, S.T.: Diverse utterances generation with GPT to improve task-oriented chatbots and APIS integration. In: 2024 2nd International Conference on Foundation and Large Language Models (FLLM), pp. 42–50 (2024). https://doi.org/10.1109/FLLM63129.2024.10852454
27. dos Santos, V.G., Santos, G.L., Lynn, T., Benatallah, B.: Identifying citizen-related issues from social media using LLM-based data augmentation. In: Advanced Information Systems Engineering. CAiSE 2024, pp. 531–546. Springer (2024). https://doi.org/10.1007/978-3-031-61057-8_31
28. Su, Y., et al.: On generating characteristic-rich question sets for QA evaluation. In: Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, pp. 562–572. Association for Computational Linguistics, Austin, Texas (2016). https://aclanthology.org/D16-1054
29. Thompson, B., Post, M.: Paraphrase generation as zero-shot multilingual translation: disentangling semantic similarity from lexical and syntactic diversity. In: Proceedings of the Fifth Conference on Machine Translation, pp. 561–570. Association for Computational Linguistics (2020). https://aclanthology.org/2020.wmt-1.67/
30. Wang, W.Y., Bohus, D., Kamar, E., Horvitz, E.: Crowdsourcing the acquisition of natural language corpora: methods and observations. In: 2012 IEEE Spoken Language Technology Workshop (SLT), pp. 73–78. IEEE (2012). https://doi.org/10.1109/SLT.2012.6424200
31. Wu, T., et al.: LLMs as workers in human-computational algorithms? Replicating crowdsourcing pipelines with LLMs (2023). https://arxiv.org/abs/2307.10168
32. Yaghoub-Zadeh-Fard, M.A., Benatallah, B., Casati, F., Barukh, M.C., Zamanirad, S.: Dynamic word recommendation to obtain diverse crowdsourced paraphrases of user utterances, p. 55–66. IUI '20, Association for Computing Machinery, New York, NY, USA (2020). https://doi.org/10.1145/3377325.3377486
33. Yaghoub-Zadeh-Fard, M.A., Benatallah, B., Casati, F., Barukh, M.C., Zamanirad, S.: User utterance acquisition for training task-oriented bots: a review of challenges, techniques and opportunities. IEEE Internet Comput. **24**(3), 30–38 (2020). https://doi.org/10.1109/MIC.2020.2978157
34. Zhang, T., Kishore, V., Wu, F., Weinberger, K.Q., Artzi, Y.: BERTScore: evaluating text generation with BERT (2020). https://arxiv.org/abs/1904.09675

35. Zhang, Y., et al.: Pushing the limit of LLM capacity for text classification (2024). https://arxiv.org/abs/2402.07470
36. Zhao, W.X., et al.: A survey of large language models (2025). https://arxiv.org/abs/2303.18223
37. Zhou, Y., et al.: Large language models are human-level prompt engineers (2023). https://arxiv.org/abs/2211.01910

A Conversational Framework for Faithful Multi-perspective Analysis of Production Systems

Angelo Casciani[1(✉)], Livia Lestingi[2], Andrea Marrella[1], and Andrea Matta[2]

[1] Sapienza University of Rome, Rome, Italy
{casciani,marrella}@diag.uniroma1.it
[2] Politecnico di Milano, Milan, Italy
{livia.lestingi,andrea.matta}@polimi.it

Abstract. Production systems call for analysis techniques yielding reliable diagnostic and prognostic insights in a timely fashion. To this end, numerous reasoning techniques have been exploited, mainly within the simulation and formal verification realms. However, the technological barrier between these approaches and the target end users remains a stumbling block to their effective adoption. This paper presents a framework interposing a natural language-based interface between the interpretation of the user's request and the reasoning tools. The user's natural language request is automatically translated into a machine-readable problem. The latter is then dispatched to a proper reasoning engine and either solved through a simulation or a formal verification task, thus enabling a multi-perspective analysis of the production system and certifying the correctness and transparency of the obtained solutions. The outcome is then reprocessed to be human-interpretable. State-of-the-art Large Language Models (LLMs), with their robust capability to interpret the inherent ambiguity of natural language, perform both translations. We evaluate the framework on a lab-scale case study replicating a real production system. The results of the experiments suggest that LLMs are promising complements to derive insights from faithful reasoning engines, supporting accurate analysis.

Keywords: Production System · Large Language Model · Reasoning Engine · Simulation · Formal Verification · Analysis

1 Introduction

With the advent of Industry 4.0 (I4.0), production systems are experiencing a profound digital transformation, driven by the availability of Internet-of-Things (IoT) devices that generate vast amounts of *production data*, enabling increased monitoring and traceability of the production processes. A *production process* refers to a series of activities to transform raw materials into finished products, spanning from the initial preparation of materials to quality assurance and distribution. *Production systems* encompass all the software, hardware, robotic tools, human resources, and business rules required to carry out production processes.

J. Krogstie et al. (Eds.): CAiSE 2025, LNCS 15701, pp. 163–181, 2025.
https://doi.org/10.1007/978-3-031-94569-4_10

Processes are captured as *digital models*, virtual representations that accurately reflect the manufacturing activities throughout their lifecycle [32].

Manufacturing companies' aim to leverage production data and digital models to amplify human expertise, creating a collaborative environment between human experts and production systems where decision-making is faster, more accurate, and informed by insights drawn from previously inaccessible or tangled knowledge [3]. However, achieving this goal is hindered by several real-life challenges. Current reasoning tools require highly skilled users to operate and interpret their outputs. This creates a significant barrier to adoption, as most domain experts lack the technical expertise to interact with these tools effectively. Moreover, decision-making in production systems often involves manual data selection and interpretation, which is time-consuming and prone to errors. Additionally, these tools operate in isolation, requiring users to switch between different interfaces and modalities to perform multi-perspective analyses. This fragmentation increases the risk of inconsistencies and reduces the overall efficiency of the decision-making process [20]. For example, to forecast the optimal timing for producing and delivering new orders in compliance with customer deadlines, a company ought to revise the production plan to (*i*) minimize the changes in the planned production, (*ii*) optimize resource utilization and warehouse management, while (*iii*) guaranteeing the soundness of the production system behavior. Nowadays, this kind of analysis is undertaken by expert users based on manual data selection and their interpretation through dedicated reasoning tools, mainly within the simulation [12] and formal verification realms [24]. This is time-consuming, requires a highly skilled workforce able to interact with and interpret the outcomes of multiple reasoning tools (with different interaction modalities), and increases the risk of deriving sub-optimal decisions that are poorly aligned with the production process requirements, as only a restricted subset of execution data and options are considered [8].

In this exploratory paper, we present a conversational framework interposing a natural language-based interface between the user's request and the reasoning tools used to analyze production systems from different perspectives. Our framework decomposes a reasoning problem into three operational stages: (*i*) problem formulation, (*ii*) reasoning, and (*iii*) results interpretation. During *problem formulation*, the framework uses the Large Language Model (LLM) technology to translate the user's natural language request into a machine-readable problem compatible with the specific reasoning task to solve (i.e., simulation or formal verification) and dispatch it to a reasoning engine. Subsequently, at the *reasoning* stage, a dedicated reasoning engine processes the machine-readable formulation to obtain an outcome that is translated by an LLM to be human-readable in the *results interpretation* stage.

The conversational framework is primarily designed to serve *non-technical users* (e.g., plant managers, operators) who require an intuitive, tool-agnostic, natural language-based interface to obtain clear and actionable insights from complex reasoning engines without the need for deep technical expertise [27, 45, 50]. A typical use case for our framework occurs when a sudden machine breakdown disrupts production, and the plant manager asks: "*What would be* *the average waiting time for producing more products on another machine if I redirect the production from the broken machine to it?*." The framework assists the plant manager with a natural language response, abstracting away the details of the engine that simulates the requested production scenario. This enables the manager to make an informed decision-whether to wait for the machine to be repaired or to redirect production-ensuring that the redirection does not create bottlenecks or delays in processing orders on the other machine. The framework can also assist with queue re-prioritization, i.e., evaluating trade-offs when inserting new urgent orders while accounting for warehouse constraints and existing production schedules [3]. In general, any decision involving interaction between human experts and one or more complex systems in the production process is a potential use case for management by our framework [15].

The novelty of our approach lies in its seamless integration of LLM technology with reasoning engines. We exploit the LLMs' capability in interpreting the inherent ambiguity of natural language requests into machine-readable representations that well-established reasoning engines can digest. Indeed, LLMs still struggle with complex reasoning problems, occasionally deriving unfaithful conclusions that do not follow the reasoning steps prompted by the users [7,14,37]. In contrast, reasoning engines offer mechanisms to guarantee the faithfulness and transparency of their outcomes, a critical aspect for ensuring trusted analysis of production systems.

For evaluation purposes, we select a lab-scale case study replicating a real production system within the Italian PRIN (Research Projects of National Relevance) project MOTOWN.[1] Quantitative results show that the framework is technologically flexible with respect to the selected LLMs. From a qualitative standpoint, selected users evaluated its ease of use, ability to interpret requests even when not perfectly formalized, and clarity in the responses, highlighting how LLMs complement reasoning engines.

The rest of the paper is organized as follows. Section 2 introduces the background on LLMs, simulation, and formal verification necessary to understand the paper and discusses the related work. Section 3 shows the main steps of the framework and provides details on their implementation. Section 4 assesses the framework through quantitative and qualitative evaluations performed over a lab-scale case study, while Sect. 5 concludes the paper by tracing limitations and future works.

2 Background and Related Work

2.1 Large Language Models

LLMs are computational models for Natural Language Processing (NLP), capable of understanding and generating human-like text [53]. Built on transformer-based architectures, they use self-attention mechanisms to capture relationships within input sequences and produce coherent output [44]. Models like GPT [1],

[1] http://www.open.diag.uniroma1.it/project_detail/28097.

LLaMA [11], and Mistral [18], with billions of parameters and pre-trained on massive datasets, excel in tasks ranging from question-answering [21] to code explanation and generation [9].

However, general-purpose LLMs are not optimized for domain-specific applications and can, thus, generate inaccurate information (i.e., hallucinations) [36]. Therefore, in specialized contexts such as production systems, challenges arise due to the domain knowledge inherited from potentially outdated or biased training datasets [13].

Fine-tuning is commonly adopted to adapt LLMs to particular tasks. By training pre-trained models on task-specific datasets, fine-tuning enhances their performance and answers' relevance for specialized applications. However, the resource-intensive nature of this technique, including substantial computational and data requirements, can be a limiting factor [17]. A possible alternative is embodied by *In-Context Learning* (ICL) [10]. Leveraging carefully engineered prompts containing examples and explicit instructions, ICL allows LLMs to perform zero- or few-shot learning, guiding them to generate contextually appropriate responses without additional training. ICL can be strengthened by *Retrieval-Augmented Generation* (RAG), particularly in knowledge-intensive tasks [25]. RAG operates by retrieving information from an external knowledge base, based on similarity metrics with the input, and integrating it into the prompt for the LLM, thereby leveraging ICL to utilize this additional knowledge effectively.

2.2 Simulation in Production Systems

In manufacturing, *simulation* has two main usages: performance prediction (e.g., production throughput, machine utilization, inventory levels, etc.) and what-if analysis to support managers in decision-making [52]. Despite high costs in terms of knowledge needed to create models and obtain results, time to run the experiments, and software licenses, finite element method (FEM) simulation is the dominant technique in product engineering (e.g., to estimate fatigue resistance of the product) and process engineering (e.g., to estimate strains in material removal cutting processes), whereas discrete event simulation (DES) is largely used for design and planning purposes of systems crossed by material and information flows (e.g., to estimate production capacity in an assembly line balancing problem), especially in production systems.

This paper focuses on DES for production systems. DES is a numerical approach to studying the behavior of discrete event systems, i.e., the formal framework to represent production systems in which part flows visit machines and compete for resources. Thanks to the representation power of DES, the same simulation model can be used for different purposes, and the practitioner has to initialize and execute it depending on the specific purpose. Libraries like SimPy facilitate the implementation of DES models [28].

However, designing simulation experiments and interpreting results necessitates specialized knowledge and skills that are not common in companies. For instance, controlling the bias introduced by the finite sample size in simulation experiments requires knowledge of statistical inference as well as the consideration of existing covariance in the predicted response variables of the analyzed production system. The lack of expertise often limits its broader adoption in industrial settings, underscoring the need for training and accessible methodologies to spread its application.

LLMs and Simulation The use of LLMs in production systems has grown rapidly, fueled by demands for automation, prediction, and better decision-making.

Several studies have used LLMs for automating, planning, and optimizing production processes in manufacturing [26]. For instance, [49] introduces a multi-agent system that uses LLMs to automate the parametrization of simulation models for digital twins, leveraging dynamic interaction and knowledge-based heuristics to reduce users' cognitive load. Similarly, [46] proposes an intelligent industrial production system where LLMs optimize operations and energy efficiency, and [5] integrates Computer-Aided Process Planning with GPT models and digital twins, enabling real-time optimization and adaptation of manufacturing processes. Additionally, [40] presents a multi-agent digital twin architecture driven by LLMs, integrating multimodal data and philosophical reasoning to improve adaptability and security in equipment maintenance. Furthermore, [48] combines LLMs with the ACT-R Cognitive Architecture to enhance decision-making by aligning machine and human reasoning.

Other studies combine LLMs with production system simulation or validation tools, targeting both real-time (operational) and offline (tactical) analysis. For instance, [51] proposes a hierarchical framework in which LLMs interpret production data for planning and control, facilitating task automation and flexibility in a production facility. Instead, [41] introduces a paradigm for heat treatment process design based on LLMs. This enables knowledge transfer and process recommendation, facilitating interactions between humans and the FEM software.

While the works above highlight the diverse applications of LLMs in production systems, our framework introduces an original approach by integrating LLMs with simulation and formal verification tools. Unlike existing systems that emphasize simulation or decision-making for online or offline analysis, our framework uses LLMs not to execute these tasks directly, thus avoiding concerns about their reliability, but as interfaces for routing and formalizing tasks to be handled by sound analysis engines. This design ensures greater flexibility and interpretability, allowing the framework to be customized for specific production use cases, such as the one introduced in Sect. 1, while maintaining intuitive natural language interactions with the users.

2.3 Formal Verification in Production Systems

Critical production systems require strong behavioral guarantees. *Formal verification* mathematically proves the system's compliance with its *specification*, e.g., behavioral constraints regarding safety and efficiency. To this end, verification techniques require a sound formalization of the system (i.e., the *formal model*) and the specification to be verified (i.e., the *property*). Unlike simulation, verification conclusively proves whether a property holds on a given model, but at a higher computational cost.

Production systems have time-dependent dynamics due to stochastic processing delays. We model them using Probabilistic Timed Automatons (PTAs) [6] and express properties in Timed Computation Tree Logic (TCTL) [4], enabling model checking via UPPAL [22].” In Deterministic Finite-state Automatons (DFAs), *locations* correspond to the different states of the system under modeling connected through *edges*. The alphabet—referred to as set A—contains all the events that can trigger an edge. Timed Automatons (TAs) extend DFAs by introducing variables that can only grow uniformly with time or be reset, called *clocks*. Given locations L and clocks C, we indicate with $\Gamma(C)$ the set of constraints (i.e., the guard conditions) that can be expressed on clocks and with $\mathcal{I}(l) \subseteq \Gamma(C)$ the set of invariants assigned to location $l \in L$. Given valuation $\nu \subseteq \mathbb{R}^C$ containing the current clock values, an edge can fire only if the valuation satisfies its guard condition, and a location can only be occupied if the valuation satisfies its invariant. PTAs further extend this by adding probability weights on edges, thus defining a probability transition function.

UPPAL supports a subset of TCTL formulae, which defines the properties that can be verified through our approach. Specifically, we support properties of the form $\phi ::= \forall\Box\ \beta \mid \exists\Diamond\ \beta \mid \forall\Diamond\ \beta \mid \exists\Box\ \beta$, $\beta ::= a \mid \beta_1 \wedge \beta_2 \mid \neg\beta$, where $\Box$ and $\Diamond$ are the “always” and “eventually” operators, respectively, and a is either an atomic clock constraint (e.g., $1 \leq x \leq 5$ with $x \in C$) or a location (e.g., $\mathcal{A}$ $\texttt{at}$ l where $\mathcal{A}$ is a PTA and $l \in L$ holds). Notable properties compatible with this grammar are *safety* ($\forall\Box\ \neg\beta$ where β is an unsafe state) and *reachability* properties ($\exists\Diamond\ \beta$ where β is the desired state).

LLMs and Verification. Despite LLMs demonstrating impressive performance across diverse tasks and benchmarks, emerging research highlights critical shortcomings in their reasoning and verification abilities. Studies demonstrate that even state-of-the-art LLMs struggle with basic reasoning tasks, exhibiting high performance variability due to minor problem variations, overconfidence in incorrect solutions, and an inability to benefit from multi-step re-evaluation [33]. Similarly, LLMs display significant performance drops in mathematical reasoning when faced with slightly altered problem parameters or increased complexity, suggesting that their reasoning is more simulated than logical [31]. Indeed, LLMs are argued to function more as approximate knowledge retrievers than true reasoning agents, raising doubt on their capacity for planning and self-verification tasks [19]. These findings emphasize the need for more robust frameworks and rethinking LLMs’ application in reasoning-intensive domains.

Recent research has integrated LLMs with formal reasoners and verifiers to address their limitations. For instance, combining LLMs with Linear Temporal Logic (LTL) representations has enhanced their ability to interpret and execute natural language instructions in temporally extended tasks, demonstrating superior performance in text-based games [43]. The LLM-Modulo framework proposes a tighter integration of LLMs with symbolic verifiers, enabling bidirectional interactions that extend planning and reasoning capabilities beyond mere translation tasks [19]. Other approaches incorporate Answer Set Programming (ASP) solvers to structure user queries for classical AI planners, outperforming

LLM-only strategies in orchestrating API tasks [2]. Neuro-symbolic systems such as AlphaGeometry further highlight the potential of integrating neural LLMs with symbolic deduction engines, achieving state-of-the-art results in mathematical theorem proving [42].

Building on these advancements that underscore the potential of hybrid architectures in overcoming the inherent reasoning deficits of standalone LLMs, our work introduces a novel framework integrating LLMs with simulation and formal verification tools for production processes. This approach leverages simulation parameters and automata representations of processes to ensure accurate and faithful responses that bridge the gap between natural language understanding, event-based simulation, and rigorous formal verification in a unified system.

3 Framework

The overall framework is designed to provide grounded and interpretable answers to natural language requests concerning a production process, i.e., the representation of the activities performed within a production system. It achieves this through the integration of a *Conversational Layer* and a *Reasoning Layer*. The former tackles the formulation of the problem to be fed to the Reasoning Layer and the interpretation of the results in response to the user. The latter exploits either a digital twin simulating the production process or a formal verifier reasoning on its automaton. Therefore, the approach assumes the availability of the simulation parameters and the automaton modeling the production process, provided by a domain expert rather than being LLM-generated to ensure their correctness.

As illustrated in Fig. 1, the Conversational Layer includes a set of LLMs: the *Gateway LLM*, which routes the user's questions, and the *Translator LLMs for Simulation* and *Verification*, which translate these requests into machine-readable representations compatible with the corresponding components' syntax.

A critical decision in our framework is the adoption of three specialized LLMs rather than a single model handling both routing and formalization tasks. Sev-

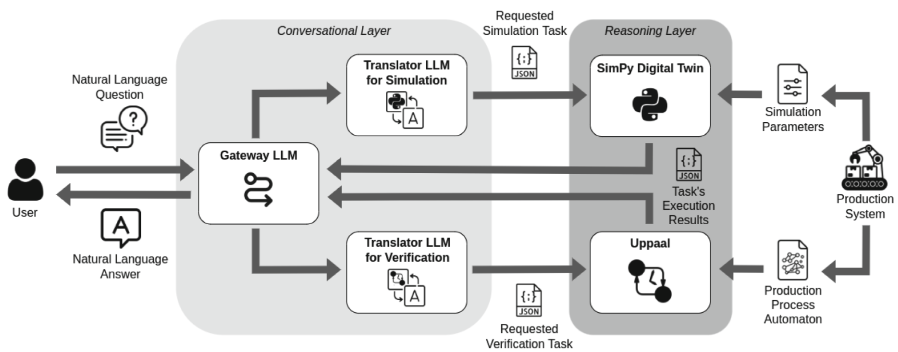


Fig. 1. The architecture of the proposed framework.

Table 1. Composition of prompts for the LLM within the framework. Each column represents a phase of the interaction, while each row corresponds to a prompt's component.

Prompt Section	**Question Routing**	**Simulation**	**Verification**	**Answer Generation**
Role&Task	Gateway	Formalizer for Digital Twin	Formalizer for Verifier	Interface
Context	Rules for Routing	Allowed Simulation Tasks	Uppaal Query Syntax	User's question
Examples	-	(Natural language question, JSON with simulation parameters)	(Natural language question,JSON with Uppaal query)	-
Input	User's question	User's question	User's question	Task's results

eral considerations drive this design choice. First, employing a single LLM for multiple tasks can result in *interference effects*, where the model's effectiveness on one task potentially reduces its performance on another [16]. Specializing each LLM for a specific task mitigates this risk. Moreover, a modular approach with multiple specialized LLMs is inherently more scalable. If a new task is introduced, another LLM can be integrated without retraining or compromising existing models. Modularity also facilitates troubleshooting, as each LLM is dedicated to a single task. This makes it easier to identify and resolve failures or inaccuracies within the system.

Additionally, we chose not to allocate time and computational resources to fine-tune the specialized models, as their performance on their respective tasks proved satisfactory using ICL alone (refer to Sect. 4 for further details). It is important to note that the approach is entirely flexible in terms of LLM choice, as any model can be utilized for all three roles, even a combination of different ones.

The prompts[2] designed for the various LLMs involved in this approach follow the structure detailed in Table 1. We employed several well-established prompt engineering techniques to ensure high-quality responses [47]. These include explicitly defining the LLM role to clarify its task and output style [39], as well as incorporating multiple examples of question-answer pairs to facilitate ICL in a few-shot manner [34]. The complete prompt includes four main sections: (*i*) the definition of the LLM role and the specific task it needs to perform within the framework; (*ii*) the contextual information required to carry out the task; (*iii*) illustrative examples to guide the model in understanding how to perform its task; and (*iv*) the user's input, processed by the LLM according to the preceding sections. An instance of the above prompt structure for the Translator LLM for Simulation is provided in Fig. 2.

We implemented the proposed framework using LangChain[3] to facilitate communication with and between the LLMs.

[2] Prompts are available at https://github.com/angelo-casciani/conv_automata.
[3] Langchain Website: https://www.langchain.com/.

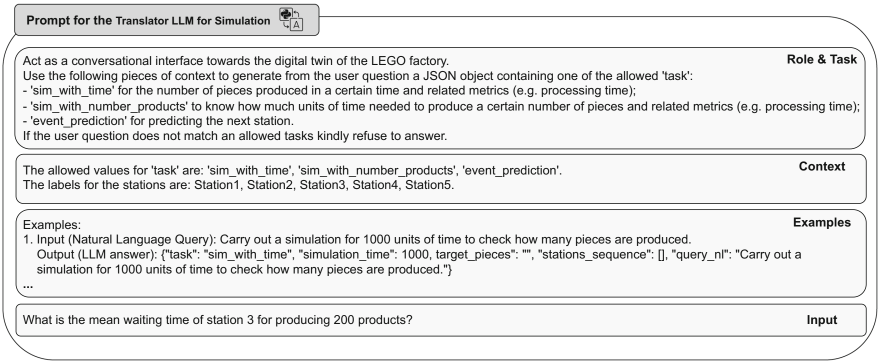


Fig. 2. Example of a prompt for the Translator LLM for Simulation.

3.1 Question Routing

First, the factory worker submits to the framework a natural language request about the production process. The *Gateway LLM* receives this question, and it is responsible for routing it to the appropriate branch depending on its content.

Notably, such frontier LLM identifies the intent of the question and directs it to the relevant Translator LLM or declines to respond if the request is outside the scope of the framework, i.e., if it pertains neither to simulation tasks involving the digital twin nor to verification tasks over the automaton.

The prompt used to instruct the Gateway LLM includes specific rules for routing the user's questions, reporting the criteria that determine whether a question should be forwarded to a particular branch or classified as out of scope.

Regarding the choice of relying on an LLM for this step, a rule-based matching approach could also dispatch the user's question. However, an LLM offers a higher degree of semantic flexibility, allowing the framework to understand requests that do not conform to a given pattern or use predefined terms to select the correct branch.

3.2 Simulation Task Generation

When a question is identified as a simulation task, it is redirected to the *Translator LLM for Simulation*. This LLM formulates the natural language question into a JSON file containing all the needed information for the digital twin of the production process to carry out the requested simulation task (see Fig. 3).

The LLM operates in response to a specific prompt (see Table 1), which reports the supported simulation tasks and provides instructions on how to generate a JSON conforming to the format expected by the digital twin. The prompt also provides multiple examples of natural language questions and their corresponding JSON translations, enabling effective few-shot learning.

The digital twin, which was implemented relying on the SimPy library, accepts as input the main simulation parameters of the production process. These

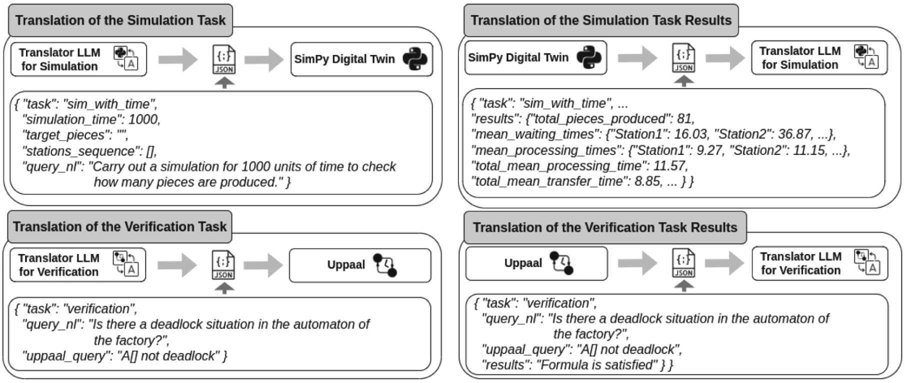


Fig. 3. The structure of the JSON files exchanged between the components. The user's initial prompt is stored as the value of the "*query_nl*" key, while the remaining key values are populated by the respective Translator LLM based on its interpretation of the request (on the right) or of the results to report back to the user (on the left).

parameters encompass the inter-arrival time of new instances to process, the stations composing the production line, their processing times (modeled as normal distributions characterized by their mean and standard deviation), and the probabilities of the outgoing branches to subsequent production stations along with the associated transfer times (also expressed as normal distributions). The digital twin supports several tasks such as simulating the production process to determine the number of pieces that can be produced within a specified time frame, simulating the production line's execution for computing the time required to produce a specified number of pieces, and predicting the next station based on a given sequence of stations. For each simulation, the digital twin computes detailed statistics, including processing and waiting times for each station, as well as the aggregate mean waiting, processing, and transfer times for the entire production process based on the specified input parameters.

3.3 Verification Task Generation

If the request is routed to the *Translator LLM for Verification*, the natural language question is processed and converted into a TCTL property expressed in the UPPAL-specific query notation. Potential queries in this context regard all properties that can be expressed in TCTL, such as the reachability of a specific state, the detection of deadlocks, or the verification of temporal properties within a specified time bound. Similarly to simulation tasks, the prompt provided to the Translator LLM includes a system message describing the TCTL syntax and the states of the PTA model. A few-shot prompting approach is employed, presenting several examples of natural language questions along with their corresponding TCTL queries to be submitted to UPPAL for formal verification.

The resulting TCTL property is then bundled in a JSON file (see Fig. 3). UPPAL then checks the TCTL property against the production process's PTA model.

3.4 Answer Generation

Once the outcomes of the requested task are computed by the Reasoning Layer, they are appended to the original JSON and sent back to the Gateway LLM (see Fig. 3). In this second iteration, the Gateway LLM is tasked with the interpretation of the results, extracting the relevant information needed to formulate a precise and coherent natural language response for the user's initial request.

The prompt is tailored to provide the LLM with the reasoning outcomes as context for the response's generation. It also includes instructions to ensure that the LLM answers the original question accurately by leveraging the provided results. If the LLM cannot extract the necessary information from the context to respond appropriately, it refuses to answer without reporting non-factual information crafted on the fly.

4 Evaluation

Given its intrinsic interactive nature, we conducted a twofold evaluation of the framework, encompassing a quantitative analysis of the performance of various language models in executing the routing and translation phases, along with a preliminary qualitative assessment involving users for judging its natural language answers. This evaluation provided a comprehensive understanding of the framework's performance at each step, determining its generalizability and dependency on the technology (regarding the used LLM for each step), and examined human users' perceptions of its responses.

4.1 Case Study

The case study selected for the experimental campaign is grounded in the Italian PRIN project MOTOWN, which focuses on replicating a real processing plant from the automotive field on a small scale using the Lego Mindstorms [30] kit for educational purposes. While the original plant serves as the foundation for this study, specific details about it cannot be disclosed due to a non-disclosure agreement with the involved company. The small-scale replication allows us to realistically test and evaluate the framework while adhering to these constraints.

A schematic representation of the plant is shown in Fig. 4, specifically highlighting the processing stations composing the line and how the conveyor belt connects them. The plant consists of five stations. The processing of items flowing through the plant begins from stations 1 and 2, then either continues with station 3 or 4. The policy with which items are sent to station 3 or 4 is not assumed to be known (thus, not explicitly modeled) but approximated through probability weights modeling the bias of the system toward the connection to station 3 compared to station 4. Processing always ends with station 5.

Stations are further characterized by their maximum capacity c, and the distribution of the associated processing time, which is assumed to be normal with mean μ_p and standard deviation σ_p. Conveyor belt segments connecting two stations are characterized by the distribution of the associated transfer time, also

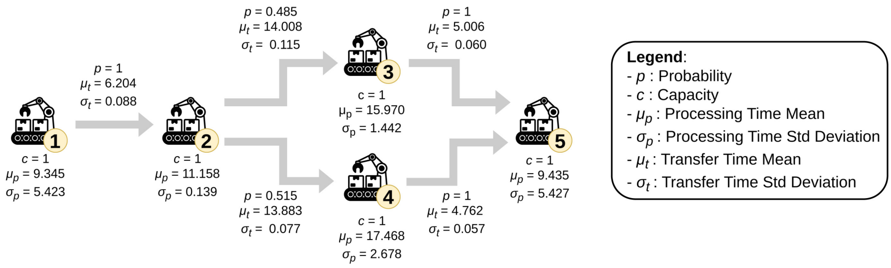


Fig. 4. The LEGO factory annotated with the corresponding simulation parameters for each production station and conveyor belt. Time-related parameters are in seconds (s).

assumed to be normal with mean μ_t and standard deviation σ_t. Each segment is also characterized by frequency p with which items traverse it compared to its alternative (thus, $p = 1$ holds for all segments except those connecting station 2 to 3 and 4). Figure 4 reports the specific values for all the listed parameters mined from a dataset of 3 hours of event logs.

4.2 Design of the Evaluation

Quantitative Evaluation: The approach was evaluated by executing the routing and translation steps. For the routing phase, the Gateway LLM was expected to direct the user's question to the appropriate branch or reject the ones that were out of scope. The Translator LLMs were tested on their ability to convert the natural language request into a syntax compatible with the digital twin for simulation or UPPALfor verification. Both LLMs should produce a valid JSON embedding the correct translation of the requested task. Performance was measured using the *accuracy* metric, defined as the ratio of correct LLM-generated answers over the total number of questions. An LLM-generated answer is considered correct when it contains the exact expected output for a given question, i.e., the precise branch categorization for routing (simulation, verification, or no answer at all) and the valid JSON for the translations.

We created three datasets, one for each step under test, each composed of 300 samples with a consistent structure of question, answer, and evaluation type triplets. These synthetic datasets were automatically constructed by randomly selecting numeric inputs to complement generic templates of requests in natural language designed to test specific tasks supported by the framework. Each crafted question for the translation steps has a corresponding answer, composed by the JSON object containing the question's translation for the branch, stated by the evaluation type field. For the routing phase dataset, one-third of the samples (100 triplets) pertained to simulation, another third to verification, and the remaining third consisted of out-of-scope questions that the Gateway LLM was not expected to route.

Evaluation subjects include open-source models from the Llama [11], Mistral [18], Qwen [35], and Gemma [38] families, which were run locally, and proprietary ones like GPT-4 [1]. These tests aimed to determine the extent to which the framework is generalizable and whether each step works independently of the underlying technology, which is a fundamental aspect given the rapid advancements in LLM research.

We conducted the experiments with local open-source LLMs on a workstation running the Linux/Ubuntu 22.04.3 LTS operating system and utilizing an NVIDIA A100 GPU. For testing with the GPT model, we relied upon the API services provided by OpenAI.[4] Each evaluation step was repeated five times to ensure robustness against the statistical variability of LLM outputs.

Qualitative Evaluation: To complement the quantitative evaluation, we conducted a preliminary qualitative study following the guidelines outlined in [23]. Our goal was to gain qualitative insights into the answers provided by the framework, understand users' perceptions of its strengths and weaknesses, and identify areas for improvement. Specifically, we conducted a survey in controlled settings to assess the quality of the answers provided by the framework. The metrics we used for the evaluation are: (*i*) *coherence*, the extent to which the response logically aligns with the query; (*ii*) *understandability*, the clarity and ease of interpreting the response; (*iii*) *relevance*, the degree to which the response addresses the query; and (*iv*) *usefulness*, the practical value of the response in addressing the query [23]. The chosen setup featured OpenAI's *gpt-4o-mini* for all steps since it performed best in both routing and translation tasks (see Sect. 4.3).

The survey began with background information about the framework's utility in the context of production systems, details about the LEGO factory, and an overview of the supported tasks. Participants were then invited to experiment with the tool by posing natural language requests about the LEGO factory and, consequently, to complete the survey. This survey consisted of four closed-ended questions evaluating the coherence, understandability, relevance, and usefulness of the framework's answers through a Likert scale, along with two optional open-ended questions asking for feedback on what participants appreciated most and potential suggestions for improvement. We distributed the survey to a total of 14 participants from Sapienza University of Rome, including 3 Master's students and 11 academics (10 doctoral students and one professor), all with a background in Artificial Intelligence (AI) and Computer Science. While these participants are not domain experts in production systems, we selected them based on their technical expertise, with prerequisite knowledge in Process Management and Mining, Production Process Optimization, Symbolic AI, and Human-Computer Interaction. This allowed them to critically evaluate the framework's functionality and provide meaningful feedback for improvement. Eventually, we collected the responses maintaining participants' anonymity.

4.3 Evaluation Results

Quantitative Evaluation: Table 2 presents the quantitative results of the evaluation assessing the ability of LLMs to perform the request's routing and the

[4] https://platform.openai.com/docs/overview.

Table 2. Average accuracy of the LLMs over 5 runs in the routing of the user's question, in the translation of the simulation task, and in the translation of the verification task. The best results for each category are highlighted in bold.

Language Model	Routing	Simulation	Verification
meta-llama/Meta-Llama-3-8B-Instruct	96.11%	77.56%	53.77%
meta-llama/Meta-Llama-3.1-8B-Instruct	99.45%	82.00%	69.99%
meta-llama/Llama-3.2-1B-Instruct	45.66%	16.67%	0.78%
meta-llama/Llama-3.2-3B-Instruct	89.67%	57.33%	52.55%
mistralai/Mistral-7B-Instruct-v0.2	91.00%	74.33%	81.55%
mistralai/Mistral-7B-Instruct-v0.3	96.66%	74.00%	45.67%
mistralai/Mistral-Nemo-Instruct-2407	96.83%	71.87%	62.22%
mistralai/Ministral-8B-Instruct-2410	87.11%	72.00%	59.78%
Qwen/Qwen2.5-7B-Instruct	**100.00**%	84.67%	85.00%
google/gemma-2–9b-it	71.56%	87.00%	70.11%
OpenAI/gpt-4o-mini	**100.00**%	**99.56**%	**96.78**%

translation of the simulation and the verification tasks. In particular, regarding the routing task, the tested language models demonstrated satisfactory performance, with *Qwen2.5-7B-Instruct* and *gpt-4o-mini* achieving perfect accuracy and demonstrating strong capabilities in properly categorizing user's requests. The main exception was the smallest tested model, *Llama-3.2-1B-Instruct*, which struggled to generalize effectively across diverse question patterns. These findings suggest that both open- and closed-source language models can be reliably employed for the routing task, with minimal variability in outcomes among well-performing LLMs.

On the other hand, the results in Table 2 for translating the requests into the corresponding simulation task suggest substantial differences across the models, defining a notable gap between open-source and proprietary LLMs. The *gpt-4o-mini* model demonstrated the highest accuracy, significantly outperforming all other models and showcasing the superior reliability of advanced proprietary models in handling this translation. Among open-source models, *gemma-2–9b-it* achieved the best performance. Conversely, the smallest tested models, both variants of *Llama 3.2*, delivered the weakest results, indicating that this conversion does not exhibit the inverse scaling phenomenon, sometimes observed when smaller models outperform larger ones due to issues in training objectives and data [29].

Similarly to the simulation task translation, there is significant variability in accuracy between open-source and proprietary models for verification tasks (see last column of Table 2). The findings indicate that the verification tasks translation requires advanced language comprehension and formalization capabilities, in which closed-source models such as *gpt-4o-mini* seem to perform better. Among open-source LLMs, the sufficient performance of *Qwen2.5-7B-Instruct* and *Mistral-7B-Instruct-v0.2* underlines the availability of competitive alterna-

Table 3. Results from the administered survey for the qualitative evaluation of the answers.

Question	**1**	**2**	**3**	**4**	**5**
How coherent is the answer with the provided question?		1	1	8	4
How easy is it to understand the system's response?				6	8
How relevant is the system's answer to your question?		1	2	3	8
How useful is the system's answer in addressing your question?		2		9	3

tives, even if with notable trade-offs for accuracy compared to state-of-the-art proprietary LLMs.

In conclusion, the outcomes of the quantitative evaluation reveal that while all models perform effectively in routing, proprietary models consistently outperform open-source alternatives in the translations, particularly for formal verification. This suggests that the framework can be implemented with proprietary models such as GPT (as we did in the qualitative evaluation) or with a combination of open-source models that demonstrate satisfactory accuracy for specific tasks. For instance, *Qwen2.5-7B-Instruct* could be used for routing, *gemma-2–9b-it* for simulation task translation, and *Mistral-7B-Instruct-v0.2* for verification task translation. This underscores the framework's adaptability to different technologies, enhancing its overall flexibility.

Qualitative Evaluation: The qualitative evaluation results in Table 3 indicate that the framework provided coherent answers, with only a few participants rating its coherence as suboptimal. No participant judged the understandability of the responses below 4 on the Likert scale, suggesting that the outputs are clear and easily comprehensible, even for users outside the manufacturing domain. Additionally, although the framework performs well in relevance, occasional cases arise where responses do not entirely align with users' expectations. Similarly, while the generated answers are deemed useful, certain responses may lack the depth required to be fully actionable or satisfactory.

The feedback gathered from the open-ended questions pointed out several strengths and weaknesses of the framework. Participants appreciated its ease of use, ability to understand queries even when not perfectly formulated, clarity of the answers, and responsiveness in handling the requests. However, they also provided valuable suggestions for improvement. Key recommendations included supporting a guided interaction feature to help users refine their questions when the framework fails to answer, enabling it to respond. Additionally, they suggested enhancing accessibility for verification tasks, particularly for users without a background in automata theory. In particular, they advised including a visual representation of the production process' automaton when prompting verification questions and offering more context about the supported tasks. Another proposed improvement was to enable a conversational mode that retains the context of previous questions and answers, moving beyond the current transactional interaction supported by the framework.

5 Conclusion

This paper introduces a conversational framework combining LLMs with reasoning engines for accurate, multi-perspective analysis of production systems. Experimental results highlight LLMs as valuable complements to reliable reasoning engines.

Future work will expand the framework with additional reasoners for advanced analyses and explore leveraging reasoning tool feedback to enhance problem formulation accuracy. This can be achieved through a dual approach: allowing expert users to review and refine LLM-generated translations, and implementing an automatic feedback loop between the LLM and the reasoner's debugger for the non-technical users, enabling iterative refinement and error resolution. We also plan to explore larger-scale automata of production processes and investigate potential bottlenecks in the reasoning engines to enhance the framework's scalability and performance. In addition, since our framework allows reasoning engines to operate independently and handle very specific analysis requests, we aim to explore new solutions for interpreting requests involving multiple reasoning engines at once and unifying their results into a cohesive output. In a future extension, we also plan to conduct a comprehensive qualitative evaluation involving experts in production systems and real users from industry.

A potential threat to the validity of this work lies in the rapid advancements in LLM development, potentially rendering the results obsolete over time. To address this, we evaluated numerous state-of-the-art models, both open-source and proprietary, demonstrating the framework's versatility and independence from specific technologies. Another concern arises from the statistical variability in LLM outputs for identical prompts: to mitigate this, we conducted multiple runs for each experiment and made the employed prompts available online.

Reproducibility. The source code, datasets, and instructions for replicating the experiments are available at: https://github.com/angelo-casciani/conv_automata.

Acknowledgments. This work was supported by the H2020 project DataCloud (Grant 101016835), the Sapienza project FOND-AIBPM, the PRIN 2022 project MOTOWN, the PNRR MUR project PE0000013-FAIR, and the EU Horizon Europe Research and Innovation Programme under Grant 101092021 (AutoTwin). The work of A. Casciani is in the range of the Italian National Doctorate on AI run by Sapienza.

References

1. Achiam, J., Adler, S., Agarwal, S., Ahmad, L., et al.: GPT-4 Technical Report. arXiv preprint (2023). https://arxiv.org/abs/2303.08774
2. Agarwal, S., Sreepathy, A., Alonso, D.H., Lamba, P.: LLM+reasoning+planning for supporting incomplete user queries in presence of APIS. arXiv preprint (2024). https://doi.org/10.48550/ARXIV.2405.12433

3. Agostinelli, S., et al.: A context-aware framework to support decision-making in production planning. In: 36th International Conference on Advanced Information Systems Engineering, CAiSE 2024, pp. 248–264 (2024). https://doi.org/10.1007/978-3-031-61057-8_15
4. Alur, R., Courcoubetis, C., Dill, D.L.: Model-checking in dense real-time. Inf. Comput. **104**(1), 2–34 (1993). https://doi.org/10.1006/INCO.1993.1024
5. Azab, A., Osman, H., Baki, F.: CAPP-GPT: a computer-aided process planning-generative pretrained transformer framework for smart manufacturing. Manufact. Lett. **41**, 51–62 (2024)
6. Beauquier, D.: On probabilistic timed automata. Theor. Comput. Sci. **292**(1), 65–84 (2003). https://doi.org/10.1016/S0304-3975(01)00215-8
7. Casciani, A., Bernardi, M.L., Cimitile, M., Marrella, A.: Conversational systems for AI-augmented business process management. In: Research Challenges in Information Science, pp. 183–200. Springer (2024). https://doi.org/10.1007/978-3-031-59465-6_12
8. Chapela-Campa, D., Dumas, M.: From process mining to augmented process execution. Softw. Syst. Model. **22**(6), 1977–1986 (2023). https://doi.org/10.1007/s10270-023-01132-2
9. Chirkova, N., Troshin, S.: Empirical study of transformers for source code. In: ESEC/FSE '21: 29th ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering, pp. 703–715. ACM (2021). https://doi.org/10.1145/3468264.3468611
10. Dong, Q., Li, L., Dai, D., Zheng, C., et al.: A survey on in-context learning. In: 2024 Conference on Empirical Methods in Natural Language Processing, EMNLP, pp. 1107–1128 (2024). https://aclanthology.org/2024.emnlp-main.64
11. Dubey, A., Jauhri, A., Pandey, A., Kadian, A., et al.: The llama 3 herd of models. arXiv preprint (2024). https://arxiv.org/abs/2407.21783
12. Frigerio, N., Tan, B., Matta, A.: Simultaneous control of multiple machines for energy efficiency: a simulation-based approach. Int. J. Prod. Res. **62**(3), 933–948 (2024). https://doi.org/10.1080/00207543.2023.2175175
13. Gao, Y., Xiong, Y., Gao, X., Jia, K., et al.: Retrieval-augmented generation for large language models: a survey. arXiv preprint (2024). https://arxiv.org/abs/2312.10997
14. Golovneva, O., et al.: ROSCOE: a suite of metrics for scoring step-by-step reasoning. In: The Eleventh International Conference on Learning Representations, ICLR 2023 (2023)
15. Goujon, A., Rosin, F., Magnani, F., Lamouri, S., Pellerin, R., and, L.J.: Industry 5.0 use cases development framework. Int. J. Prod. Res. **62**(17), 6064–6089 (2024). https://doi.org/10.1080/00207543.2024.2307505
16. Gupta, A., Sheth, I., Raina, V., Gales, M., Fritz, M.: LLM task interference: an initial study on the impact of task-switch in conversational history. arXiv preprint (2024). https://arxiv.org/abs/2402.18216
17. Han, Z., Gao, C., Liu, J., Zhang, J., Zhang, S.Q.: Parameter-efficient fine-tuning for large models: a comprehensive survey. arXiv preprint (2024). https://arxiv.org/abs/2403.14608
18. Jiang, A.Q., Sablayrolles, A., Mensch, A., Bamford, C., et al.: Mistral 7B. arXiv preprint (2023). https://doi.org/10.48550/ARXIV.2310.06825
19. Kambhampati, S., Valmeekam, K., Guan, L., Verma, M., et al.: Position: LLMs can't plan, but can help planning in LLM-modulo frameworks. In: Forty-first International Conferenc on Machine Learning, ICML (2024). https://openreview.net/forum?id=Th8JPEmH4z

20. Keskin, Z., et al.: LLM-enhanced human-machine interaction for adaptive decision-making in dynamic manufacturing process environments. IEEE Access **13**, 44650–44661 (2025). https://doi.org/10.1109/ACCESS.2025.3549529
21. Kojima, T., Gu, S.S., Reid, M., Matsuo, Y., Iwasawa, Y.: Large language models are zero-shot reasoners. In: Advances in Neural Information Processing Systems (NeurIPS) (2022). https://dl.acm.org/doi/10.5555/3600270.3601883
22. Larsen, K.G., Pettersson, P., Yi, W.: UPPAAL in a nutshell. Int. J. Softw. Tools Technol. Transf. **1**(1–2), 134–152 (1997). https://doi.org/10.1007/S100090050010
23. van der Lee, C., Gatt, A., van Miltenburg, E., Krahmer, E.: Human evaluation of automatically generated text: current trends and best practice guidelines. Comput. Speech Lang. **67**, 101151 (2021). https://doi.org/10.1016/J.CSL.2020.101151
24. Lestingi, L., et al.: Analyzing the impact of human errors on interactive service robotic scenarios via formal verification. Softw. Syst. Model. **23**(2), 473–502 (2024). https://doi.org/10.1007/S10270-023-01125-1
25. Lewis, P.S.H., Perez, E., Piktus, A., Petroni, F., et al.: Retrieval-augmented generation for knowledge-intensive NLP tasks. Adv. Neural Inf. Process. Syst. **33** (NeurIPS) (2020). https://dl.acm.org/doi/abs/10.5555/3495724.3496517
26. Li, Y., Zhao, H., Jiang, H., Pan, Y., et al.: Large language models for manufacturing. arXiv preprint (2024). https://arxiv.org/abs/2410.21418
27. Liu, M.X., et al.: We need structured output: towards user-centered constraints on large language model output. In: Ext. Abstracts of the CHI Conference on Human Factors in Computing Systems, CHI, pp. 10:1–10:9. ACM (2024). https://doi.org/10.1145/3613905.3650756
28. Matloff, N.: Introduction to discrete-event simulation and the simPY language. Davis CA. Dept. Comput. Sci . Univ. Calif. Davis **2**(2009), 1–33 (2008)
29. McKenzie, I.R., Lyzhov, A., Pieler, M., Parrish, A., et al.: Inverse scaling: when bigger isn't better. Trans. Mach. Learn. Res. **2023** (2023). https://openreview.net/forum?id=DwgRm72GQF
30. Mindell, D., Beland, C., Chan, W., Clarke, D., Park, R., Trupiano, M.: LEGO Mindstorms. In: The Structure of an Engineering (R) evolution (2000)
31. Mirzadeh, S., Alizadeh, K., Shahrokhi, H., Tuzel, O., et al.: GSM-symbolic: understanding the limitations of mathematical reasoning in large language models. arXiv preprint (2024). https://doi.org/10.48550/ARXIV.2410.05229
32. Negri, E., Berardi, S., Fumagalli, L., Macchi, M.: MES-integrated digital twin frameworks. J. Manuf. Syst. **56**, 58–71 (2020)
33. Nezhurina, M., Cipolina-Kun, L., Cherti, M., Jitsev, J.: Alice in wonderland: simple tasks showing complete reasoning breakdown in state-of-the-art large language models. arXiv preprint (2024). https://doi.org/10.48550/ARXIV.2406.02061
34. Parnami, A., Lee, M.: Learning from few examples: a summary of approaches to few-shot learning. arXiv preprint (2022). https://doi.org/10.48550/ARXIV.2203.04291
35. Qwen Team: Qwen2.5: A party of foundation models (2024). https://qwenlm.github.io/blog/qwen2.5/
36. Rawte, V., Sheth, A., Das, A.: A survey of hallucination in large foundation models. arXiv preprint (2023). https://arxiv.org/abs/2309.05922
37. Ribeiro, D.N., et al.: STREET: a multi-task structured reasoning and explanation benchmark. In: The Eleventh International Conference on Learning Representations, ICLR 2023 (2023)
38. Rivière, M., Pathak, S., Sessa, P.G., Hardin, C., et al.: Gemma 2: improving open language models at a practical size. arXiv preprint (2024). https://doi.org/10.48550/ARXIV.2408.00118

39. Shanahan, M., McDonell, K., Reynolds, L.: Role play with large language models. Nature **623**(7987), 493–498 (2023). https://doi.org/10.1038/S41586-023-06647-8
40. Sun, Y., Zhang, Q., Bao, J., Lu, Y., Liu, S.: Empowering digital twins with large language models for global temporal feature learning. J. Manuf. Syst. **74**, 83–99 (2024)
41. Sun, Y., et al.: Development of an intelligent design and simulation aid system for heat treatment processes based on LLM. Mater. Des. **248**, 113506 (2024). https://doi.org/10.1016/j.matdes.2024.113506
42. Trinh, T.H., Wu, Y., Le, Q.V., He, H., Luong, T.: Solving Olympiad geometry without human demonstrations. Nature **625**(7995), 476–482 (2024). https://doi.org/10.1038/S41586-023-06747-5
43. Tuli, M., Li, A.C., et al.: Learning to follow instructions in text-based games. Adv. Neural Inf. Process. Syst. **35** (NeurIPS) (2022). https://dl.acm.org/doi/10.5555/3600270.3601683
44. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., et al.: Attention is all you need. Adv. Neural Inf. Process. Syst. (NeurIPS) **30** (2017)
45. Wang, J., Ma, W., Sun, P., Zhang, M., Nie, J.: Understanding user experience in large language model interactions. arXiv preprint (2024). https://doi.org/10.48550/ARXIV.2401.08329
46. Wang, Z., Qin, H.: Intelligent industrial production process automatic regulation system based on LLM agents. In: 2024 5th International Conference on Artificial Intelligence and Electromechanical Automation (AIEA), pp. 133–137. IEEE (2024)
47. White, J., Fu, Q., Hays, S., Sandborn, M., et al.: A prompt pattern catalog to enhance prompt engineering with ChatGPT. arXiv preprint (2023). https://doi.org/10.48550/ARXIV.2302.11382
48. Wu, S., Oltramari, A., Francis, J., Giles, C.L., Ritter, F.E.: Cognitive LLMs: towards integrating cognitive architectures and large language models for manufacturing decision-making. arXiv preprint (2024). https://arxiv.org/abs/2408.09176
49. Xia, Y., Dittler, D., Jazdi, N., Chen, H., Weyrich, M.: LLM experiments with simulation: large language model multi-agent system for simulation model parametrization in digital twins. In: 2024 IEEE 29th International Conference on Emerging Technologies and Factory Automation (ETFA), pp. 1–4 (2024)
50. Xia, Y., Shenoy, M., Jazdi, N., Weyrich, M.: Towards autonomous system: flexible modular production system enhanced with large language model agents. In: 28th IEEE International Conference on Emerging Technologies and Factory Automation, ETFA, pp. 1–8 (2023). https://doi.org/10.1109/ETFA54631.2023.10275362
51. Xia, Y., Zhang, J., Jazdi, N., Weyrich, M.: Incorporating large language models into production systems for enhanced task automation and flexibility. arXiv preprint (2024). https://doi.org/10.48550/ARXIV.2407.08550
52. Zhang, L., Zhou, L., Ren, L., Laili, Y.: Modeling and simulation in intelligent manufacturing. Comput. Ind. **112**, 103123 (2019). https://doi.org/10.1016/j.compind.2019.08.004
53. Zhao, W.X., Zhou, K., Li, J., Tang, T., et al.: A survey of large language models. arXiv preprint (2023). https://doi.org/10.48550/ARXIV.2303.18223

Process Monitoring

Achieving Group Fairness Through Independence in Predictive Process Monitoring

Jari Peeperkorn(✉) and Simon De Vos

Research Center for Information Systems Engineering (LIRIS), KU Leuven, Leuven, Belgium
jari.peeperkorn@kuleuven.be

Abstract. Predictive process monitoring focuses on forecasting future states of ongoing process executions, such as predicting the outcome of a particular case. In recent years, the application of machine learning models in this domain has garnered significant scientific attention. When using historical execution data, which may contain biases or exhibit unfair behavior, these biases may be encoded into the trained models. Consequently, when such models are deployed to make decisions or guide interventions for new cases, they risk perpetuating this unwanted behavior. This work addresses group fairness in predictive process monitoring by investigating independence, i.e. ensuring predictions are unaffected by sensitive group membership. We explore independence through metrics for demographic parity such as ΔDP, as well as recently introduced, threshold-independent distribution-based alternatives. Additionally, we propose a composite loss function existing of binary cross-entropy and a distribution-based loss (Wasserstein) to train models that balance predictive performance and fairness, and allow for customizable trade-offs. The effectiveness of both the fairness metrics and the composite loss functions is validated through a controlled experimental setup.

Keywords: Process Mining · Predictive Process Monitoring · Fairness · Machine Learning

1 Introduction

Predictive Process Monitoring (PPM) is a branch of process mining that aims to predict the future state of ongoing business processes based on historical event data. A specialized subset of PPM, known as Outcome-Oriented Predictive Process Monitoring (OOPPM), focuses on predicting specific outcomes or labels of process instances. By leveraging historical cases, OOPPM enables organizations to anticipate critical outcomes and intervene at earlier stages of the process, enabling earlier interventions that improve efficiency or reduce risks. Recent advancements in OOPPM have predominantly utilized machine and deep learning models trained on labeled historical data to achieve accurate predictions.

J. Krogstie et al. (Eds.): CAiSE 2025, LNCS 15701, pp. 185–203, 2025.
https://doi.org/10.1007/978-3-031-94569-4_11

A significant ethical and legal challenge arises when these models are trained on biased data, which may encode systemic inequalities, such as those based on gender or ethnicity. Predictive models can inadvertently reproduce or even exacerbate these disparities if fairness considerations are not addressed. In recent years, different works have focused on using process mining to detect or discover such fairness concerns within a process [22,29]. This work focuses on ensuring OOPPM models produce fair and unbiased predictions. While improving the fairness of the process execution itself is not the primary goal of these approaches, it is assumed that efforts are made in parallel to enhance the fairness of the process. Consequently, our approaches can then be used to measure and improve the extent to which the classifier's outcomes reflect these fairer executions, rather than perpetuating historical biases. For example, in a hiring process, if a model is trained on historically biased data, such as gender bias, it may carry that bias into its predictions, potentially reinforcing discrimination during early interventions. These biases can be direct, if the model decisions are (partially) based on protected attributes, or indirect, if decisions affect protected groups even without explicitly using their related attributes. From a legal and ethical perspective, tackling these is essential for organizations to align with fairness mandates, such as those outlined in regulations like the EU AI Act [10].

A key fairness criterion discussed in this work is group fairness through independence, which ensures equal predictive outcomes across protected groups (e.g., defined by gender or ethnicity). Achieving fairness often involves trade-offs, as enforcing fairness constraints may reduce predictive performance [5,17]. To navigate this tension, this work evaluates the fairness-performance trade-offs, providing a principled framework for balancing these competing objectives. This approach allows stakeholders to make informed policy decisions about the desired level of fairness enforcement, aligning with a risk-based perspective that can vary depending on the application and its regulatory environment. By addressing these challenges, this paper aims to contribute to the development of fair and effective OOPPM models that align predictive capabilities with ethical and legal fairness standards and to offer tools for practitioners and researchers to do so as well. To this extent, the main contributions of this work are:

– Introducing group fairness into predictive process monitoring, with independence as the fairness criterion.
– Proposing and evaluating metrics for *demographic parity* such as ΔDP, alongside more advanced, threshold-independent alternatives *area between probability density function curves* (ABPC) and *area between cumulative density function curves* (ABCC).
– Incorporating *integral probability metrics* (IPMs) into a composite loss function, complementing traditional loss functions such as binary cross-entropy. Experiments demonstrate that balancing IPMs with traditional loss functions enables flexible trade-offs between fairness and predictive accuracy.

The rest of this work is structured as follows: related work and notation are introduced in Sect. 2, followed by an in-depth introduction to the independence measures used in Sect. 3. The predictive setup is discussed in Sect. 4, followed by a proof-of-concept in 5. Some practical considerations and guidelines are considered in Sect. 6, before concluding in Sect. 7. The adjustable and modular code is made available online, together with full experimental results[1].

2 Background

2.1 Preliminaries

Executed *activities* in a process are recorded as events in an event log L. Each *event* belongs to one *case*, identified by its *CaseID* $c \in C$. An event e can be expressed as a tuple $e = (c, a, t, d, s)$, where $a \in A$ represents the *activity* (i.e., the *control-flow* attribute) and t is the timestamp of the event. Optionally, an event might have associated *event-related attributes* $\mathbf{d} = (d_1, d_2, \ldots, d_{m_d})$, which are dynamic attributes that are event-specific (such as the resource executing the activity). Conversely, *static attributes* $\mathbf{s} = (s_1, s_2, \ldots, s_{m_s})$ are case-level attributes that do not change during the execution of a case (such as customer information). A sequence of events that belong to a single case is referred to as a *trace*. The outcome y of a trace is an attribute defined by the process owner, often binary in nature, indicating whether a specific criterion has been met [39]. A *prefix* is a portion of a trace, consisting of the first l events, where l is an integer smaller than the trace length. In summary, we are working with a data set $\mathcal{D} = \{(\mathbf{x}_i, y_i, s_i)\}_{i=1}^{N}$. Here, $\mathbf{x}_i \in \mathbb{R}^d$ is the (encoded) input feature set for one sample, in this setup the full prefix sequence and y_i is the true outcome of the case from which that prefix is derived. Let $s_i \in \{0, 1\}$ be the sensitive attribute (i.e., a static case feature) value of sample i that defines $\mathcal{S}_0 = \{i : s_i = 0\}$ and $\mathcal{S}_1 = \{i : s_i = 1\}$ as index sets of entities that, according to this sensitive feature S, belong to group 0 or group 1 respectively. We train a model $m : \mathbb{R}^d \rightarrow [0, 1]$ that provides a propensity $\hat{y} \in [0, 1]$. The model's output propensity scores may not always represent true probabilities, particularly if the model is not well-calibrated. In this work, however, due to its model-agnostic approach, we work with the direct output propensities.

2.2 Related Works

Predictive process monitoring (PPM) addresses various tasks such as predicting the remaining time of a process [42], identifying the next activity [3,38], or determining the final outcome of a process [8,18,39]. Different approaches in the literature range from using finite state machines [40] and stochastic Petri nets [31] to machine learning techniques such as regression trees [7] and ensembles [35]. In recent years, lots of focus has been placed on deep learning techniques. Due to the natural fit with the sequential process data, recurrent neural networks [3,38] and, more recently, transformer nets [46] have garnered a lot of attention. In addition to technical advancements in PPM, the importance of addressing fairness

[1] https://github.com/jaripeeperkorn/Group-Fairness-in-Predictive-Process-Monitoring.

within process mining has gained attention. Fairness in process mining focuses on among others identifying and mitigating biases that may lead to discriminatory outcomes in processes or their analyses, within the broader context of AI regulatory compliance [26]. [29] categorizes fairness concepts from machine learning, applies them to process mining, and highlights key fairness-related challenges for process mining. Other work proposes a fair classifier for root cause analysis in processes [30], or an adapted genetic process discovery algorithm optimized for group fairness [22]. Recently, [28] introduced a collection of simulated event logs designed to address the scarcity of fairness-aware datasets in process mining. Within PPM, integrating adversarial debiasing has been proposed to mitigate the influence of certain variables on biased predictions [19], and different metrics have been explored as well [34].

3 Group Fairness Through Independence

Predictive models used in PPM can exhibit discriminatory behavior toward specific groups. Such biases often stem from historical patterns of systematic disadvantage faced by certain populations. This work focuses on group-level fairness measures, like independence, which aim to equalize outcomes across protected groups, addressing systemic inequalities at the population level. Next to this, individual fairness, ensuring similar treatment for similar cases, provides a more granular perspective and requires different approaches such as generating counterfactuals [9]. Features defining the protected group(s) may be present explicitly or implicitly. When explicitly included, removing sensitive attributes alone is still often insufficient to ensure fairness, as the principle of *"fairness through unawareness"* has been shown to be ineffective [1,4,9]. Other features in the dataset could correlate with the *sensitive* attribute, allowing its value to be inferred through these so-called proxy features, which can perpetuate bias. In PPM, such inference might stem from complex control-flow patterns in historical cases. Sometimes fully excluding these features can also lead to unexpected results, such as in the case of Simpsons's paradox (e.g., when gender correlates with physical attributes such as length) [24]. Fairness in machine learning can be defined in various ways. Two other prominent approaches, other than independence, are separation and sufficiency. Separation corresponds to the idea of *error rate parity* (for both false positive and false negative rates). Sufficiency, in short, means that predictions should be independent of the sensitive attribute given the actual outcome, ensuring *calibration fairness*. Comprehensive reviews of these fairness definitions can be found in works such as [21] and [1]. For the rest of this work, we will focus on independence.

3.1 Demographic Parity Metrics

In machine learning, demographic or statistical parity is defined as independence on group-level [21]. This requires that predicted probabilities are independent of sensitive attributes. Commonly used metrics to measure demographic parity violations include the average propensity difference between the two groups, denoted as ΔDP_c (continuous) and defined by Eq. 1, and the difference in the proportion of positive predictions between the two groups, dubbed ΔDP_b^t (binary at threshold t) and defined by Eq. 2 [13,15,47,48]. Here, $\mathbf{1}(\hat{y}_n > t)$ is the indicator function that equals 1 if the predicted value exceeds threshold t and 0 otherwise.

$$\Delta DP_c = \left| \frac{\sum_{n \in \mathcal{S}_0} \hat{y}_n}{|\mathcal{S}_0|} - \frac{\sum_{n \in \mathcal{S}_1} \hat{y}_n}{|\mathcal{S}_1|} \right| \quad (1)$$

$$\Delta DP_b^t = \left| \frac{\sum_{n \in \mathcal{S}_0} \mathbf{1}(\hat{y}_n > t)}{|\mathcal{S}_0|} - \frac{\sum_{n \in \mathcal{S}_1} \mathbf{1}(\hat{y}_n > t)}{|\mathcal{S}_1|} \right| \quad (2)$$

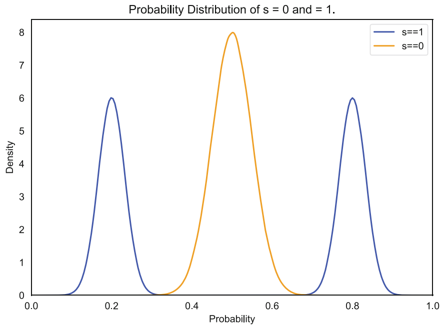
To clarify consider an early intervention PPM model in hiring, where a process is automatically accepted or rejected if the prediction exceeds or falls below a threshold t. ΔDP_b^t measures the difference in intervention rates between e.g. gender groups, while ΔDP_c captures the difference in the average model propensity for both groups. A limitation of ΔDP_b^t is its dependence on the classification threshold t. Threshold-sensitive fairness metrics might not fully capture disparities across the entire range of outputs, which is particularly important in dynamic settings like PPM where t can vary due to application-specific properties such as changing cost-benefit ratios or scarce resources. Such settings require distribution-based approaches to ensure flexibility post-training or even post-deployment. For example, when optimizing profit, cost-sensitive thresholding has been demonstrated to be effective [41]. To address these limitations, recent work has shifted towards evaluating fairness across the entire output distribution. This approach allows predictive models independently of the chosen threshold [13]. Therefore, following [13], we adopt the metrics *area between probability density function curves* (ABPC) and *area between cumulative density function curves* (ABCC).

$$\text{ABPC} = \int_0^1 |f_0(x) - f_1(x)| \, \mathrm{d}x \quad (3) \qquad \text{ABCC} = \int_0^1 |F_0(x) - F_1(x)| \, \mathrm{d}x \quad (4)$$

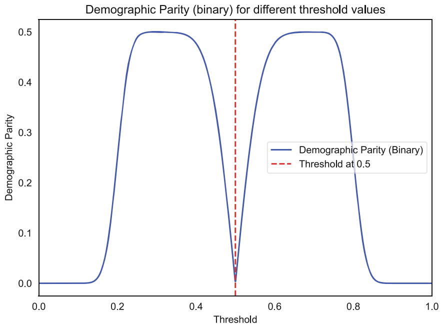
Here, $f_0(\cdot)$ and $f_1(\cdot)$ represent the probability density functions (PDFs) of protected groups 0 and 1, respectively, and $F_0(\cdot)$ and $F_1(\cdot)$ denote their corresponding cumulative density functions (CDFs). ABPC values range between 0 (full parity) and 2, and ABCC values between 0 and 1. Since we are working with finite samples, we make estimations in practice. For ABPC we estimate the PDFs via kernel density estimation (KDE) [32], and for ABCC directly use the empirical CDFs [44], following [13]. Both metrics are calculated using the composite trapezoidal rule[2].

Figure 1 demonstrates the advantage of using distribution-based metrics, next to their threshold-independence. The left subplot shows the propensity distributions for two groups (e.g. for male and female candidates in a hiring process).

[2] Sufficient precision is ensured by using 10,000 steps per integration [13].



(a) Propensity distributions for $\mathcal{S}_0$ and $\mathcal{S}_1$



(b) ΔDP_b^t scores in function of t

Fig. 1. A toy example showing the need for threshold-free independence metrics.

Although $\Delta DP_c = 0$, the distributions are clearly not independent of the sensitive attribute, as reflected by ABPC and ABCC values of 2.0 and 0.26, respectively. The right subplot illustrates how ΔDP_b^t varies significantly with threshold t, highlighting its sensitivity to the choice of t.

3.2 Integral Probability Metric Loss

To incorporate fairness as defined above into the learning process, we include IPMs [23,36] to quantify the distance between the two prediction distributions $p(\hat{y}|s = 0)$ and $p(\hat{y}|s = 1)$. IPMs have been earlier applied in machine learning to, e.g., match output and true probability densities of binary classifiers [37], or to learn balanced covariate representations in causal inference [33]. Additionally, they have been recently employed to improve fairness metrics like demographic parity [15]. During training, IPMs are incorporated into the model's objective function through a composite loss formulation:

$$\mathcal{L}_{\text{total}} = (1 - \lambda) \cdot \mathcal{L}_{\text{BCE}} + \lambda \cdot \mathcal{L}_{\text{IPM}} \tag{5}$$

Here, $\mathcal{L}_{\text{BCE}}$ denotes a standard supervised learning loss, such as binary cross-entropy (BCE), which measures the predictive performance of the model. On the other hand, $\mathcal{L}_{\text{IPM}}$ penalizes discrepancies between the prediction distributions across protected groups. The hyperparameter $\lambda \in [0, 1]$ governs the trade-off between maximizing predictive accuracy and ensuring fairness. By incorporating $\mathcal{L}_{\text{IPM}}$ into the training process, the model is guided not only to minimize traditional predictive losses but also to satisfy a soft fairness constraint, encouraging more equal outcomes across protected groups. In the hiring process example, $\mathcal{L}_{\text{BCE}}$ trains early predictions to match (potentially biased) outcomes, while $\mathcal{L}_{\text{IPM}}$ encourages similar propensity distributions across groups. Adding a penalty proportional to the difference between the two propensity distributions nudges the model toward aligning the distributions more closely, thereby reducing independence violations. In this work, we employ the Wasserstein distance (also known as Earth Mover's Distance) as the IPM. The Wasserstein distance is a measure of the effort required to transform one probability distribution into another, taking into account where (at which probabilities) in the distribution the differences occur, i.e., if further apart, the effort should be higher. The optimal mass transformation from one distribution to the other is determined by the area between the cumulative distribution functions. In this way, using Wasserstein as IPM loss is more aligned with optimizing ABCC than with optimizing ABPC. For a formal definition, the reader is referred to [43]. The Wasserstein distance provides a geometrically intuitive and robust way to measure the distributional discrepancy, making it well-suited for enforcing group fairness. In this work, the more efficient Sinkhorn approximation is used, as implemented by [33], allowing it to be used inside the training loop as well. Alternatives to measure distributional differences include Kullback-Leibler (KL) Divergence [2] and the kernel-based Maximum Mean Discrepancy (MMD) [11].

4 Methodology

4.1 Preprocessing

The event logs are divided into training and test sets at the case level, with an 80–20% split. While fully correct practice involves removing overlapping cases and biased cases near the end of the test set (as described in [45]), this step is omitted due to the use of artificially simulated data and the focus on conceptual demonstrations rather than predictive performance. Cases are converted into prefix-outcome pairs. Prefixes are handled by defining a maximum length: shorter prefixes are right-padded, and longer ones are left-truncated. Outcomes are defined based on the presence of specific activities in the process executions. Only prefixes up to but not including the target activity are included in the dataset. A sensitive feature is selected to determine the protected groups. The implementation allows for including or excluding this feature from the input space. Static case features are included as input at every event in a case. Numerical features are min-max scaled to the range $[0, 1]$, while non-binary categorical features (e.g., activity, resource) are one-hot encoded and passed through separate embedding layers. A validation set is used for hyperparameter tuning, early stopping, and threshold tuning. This validation set is created by splitting 20% of the prefix-outcome pairs from the training samples.

4.2 Model

An LSTM model [14] was selected for these conceptual experiments, due to its widespread use in PPM research and its natural suitability for sequential data. However, the metrics and adjusted loss functions proposed in this work are compatible with various other models. A graphical representation of the LSTM classifier used in our experiments is shown in Fig. 2. Categorical features are passed through individual trainable embedding layers with adjustable sizes. At each timestep, the outputs of these embeddings are combined with binary and

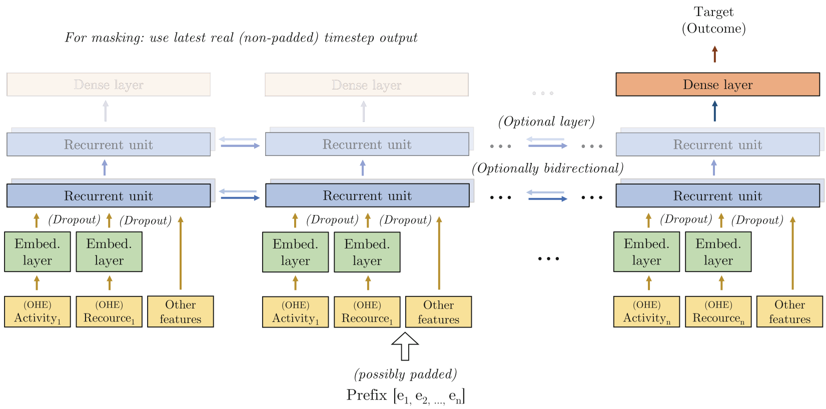


Fig. 2. A graphical depiction of the LSTM model used.

numerical features to form the input to a recurrent LSTM layer, with dropout applied for regularization. The LSTM layer can be configured as bidirectional (processing the sequence in both forward and backward directions), and multiple layers can be stacked. The output from the final LSTM unit is fed into a dense layer to produce a binary propensity. To handle right-padded sequences, masking is employed, ensuring that the output of the LSTM corresponds to the last valid (unpadded) event.

The models are trained using the AdamW optimizer [20]. A learning rate scheduler is applied to reduce the learning rate by a factor of 0.75 every 10 epochs without improvement (greater than 0.001). Early stopping based on validation loss is employed, with a patience setting of 20 for hyperparameter tuning and 50 for the main experiments. Training is capped at 300 epochs. The loss function used is BCE, or, if $\lambda > 0$, a combination of BCE and the Wasserstein distance as IPM (as defined in Eq. 5).

Hyperparameters. The model has several hyperparameters that were optimized via grid search, including: the number of LSTM layers (1 or 2), the bidirectionality of this layer, and its size (16, 32 or 64), the mini-batch size (128, 256, 512), the starting learning rate (0.0001 or 0.001) and the dropout (0.2 or 0.4). The embedding layer size is set to the square root of the vocabulary size for each categorical feature. The grid search is performed once per event log using the BCE loss, and hyperparameters are selected using the validation set Area-under-the-curve (AUC) score. These are subsequently used in all experiments involving the corresponding event log. For experiments involving the IPM-composite loss functions, it was opted to keep the batch size fixed at 512 to ensure reliable computation of batch-level statistics.

5 Experimental Setup and Results

To validate the proposed metrics and the use of the composite loss, we conducted proof-of-concept experiments. These are not aimed to showcase predictive performance, but to demonstrate the potential of the fairness metrics and IPM loss, which are modular and adaptable across models. Experiment 1 demonstrates how the various independence metrics described above can be applied in a predictive process monitoring context, highlighting their potential differences. Experiment 2 illustrates how independence can be improved by incorporating the IPM loss.

5.1 Datasets

To ensure a controlled setup, we utilized several artificial datasets introduced in [28], which were designed to represent varying degrees of discrimination [27]. These event logs were selected because of the different levels of bias they display within the same process. Since the focus of this experiment is to demonstrate the trade-off between independence and predictive performance, the selected event logs are well-suited despite their artificial nature. Their varying levels of bias within the same process effectively showcase how to measure and navigate this trade-off. Three processes were selected from the available event logs[3]. The first is a hiring process, which simulates recruitment workflows where the successful outcome is marked by the *Make Job Offer* activity. The second process concerns loan applications, where a positive outcome corresponds to cases containing the *Sign Loan Agreement* activity. The third process involves rental applications, with the *Sign Contract* activity indicating a successful outcome. Some process executions include events occurring after the outcome-determining activity; these were excluded when creating prefixes in the dataset.

Table 1. Information on the event logs [27], sensitive feature is *protected*.

Name	Training (& Validation) Set					Test Set				
	#Pref.	%+	% $\mathcal{S}_1$	%$\mathcal{S}_0^+$	%$\mathcal{S}_1^+$	#Pref.	%+	% $\mathcal{S}_1$	%$\mathcal{S}_0^+$	%$\mathcal{S}_1^+$
hiring_high	40268	41.24	19.94	48.86	10.66	10090	40.58	20.46	48.02	11.68
hiring_medium	43753	46.05	15.71	50.61	21.58	10983	46.04	16.59	50.54	23.44
hiring_low	45633	50.01	9.24	51.44	35.97	11566	50.07	9.43	51.27	38.50
lending_high	36999	24.51	30.44	32.33	6.63	9196	25.13	32.11	33.69	7.04
lending_medium	36582	27.64	21.16	31.51	13.23	9101	28.48	21.02	32.62	12.91
lending_low	37994	30.52	9.78	31.73	19.35	9480	30.75	9.48	31.33	25.25
renting_high	40516	28.91	25.67	34.49	12.78	10132	30.72	26.32	36.33	15.04
renting_medium	42089	42.21	9.68	43.56	29.66	10533	41.37	10.29	43.17	25.65
renting_low	41723	34.87	18.06	37.91	21.08	10413	34.50	18.79	37.54	21.41

[3] The hospital process was excluded because its outcome variable did not correlate directly with the sensitive features.

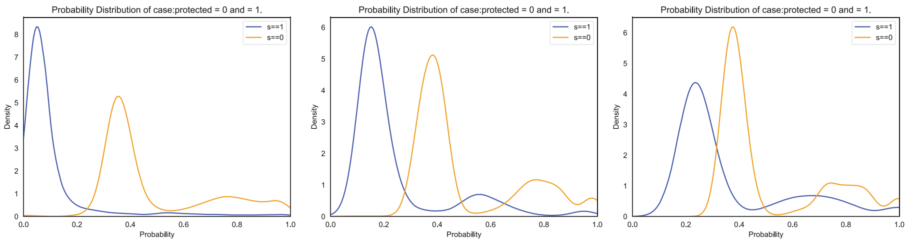
Each event log comprises 10,000 cases. In addition to activity labels and resource event features, each log contains various binary features that could serve as sensitive attributes, such as *gender* and *religious affiliation*, as well as a *protected* feature that identifies the protected groups. Some logs also include continuous features, such as *age* or *years of education*. For the experiments presented in this paper, the *case:protected* feature was used as the sensitive attribute. However, experiments were also conducted using other binary features (static or case-level); the results of these additional experiments are available in the online repository. Table 1 provides an overview of the event logs, including the number of prefixes in both the training and test datasets, the percentage of prefixes with positive (successful) outcomes, and the percentage of prefixes corresponding to cases belonging to the protected group $\mathcal{S}_1$, corresponding to cases for which *case:protected* = *True*. Additionally, the metrics $\%\mathcal{S}_0^+$ and $\%\mathcal{S}_1^+$ denote the percentages of positive outcomes within the two groups. For all of the experiments we kept the maximum prefix length at 6.

5.2 Experiment 1: Assessing Fairness Metrics

The first experiment demonstrates the use of various independence fairness metrics within the OOPPM framework. An LSTM outcome classifier is trained for each event log from Table 1, and evaluated on the test set. The evaluation included standard performance metrics AUC, accuracy, F1 scores, as well as the fairness metrics introduced in Sect. 3.1. For threshold-dependent metrics, results were computed for a fixed threshold of 0.5 ($\mathrm{F1}_{0.5}$, $\mathrm{Acc}_{0.5}$ and $\Delta DP_b^{0.5}$) and the threshold maximizing the F1 score on the validation set ($\mathrm{F1}_{\mathrm{opt.}}$, $\mathrm{Acc}_{\mathrm{opt.}}$ and $\Delta DP_b^{\mathrm{opt.}}$). This experiment not only introduces independence metrics to OOPPM but also highlights the advantages of distribution-based metrics like ABPC and ABCC. Two settings were tested: one where the sensitive feature was included in the input space, and another where it was removed.

Table 2. Results Exp. 1, sensitive feature is *protected* (no removal).

Log	AUC	$\mathrm{F1}_{0.5}$	$\mathrm{F1}_{\mathrm{Opt.}}$	$\mathrm{Acc.}_{0.5}$	$\mathrm{Acc.}_{\mathrm{Opt.}}$	$\Delta DP_b^{0.5}$	$\Delta DP_b^{\mathrm{opt.}}$	ΔDP_{c}	ABPC	ABCC
hiring_high	0.75	0.56	0.64	0.71	0.57	0.25	0.90	0.38	1.71	0.38
hiring_medium	0.72	0.58	0.66	0.69	0.55	0.17	0.80	0.27	1.66	0.27
hiring_low	0.70	0.59	0.67	0.67	0.53	0.05	0.69	0.14	1.30	0.14
lending_high	0.71	0.00	0.53	0.75	0.60	0.00	0.89	0.24	1.85	0.24
lending_medium	0.65	0.00	0.52	0.72	0.53	0.00	0.89	0.18	1.81	0.18
lending_low	0.59	0.07	0.51	0.68	0.43	0.02	0.22	0.11	1.02	0.11
renting_high	0.65	0.12	0.53	0.69	0.55	0.06	0.81	0.22	1.70	0.22
renting_medium	0.61	0.28	0.61	0.59	0.52	0.17	0.42	0.13	1.51	0.13
renting_low	0.63	0.00	0.55	0.65	0.53	0.00	0.86	0.13	1.66	0.13



(a) hiring_high (b) hiring_medium (c) hiring_low

Fig. 3. The LSTM propensity densities for the *hiring* event log.

Table 2 displays results for models trained with sensitive features included. While this may not always reflect realistic use cases, the goal here was to explore the metrics' ability to measure demographic parity violations, even under conditions where the models are explicitly trained on biased data. For the hiring process we can see a clear trend: as bias in the event log decreases, both predictive scores and demographic parity violations reduce. For both the renting and lending logs these trends are a little less pronounced. The renting_medium the DP metrics indicate less bias than renting_low, but this is in line with the higher $\mathcal{S}_1^+$ value in Table 1. Some classifiers produced low accuracy at F1-tuned thresholds, while others resulted in F1 scores of zero at $t = 0.5$, as all samples' propensity scores $\hat{y}$ fell either above or below the threshold, therefore also resulting in $\Delta DP_b^{0.5} = 0$. The scores of $\Delta DP_b^{\text{opt.}}$ are also noticeably larger than $\Delta DP_b^{0.5}$. Figure 3 visualizes propensity distributions for hiring process classifiers across varying bias levels. The distributions show increased overlap, i.e., predictions are more fair, as bias in the event log, and consequently in the classifier, decreases.

Table 3. Results Exp. 1, sensitive feature is *protected* (with removal).

Log	AUC	$F1_{0.5}$	$F1_{\text{Opt.}}$	$\text{Acc.}_{0.5}$	$\text{Acc.}_{\text{Opt.}}$	$\Delta DP_b^{0.5}$	$\Delta DP_b^{\text{opt.}}$	ΔDP_{c}	ABPC	ABCC
hiring_high	0.73	0.56	0.62	0.71	0.57	0.23	0.61	0.22	1.11	0.22
hiring_medium	0.71	0.58	0.64	0.69	0.59	0.15	0.40	0.13	0.75	0.13
hiring_low	0.70	0.59	0.67	0.67	0.52	0.07	0.46	0.10	0.87	0.10
lending_high	0.66	0.00	0.46	0.75	0.54	0.00	0.49	0.09	0.99	0.09
lending_medium	0.62	0.00	0.49	0.72	0.49	0.00	0.45	0.04	0.84	0.04
lending_low	0.59	0.08	0.51	0.68	0.42	0.03	0.02	0.07	0.73	0.07
renting_high	0.60	0.02	0.49	0.69	0.43	0.00	0.06	0.03	0.40	0.03
renting_medium	0.61	0.29	0.61	0.59	0.51	0.14	0.25	0.07	0.93	0.07
renting_low	0.61	0.00	0.54	0.65	0.46	0.00	0.07	0.03	0.39	0.03

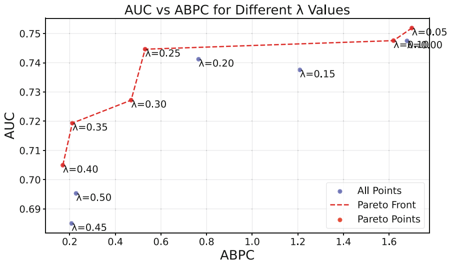
For completeness, we also include results where the sensitive feature was excluded from the dataset, as shown in Table 3. These results reveal significantly lower demographic parity violation scores, at the cost of slightly reduced predictive performance. However, since the protected cases in the event log generation were selected based on other features (e.g., gender), sensitive information persists indirectly through these proxy variables and DP metrics do not fall to 0. This mirrors real-world scenarios where biases often remain embedded in correlated attributes, even after explicit sensitive features are removed. In this experimental setup, removing all binary features to eliminate indirect bias would leave the dataset too sparse for meaningful prediction.

5.3 Experiment 2: Testing IPM Loss

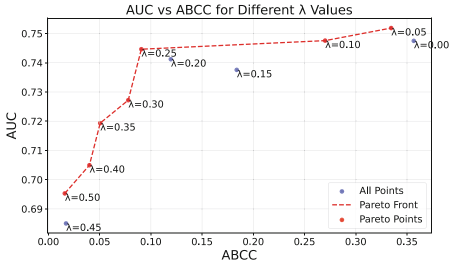
The second experiment evaluates the impact of incorporating the IPM loss, specifically the Wasserstein loss, into the training process. This experiment demonstrates how integrating this loss component reduces demographic parity violations while exploring its trade-offs with predictive performance. To achieve this, the setup uses the three event log variants with the highest bias levels. For each event log, classifiers are trained with varying λ values, controlling the weight of the IPM loss relative to the BCE loss. The λ values range from 0 to 0.5, incremented in steps of 0.05, providing a detailed exploration of its effects. The trained models undergo evaluation on the test set using two primary threshold-independent metrics: AUC to measure predictive capability and ABPC and ABCC to assess demographic parity violations. The results are presented in Fig. 4. Next to a scatter plot, indicating the results for all values of λ, the Pareto points are indicated in red, and connected to show a Pareto front. For points not on the curve, there is at least one result for another λ that scores better on both metrics. Since most points are either on the curve or relatively close to it, the results highlight the trade-off between predictive performance and fairness. Models with low λ values achieve high AUC scores, reflecting strong predictive accuracy, but show significant demographic parity violations. In contrast, higher λ values result in lower AUC scores, indicating reduced predictive performance, but successfully minimizing demographic parity violations, as measured by ABPC and ABCC.

5.4 Discussion

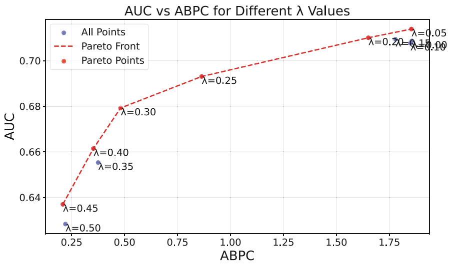
The results for both experiments 1 and 2 underscore the potential utility of the proposed metrics and the IPM loss function respectively. However, some limitations and challenges arise that warrant discussion. The experiments reveal that the predictive information contained in the prefixes may be limited, as evidenced by the low F1 scores and the shape of the propensity density curves. For instance, the propensity distributions in Fig. 3 lack the ideal bi-normal shape characteristic of binary classification. This suggests that class imbalance countermeasures or other probability adjustments may be necessary to optimize classifier performance. It also shows the limitations of using artificial data for classifier



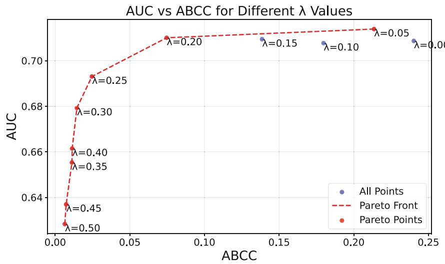
(a) AUC vs ABPC of hiring_high log.



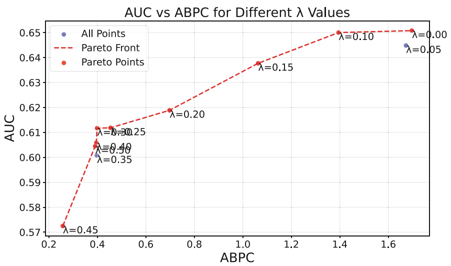
(b) AUC vs ABCC of hiring_high log.



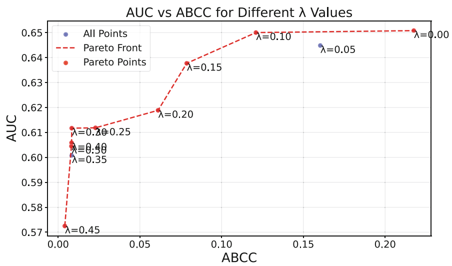
(c) AUC vs ABPC of lending_high log.



(d) AUC vs ABCC of lending_high log.



(e) AUC vs ABPC of renting_high log.

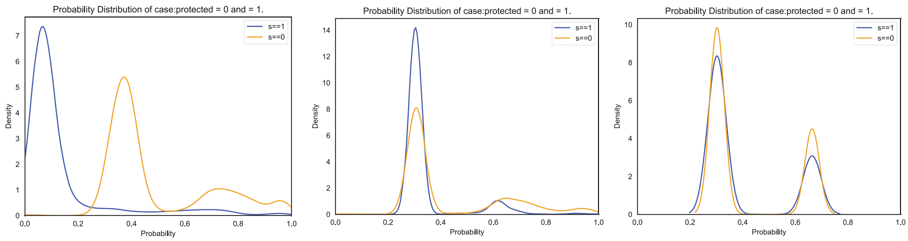


(f) AUC vs ABCC of renting_high log.

Fig. 4. The results for including IPM loss with different values for λ.

experiments. Nevertheless, since the primary objective of the experiments is to demonstrate proof-of-concept for the metrics and loss functions, these limitations do not undermine the core conclusions of the study. The large observed difference between the $\Delta DP_b^{0.5}$ and $\Delta DP_b^{\text{opt.}}$ values, highlight the sensitivity of binary ΔDP to threshold selection. This underscores the advantage of adopting threshold-independent metrics like ABPC and ABCC. One remarkable observation from Tables 2 and 3 is the similarity between ABCC and ΔDP_c scores, which are often rounded to equal values. This similarity is in line with experimental results in the original work [13].

The results of Experiment 2, as illustrated in Fig. 4, clearly demonstrate the trade-off between predictive quality and demographic parity violation when balancing the IPM and BCE loss functions. Most data points align along a clear Pareto front, with minor outliers remaining close to this frontier. Interestingly, for some event logs, small values of λ yielded slight improvements in predictive performance compared to no IPM, potentially due to the regularization effect of IPM. To illustrate the effect of including the IPM loss on the PPM output, in Fig. 5 we plot the propensity distributions of the LSTM model for both protected groups, when trained on the *hiring_high* event log, with different values of λ (0.0, 0.25, and 0.5). We can see the effect of increasing the importance of the IPM loss component, as the distributions of both groups show increased overlap. We also see that the shape of both distributions changes, which might additionally indicate a possible decrease in calibration (difference between propensity and true probability).



(a) $\lambda = 0.0$ (b) $\lambda = 0.25$ (c) $\lambda = 0.5$

Fig. 5. The LSTM propensity densities for the *hiring_high* event log.

6 Practical Considerations

Our work provides a practical framework for practitioners and researchers to assess and improve group fairness in OOPPM models using independence. Simpler methods such as threshold adjustments for different groups or batch-based groupings, are often insufficient for OOPPM. Processes often involve continuous data streams, and predictions are needed as early indicators for all running cases. We propose a principled approach to measuring and enhancing group-level independence using propensity scores or modifying the training process directly. Experimental results indicate that both ΔDP_c and distribution-based metrics ABPC and ABCC can effectively measure demographic parity violations (independent of the chosen threshold). Supported by qualitative reasoning in Sect. 3 and the literature [13], distribution-based metrics are recommended for a more robust evaluation. These metrics can be applied to any predictive model that outputs propensity scores, making them a low-effort addition to existing evaluation pipelines. Practitioners could use these metrics to check for potential biases, as they require only group identification labels and do not necessitate changes to testing workflows. It is important to note that the metrics should ideally be applied to the probability scores used for decision-making, even if these differ from the model's original propensity scores, such as in cases where calibration is applied. One key advantage of using threshold-independent metrics is flexibility. Threshold-independent metrics allow models to be evaluated for fairness while retaining the ability to adjust thresholds down the line for e.g. cost-sensitive purposes [41], without altering conclusions on demographic parity.

To achieve models with greater fairness across protected groups, incorporating the IPM loss into composite loss functions offers a practical trade-off between independence and predictive performance. In a production setting, an optimal choice for λ could be determined based on validation result. Moreover, the trade-off can be tailored to the specific use case and its associated risks regarding independence fairness. However, practitioners may face challenges in choosing an optimal value without extensive validation. Future work could explore automated or adaptive tuning methods to alleviate this dependency. Our implementation of the Wasserstein distance as an IPM loss, built-in PyTorch[4], is readily available to train compatible models (with a gradient-based optimization method). However, we recommend sufficiently large batch sizes (or full batch training) for distribution-based losses, as these rely on batch-level statistics. While computational constraints were not an issue in these experiments, scaling these methods to very large datasets (and batch sizes) could present challenges. Calculating Wasserstein distances or other IPMs for large datasets can be computationally intensive, though approximation techniques like Sinkhorn distances [6] can help.

It is important to note that this work assumes OOPPM models are deployed in contexts where process fairness has been, or is actively being improved. Without such improvements, deploying these models risks perpetuating existing biases rather than mitigating them. For instance, if OOPPM models are used to trigger early interventions that disproportionately keep cases from one protected (minority) group, the remaining cases for this group may exhibit an even lower fraction of positive outcomes compared to counterparts not belonging to that group and the inclusion of other fairness metrics is essential, such as separation or sufficiency. Furthermore, focusing exclusively on group independence during training might unintentionally reduce accuracy for certain groups, potentially even conflicting with other fairness definitions [1]. Balancing these competing objectives requires careful consideration and alignment with the specific goals and risks of the application context.

As mentioned earlier, another important aspect to note is that model outputs (propensities) should only be interpreted as probabilities for well-calibrated models. Calibration depends on the classifier, and neural network models (e.g., LSTMs) have shown mixed results [12]. Miscalibration may worsen when models are trained using multi-objective criteria such as done here with the composite loss function, since fairness objectives can reshape output distributions at the cost of calibration. Look for example at the propensity distributions found in Fig. 5, where increasing the value for λ alters the shape of both distributions. Hence, decision-makers should interpret such outputs with caution, and further investigation on the effects of calibration on decision making in PPM would be interesting.

[4] Version 2.5.1.

7 Conclusion and Future Work

This work introduces group independence and demographic parity violation metrics to ensure fairness in OOPPM classifiers. In addition to traditional metrics like ΔDP, which quantifies the average difference in classifier outputs between two protected groups, this study incorporates metrics based on probability density distributions. These novel metrics, inspired by recent advancements in machine learning [13], offer a more robust threshold-independent evaluation of demographic parity. Furthermore, the study introduces the use of a composite loss function, including an IPM loss, specifically the Wasserstein distance, in combination with BCE loss. Both the applicability of the metrics and the effectivity of the adapted loss functions are shown in controlled experiments using LSTM neural networks. However, the methodology is designed to be compatible with a wide range of classifiers. By adjusting the weight of the IPM loss relative to the BCE loss, a Pareto front is formed, exploring the trade-off between improved predictive capabilities and reduced independence violations. This framework allows for informed decision-making, enabling stakeholders to select optimal trade-offs that align with varying application-specific fairness requirements.

Future research could expand this work in several directions. A first option involves incorporating other forms of group fairness, such as separation (error rate parity) and sufficiency (calibration fairness), or exploring individual fairness techniques, such as counterfactual methods. The proposed metrics and loss functions could also be adapted to handle continuous sensitive attributes or a combination of multiple (intersecting) sensitive attributes. Another possibility is to explore multiclass tasks, such as next-event or suffix prediction. Alternative strategies, including data augmentation, debiasing, or the use of generative models [19], could also be evaluated using the fairness metrics introduced here. A more holistic approach might address related challenges such as class imbalance and label uncertainty for both outcomes and sensitive parameters [16,25]. It is important to note that our experimental setup uses simplified, artificial datasets. While this choice aids in illustrating the core concepts of fairness measurement and loss function integration, real-world scenarios often involve complexities such as imbalanced data and multiple intersecting sensitive attributes. Addressing these challenges will be a key focus for future research.

Acknowledgments. This work was supported in part by the Research Foundation Flanders (FWO) under Project 1294325N, and by the Flemish Government, through Flanders Innovation & Entrepreneurship (VLAIO, project HBC. 2021.0833).

References

1. Barocas, S., Hardt, M., Narayanan, A.: Fairness and machine learning: Limitations and opportunities. MIT Press (2023)
2. Boyd, S., Vandenberghe, L.: Convex optimization. Cambridge university press (2004)

3. Camargo, M., Dumas, M., González-Rojas, O.: Learning accurate LSTM models of business processes. In: Business Process Management, pp. 286–302, Springer International Publishing, Cham (2019), ISBN 978-3-030-26619-6
4. Chen, J., Kallus, N., Mao, X., Svacha, G., Udell, M.: Fairness under unawareness: assessing disparity when protected class is unobserved. In: Proceedings of the Conference on Fairness, Accountability, and Transparency, pp. 339–348, FAT* '19, Association for Computing Machinery, New York, NY, USA (2019), ISBN 9781450361255, https://doi.org/10.1145/3287560.3287594
5. Corbett-Davies, S., Pierson, E., Feller, A., Goel, S., Huq, A.: Algorithmic decision making and the cost of fairness. In: Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 797–806 (2017)
6. Cuturi, M.: Sinkhorn distances: Lightspeed computation of optimal transportation distances (2013). https://arxiv.org/abs/1306.0895
7. de Leoni, M., van der Aalst, W.M., Dees, M.: A general process mining framework for correlating, predicting and clustering dynamic behavior based on event logs. Inform. Syst. **56**, 235–257 (2016), ISSN 0306-4379, https://doi.org/10.1016/j.is.2015.07.003, https://www.sciencedirect.com/science/article/pii/S0306437915001313
8. Di Francescomarino, C., Dumas, M., Maggi, F.M., Teinemaa, I.: Clustering-based predictive process monitoring. IEEE Trans. Serv. Comput. **12**(6), 896–909 (2019). https://doi.org/10.1109/TSC.2016.2645153
9. Dwork, C., Hardt, M., Pitassi, T., Reingold, O., Zemel, R.: Fairness through awareness. In: Proceedings of the 3rd Innovations in Theoretical Computer Science Conference, p. 214-226, ITCS '12, Association for Computing Machinery, New York, NY, USA (2012), ISBN 9781450311151, https://doi.org/10.1145/2090236.2090255
10. European Commission: Proposal for a Regulation of the European Parliament and of the Council Laying Down Harmonised Rules on Artificial Intelligence (Artificial Intelligence Act) and Amending Certain Union Legislative Acts (2021), cOM(2021) 206 final
11. Gretton, A., Borgwardt, K.M., Rasch, M.J., Schölkopf, B., Smola, A.: A kernel two-sample test. J. Mach. Learn. Res.**13**(25), 723–773 (2012). http://jmlr.org/papers/v13/gretton12a.html
12. Guo, C., Pleiss, G., Sun, Y., Weinberger, K.Q.: On calibration of modern neural networks. In: Precup, D., Teh, Y.W. (eds.) Proceedings of the 34th International Conference on Machine Learning, Proceedings of Machine Learning Research, vol. 70, pp. 1321–1330, PMLR (06–11 Aug 2017). https://proceedings.mlr.press/v70/guo17a.html
13. Han, X., Jiang, Z., Jin, H., Liu, Z., Zou, N., Wang, Q., Hu, X.: Retiring δ DP: New distribution-level metrics for demographic parity. Transactions on Machine Learning Research (2023), ISSN 2835-8856
14. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural Comput. **9**(8), 1735-1780 (Nov 1997), ISSN 0899-7667
15. Kim, D., Kim, K., Kong, I., Ohn, I., Kim, Y.: Learning fair representation with a parametric integral probability metric. In: Proceedings of the 39th International Conference on Machine Learning, Proceedings of Machine Learning Research, vol. 162, pp. 11074–11101, PMLR (2022)
16. Kim, J., Comuzzi, M.: A diagnostic framework for imbalanced classification in business process predictive monitoring. Expert Syst. Appl. **184**, 115536 (2021), ISSN 0957-4174, https://doi.org/10.1016/j.eswa.2021.115536, https://www.sciencedirect.com/science/article/pii/S095741742100943X

17. Kleinberg, J., Mullainathan, S., Raghavan, M.: Inherent trade-offs in the fair determination of risk scores. arXiv preprint arXiv:1609.05807 (2016)
18. Kratsch, W., Manderscheid, J., Röglinger, M., Seyfried, J.: Machine learning in business process monitoring: a comparison of deep learning and classical approaches used for outcome prediction. Business Inform. Syst. Eng. **63**(3), 261–276 (2021)
19. de Leoni, M., Padella, A.: Achieving fairness in predictive process analytics via adversarial learning (extended version) (2024). https://arxiv.org/abs/2410.02618
20. Loshchilov, I., Hutter, F.: Fixing weight decay regularization in adam. CoRR **abs/1711.05101** (2017). http://arxiv.org/abs/1711.05101
21. Makhlouf, K., Zhioua, S., Palamidessi, C.: On the applicability of machine learning fairness notions. ACM SIGKDD Explor. Newsl **23**(1), 14–23 (2021)
22. Muskan, M., Mannhardt, F., van Dongen, B.: Extending genetic process discovery to reveal unfairness in processes. In: Process Mining Workshops, Springer Nature Switzerland, Cham (2025)
23. Müller, A.: Integral probability metrics and their generating classes of functions. Adv. Appl. Probab. **29**(2), 429–443 (1997). https://doi.org/10.2307/1428011
24. Pearl, J.: Comment: Understanding Simpson's Paradox, pp. 399–412. Association for Computing Machinery, New York, NY, USA, 1 edn. (2022), ISBN 9781450395861, https://doi.org/10.1145/3501714.3501738
25. Peeperkorn, J., Ortega Vázquez, C., Stevens, A., De Smedt, J., vanden Broucke, S., De Weerdt, J.: Outcome-oriented predictive process monitoring on positive and unlabelled event logs. In: Process Mining Workshops, pp. 255–268, Springer Nature Switzerland, Cham (2023), ISBN 978-3-031-27815-0
26. Pery, A., Rafiei, M., Simon, M., van der Aalst, W.M.P.: Trustworthy artificial intelligence and process mining: Challenges and opportunities. In: Process Mining Workshops, pp. 395–407, Springer International Publishing, Cham (2022), ISBN 978-3-030-98581-3
27. Pohl, T., Berti, A.: (un)fair process mining event logs (Jun 2023). https://doi.org/10.5281/zenodo.8059489
28. Pohl, T., Berti, A., Qafari, M.S., van der Aalst, W.M.P.: A collection of simulated event logs for fairness assessment in process mining (2023). https://arxiv.org/abs/2306.11453
29. Pohl, T., Qafari, M.S., van der Aalst, W.M.P.: Discrimination-aware process mining: a discussion. In: Process Mining Workshops, pp. 101–113, Springer Nature Switzerland, Cham (2023), ISBN 978-3-031-27815-0
30. Qafari, M.S., van der Aalst, W.: Fairness-aware process mining. In: Panetto, H., Debruyne, C., Hepp, M., Lewis, D., Ardagna, C.A., Meersman, R. (eds.) OTM 2019. LNCS, vol. 11877, pp. 182–192. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-33246-4_11
31. Rogge-Solti, A., Weske, M.: Prediction of remaining service execution time using stochastic petri nets with arbitrary firing delays. Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics) **8274 LNCS**, 389–403 (2013)
32. Rosenblatt, M.: Remarks on some nonparametric estimates of a density function. Ann. Math. Stat. **27**, 832–837 (1956)
33. Shalit, U., Johansson, F.D., Sontag, D.: Estimating individual treatment effect: generalization bounds and algorithms. In: Proceedings of the 34th International Conference on Machine Learning, Proceedings of Machine Learning Research, vol. 70, pp. 3076–3085, PMLR (06–11 Aug 2017)

34. da Silva, M.C., Fantinato, M., Peres, S.M.: Towards fairness-aware predictive process monitoring: Evaluating bias mitigation techniques. In: Comuzzi, M., Grigori, D., Sellami, M., Zhou, Z. (eds.) Cooperative Information Systems, pp. 150–166, Springer Nature Switzerland, Cham (2025), ISBN 978-3-031-81375-7
35. Spoel, S.V.D., Keulen, M.V., Amrit, C.: LNBIP 162 - Process prediction in noisy data sets: a case study in a Dutch hospital. In: International Symposium on Data-Driven Process Discovery and Analysis, pp. 60–83 (2013)
36. Sriperumbudur, B.K., Fukumizu, K., Gretton, A., Schölkopf, B., Lanckriet, G.R.G.: On the empirical estimation of integral probability metrics. Electron. J. Stat. **6**(none), 1550 – 1599 (2012). https://doi.org/10.1214/12-EJS722
37. Sriperumbudur, B.K., Fukumizu, K., Gretton, A., Schölkopf, B., Lanckriet, G.R.G.: On integral probability metrics, ψ-divergences and binary classification (2009). https://arxiv.org/abs/0901.2698
38. Tax, N., Verenich, I., Rosa, M.L., Dumas, M.: Predictive business process monitoring with lstm neural networks. In: International Conference on Advanced Information Systems Engineering, pp. 477–492, Springer (2017)
39. Teinemaa, I., Dumas, M., Rosa, M.L., Maggi, F.M.: Outcome-oriented predictive process monitoring: Review and benchmark. TKDD **13**(2), 17:1–17:57 (2019)
40. van der Aalst, W.M.P., Schonenberg, M.H., Song, M.: Time prediction based on process mining. Inform. Syst. **36**(2), 450–475 (2011), ISSN 03064379
41. Vanderschueren, T., Verdonck, T., Baesens, B., Verbeke, W.: Predict-then-optimize or predict-and-optimize? an empirical evaluation of cost-sensitive learning strategies. Inf. Sci. **594**, 400–415 (2022)
42. Verenich, I., Dumas, M., Rosa, M.L., Maggi, F.M., Teinemaa, I.: Survey and cross-benchmark comparison of remaining time prediction methods in business process monitoring. ACM Trans. Intell. Syst. Technol. (TIST) **10**(4), 1–34 (2019)
43. Villani, C., et al.: Optimal transport: old and new, vol. 338. Springer (2009)
44. Wasserman, L.: All of Nonparametric Statistics. Springer Texts in Statistics, Springer, New York (2006),ISBN 9780387306230, https://books.google.be/books?id=MRFlzQfRg7UC
45. Weytjens, H., De Weerdt, J.: Creating unbiased public benchmark datasets with data leakage prevention for predictive process monitoring. In: Business Process Management Workshops, pp. 18–29, Springer International Publishing, Cham (2022), ISBN 978-3-030-94343-1
46. Wuyts, B., vanden Broucke, S., De Weerdt, J.: Sutran: an encoder-decoder transformer for full-context-aware suffix prediction of business processes. In: 2024 6th International Conference on Process Mining (ICPM), pp. 17–24 (2024). https://doi.org/10.1109/ICPM63005.2024.10680671
47. Zemel, R., Wu, Y., Swersky, K., Pitassi, T., Dwork, C.: Learning fair representations. In: Proceedings of the 30th International Conference on Machine Learning, Proceedings of Machine Learning Research, vol. 28, pp. 325–333, PMLR (2013)
48. Zeng, X., Dobriban, E., Cheng, G.: Fair bayes-optimal classifiers under predictive parity. In: Advances in Neural Information Processing Systems, vol. 35, pp. 27692–27705, Curran Associates, Inc. (2022)

On the Use of Steady-State Detection for Process Mining: Achieving More Accurate Insights

Alexander Kraus[1](✉), Keyvan Amiri Elyasi[1], and Han van der Aa[2]

[1] Data and Web Science Group, University of Mannheim, Mannheim, Germany
{alexander.kraus,keyvan}@uni-mannheim.de
[2] Faculty of Computer Science, University of Vienna, Vienna, Austria
han.van.der.aa@univie.ac.at

Abstract. Steady-state detection (SSD) is a critical task in the analysis of dynamic systems, as it enables the reliable evaluation of system behavior by differentiating between stable and unstable states. While SSD techniques have been developed and tested in domains such as signal processing and industrial systems, their application in the information systems domain, particularly in process mining, has been largely overlooked. Specifically, event logs that record the executed behavior of a business process often contain data from both steady and non-steady states, which can distort process mining results, such as performance analysis and remaining time prediction. This paper highlights the importance of SSD in the process mining domain and investigates the applicability of existing SSD solutions. To operationalize this, we propose a two-step framework for detecting steady states in business processes. The framework extracts relevant process characteristics from an event log and applies established SSD techniques to identify periods during which a business process operated in a steady state. We evaluate the framework through experiments that assess its accuracy within a controlled environment using simulated event logs and that demonstrate the benefits of SSD for a downstream process mining task: remaining time prediction. The findings emphasize the potential of SSD for obtaining more accurate process mining insights.

Keywords: Process mining · Business Process · Steady-state detection

1 Introduction

Business processes are often supported by information systems that record execution data in event logs, which are then used in process mining to extract data-driven insights [1]. However, these event logs often capture business processes executed in both steady and non-steady states. *Steady states* refer to periods when process behavior remains stable and consistent over time [10], while *non-steady states* are marked by fluctuations and irregularities due to the dynamic

J. Krogstie et al. (Eds.): CAiSE 2025, LNCS 15701, pp. 204–220, 2025.
https://doi.org/10.1007/978-3-031-94569-4_12

environments in which processes operate. These non-steady states can arise from factors such as increased case arrivals during peak seasons or reduced resource availability during holidays, causing the process to deviate from its usual operations and performance levels.

The distinction between steady and non-steady states of processes is crucial for various process mining tasks. As shown later in this work, failing to distinguish between such states can, for instance, distort performance insights obtained through lead-time analysis or hurt the accuracy of predictive process monitoring models. Recognizing the impact that state fluctuations have in dynamic environments, the task of *steady-state detection* (SSD) aims to identify periods when a system operates in a steady state (or when it does not). Various techniques for this task have already been developed and tested in different application contexts, such as industrial systems [14] and signal processing [6]. However, their application in process mining has so far been largely overlooked, despite the potential of SSD to improve the accuracy of process mining insights.

Therefore, this paper highlights the importance of SSD in process mining and investigates the applicability of existing SSD solutions within this domain. To operationalize this, we propose a framework designed to identify steady states in business processes based on event data. The framework consists of two steps: (1) extracting time series from an event log that capture the progression of relevant process characteristics and (2) applying an established SSD technique to detect steady and non-steady states per process characteristic and aggregating these results to detect steady states at the process level. The effectiveness of our framework is evaluated in two experiments: one assessing its accuracy in a controlled environment based on simulated event logs and the other demonstrating its practical benefits in a downstream process mining task, specifically for remaining time prediction. Our findings showcase that our framework indeed enables the use of SSD for process mining and highlight the potential of SSD to provide more accurate insights into the operations of organizations.

The remainder of this paper is organized as follows. Section 2 provides background and illustrates the importance of SSD in process mining. Section 3 introduces our proposed framework for SSD for business processes. In Sect. 4, we present the results of our evaluation experiments, demonstrating the framework's accuracy and usefulness. Finally, Sect. 5 discusses the relationship between SSD and other related problems in process mining, while Sect. 6 summarizes our findings and suggests potential directions for future work.

2 Background and Problem Illustration

In this section, we provide background information on steady states and demonstrate the importance of their consideration in process mining.

Steady States and the SSD Problem. A steady state refers to a condition in which the behavior of a system remains constant over time [10], making its behavior predictable and allowing for more precise and meaningful analysis. The

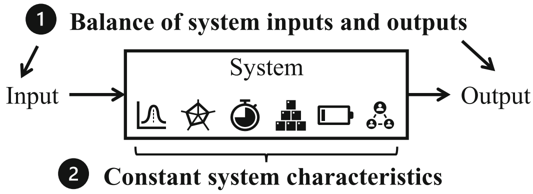


Fig. 1. Two key properties of a steady state.

study of steady states has a long history and has proven to be important in various fields, including mechanics [4], biology [11], ecology [22], and economics [5].

Steady State of a Business Process. In process mining, we define a business process to be in a steady state if its system-level behavior remains stable and consistent over time. As shown in Fig. 1, a business process can be represented as a system with its inputs, internal characteristics, and outputs. The steady state of a business process can then be characterized by the following two properties:

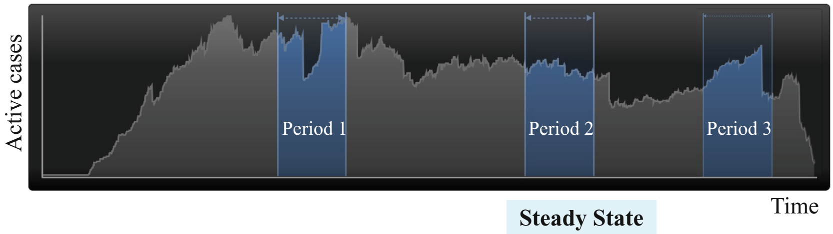
1. *Balance of system inputs and outputs*: A system in a steady state maintains a balance between input and output, ensuring that no significant fluctuation occurs over time. In the context of a business process, this means, e.g., that the number of incoming and completed cases remains consistent over time.
2. *Constant system characteristics*: The characteristics of a system in a steady state remain consistent. For a business process, this could mean that, e.g., the number of active cases and available resources remain stable.

It is important to note that when examining the system-level behavior of a process, we focus on process characteristics that provide a holistic description of its behavior that evolves over time, exhibiting notable fluctuation.

The SSD Problem. In the context of process mining, we define the SSD problem as the task of detecting periods when system-level process behavior, derived based on information recorded in an event log, remains in a steady state.

Importance of SSD in Process Mining. To illustrate the importance of SSD in process mining, we examine how the performance of a business process, measured by average and median lead times, can differ between steady and non-steady periods, and the implications this may have on a downstream process mining task. For this purpose, we use a real-life event log describing a permit application process at a municipality (BPIC2015-2) [7] as a running example. The event log contains 44,354 events, capturing the execution of 832 cases over a period of approximately 5 years. During this period, the process exhibits an average lead time of 22.9 weeks, with a median lead time of 15.5 weeks. For simplicity, we focus on a single system-level process characteristic, namely the number of active cases, when examining the steady state of a business process.

The number of active cases, shown in Fig. 2, indicates that the process was not steady throughout the recorded timeframe, with both stable and unstable periods. For instance, in Period 1, spanning 5 months and involving 23 cases,



			Steady State	
Measure	**Event log**	**Period 1**	**Period 2**	**Period 3**
Number of traces	832	23	23	25
Duration	56 months	5 months	5 months	5 months
Average lead time	22.9 weeks	10.7 weeks	5.6 weeks	12.1 weeks
Median lead time	15.5 weeks	11.3 weeks	3.6 weeks	13.9 weeks

Fig. 2. Comparison of process performance between different periods.

the process shows instability, marked by a significant drop in active cases. The average lead time is 10.7 weeks, with a median of 11.3 weeks. Period 2, also 5 months long with 23 cases, is more stable, with fewer fluctuations in active cases. The average lead time is 5.6 weeks, and the median is 3.6 weeks. Lastly, Period 3 exhibits a rise and fall in active cases, indicating a non-steady state. It has an average lead time of 12.1 weeks and a median lead time of 13.9 weeks.

As observed, performance in the steady state (Period 2) is nearly twice as good as in the other periods and about four times better than the overall average across all recorded cases. Such differences are particularly relevant for process mining tasks such as remaining time prediction, as demonstrated in our evaluation. Specifically, when significant performance differences exist between steady and non-steady states, it may be beneficial to use SSD as a bucketing method to split the event log into sublogs representing steady and non-steady states. Separate models can then be trained for each sublog, allowing the appropriate model to be applied based on whether the process is currently in a steady or non-steady state, improving the accuracy of predictions for ongoing cases.

3 Steady-State Detection Framework for Process Mining

This section describes our proposed SSD framework. As illustrated in Fig. 3, the framework takes an event log as input and then extracts time series that represent the progression of relevant process characteristics over time. These time series are then analyzed using existing SSD techniques to identify periods when a process is in a steady state. As output, the framework provides the detected steady-state periods along with a sublog of traces corresponding to them. In the following, we describe these two main steps.

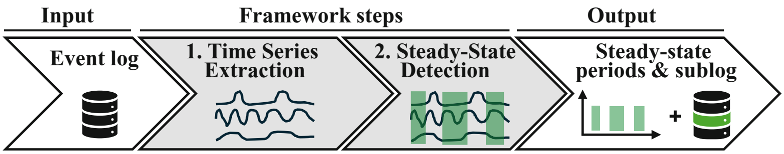


Fig. 3. Overview of the main steps of our framework.

3.1 Time Series Extraction

In Step 1, we generate time series from an event log to capture the evolution of process characteristics relevant to SSD. Such transformations are widely used in process mining, for purposes including business process simulation [19], assessing process resilience [13], and evaluating process complexity [26]. Below, we outline the specifics of this step.

Input. Our approach takes an *event log* L, which we define as a collection of events recorded by a process-aware information system. Each *event* $e \in L$ is represented as a tuple with at least three attributes $e =$ (caseID, activity, timestamp), where *caseID* is the unique identifier for the executed case, *activity* indicates the executed process activity, and *timestamp* denotes the event moment. A *trace* σ is a sequence of events from L with the same caseID, ordered by their timestamps. We denote Σ_L as the ordered collection of all traces from L, arranged according to the timestamp of their first event.

Windowing. We divide the entire timeframe of an event log L into $n \in \mathbb{N}$ equally spaced *time windows* $W_l = \langle w_1, \ldots, w_n \rangle$, each with a fixed length l (e.g., a day or a week). Consequently, each event $e \in L$ is assigned to exactly one time window w_t, where $t \in \{1, \ldots, n\}$.

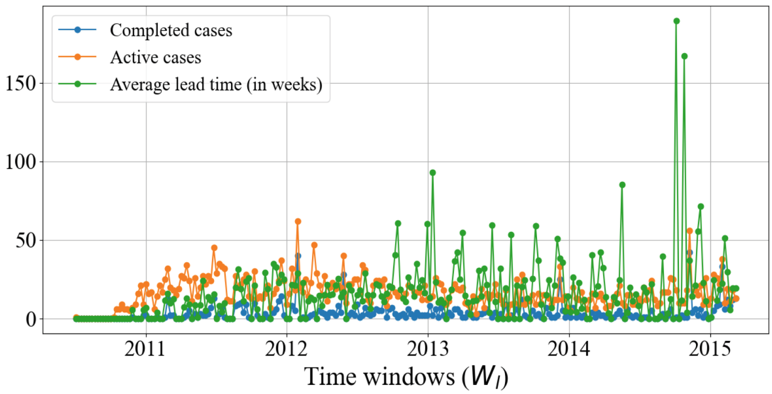


Fig. 4. Outcome of the first framework step.

Time Series Construction. Next, we construct time series over $w_t \in W_l$ for different process characteristics. In our framework, we consider 3 process characteristics that are relevant for SSD and can be derived from a standard event log L: *the number of active cases* (ac), *the number of completed cases* (cc), and *the average lead time* (alt) of completed cases during a time window w_t. If the event log includes further information, such as resource details, additional process characteristics can be considered to enrich the process representation.

We use $y^f_{w_t} \in \mathbb{R}$ to denote the value of a characteristic (or *feature)* $f \in F = \{ac, cc, alt\}$ during a time window w_t. For each feature, we concatenate these values into a *time series* $\{y^f_{w_t}\}^n_{t=1}$, which captures the evolution of f over the time windows in W_l. Figure 4 shows the outcome of this step with weekly windowing for the BPIC2015-2 event log, serving as a running example.

3.2 Steady-State Detection

After extracting time series, the next step is to identify time windows when a process is in a steady state. This involves performing SSD at the time series level (i.e., per characteristic) and then at the process level.

SSD at Time Series Level. For each time series $\{y^f_{w_t}\}^n_{t=1}$, we derive a corresponding *binary time series* $\{p^f_{w_t}\}^n_{t=1}$, with $p^f_{w_t} \in \{0, 1\}$ for each time window w_t using an existing SSD technique. This binary time series indicates whether the corresponding process characteristic is in a steady state during w_t, where $p^f_{w_t} = 1$ signifies a steady state and $p^f_{w_t} = 0$ indicates a non-steady state.

To obtain $\{p^f_{w_t}\}^n_{t=1}$, we can use an SSD technique from a range of existing ones. Our framework's implementation currently supports the following options:

- *Rolling Window (RW)* [28]: The RW technique detects steady states in a time series by comparing the short-term and long-term rolling averages of its values. It identifies a drift when the deviation between the short-term and long-term averages exceeds a threshold that is scaled by the standard deviation of the long-term average.
- *Cumulative Sum (CS)* [8]: The CS algorithm monitors cumulative increases and decreases in the data and flags a change when these values exceed a predefined threshold. Once a change is identified, the cumulative calculation resets to ensure continued monitoring.
- *Variance Filter (VR)* [23]: The VR method proposed by Rhinehart uses a variance filter to distinguish between steady and non-steady states based on statistical analysis. It applies a filter that evaluates the ratio of the variance of the signal, with thresholds used to identify steady states.
- *ED Pelt with Transitions (EDP)* [9]: The EDP technique identifies steady states in a time series by splitting the time series into "statistically homogeneous" segments using the pruned exact linear time (Pelt) change point detection algorithm. The Pelt method guarantees optimal segmentation while maintaining a linear computational complexity.

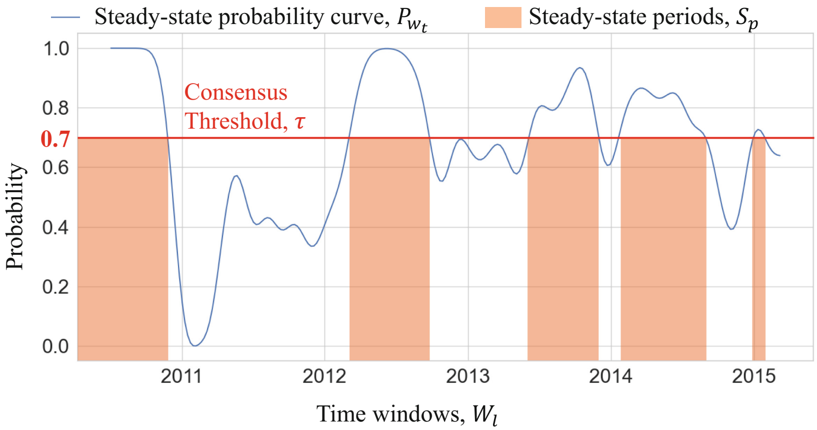


Fig. 5. SSD using the probability curve and consensus threshold.

Beyond these techniques, our framework is compatible with any SSD method that accepts a real-valued time series and generates a binary time series.

SSD at Process Level. After performing SSD per process characteristic, we next aggregate the information from the binary time series to determine if indeed the entire process can be considered to be in a steady state during a given time window. Our framework supports several aggregation techniques for this:

Kernel-based aggregation computes a *steady-state probability curve* as a time series $\{P_{w_t}\}_{t=1}^{n}$ with $P_{w_t} \in [0,1], \forall w_t$ that represents the likelihood of a time window w_t to record a steady state of a process. To do this, we first aggregate insights from different process characteristics by calculating the average value across all binary time series $\{p_{w_t}^{f}\}_{t=1}^{n}$ for each time window. We then apply a Gaussian filter [17] with a kernel of 4 standard deviations to smooth the curve and reduce fluctuations. After smoothing, the time series is rescaled using Min-Max normalization to ensure that the values lie between 0 and 1. Finally, to identify the steady states of a business process, we compare the values of the steady-state probability curve with a *consensus threshold* $\tau \in [0,1]$. If $P_{w_t} \geq \tau$, the time window w_t is considered to be part of a steady state; otherwise, as non-steady. Figure 5 illustrates the outcome of this transformation for the time series shown in Fig. 4, assuming a consensus threshold $\tau = 0.7$.

In addition to the kernel-based aggregation technique, our framework also supports more straightforward aggregation techniques. *Consensus-based aggregation* considers a time window w_t as a steady state if $p_{w_t}^{f} = 1$ for all process characteristics. *Majority-based aggregation* deems a time window w_t as a steady state if $p_{w_t}^{f} = 1$ holds for at least 50% of the process characteristics. Finally, *single-source aggregation* classifies a time window w_t as a steady state if $p_{w_t}^{f} = 1$ holds for at least one process characteristic f.

As output, we obtain the time windows $W_l' = \langle w_{i_1}, \ldots, w_{i_m} \rangle$ as a subsequence of W_l with $1 \leq i_1 < i_m \leq n$, where the process is in a steady state.

Detection of Steady-State Periods and a Sublog. Finally, after identifying time windows that characterize a process in a steady state, we define steady-state periods $S_p = \langle s_1, \ldots, s_d \rangle$ by merging consecutive time windows from W_l' into continuous intervals. In Fig. 5, we detect a total of 5 steady-state periods.

In addition, to enable downstream process mining task, we identify traces that correspond to the detected steady-state periods. To do this, we analyze each trace $\sigma \in \Sigma_L$ and check whether the timestamps of its events fall within the identified periods in S_p. If the proportion of such events relative to the total number of events in σ exceeds a predefined *trace acceptance threshold* $\theta \in [0, 1]$, the trace is classified as a trace that belongs to a steady-state sublog. For example, if 3 out of 5 events in a trace fall within one or more periods from S_p and $\theta = 0.5$, the trace is classified as belonging to a steady state since $3/5 > \theta$. Otherwise, the trace does not belong to a steady state. This results in a sublog $\Sigma_L^S \subseteq \Sigma_L$, containing traces associated with the process in a steady state.

Depending on the downstream process mining task, the assignments of traces to steady-state periods can also be done at the event or sub-trace level if more fine-granular information is desired.

4 Evaluation

This section presents two conducted evaluation experiments. In the first experiment, detailed in Sect. 4.1, we evaluate the accuracy of our framework in detecting steady states using synthetic data. The second experiment, explained in Sect. 4.2, demonstrates the usefulness of the framework using real-life event logs and a concrete process mining task, i.e., the prediction of the remaining time for ongoing cases. To ensure reproducibility, we have provided the data, implementation, configurations, and raw results in a publicly accessible repository[1].

4.1 Experiment 1: Accuracy

In the first experiment, we assess the ability of our framework to identify steady states in event logs. In the following, we discuss the data collection, setup, evaluation measure, and obtained results.

Data Collection. In this experiment, we generate data by simulating an order-to-cash process for a medium-sized company, as described in the work by Zahoransky et al. [27]. The simulation model is built using the CIW library [18], an open-source tool for discrete event simulation.[2] To introduce steady and non-steady states, we vary the number of incoming cases during the simulation, ensuring a balanced distribution between steady and non-steady periods. Specifically, we create non-steady states by applying periods of linear increases and decreases in the arrival rate, followed by periods of constant arrival rate to establish steady states. We implement up to 5 changes in the arrival rates,

[1] Project repository: https://gitlab.uni-mannheim.de/processanalytics/ssd.
[2] Available online: https://ciw.readthedocs.io/en/latest/index.html.

starting with either increases or decreases, resulting in 10 distinct scenarios, as shown in Fig. 6. To ensure robust evaluation, we generate 10 event logs for each scenario, producing a total of 100 event logs.

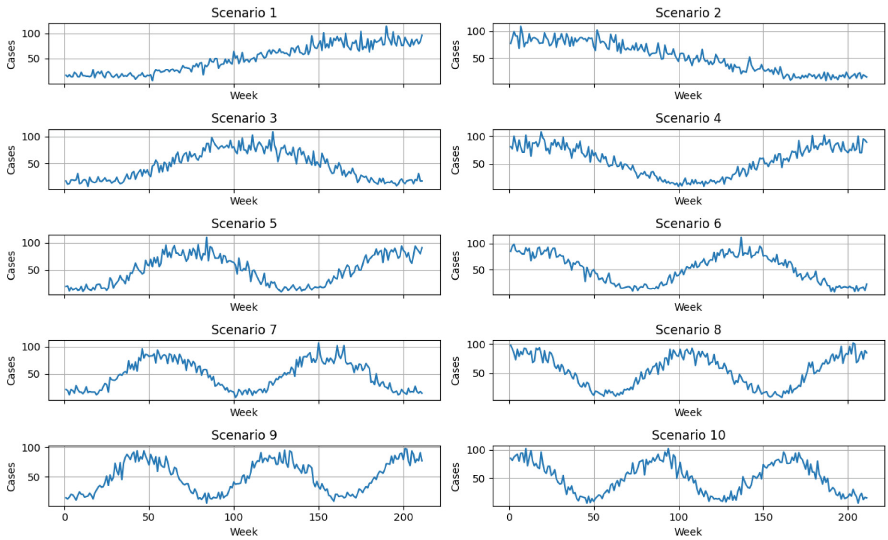


Fig. 6. Simulated number of arrived cases for each scenario.

Setup. In Step 1 of our framework, we apply weekly windowing and consider 3 process characteristics: the number of active cases, the number of completed cases, and the average lead time of completed cases, i.e., $f \in \{\text{ac}, \text{cc}, \text{alt}\}$.

In Step 2, we evaluate all four implemented techniques for SSD: Rolling Window (RW), Cumulative Sum (CS), Variance Filter (VR), and ED Pelt with Transitions (EDP). For each technique, we test a variety of parameter combinations,[3] resulting in a total of 564 evaluations per event log. To detect steady states of the process, we evaluate 4 aggregation techniques (i.e., aggregation-based SSD) with a trace acceptance threshold of $\theta = 0.8$: kernel-based, consensus-based, majority-based, and single-source. For the kernel-based approach, we set the consensus threshold to $\tau = 0.7$. Additionally, we compare the results of our framework when the decision about steady states is made based solely on a single process characteristic (i.e., feature-based SSD).

Evaluation Measure. To measure our framework,s accuracy in classifying each time window as a steady or non-steady state, we use the ϕ coefficient [15], a widely used binary classification metric for assessing the strength of observed

[3] The exact parameters tested for each technique are specified in our repository.

associations. This metric offers a balanced evaluation by considering all components of the confusion matrix. It is defined as follows::

$$\phi = \frac{(TP \times TN) - (FP \times FN)}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}},$$

where TP, FP, TN, and FN represent true positives (correctly predicted steady-state windows), false positives (incorrectly predicted steady-state windows), true negatives (correctly predicted non-steady state windows), and false negatives (incorrectly predicted non-steady state windows), respectively. The ϕ coefficient ranges from -1 to $+1$, where $+1$ indicates perfect classification, 0 indicates random guessing, and -1 indicates complete disagreement.

Table 1. Results of Experiment 1: The average ϕ coefficient along with its standard deviation.

SSD configuration	SSD technique			
	RW	CS	VF	EDP
Feature-based				
Active cases	0.35 ± 0.03	0.27 ± 0.08	0.19 ± 0.06	**0.43 ± 0.12**
Avg. lead time	0.21 ± 0.06	**0.38 ± 0.12**	0.37 ± 0.10	0.04 ± 0.07
Case completions	0.36 ± 0.03	0.26 ± 0.05	0.25 ± 0.06	**0.43 ± 0.08**
Aggregation-based				
Kernel-based	**0.47 ± 0.04**	0.33 ± 0.06	0.32 ± 0.10	0.35 ± 0.09
Consensus-based	0.32 ± 0.03	0.25 ± 0.05	0.32 ± 0.13	**0.35 ± 0.09**
Majority-based	0.34 ± 0.03	0.30 ± 0.06	0.18 ± 0.05	**0.42 ± 0.08**
Single-source	0.25 ± 0.04	**0.40 ± 0.10**	0.20 ± 0.07	0.07 ± ± 0.09

Note: The highlighted values show the best results in each row.

Results. Table 1 presents the results obtained on our data collection, showing the average and standard deviations of the ϕ coefficient for both feature-based and aggregation-based configurations. The table shows that the SSD technique using rolling windows (RW) achieves the highest ϕ coefficient of 0.47 with kernel-based aggregation, indicating a moderate positive association between the predicted steady and non-steady states. The EDP technique produces similar outcomes, with a ϕ coefficient around 0.43 when using either the number of active cases or the number of case completions. In contrast, the VF technique demonstrates the lowest performance, consistently underperforming relative to other techniques across all configurations.

When comparing these results with the results observed in other domains [25], we see that the accuracy is slightly lower. The main reason for this is the specific and more complex interrelations between different process characteristics in a business process, compared to other domains where relationships typically follow well-defined laws or equations. For example, in a business process, an increase in the arrival rate does not automatically lead to an increase in the average lead time, since the process may have additional capacities that allow it to handle the increased workload without significant changes in system behavior.

Overall, existing SSD techniques can be applied in process mining, but the evaluation shows room for improvement due to the unique properties and interrelations of process characteristics, requiring domain-specific adjustments to SSD for better accuracy and applicability.

4.2 Experiment 2: Usefulness

In this experiment, we demonstrate the usefulness of our SSD framework by considering a well-known task in process mining, namely the remaining time prediction problem. Specifically, we compare the prediction accuracy of various state-of-the-art approaches applied to entire event logs with their accuracy when using only data from steady states. In the following, we discuss the data collection, setup, and obtained results.

Table 2. Characteristics of the employed event logs.

Event log	Number of				Case length		Duration (days)	
	Cases	Variants	Events	Classes	Avg	Max	Avg	Max
Steady and non-steady states (Σ_L)								
Hospital	100000	1020	451359	18	4.5	217	127.2	1035
Sepsis	1050	846	15214	16	14.5	185	28.5	422
Helpdesk	4580	226	21348	14	4.7	15	40.9	60
BPIC12	13087	4366	262200	36	20.0	175	8.6	137
BPIC15-1	1199	1170	52217	398	43.6	101	95.9	1486
BPIC15-2	832	828	44354	410	53.3	131	160.3	1326
BPIC15-3	1409	1349	59681	383	42.4	123	62.2	1512
BPIC15-4	1053	1049	47293	356	44.9	115	116.9	927
BPIC15-5	1156	1153	59083	389	51.1	153	98.0	1344
Steady states (Σ_L^S)								
Hospital	8315	176	27117	15	3.3	217	54.3	773
Sepsis	439	378	6242	16	14.2	170	35.8	422
Helpdesk	745	92	3742	10	5.0	14	40.4	60
BPIC12	5692	1417	81125	36	14.2	142	5.0	67
BPIC15-1	682	667	29956	377	43.9	93	99.8	1486
BPIC15-2	311	310	17823	341	57.3	132	152.9	1171
BPIC15-3	521	505	22363	303	42.9	101	58.3	1261
BPIC15-4	677	674	30813	321	45.5	116	104.9	831
BPIC15-5	520	519	27462	329	52.8	108	86.9	812

Data Collection. Our data collection consists of 9 publicly available real-life event logs that are commonly used for predicting the remaining runtime of ongoing cases.[4] As summarized in Table 2, these logs represent the execution of various processes and display diverse characteristics across multiple dimensions, including the number of cases, variants (i.e., unique traces), recorded events, event classes (i.e., unique activities), average case lengths and durations. In addition to the characteristics of the original event logs that include all traces (Σ_L), we include the characteristics of the sublogs with traces that correspond to steady states (Σ_L^S), as identified using our framework.

Setup. Next, we discuss the framework configurations, employed data split, and used remaining time prediction approaches.

Configurations. In Step 1 of our framework, we apply weekly windowing for all event logs, except for the BPIC12 event log, which covers a relatively short time period. For this log, we use daily windowing instead. We again consider 3 process characteristics, i.e., $f \in \{\text{ac}, \text{cc}, \text{alt}\}$. In Step 2, we use the configuration that yielded the best results in Experiment 1, specifically the rolling window (RW) and kernel-based aggregation with a consensus threshold of $\tau = 0.7$ and a trace acceptance threshold of $\theta = 0.8$.

Data Split and Prefix Generation. We use a 64%–16%–20% chronological holdout split that divides data into training, validation, and testing sets while preserving the natural chronological order. This method mitigates data leakage and simulates real-world scenarios where predictions are made based on historical data [24]. For each trace σ in a split, we extract all prefixes between lengths 2 and $|\sigma| - 1$ to establish prediction problems.

Approaches. We consider 3 remaining time prediction approaches that estimate the remaining time of an ongoing case based on the sequence of already executed activities (and possibly other available attributes):

- DUMMY: A simple baseline that predicts the remaining time of an ongoing case by averaging the remaining time of training cases that share the same sequence of executed activities.
- DALSTM: This deep learning model, based on the LSTM architecture, outperforms other LSTM-based approaches in remaining time prediction [21].
- PGTNet: This approach employs graph transformers to balance learning from the local contexts with capturing long-range dependencies [2], demonstrating state-of-the-art results.

For DALSTM and PGTNet, we use the settings reported in the original papers.

Evaluation Measures. To evaluate the impact of SSD on prediction accuracy, we consider three evaluation measures:

[4] We excluded event logs from the BPI Challenge 2013 and 2020 due to long periods of process inactivity, the Traffic Fine log for its strong batching behavior, and the Environment Permit log for having too few events, making further segmentation unsuitable for training a prediction model.

- *Mean Absolute Error* (MAE) quantifies the average magnitude of absolute errors between predicted and actual remaining time. It is formally defined as: $\text{MAE} = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|$, where n is the number of predictions, y_i represents the actual observed values, and $\hat{y}_i$ denotes the predicted values. Lower MAE values indicate higher predictive accuracy.
- *Average Performance Change* (APC) measures the average change in MAI across all approaches when comparing values obtained for all traces in an event log, denoted as Σ_L, with those obtained for traces belonging to steady-state periods, denoted as Σ_L^S. It is defined as: $APC = \frac{1}{3}\sum_{i=1}^{3}\frac{MAE_i^{\Sigma_L^S}}{MAE_i^{\Sigma_L}} - 1$, where the index i iterates over the three remaining time prediction approaches considered in this experiment. An APC value closer to zero indicates smaller differences in prediction accuracy between Σ_L and Σ_L^S.
- *Steady-State Ratio* (SSR) represents the proportion of steady-state traces relative to the total number of traces in each event log. It is calculated as: $SSR = \frac{|\Sigma_L^S|}{|\Sigma_L|}$. This measure provides context for the APC by showing the prevalence of steady-state traces within the event log.

Results. Table 3 presents the MAE values for all traces in an event log (Σ_L) and those associated with steady-state periods (Σ_L^S), along with the corresponding APC and SSR values for each event log. To enhance interpretability, the rows are sorted in ascending order based on the APC values.

First, we observe that for most event logs, the APC is negative, ranging from −27.1% to −5.7%. This indicates that the MAE for traces executed in a steady state (columns Σ_L^S) is, on average, lower than when predictions are made using the entire dataset (columns Σ_L). This is expected, as many processes in steady states have shorter lead times with less fluctuation, allowing for more accurate predictions of the remaining time for ongoing cases compared to non-steady states[5]. In some event logs, such as BPIC15 Municipalities 2, 3, and 5, this trend holds consistently across different approaches. For the remaining time prediction task, this finding highlights the importance of training separate models: one tailored for steady states and another optimized for non-steady states. This strategy is likely to provide more accurate predictions in terms of MAE compared to using a single model that is trained on the entire event log.

However, in some event logs, the APC is positive, meaning the MAE has increased. This can be attributed to the specific characteristics of the recorded processes. In the Sepsis event log, the APC is 40%, due to the process's long warm-up and cool-down phases, which together account for over 50% of the total recorded time. As a result, steady states are detected too early, misclassifying the warm-up period and causing inaccurate detection. This highlights a challenge in business process mining, where SSD techniques from other fields may struggle to accurately detect and differentiate steady states from warm-up and cool-down

[5] SSD remains useful when traces in a non-steady state have shorter lead times. For example, an emergency call center may receive a surge of calls during a disaster, prompting faster processing to assist more people, reducing average lead time.

Table 3. Mean Absolute Error for remaining time prediction

Event log	DUMMY		DALSTM		PGTNet		APC	SSR
	Σ_L	Σ_L^S	Σ_L	Σ_L^S	Σ_L	Σ_L^S	in %	in %
BPIC15-4	77.2	86.5	72.7	45.4	82.7	36.6	-27.1	64.3
BPIC15-3	24.2	22.5	15.1	10.6	15.0	10.3	-22.6	37.0
BPIC15-5	48.3	45.4	43.6	32.9	36.3	32.6	-13.6	44.0
BPIC15-2	72.9	61.4	47.0	43.8	68.8	59.2	-12.1	37.4
BPIC12	7.6	7.5	8.0	4.8	5.5	5.9	-11.3	43.5
Helpdesk	12.3	10.6	12.9	10.9	5.4	6.1	-5.7	16.3
BPIC15-1	38.2	40.9	29.3	37.9	20.4	27.3	23.6	56.9
Sepsis	32.7	43.4	15.7	22.2	16.4	24.9	41.8	41.8
Hospital	47.9	66.8	36.7	35.9	24.2	59.0	60.4	8.3

phases. In the Hospital event log, the APC is also positive. However, the SSR is only 8%, indicating that the proportion of traces belonging to a steady state is very low. Consequently, the steady-state sublog may be too small to yield reliable results. Finally, for the BPIC15-1 event log, the APC is approximately 24%. A closer analysis of the detected steady states reveals that the event log may contain multiple steady states with different properties. The detected steady state at the beginning of the event log occurs when the number of active cases is high, while the second part features a lower number of active cases, leading to a qualitatively different steady state. In this case, a more appropriate approach would be to consider these two steady states independently.

Overall, this experiment demonstrates that our SSD framework can notably impact the insights for a downstream process mining task, making it a valuable preprocessing step, such as bucketing in the case of remaining time prediction. While the applicability of our framework may be influenced by certain specific characteristics of the recorded process behavior, it remains a highly effective approach for many business processes.

5 Related Work

In this section, we relate the SSD problem to other problems in process mining.

Concept Drift Detection. The problem of SSD is related to concept drift detection in process mining, but they address different aspects. Concept drift detection identifies changes in the process that lead to a new process version [3], which operates for a certain period. In contrast, SSD focuses on the system-level behavior of the process, identifying periods where key process characteristics remain stable over time. These aspects are not necessarily correlated. For example, if a new activity (drift in the control flow) creates a bottleneck due to limited resource capacity, it is likely to impact system-level characteristics such as the average lead time. This would disrupt the steady state, potentially leading to a non-steady state or another steady state. However, if the new activity does not create a bottleneck, the system may remain in the same steady state despite transitioning to a new process version. Conversely, a business process can transition from a steady state to a non-steady state without changing its process version, for example, due to fluctuations in the arrival rate.

Business Process Simulation. In business process simulation, SSD can be used to address the initialization bias (or startup issue) of simulation models [20]. Many simulations begin from an empty state, causing early fluctuations that distort results and limit analysis. The primary goal of SSD in process simulation is to identify when a process reaches a steady state, which is essential for predicting reliable long-term insights. Despite the similar terminology, the SSD problem discussed in this paper is distinct, focusing on detecting the steady state of a business process based on past recorded behavior in an event log, and serving as a crucial preprocessing step for various offline process mining techniques. We believe that the SSD problem discussed in this paper could also impact future applications in business process simulation, particularly in the automated extraction of business process simulation models from event logs.

Anomaly Detection. Anomaly detection seeks to identify outliers or unusual patterns at the case level that deviate from expected process behavior [12]. In contrast, SSD focuses on identifying periods of stable, consistent process behavior across all active cases for a given period. However, SSD can provide a baseline for anomaly detection, making it easier to identify and explain unexpected behaviors. Once a steady state is reached, significant deviations can signal potential irregularities, while anomalies during non-steady states can often be explained by the process's inherent instability during that period.

Statistical Quality Control. The problem of SSD is closely related to statistical quality control (SQC) [16], with both aiming to monitor process stability over time. However, SSD focuses on identifying when a process has reached a steady state, where its characteristics remain relatively stable. In contrast, SQC emphasizes detecting deviations from a desired range, typically defined by specific process characteristics that reflect the process's quality or efficiency. Moreover, it is important to note that reaching a steady state does not necessarily mean the process is operating within the optimal performance range that SQC seeks to maintain. A process can be stable but still fall outside the desired limits.

6 Conclusion

This paper addresses the problem of steady-state detection (SSD) in business processes, emphasizing its importance in process mining and examining the applicability of existing SSD solutions within this domain. We propose a framework for identifying when a process is in a steady state in a data-driven manner using information recorded in event logs. The framework first generates time series to represent key process characteristics and applies established SSD techniques to identify steady states in the time series and process levels, producing a sublog that captures the process behavior during these periods. The evaluation demonstrates that the framework effectively detects steady states in many real-life business processes and can enhance the accuracy and reliability of insights derived from a downstream process mining task.

In future work, we plan to pursue two key directions: enhancing the proposed framework and further investigating how SSD affects process mining tasks. To strengthen the SSD framework, we aim to develop a technique specifically tailored to the unique characteristics of business processes. Our evaluation has demonstrated that existing generic SSD techniques from other domains are not fully effective, highlighting the need for a specialized approach. This tailored SSD technique would be applicable to any event log and account for atypical behaviors, such as extended warm-up periods or periods of inactivity that may occur within event logs. To develop a more comprehensive understanding of the impact of SSD on process mining, we plan to investigate its effects across a wider range of tasks, including process discovery, conformance checking, concept drift detection, and more. This investigation will further highlight the importance of SSD and demonstrate its value in process mining research and practice.

References

1. Aalst, W.: Data science in action. In: Process Mining, pp. 3–23. Springer, Heidelberg (2016). https://doi.org/10.1007/978-3-662-49851-4_1
2. Amiri Elyasi, K., van der Aa, H., Stuckenschmidt, H.: PGTNet: A process graph transformer network for remaining time prediction of business process instances. In: Guizzardi, G., Santoro, F., Mouratidis, H., Soffer, P. (eds.) International Conference on Advanced Information Systems Engineering, pp. 124–140. Springer, Cham (2024). https://doi.org/10.1007/978-3-031-61057-8_8
3. Bose, R.P.J.C., van der Aalst, W.M.P., Žliobaitė, I., Pechenizkiy, M.: Handling concept drift in process mining. In: Mouratidis, H., Rolland, C. (eds.) CAiSE 2011. LNCS, vol. 6741, pp. 391–405. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-21640-4_30
4. Capecchi, D., Vestroni, F.: Steady-state dynamic analysis of hysteretic systems. J. Eng. Mech. **111**(12), 1515–1531 (1985)
5. Daly, H.E.: The economics of the steady state. Am. Econ. Rev. **64**(2), 15–21 (1974)
6. Derzsi, Z.: Optimal approach for signal detection in steady-state visual evoked potentials in humans using single - Channel EEG and stereoscopic stimuli. Front. Neurosci. **15**, 600543 (2021)
7. van Dongen, B.B.: BPI challenge 2015. 4TU ResearchData collection (2015)
8. Gustafsson, F., Gustafsson, F.: Adaptive Filtering and Change Detection, vol. 1. Wiley, New York (2000)
9. Haynes, K., Fearnhead, P., Eckley, I.A.: A computationally efficient nonparametric approach for changepoint detection. Stat. Comput. **27**, 1293–1305 (2017)
10. Kast, F.E., Rosenzweig, J.E.: General systems theory: applications for organization and management. Acad. Manag. J. **15**(4), 447–465 (1972)
11. Kitano, H.: Systems biology: a brief overview. Science **295**(5560), 1662–1664 (2002)

12. Ko, J., Comuzzi, M.: A systematic review of anomaly detection for business process event logs. Bus. Inf. Syst. Eng. **65**(4), 441–462 (2023)
13. Kraus, A., Rehse, J.R., van der Aa, H.: Data-driven assessment of business process resilience. Process Sci. **1**(1), 4 (2024)
14. Lipo, T.A., Cornell, E.P.: State-variable steady-state analysis of a controlled current induction motor drive. IEEE Trans. Ind. Appl. **IA-11**(6), 704–712 (1975)
15. Matthews, B.W.: Comparison of the predicted and observed secondary structure of T4 phage lysozyme. Biochimica et Biophysica Acta (BBA)-Protein Structure **405**(2), 442–451 (1975)
16. Montgomery, D.C.: Introduction to Statistical Quality Control, 8th edn. Wiley, Hoboken, NJ (2019)
17. Oppenheim, A.V.: Discrete-Time Signal Processing. Pearson Education India (1999)
18. Palmer, G.I., Knight, V.A., Harper, P.R., Hawa, A.L.: CIW: an open-source discrete event simulation library. J. Simul. **13**(1), 68–82 (2019)
19. Pourbafrani, M., van der Aalst, W.: Discovering system dynamics simulation models using process mining. IEEE Access **10**, 78527–78547 (2022)
20. Pourbafrani, M., Lücking, N., Lucke, M., van der Aalst, W.: Steady state estimation for business process simulations. In: Di Francescomarino, C., Burattin, A., Janiesch, C., Sadiq, S. (eds.) International Conference on Business Process Management, pp. 178–195. Springer, Cham (2023). https://doi.org/10.1007/978-3-031-41623-1_11
21. Rama-Maneiro, E., Vidal, J.C., Lama, M.: Deep learning for predictive business process monitoring: review and benchmark. IEEE Trans. Serv. Comput. **16**(1), 739–756 (2023)
22. Ranta, E., Lundberg, P., Kaitala, V.: Ecology of Populations. Cambridge University Press (2005)
23. Rhinehart, R.R.: Automated steady and transient state identification in noisy processes. In: 2013 American Control Conference, pp. 4477–4493. IEEE (2013)
24. Tax, N., Verenich, I., La Rosa, M., Dumas, M.: Predictive business process monitoring with LSTM neural networks. In: Dubois, E., Pohl, K. (eds.) CAiSE 2017. LNCS, vol. 10253, pp. 477–492. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-59536-8_30
25. Turan, E.M., Jäschke, J.: A simple two-parameter steady-state detection algorithm: Concept and experimental validation. In: Computer Aided Chemical Engineering, vol. 52, pp. 1765–1770. Elsevier (2023)
26. Vidgof, M., Wurm, B., Mendling, J.: The impact of process complexity on process performance: A study using event log data. In: International Conference on Business Process Management. pp. 413–429. Springer (2023)
27. Zahoransky, R.M., Brenig, C., Koslowski, T.: Towards a process-centered resilience framework. In: ARES, pp. 266–273. IEEE (2015)
28. Zivot, E., Wang, J.: Modeling Financial Time Series with S-PLUS, vol. 2. Springer, New York (2006). https://doi.org/10.1007/978-0-387-32348-0

Automating Performance Insights: Suggesting and Computing Process Performance Indicators from Event Logs

Simone Agostinelli[1(✉)], Adela del-Río-Ortega[2], Rocío Goñi-Medina[2], Andrea Marrella[3], Manuel Resinas[2], and Jacopo Rossi[3]

[1] Universitas Mercatorum, Rome, Italy
simone.agostinelli@unimercatorum.it
[2] SCORE Lab, I3US, Universidad de Sevilla, Seville, Spain
{adeladelrio,rgoni,resinas}@us.es
[3] Sapienza Universitá di Roma, Rome, Italy
{marrella,j.rossi}@diag.uniroma1.it

Abstract. The increasing availability of event data tracing the execution of business processes represents an excellent opportunity for organizations to create relevant and measurable Process Performance Indicators (PPIs). PPIs are a tool to assess how well an organization achieves its key business objectives at an operational level and to support informed decision-making. To mitigate the risk of extracting mislabeled and misused PPIs with few or no connection with the available event data, in this paper, we present an approach and an implemented tool called PPIPilot to automatically suggest a list of measurable PPIs against a pursued organizational goal by providing an event log and a business process textual description as inputs. PPIPilot leverages the domain knowledge embedded in large-language models (LLMs) to suggest relevant PPIs from the event log and relies on the PPINAT definition model to compute them from the available data. We report on the results of a qualitative evaluation to investigate the feasibility and perceived usefulness of PPIPilot and a quantitative assessment to measure the extent to which PPIPilot is able to correctly suggest and compute PPIs from event logs.

Keywords: Process Performance Indicator · Event log · Business Process Analysis · Large Language Model · Natural Language Processing

1 Introduction

Measuring the performance of business processes has become a central concern for organizations, which are constantly challenged to evaluate how well they achieve their key business objectives at an operational level. To meet this purpose, the definition, computation, and analysis of measurable Process Performance Indicators (PPIs) –which are quantifiable metrics used to evaluate the

J. Krogstie et al. (Eds.): CAiSE 2025, LNCS 15701, pp. 221–237, 2025.
https://doi.org/10.1007/978-3-031-94569-4_13

efficiency and effectiveness of one or more business processes and can be measured directly from data generated during the process execution [9]– provide valuable insights into whether a business process's results align with the strategic objectives within an organization by supporting informed decision-making and continuous process optimization [9].

The growing availability of event data stored in dedicated event logs that trace the execution of business processes has led to the emergence of research approaches investigating how PPIs can be quantified directly from the event logs [29,38]. These approaches have proven effective in computing the *value* of a PPI from an event log (e.g., *average time until reimbursement*: *3 weeks*), with the assumption that its *definition* ("*average time until reimbursement*") was already being identified based on well-established reference models [28,32], and its connection with the event log content made explicit.

However, they approaches address just a small part of the problem. The decision of which performance measures to adopt is a manual, complex and time-consuming process [25] usually performed by managers. Although best-practice frameworks like ITIL or SCOR provide an initial set of PPI definitions, they must be adapted to the organization's goals and somehow connected with the event log at hand. Yet, managers rarely have in-depth knowledge of the organization's event data, increasing the risk of focusing on mislabeled and misused PPIs with few or no connection with the business process event log. For example, suppose the event log under analysis does not include relevant details about the reimbursements or such details are incomplete. In that case, quantifying the PPI "*average time until reimbursement*" may lead to a biased measurement, suggesting a partial or invalid picture of the business process execution performance.

In this paper, we tackle this challenge by presenting an approach and an implemented tool, called PPIPilot, which provides managers with a convenient solution to obtain a list of measurable PPIs and their computation from a business process description and its corresponding event log against a pursued organizational goal, thus addressing the following research question:

(RQ): "*How relevant PPIs can be automatically identified and effectively computed from a given event log?*"

PPIPilot takes advantage of the creative[1] capabilities of large language models (LLMs) to support decision-making [5,19,39], enriched by their extensive domain knowledge spanning various fields [8,18,33], and acts as a co-pilot for managers to suggest a list of PPIs that can be derived from the event log. While LLMs are a powerful tool for generating insightful recommendations, they require human oversight to increase the accuracy and precision of the results, fostering a collaborative approach that merges the benefits of automation with the nuance of human judgment [35]. PPIPilot relies on the well-known PPI definition model of PPINAT [29] to compute a value for the suggested PPIs from the event log, useful to measure the performance of the business process execution observed

[1] With the term 'creative' we refer to the ability of LLMs to generate text, rather than implying human-like reasoning.

in the log. To showcase the feasibility and perceived usefulness of PPIPilot, we report the results of a qualitative evaluation conducted on real-world use cases involving business process domain experts. Additionally, to investigate the extent to which PPIPilot correctly suggests and computes PPIs from event logs, we conducted a quantitative assessment to quantify and classify the list of suggested PPIs across multiple dimensions, leveraging three state-of-the-art LLMs.

The rest of the paper is organized as follows. Section 2 discusses related work on PPI definition and computation. Section 3 shows the main steps of the approach. Section 4 describes the implemented tool over a sample use case and presents the results of the qualitative evaluation and the quantitative assessment, while Sect. 5 concludes the paper by tracing limitations and future works.

2 Related Work

The process of deciding which performance measures to adopt is complex and time consuming [25]. For this reason, many attempts are available in the literature to systematize and facilitate it. In the business process field, well-established performance measurement models such as the Balanced Scorecard [15] and the Devil's Quadrangle (focused on time, cost, quality, and flexibility) [12] offer key dimensions to consider when defining PPIs. However, they lack specific indicators or guidance for their definition [20]. On the other hand, best-practice frameworks like ITIL, SCOR, and domain-specific PPI collections [20] offer analysts a useful starting set of PPIs but have limitations: they are generic, require adaptation to organizational goals, and must be operationalized using available data [20].

Specific guidance on how to define or adapt existing PPIs has also been developed. Many authors suggest that well-defined PPIs must follow the SMART properties [7], i.e., PPIs must be *specific*, *measurable*, *achievable*, *relevant*, and *time-bounded*. Other approaches involve brainstorming sessions [25] and collaborative creation [10] of PPI dictionaries, leveraging semantic information of PPIs to establish relationships and equivalence between them. There is still the risk of the lack of alignment with the event log data available, failing then to meet the *measurability* property. The notion of *measurability* aligns with the framework in [7], which assesses the measurability of PPIs based on the data quality of the event logs used to compute them. Additionally, multi-criteria decision-making techniques (MCDM) like ANP or AHP can be used to refine PPI suggestions [17]. However, the above proposals focus solely on PPI selection, leaving their identification to manual methods like expert assessment.

Finally, some approaches focus on the automation of the definition of parts of an indicator based on data. An attempt to provide guidance for the definition of achievable target values based on event data is shown in [31]. Moreover, in [13, 27], the authors define methodologies that combine data mining techniques to semi-automatically identify potentially relevant lead indicators. However, they approaches rely on domain experts to initially define PPIs based on raw data, goals, and experience, a step that our article aims to streamline. The closest work to PPIPilot is [1], which uses an Ontological Enterprise Model (OEM) to

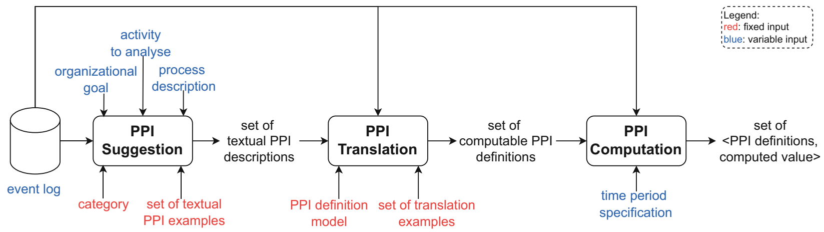


Fig. 1. Overview of PPIPilot including its three stages, namely: suggestion, translation and computation of PPIs.

automatically derive PPIs via predefined patterns, calculates their values from runtime data, and visualizes them in a dashboard. While similar to our goal, it requires building and maintaining complex PPI derivation patterns and an OEM. In contrast, PPIPilot only needs an event log and simple natural language input that can be provided by any business user.

Concerning PPI computation, most process mining tools support the quantification of some PPIs. In most cases, however, they just support a predefined set of metrics, mainly related to time. Some exceptions to it are *Celonis* Process Query Language (*PQL*) [38] and *ppinot4py* [30]. The main issue of these proposals is that users need to know low-level details of the log involved as well as technical aspects of the definition of PPIs. To overcome these limitations, other existing work aims to measure process performance by using textual descriptions and event logs, integrating state-of-the-art natural language processing techniques with matching strategies customized for the specific task [3,16,29]. PPIPilot builds on some of these proposals in its approach and extends them to suggest, from an event log, the set of PPIs that could potentially be computed.

3 PPIPilot Approach

From a methodological perspective, the approach has been conceptualized and designed to address the RQ identified and discussed in Sect. 1. The PPIPilot approach consists of 3 operational stages to be applied in sequence: *(i)* PPI Suggestion, *(ii)* PPI Translation, and *(iii)* PPI Computation, as shown in Fig. 1. The design of the approach has been driven by three principles:

1. The PPIs generated by PPIPilot must follow the well-established SMART guidelines to define PPIs [7]. The *Achievable* property is out of the scope of the paper because it is related to the target defined for the PPI instead of the metric, although proposals that automatically compute target values like [31] can be easily integrated.
2. The PPIs generated are intended to serve as suggestions for human users that need to be checked and refined. They are not intended to be a complete set of PPIs to be used as is in a performance management system.

3. The generated PPIs must be provided to the user together with their computation to support the domain expert in making decisions on their relevance.

PPI Suggestion. This stage lies its foundation in LLMs. It relies on prompt engineering as a source of creativity for suggesting textual PPI descriptions to be computed. The input to this stage involves the following elements:

– A ***Process Event Log*** for which we want to obtain the suggested and computed PPIs. This event log will also be used in the other two stages. In this stage, the event log serves to craft the names of activities, process variants, and process attributes into the prompt provided to the LLM.
– A textual ***process description*** of the event log that provides additional context about the business process behaviour underlying the recorded event log, and helps to further constrain the PPI Suggestion Stage.
– The ***pursued organizational goal***, such as improving efficiency, reducing costs, or increasing customer satisfaction. This is included to encourage the suggestion of relevant PPIs as recommended by the SMART guidelines.
– The specific ***activity*** to focus the analysis on among the set stored within the event log. This information helps identify which parts of the business process to examine more closely for process performance evaluation.
– The ***category*** of PPIs the user wants to focus on. In this paper, we support either the *(i)* time perspective (e.g., average duration of an activity) or the *(ii)* occurrence perspective (e.g., frequency/percentage of a particular activity), but it can easily be extended to other categories.

These inputs form the basis of the prompt provided to the LLM. To further guide the LLM to generate specific PPIs and mitigate the problem of hallucinations, we also provide in the prompt a carefully crafted ***set of textual PPI examples*** that provide additional context to the LLM and leads the LLM to: *(i)* suggest PPIs that are specific, as recommended by the SMART guidelines, by giving examples of specific PPIs using the category selected in the input; and *(ii)*make the suggested PPIs to refer to the activities and attributes of the event log by providing placeholders in the examples that the LLM must replace with activities or attributes from the log. This strategy is useful to increase the quality of the PPIs obtained, but it does not fully prevent the output of the PPI Suggestion Stage from including a few PPIs that are hallucinated and for which it is not possible to compute any value. To mitigate their impact, two additional strategies, detailed at the end of this section, are applied in the third stage.

The output returned by the LLM is a set of textual PPI descriptions. To this end, the LLM leverages on the domain knowledge that it implicitly possess due to their extensive training as shown in previous research [33].

Various types of general-purpose LLMs can be utilized for the enactment of this stage, including: GPT-4 [26], Mistral AI [23], Claude 2 [2], Gemini Pro [14], and LLaMA [21]. All these LLMs are designed to handle a wide range of natural language processing tasks and can generate diverse textual outputs.

PPI Translation. This stage translates the list of textual PPIs generated by the LLM to a computable PPI definition. Two approaches can be followed to implement the translation process. One approach is to generate a computable PPI definition directly in the format that is used by the PPI computation tool. This would mean, for instance, generating queries using Celonis PQL [38] for Celonis, or PPI definitions using PPINOT for ppinot4py [30]. An alternative approach is the one taken in [29], which involves creating an intermediate PPI definition model that has a narrower semantic gap with the textual PPI description than query languages like Celonis PQL, at the expense of losing expressiveness. The PPIs specified in this intermediate PPI definition model can be automatically translated into the format used by the PPI computation tool. The advantage of this approach is that, by reducing the semantic gap, it makes the translation task simpler and less prone to errors.

For this reason, in this paper, we use the computable PPI definition model of PPINAT [29], which supports three types of base measures, namely: *count measures* that include a condition that specifies when to count; *time measures* that are composed of two conditions that specify when the time measure starts and stops; and *data measures* that specify the attribute of the event log whose value we want to obtain. The conditions of count and time measures refer to conditions on the attributes of the event log. For instance, a count measure with a condition like *activity* = *payment handled* counts the number of times activity *payment handled* is performed in each case. In addition, these three types of base measures may include an optional condition on the result of the measure. For instance, a measure condition > 1 on the previous example makes the metric to return *true* if activity *payment handled* occurs once or more in each case, and $false$ otherwise. Finally, base measures are aggregated over a period of time, like one week or one month, using an aggregation function (e.g., sum or average). Furthermore, these aggregations can specify an attribute of the event log to group the results, and a condition to filter the cases included in the aggregation.

The implementation of this stage, however, differs from the one in [29], which involved three steps before the PPI computation: entity extraction, entity matching and PPI completion. In this paper, we replace the entity extraction and matching steps with a second prompt[2] including the following elements:

- The list of activities and attributes extracted from the input ***event log***, which ensure the LLM to use them in the computable PPI definitions.
- The set of ***textual PPIs suggested*** in the previous stage to be translated.
- The ***PPI definition model*** in PPINAT in JSON format, which is typically used when asking an LLM to generate a structured output, together with some guidelines on how to fill it.
- Some ***examples of translation*** from textual PPI descriptions according to the PPI definition model in PPINAT in JSON format. These examples exploit the few-shot learning abilities of LLMs [6] by supporting them to understand how to structure the output.

[2] The structure of the first and second prompt can be inspected at the following link: https://github.com/bpm-diag/PPIPilot.

After executing the prompt, we obtain a set of computable PPI definitions for the event log provided. The last step in the PPI Translation stage is to apply the PPI completion step defined in [29] to fill details that are left implicit in the textual PPI with common-sense interpretations. Specifically, this step involves filling missing time points, conditions and aggregations. For instance, in textual PPIs like "*The amount of time until reimbursement*", which define a time measure with only one end point, we fill the start point to refer to the beginning of the case. The interested reader can find more details in [29].

It is worth noticing we have introduced a separated step for PPI Translation instead of generating computable PPIs directly in the PPI Suggestion step for two reasons. First, it alleviates the complexity of the task, which is a strategy commonly used in LLMs to improve performance. Second, it allows to use different approaches for the PPI Translation stage that does not necessarily require LLMs as shown in [29].

PPI Computation. The last step of the framework is to compute a measure for the set of computable PPIs obtained in the previous step using the information provided by the process ***event log*** as input. Optionally, the user can provide as input the ***time period*** for which the PPI is going to be computed. By doing so, we cover the *Time-bounded* property of the SMART guidelines. PPIPilot implements the PPI Computation stage using ppinot4py [30] because there is a direct correspondence between the elements of the PPINAT computable definitions and the PPINOT model, so almost no translation is needed. However, other PPI computation tools like Celonis PQL could be easily integrated. The output of this stage is a set of PPI definitions along with their computed values.

Mitigating Hallucinations and Errors. Not all the PPIs obtained in the PPI Suggestion Stage can be computed, as few may be affected by hallucinations and, for instance, may not refer to elements of the event log. In addition, the PPI Translation step may produce invalid computable PPI definitions, or the textual PPI may not have a valid translation to a computable PPI definition because of limitations in the expressiveness of the model used to define computable PPIs. Moreover, since the process from suggestion to computation is fully automated, even if the textual PPI is computed, we cannot guarantee that the computed PPI actually corresponds to the textual PPI suggested. To mitigate these problems, we perform two actions:

1. We discard all PPIs whose computation leads to an error, either because the computable PPI definition is not well-formed or because it refers to activities or attributes that do not belong to the event log. By doing so, we ensure that all PPIs suggested by our approach meet the *Measurable* property of the SMART guidelines.
2. The textual PPI description that is returned by PPIPilot is not the one that was generated by the LLM in the PPI Suggestion Stage, but one that we generate directly from the computable PPI definition. By doing so, we can control the textual PPI description provided to the user and ensure that it accurately matches what is being computed.

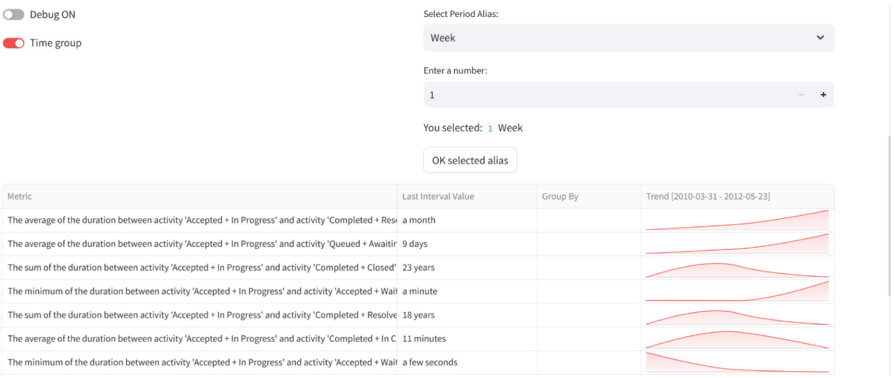


Fig. 2. Screenshot of the PPIPilot tool depicting the suggested PPIs together with their computation in the bottom of the figure. It also depicts the configuration of the time period and several configuration toggles.

We decided to adopt this strategy because the goal of PPIPilot is to suggest a set of useful PPIs that human experts can later refine. Therefore, we chose to prioritize outputs that contain valid and computed PPIs over unclear PPI descriptions without any quantified value for them.

4 Evaluation

The evaluation of PPIPilot involves two steps. First, a qualitative evaluation was conducted to get feedback from domain experts on the *feasibility* and *usefulness* of PPIPilot (see Sect. 4.2). Second, a quantitative assessment was carried out to understand the extent to which PPIPilot is able to suggest and compute PPIs from event logs (see Sect. 4.3). Specifically, we conducted experiments aimed at: *(i)* quantifying and classifying the list of PPIs generated by PPIPilot across multiple dimensions; *(ii)* analysing the performance of three state-of-the-art LLMs when generating computable PPIs (GPT-4, Mistral AI, and LLaMa); *(iii)* measuring the variability of PPIPilot in suggesting computable PPIs, considering the inherent creativity of LLMs; and *(iv)* investigating the impact on the quality of the results by including or excluding contextual information in the prompt provided in the PPI Suggestion Stage. The next section describes the software tool that implements the approach and is used for the evaluation.

4.1 Tool Implementation

PPIPilot has been implemented as a stand-alone tool using Streamlit[3], a free open-source framework to rapidly build web apps that ensures an interactive

[3] https://streamlit.io/.

and user-friendly interface. The tool is available for download at https://github.com/bpm-diag/PPIPilot. The current version uses the LLM *gpt-4-0125-preview*.

Figure 2 shows a screenshot of the tool. The time period used in the computation can be configured using the input boxes in the top-right side of the figure. The PPI can also be computed for the whole process event log by disabling the "*Time group*" toggle in the left top side of the figure. At the bottom of the figure, a table that contains the list of suggested PPIs and their values computed from the event log is depicted. In addition, the sparkline graph in the right column shows the evolution of the PPI over time. For instance, we can see that the trend of the PPI "*The average of the duration between activity 'Accepted + In Progress' and activity 'Completed + Resolved'*" is increasing week by week, with the latest interval value equal to one. The user can also see the PPIs for which it was not possible to compute a value by enabling the *"Debug ON"* toggle. Screencasts of the tool are available at the following link: https://github.com/bpm-diag/PPIPilot/tree/main/screencasts.

Table 1. Questions and their corresponding labels.

Question	Label
To what extent do you find PPIPilot useful for obtaining a set of computable PPIs against a given event log?	Q1
To what extent do you consider PPIPilot helps reduce the Knowledge on PPI definition required?	Q2
To what extent do you consider PPIPilot helps reduce the Knowledge about the event log required for the definition and computation of PPIs?	Q3
To what extent do you consider PPIPilot to be an intuitive tool for the definition and computation of PPIs?	Q4
To what extent were the suggestions made by PPIPilot useful?	Q5
To what extent did the PPIs suggested and computed by PPIPilot resemble your expected results?	Q6
To what extent do you consider PPIPilot can be used to define and compute PPIs in your business domain?	Q7

4.2 Qualitative Evaluation

An evaluation was conducted to get feedback from domain experts on the *feasibility* and *usefulness* of PPIPilot for suggesting and computing PPIs from event logs. The evaluation involved an online questionnaire comprising 18 questions. Initially, participants answered eight 5-point Likert Scale questions to provide information about their prior experience and knowledge in several key areas relevant to the project (e.g., process domain, PPI definitions) and their perspectives on the difficulty of defining appropriate and computable PPIs. Following these questions, participants viewed a demonstration of PPIPilot applied to a specific event log. Subsequently, they responded to seven 5-point Likert Scale questions regarding the feasibility and usefulness of PPIPilot (see Table 1). Lastly, three open-ended questions were included to gather more extensive feedback on general opinions, identified challenges, and potential areas for improvement.

Three different questionnaires were developed, each corresponding to a different event log and business process, with dual objectives: *(i)* to engage various profiles from different domains in the evaluation, and *(ii)* to assess our approach across diverse event logs. The event logs used were: IT incident management (BPI Challenge 2013) [34], Manuscript review management (anonimyzed event log obtained by systematically recording the review process enacted by an ACM journal), and Domestic Declarations (BPI Challenge 2020) [11]. The first questionnaire was distributed to 12 IT Service Management experts from industry, the second to 15 academic professionals with expertise in manuscript review management, and the third to 34 master students familiar to some extent with the domestic travel declaration process.

A total of 22 participants from five countries (Italy, Spain, the Netherlands, Estonia, and Brazil) participated in the study: seven IT service management experts, six academics, and nine master students, with a response rate of 36%. All participants in the first two groups reported being completely familiar with the domain (score 4 or higher). In contrast, 78% of students declared themselves moderately familiar (score 3 or higher), while only 56% of students were completely familiar. All participants joined the study voluntarily and anonymously, with all data treated with strict confidentiality post-collection[4].

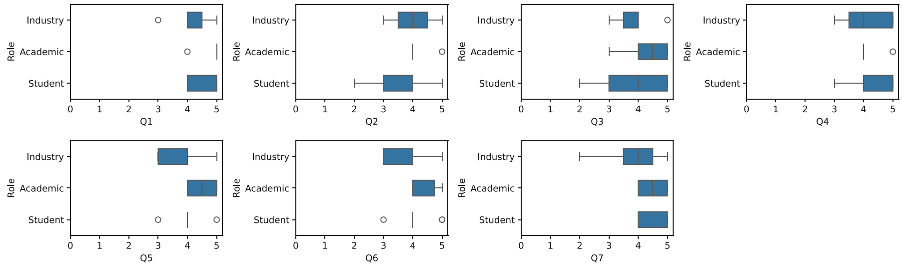


Fig. 3. Boxplots of the different variables related to the 7 questions in Table 1 about PPIPilot feasibility and usefulness, distinguishing the participant role.

Figure 3 displays a summary of the answers given by the participants according to their role: IT service management expert (referred to as "industry"), academic or (master) student. Regarding Q1, most participants reported a high

[4] Further information about the evaluation and links to the questionnaires can be found at https://github.com/bpm-diag/PPIPilot.

Table 2. GPT-4: avg. values over 10 runs

Logs	A	B	C	D
L1	134.9	104.6	10.6	47.9
L2	27.3	28.3	6.1	10.5
L3	59.9	31.2	5.3	4.6

Table 3. Mistral AI: avg. values over 10 runs

Logs	A + B	C + D
L1	520.7	108.3
L2	437.4	46.7
L3	595.5	85.7

Table 4. LLaMa: avg. values over 10 runs

Logs	A + B	C + D
L1	166.8	110.8
L2	174.2	82.4
L3	262.4	86.0

satisfaction (Score 4 or 5) with the usefulness of the tool to obtain a set of computable PPIs. Academics were especially positive, with all but one with a score of 5. Students were the least satisfied with the knowledge required for both PPI definition (Q2) and event log use (Q3), showing the most score variability, while academics remained highly optimistic. In relation to how intuitive PPIPilot is (Q4), although industry experts were slightly less positive (some scored it as 3), most participants found PPIPilot intuitive or very intuitive. The usefulness of the suggested PPIs and their similarity to the expected ones (Q5 and Q6 respectively) are considered high or very high by academics and students, while slightly less positive by industry experts (scores between 3 and 4). Finally, regarding the use of PPIPilot in the business domain of the participants (Q7), scores are quite high, between 4 and 5 for students and academics, with slightly more variability for industry experts.

Overall, the feedback from participants in the open-ended questions has been positive, highlighting that PPIPilot is intuitive and useful. Participants appreciated the tool's ability to support the definition and comparison of PPIs, particularly highlighting its intuitive interface and usefulness. However, they also provided valuable suggestions for improvement. Some mentioned the need for better explainability to help users determine which PPIs are most relevant. Others suggested integrating graphical representations of processes to improve user understanding. Additionally, participants found the trend overview feature particularly useful for identifying potential issues or unexpected patterns in the data. These insights are instrumental for future enhancements of PPIPilot.

4.3 Quantitative Assessment

The qualitative evaluation, which shows promising results, has been complemented by a quantitative assessment to analyze four different aspects, namely: *(i)* the extent to which PPIPilot is able to suggest and compute PPIs from event logs, *(ii)* the performance of different LLMs in terms of generating PPIs that can be computed, *(iii)* the variability of the PPIs generated across different runs, and *(iv)* the impact of removing elements from the input for the PPI Suggestion Stage. Next, we detail the experiments performed and the results obtained. All experiments are reported in an online appendix, available at https://github.com/bpm-diag/PPIPilot.

Ability to Suggest and Compute PPIs. To evaluate the ability of PPIPilot to suggest and compute PPIs, we performed ten runs of PPIPilot using *gpt-4-0125-preview* as LLM for three different event logs, and for both categories: time and occurrence. The three event logs used were the Domestic Declarations log (**L1**), which contains 17 activities; the IT incident management log (**L2**), which contains 13 activities, and the Manuscript review management log (**L3**), which contains 20 activities. In each *run* we executed PPIPilot for all the activities stored in the input event log. We opted for ten runs because we think it represents the right trade-off to show the extent to which PPIPilot is able to suggest and compute PPIs, considering the inherent creativity of LLMs. In the following, we present the aggregated results by computing the average values across the 10 runs, assessing four metrics:

- **A** #PPIs correctly translated from the Suggestion Stage and computed;
- **B** #PPIs incorrectly translated from the Suggestion Stage and computed;
- **C** #PPIs that are not well-formed according to the computable PPI definition (resulting in an error);
- **D** #PPIs that do not meet event log conditions (computed with empty values).

To compute these metrics, for **L1**, we manually tagged each PPI in the complete list generated by each run of PPIPilot for the 17 activities in the log, assigning a value type among A, B, C, or D. For **L2** and **L3**, we manually tagged a randomly selected subset comprising 30% of the PPIs generated by each PPIPilot run for 13 and 20 activities, respectively.

Table 2 presents the results of the time perspective only because of space limitations. We observe that on average, the number of PPIs that were correctly/incorrectly (A/B) translated from the Suggestion Stage and then correctly computed exceeds the number of PPIs resulting in an error or empty value (C/D). Furthermore, if we turn our attention to the comparison between columns A and B, it appears that the A values are slightly higher than the B values, except for **L2**, where they are quite similar. This indicates that PPIPilot is able to provide a list of PPIs that are correctly translated from the Suggestion Stage and then successfully computed. This suggests that, in most cases, the PPI textual descriptions (output of the Suggestion Stage) align with the computable PPI definitions (output of the Translation Stage). On the other hand, the C values are lower than the D values, except for **L3**, where again they are quite similar, which means that PPIPilot is able to limit the number of PPIs that cannot be computed due to formatting issues in the computable PPI definition.

Indeed, hallucinations if present during the Suggestion Stage, are mitigated in the Translation Stage. This implies that hallucinated PPIs can fall into dimensions B, C, or D once they have been translated into PPI computable definitions.

Performance with Different LLMs. To demonstrate that our approach is also feasible with other LLMs, we repeated the previous experimentation with Mistral AI (*mistral-large-latest*) [24], and LLama (*llama-3.2-90b-vision-preview*) [22]. To streamline the tagging process for the PPIs, we combined dimensions A and B, as well as C and D. Consequently, we will now refer to these combined columns as A + B and C + D. Thus, A + B represents the number of PPI definitions computed with a value, while C + D denotes the number of PPI definitions that resulted in an error or were computed with an empty value.

The experiments in Table 3 show in average a high number of A + B values compared to C + D, indicating that, even in the case of Mistral AI, for **L1**, **L2** and **L3**, the PPIPilot approach is able to provide a high number of PPIs computed with a value while keeping the number of PPIs with errors or empty values relatively low. Similarly, the experiments in Table 4 for LLaMa show a greater number of PPIs computed with a value with respect to the ones resulted in an error or an empty value. However, in this case, the ratio of PPIs with errors or empty values to the total number of PPIs computed with a value (0.46) is three times higher than that for Mistral AI (0.15). We emphasize that the purpose of this experimentation is not to claim that Mistral AI outperforms GPT-4 or LLaMa. Instead, our goal is to demonstrate the feasibility of PPIPilot in suggesting and computing PPIs across state-of-the-art LLMs. The specific reasons why Mistral AI performs better are beyond the scope of this paper.

Table 5. number of distinct PPIs that appear in <n> run(s)

PPIs	320	75	59	45	44	0	30	22	29	37	/ 661
appearing in	1 run	2 runs	3 runs	4 runs	5 runs	6 runs	7 runs	8 runs	9 runs	10 runs	-

Variability in Suggested PPIs. In this experiment, we want to determine the *degree of variability* of the computable PPIs suggested by PPIPilot, given the inherent creativity of the underlying LLM in generating textual PPI descriptions. To this end, we executed 10 complete runs of PPIPilot using GPT-4 on **L1** for the time perspective and we compared the PPIs generated by PPIPilot in each run. To compare the PPIs we used the computable PPI definitions instead of the textual PPI descriptions generated in the PPI Suggestion Stage because the goal of PPIPilot is to suggest a set of useful PPIs that humans can later consider and refine. Therefore, we chose to prioritize the output that contains valid and computable PPI definitions (e.g., PPIs of type A and B) over the unclear PPI definitions without any computed values (e.g., PPIs of type C and D). The results depicted in Table 5 show that 10% of the distinct PPIs (66 out of 661) are common to at least 9 runs (29 from 9 runs and 37 from 10 runs, thus a total of 66), while almost 50% (320 out of 661) appear in one run only.

This demonstrates that PPIPilot generates lists of PPIs with significant variability between runs reflecting the high level of creativity of LLMs. Indeed, a PPI appearing in only one run does not necessarily imply that it is not relevant to be considered. Examples of these PPIs are like those with "group by" that appear in only one run but they should be considered for humans refinement (e.g. "*Number of 'Declaration APPROVED by BUDGET OWNER' activities grouped by BudgetNumber*").

Impact of Input Elements. In the next set of experiments, we aim to explore the impact of removing certain elements from the prompt used as input for the PPI Suggestion Stage. Specifically, we investigate the impact of removing the process description, the organizational goal, or both from the prompt. This experiment was conducted using GPT-4 with a focus on the time perspective.

Table 6 highlights that the presence of both a description and a goal achieves the highest scores for A + B and the lowest scores for C + D across all logs, indicating that the inclusion of both elements significantly improves the quality of the results. The absence of either a description or a goal slightly reduces the scores for A + B while minimally affects the C + D values. The combined absence of both description and goal results in the lowest scores for A + B across all logs. In summary, while the individual contribution of either a description or a goal is positive, their combination is essential to maximize scores, highlighting the importance of context for LLMs like GPT-4.

Table 6. GPT-4: avg. values over 10 runs by including/excluding description/goal

	yes descr. yes goal		no descr. yes goal		yes descr. no goal		no descr. no goal	
Logs	**A + B**	**C + D**	**A + B**	**C + D**	**A + B**	**C + D**	**A + B**	**C + D**
L1	239.5	58.5	228.8	62.6	228.6	67.5	212.0	71.2
L2	168.7	48.5	162.7	50.6	165.0	58.1	161.7	68.3
L3	284.2	30.1	282.3	37.0	281.2	32.4	280.5	34.3

5 Conclusion

In this paper, we presented PPIPilot, an approach and implemented tool that by providing a process event log, a process textual description, an organizational goal, an activity to analyse and the category of PPIs to focus on is able to automatically suggest and compute a set of PPIs that follow the SMART guidelines. Thus, leveraging the embedded domain knowledge within LLMs [4], PPIPilot acts as a co-pilot, suggesting relevant PPIs in natural language and computing their values from the event log. To the best of our knowledge, this is the first attempt where the use of LLMs is explored for identifying, from an event log, the set of PPIs that could potentially be computed. By relying on the well-established computable PPI definition model of PPINAT, PPIPilot also computes the performance of the suggested PPIs. The demonstrated perceived usefulness of PPIPilot through user testing on real-world use cases underscores its potential to streamline PPI measurement and support organizations in achieving their operational and strategic objectives. Additionally, the quantitative assessment of PPIPilot highlights its effectiveness in suggesting and computing PPIs from event logs.

However, despite the positive outcomes, there are certain limitations: *(i)* PPIPilot's ability to suggest and compute a list of PPIs heavily relies on the quality and availability of the event logs [36]. Incomplete or inaccurate data may lead to inaccurate suggestions. Also, the pursued organizational goal and the process description should be coherent with the underlying recorded event log. If the user provides arbitrary or inconsistent text, it may negatively impact the quality of the suggested PPIs. Therefore, ensuring data quality and integrity is mandatory for maximizing the benefits of PPIPilot; *(ii)* the usefulness study involves a small number of participants from different domains and without actually using the tool, thus the perceived usefulness may need further validation through larger-scale user studies with an actual use of the tool.

For future work, we plan to: *(i)* fine-tune a specific LLM with a tailored set of PPIs to improve the ability of PPIPilot in suggesting and computing PPIs, *(ii)* extend the categories of PPIs supported, *(iii)* include mechanisms to filter the final set of suggested and computed PPIs based on the information available in the event log, *(iv)* develop a new feature that, by analyzing multiple runs of PPIPilot, can extract PPIs appearing in almost all runs, *(v)* extend the qualitative experiments to address the limitation mentioned above.

Finally, from a technological perspective, we aim to evaluate to what extent the recent o1 (Strawberry) model from OpenAI, called Large Reasoning Model (LRM) and specifically constructed and trained to be an approximate reasoner escaping the usual limitations of autoregressive LLMs [37], can (or cannot) improve the accuracy of the PPIPilot results.

Acknowledgments. The work of A. Marrella has been supported by the FOND-AIBPM project, the PRIN 2022 project MOTOWN, and the PNRR MUR project PE0000013-FAIR. J. Rossi is supported by Thales Alenia Space and Regione Lazio, through the fellowships 35757-22066DP000000041-A0627S0031 *Advanced Software Based on Cloud Computing and Machine Learning for Space Systems*. The work of M. Resinas and A. del-Río-Ortega has been supported by R&D projects PID2021-126227NB-C21 funded by MICIU/AEI/10.13 039/501100011033/ FEDER/UE; TED2021-131023B-C22 funded by MICIU/AEI/10.13039/501100011033/European Union NextGenerationEU/PRTR; and PID2022-140221NB-I00, funded by MICIU/AEI/10.13039/501100011033/ERDF/EU.

References

1. Aksu, U., Schunselaar, D., Reijers, H.: An approach for automatically deriving key performance indicators from ontological enterprise models. In: SIMPDA 2017 Data-driven Process Discovery and Analysis. CEUR Workshop Proceedings, vol. 2016, pp. 38–53 (2017)
2. Anthropic: Claude 2 (2024). https://www.anthropic.com/news/claude-2
3. Barbieri, L., Madeira, E.R.M., Stroeh, K., van der Aalst, W.M.: A natural language querying interface for process mining. J. Intell. Inf. Syst. (2022)
4. Bernardi, M.L., Casciani, A., Cimitile, M., Marrella, A.: Conversing with business process-aware large language models: the BPLLM framework. J. Intell. Inf. Syst. **62**(6), 1607–1629 (2024). https://doi.org/10.1007/s10844-024-00898-1

5. Bhavya, B., Xiong, J., Zhai, C.: Cam: a large language model-based creative analogy mining framework. In: Proceedings of the ACM Web Conference 2023, pp. 3903–3914 (2023)
6. Brown, T., et al.: Language models are few-shot learners. In: Advances in Neural Information Processing Systems, vol. 33, pp. 1877–1901. Curran Associates, Inc. (2020)
7. Cappiello, C., Comuzzi, M., Plebani, P., Fim, M.: Assessing and improving measurability of process performance indicators based on quality of logs. Inf. Syst. **103**, 101874 (2022)
8. Casciani, A., Bernardi, M.L., Cimitile, M., Marrella, A.: Conversational systems for AI-augmented business process management. In: Research Challenges in Information Science, pp. 183–200. Springer (2024)
9. del-Río-Ortega, A., Resinas, M., Durán, A., Bernárdez, B., Ruiz-Cortés, A., Toro, M.: VISUAL PPINOT: a graphical notation for process performance indicators. Bus. Inf. Syst. Eng. **61**(2), 137–161 (2019)
10. Diamantini, C., Potena, D., Storti, E.: SemPI: a semantic framework for the collaborative construction and maintenance of a shared dictionary of performance indicators. Futur. Gener. Comput. Syst. **54**, 352–365 (2016)
11. van Dongen, B.: BPI challenge 2020 domestic declarations (2020). https://doi.org/10.4121/uuid:52fb97d4-4588-43c9-9d04-3604d4613b51
12. Dumas, M., La Rosa, M., Mendling, J., Reijers, H.A.: Fundamentals of Business Process Management. Springer, Berlin, Heidelberg (2018)
13. Estrada-Torres, B., et al.: Measuring performance in knowledge-intensive processes. ACM Trans. Internet Technol. **19**(1), 15:1-15:26 (2019)
14. Google-DeepMind: Gemini pro (2024). https://deepmind.google/technologies/gemini/pro/
15. Kaplan, R.S., Norton, D.P.: The Balanced Scorecard: Translating Strategy into Action. Harvard Business School Press, Boston (1996)
16. Kobeissi, M., Assy, N., Gaaloul, W., Defude, B., Haidar, B.: An intent-based natural language interface for querying process execution data. In: ICPM, pp. 152–159. IEEE (2021)
17. Kucukaltan, B., Irani, Z., Aktas, E.: A decision support model for identification and prioritization of key performance indicators in the logistics industry. Comput. Hum. Behav. **65**, 346–358 (2016)
18. Li, Y., Wang, S., Ding, H., Chen, H.: large language models in finance: a survey. In: Proceedings of the Fourth ACM International Conference on AI in Finance, pp. 374–382 (2023)
19. Liu, Y., et al.: How AI processing delays foster creativity: exploring research question co-creation with an LLM-based agent. In: Proceedings of the CHI Conference on Human Factors in Computing Systems, pp. 1–25 (2024)
20. Looy, A.V., Shafagatova, A.: business process performance measurement: a structured literature review of indicators. Meas. Metrics **5**(1), 1797 (2016)
21. Meta: Llama 2 (2024). https://www.llama.com/llama2/
22. MetaAI: llama-3.2-90b-vision-preview model (2024). https://www.llama.com/llama-downloads
23. MistralAI: Frontier AI in your hands (2024). https://mistral.ai/
24. MistralAI: mistal-large-latest model (2024). https://docs.mistral.ai/getting-started/models/models_overview/
25. Neely, A., et al.: Performance measurement system design: developing and testing a process-based approach. Int. J. Oper. Prod. Manag. **20**(10), 1119–1145 (2000)

26. OpenAI: GPT-4 (2024). https://openai.com/index/gpt-4/
27. Peng, W., Sun, T., Rose, P., Li, T.: A semi-automatic system with an iterative learning method for discovering the leading indicators in business processes. In: Proceedings of the 2007 International Workshop on Domain Driven Data Mining, pp. 33–42. DDDM '07, ACM (2007)
28. Popova, V., Sharpanskykh, A.: Modeling organizational performance indicators. Inf. Syst. **35**(4), 505–527 (2010)
29. Resinas, M., del Río-Ortega, A., van der Aa, H.: From text to performance measurement: automatically computing process performance using textual descriptions and event logs. In: Business Process Management, pp. 266–283. Springer, Cham (2023)
30. Resinas, M., del Río-Ortega, A., Ruiz-Cortés, A.: PPINOT computer and ppinot4py: two libraries to compute process performance indicators. In: ICPM (Demo track) (2021)
31. del-Río-Ortega, A., García, F., Resinas, M., Weber, E., Ruiz, F., Ruiz-Cortés, A.: Enriching decision making with data-based thresholds of process-related KPIs. In: Dubois, E., Pohl, K. (eds.) CAiSE 2017. LNCS, vol. 10253, pp. 193–209. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-59536-8_13
32. del-Río-Ortega, A., Resinas, M., Cabanillas, C., Ruiz-Cortés, A.: On the definition and design-time analysis of process performance indicators. Inf. Syst. **38**(4), 470–490 (2013)
33. Singhal, K., et al.: Large language models encode clinical knowledge. Nature (2023)
34. Steeman, W.: BPI challenge 2013, incidents (2013). https://data.4tu.nl/articles/dataset/BPI_Challenge_2013_incidents/12693914/1
35. Suh, S., Chen, M., Min, B., Li, T.J.J., Xia, H.: Luminate: structured generation and exploration of design space with large language models for human-AI co-creation. In: Proceedings of the CHI Conference on Human Factors in Computing Systems, pp. 1–26 (2024)
36. ter Hofstede, A.H.M., et al.: Process-data quality: the true frontier of process mining. J. Data Inf. Qual. **15**(3) (2023). https://doi.org/10.1145/3613247
37. Valmeekam, K., Stechly, K., Kambhampati, S.: LLMs still can't plan; can LRMs? A preliminary evaluation of OpenAI's o1 on PlanBench. arXiv preprint arXiv:2409.13373 (2024)
38. Vogelgesang, T., Ambrosy, J., Becher, D., Seilbeck, R., Geyer-Klingeberg, J., Klenk, M.: Celonis PQL: a query language for process mining. In: Process Querying Methods, pp. 377–408. Springer (2022)
39. Xu, X., et al.: Jamplate: exploring LLM-enhanced templates for idea reflection. In: Proceedings of the 29th International Conference on Intelligent User Interfaces, pp. 907–921 (2024)

IS-Development and Usage

Collaborative Multi-organization Information System Engineering Based on Team Practice Agreements

Javier Fernández-Castillo(✉), José María Garcia, and Pablo Fernandez

SCORE Lab, I3US Institute, Universidad de Sevilla, Sevilla, Spain
{jfernandez26,josemgarcia,pablofm}@us.es

Abstract. The development of complex information systems (IS) often requires collaboration between multiple engineering organizations, each contributing unique methodologies, tools, and practices. Coordinating these diverse approaches presents significant challenges, particularly in achieving team alignment, fostering effective collaboration, and breaking down silos. This paper explores a real-world scenario involving three companies, each employing different agile software development methodologies, working together to engineer a complex IS for the public administration. To address the challenges inherent in multiorganizational collaboration, we make use of Bluejay, a framework designed to audit and optimize adherence to agile best practices through the use of Team Practice Agreements (TPAs). Specifically, Bluejay collects data through APIs from tools such as GitLab, Jira, or Redmine, providing real-time dashboards to monitor team performance. It describes specific metrics within TPAs to ensure alignment with agile methodologies in the organization. In this paper, we present how Bluejay was integrated into the workflows of the three companies, each using different tools and methodologies. Despite these differences, we identified common metrics that were used to design a unified TPA that could harmonize operations across organizations. The shared TPA aligns the teams with industry best practices and has the potential to promote productivity, quality, and better coordination between the different organizations involved.

Keywords: Team Practices · Team Coordination · Agile Auditing · Multi-Organizational Collaboration · Industrial Case Study

1 Introduction

The implementation of best practices in Information System (IS) engineering is a critical aspect in the Software Industry, especially given the increasing number of projects, growing system complexity, and the demand for maintaining quality and efficiency standards in agile, collaborative work environments. As organizations increasingly adopt agile methodologies, the challenge becomes not only maintaining internal standards but also fostering effective collaboration between teams and organizations with differing practices, tools, and processes.

J. Krogstie et al. (Eds.): CAiSE 2025, LNCS 15701, pp. 241–257, 2025.
https://doi.org/10.1007/978-3-031-94569-4_14

In IS engineering, best practices aim to optimize key processes such as task management, team collaboration, and continuous improvement delivery [7]. These practices help teams reduce risks [5], improve internal communication, and ensure that the code produced is robust, maintainable, and aligned with the project objectives. However, in a multiorganizational environment, achieving alignment in these practices is challenging due to differences in methodologies, cultures, and toolsets [1].

One of the main barriers to successfully implementing and adhering to best practices is the lack of standardized metrics and appropriate tools to evaluate adherence between different teams. This issue becomes even more pronounced when collaboration involves multiple organizations, each employing unique methodologies and using different tools to manage their projects. Without clear metrics or an effective way to monitor adherence to standards, it becomes difficult for companies to determine whether they are truly following best practices or where they may need to improve.

To address this challenge, we propose the usage of Bluejay [4]-an automated monitoring framework designed to collect data from tools such as GitLab, Jira, and Redmine, which are used in the daily operations of different companies. Bluejay allows for the creation of a shared understanding of agile best practices, using Team Practice Agreements (TPAs) as a mechanism to align practices across teams. In this paper, we present an empirical case study to evaluate the impact of Bluejay on aligning team practices in a multiorganizational agile environment.

Bluejay leverages APIs from project management and code repository tools to collect data, which is then presented in real-time dashboards. This data-driven approach enables engineering managers and team members to audit whether teams adhere to agile best practices. By defining specific metrics for Team Practices (TP) within TPAs, Bluejay provides a structured framework for achieving cross-team alignment, even in complex multiorganizational collaborations.

In Bluejay, a Team Practice (TP) refers to a specific action or behavior that an agile team should follow during their workflow, such as creating a feature branch when they start working on a user story. These practices often involve multiple tools, including project management systems and code repositories. A Team Practice Agreement (TPA) is a formalized set of TPs that serves as a global agreement for a team or multiple collaborating teams. TPAs specify the TPs teams must follow and define metrics to measure compliance, functioning in a similar manner to how Service Level Agreements (SLAs) manage expectations in service-oriented environments.

During the experimental phase of this study, Bluejay was integrated into the technological infrastructures of three participating companies, each of them employing different agile software development methodologies. The goal was to collect data on the adherence to TPAs and to evaluate the effectiveness of using a shared framework to align practices between diverse teams. Despite each company's distinct tools and methodologies, Bluejay successfully identified common metrics that could be used to form a unifying TPA, fostering improved collaboration and alignment.

The significance of this study lies in its ability to provide an objective, quantifiable view of adherence to agile best practices in complex software development environments. By analyzing the data obtained through Bluejay, we can accurately assess the degree to which teams align with the proposed standards and identify areas for improvement. Additionally, the use of an automated framework like Bluejay removes the biases associated with manual evaluations, providing accurate real-time data that reflect the behavior of development teams over time.

By monitoring the progress of each company in adopting TPAs, this study aims not only to improve the efficiency and quality of software development but also to develop a generalized set of best practices that can be applied across multiple organizations. This contributes to a broader understanding of how to achieve effective collaboration in multi-organizational information system engineering, ultimately providing valuable insights for future collaborative initiatives.

The paper is organized as follows. Section 2 presents the motivating scenario describing the coordination challenges amongst companies. Section 3 describes the application of Bluejay in the context of the three participating companies. Section 4 provides a discussion of the results and insights gained from the experimental phase. Section 5 reviews related work in the field of multiorganizational collaboration and agile methodologies. Finally, Sect. 6 concludes the paper and outlines directions for future work.

2 Motivating Scenario

The motivating scenario for this paper is the development of a flexible and efficient e-infrastructure to allow advanced geographic data analysis for the public administration. This ambitious initiative required the collaboration of three distinct companies, each bringing their specialized expertise, tools, and methodologies to the table. These organizations were tasked with designing and developing a series of innovative prototypes, each addressing specific components of the infrastructure to align with the project's overarching objectives. To maintain confidentiality, we will refer to this initiative as the Alpha Project and designate the three collaborating entities as Company A, Company B, and Company C throughout this document. In such context, the usage of Team Practice Agreements (TPAs) was proposed to promote the usage of Industry best practices and facilitate the alignment between different IS engineering models in each company.

Initially, to address the unique needs of each company, a customized TPA was designed for each organization, ensuring alignment with their specific methodologies and tools. These TPAs served as a framework for monitoring and promoting the adherence to best practices within the context of their respective workflows.

After monitoring and analyzing how each company implemented the best practices proposed, the objective was to create a unified framework to support software processes during the later phases of the project. By synthesizing the most effective practices from the three tailored TPAs, the project sought to develop a unifying TPA, a comprehensive standard that integrates the strengths of each approach, fostering improved collaboration and alignment between diverse methodologies. This approach is intended to break down silos between organizations and build a bridge between their distinct methodologies, enabling seamless collaboration.

In the following subsections, the paper presents the methodologies of the three organizations and identifies their proposed best practices.

2.1 Company A Methodology

Company A follows a structured methodology based on the JIRA tool to manage all project requests, tasks, and deliveries.

Request Management. Request management is a crucial process for organizing work within the company. All requests are recorded in JIRA, allowing for thorough tracking of their status and ensuring that they stay within the defined parameters of the Epics.

Work Structure: Epics, Tasks, Subtasks and Issues. The work structure is organized around Epics, which correspond to the large objectives of the project. From these epics, specific tasks are created to be investigated and developed during each sprint. Tasks may generate subtasks related to more detailed and specific activities, while issues are recorded if problems are detected during the testing phase.

Development Cycle and Deliverables. The development cycle is organized into Sprints, with each Sprint focused on a specific set of tasks and objectives. At the end of each sprint, a deliverable certification is performed, including validation and testing. The delivery of results can be done through Git or directly via JIRA, depending on the type of deliverable. The integration between JIRA and GitLab facilitates this process, as it allows linking JIRA issues and tasks with Git commits.

Monitoring and Control. Project monitoring is carried out using JIRA, which provides a global view of the status of requests, tasks, and deliverables. This tool provides key statistical data, such as the execution time of each task and the responsibilities assigned to each team member.

2.2 Company B Methodology

Company B follows a comprehensive and dynamic approach to managing development tasks and collaboration between its teams. The core of their methodology is the use of tools like GitLab and Redmine to ensure that tasks are consistently updated and aligned with the development process.

Task and Branch Management in GitLab. The development workflow in Company B is centered on GitLab for version control. Team members are expected to maintain an ongoing flow of work by continuously managing branches and commits. Branches should be created and closed regularly, and each development task in GitLab should be clearly linked to corresponding issues in Redmine. The team also integrates continuous updates and merges into the development cycle.

Issue Tracking in Redmine. Redmine serves as the central tool to manage and track tasks, with a strong emphasis on keeping issues actively updated. Team members are required to ensure that issues are continuously tracked, and that each issue is updated regularly to reflect its current status. The updates to issues must align closely with the work being done in GitLab.

Continuous Integration and Release Management. A key principle in Company B's methodology is continuous integration. The development work in GitLab is seamlessly connected to the progress of the tasks in Redmine, with the tasks and branches frequently being reviewed and updated. Releases are closely managed and tracked, ensuring that each release corresponds directly to the completion of branches and tasks.

2.3 Company C Methodology

Company C follows a methodical approach to project management with a strong emphasis on structured workflows and tracking mechanisms. The methodology of Company C revolves around two key tools: Redmine for task management and GitLab for version control.

Workflow Management. The organization has defined clear processes for managing user stories. Each user story is tracked and updated regularly in Redmine, where it is categorized, prioritized, and monitored throughout its life cycle. Team members are assigned specific user stories that are actively worked on and completed.

Task Execution and Branching Strategy. When a user story is initiated, a corresponding GitLab branch is created to handle its development. As development progresses, the feature branches are merged back into the main codebase. The company emphasizes the importance of creating these branches and merging them in a timely manner.

Continuous Integration and Delivery. The organization places a strong focus on maintaining continuous integration practices through GitLab. Developers are expected to push their changes regularly to the "develop" branch. This practice promotes consistency and prevents integration issues later in the development cycle.

Regular Monitoring and Adjustment. To maintain the efficiency of the process, the company conducts regular reviews of the progress of user stories and development tasks. This allows the team to quickly identify and address any issues, ensuring that the workflow remains uninterrupted and that all goals are met on time. The methodology encourages frequent updates and communication between team members to ensure that everyone is aligned with the overall project goals.

3 Bluejay in Alpha Project

Bluejay is an open source framework designed to audit best practices in agile software development. In the Alpha project, Bluejay was used to monitor the three anonymous companies according to their methodologies.

3.1 Bluejay in a Nutshell

Bluejay [4] infrastructure is designed to support the collection, monitoring, and auditing of agile Team Practices (TPs) by leveraging multiple development tools used by teams. Bluejay is built using a microservices architecture based on the Governify ecosystem [3] for SLA management [10]. Bluejay introduces two major enhancements: a Domain-Specific Language (DSL) for integrating various tools to define TPAs and a customizable dynamic dashboard. This dashboard monitors the adherence to practices over time between teams and is powered by multiple microservices organized into three main areas.

- **TPA Modeler**: Instructors use this component to define Team Practice Agreements (TPAs), which outline the TPs that teams need to follow and the metrics to be tracked.
- **Monitoring**: Once a TPA is defined, Bluejay begins to monitor compliance by collecting data from multiple APIs from external tools.
- **Auditing**: Bluejay processes the collected data and generates visual dashboards that allow instructors and teams to assess whether the TPs are being followed.

A TPA represents a formalized agreement for the team, consisting of a set of defined TPs. It outlines the required practices the team must adhere to and establishes metrics to measure compliance. By providing a structured framework, TPAs help teams maintain alignment with agile methodologies. Each TP is characterized by a specific objective (condition), a defined scope, and a time window for its completion. The time window specifies the period in which specific actions must occur. For example: hourly, daily, weekly, monthly, or annually. The scope determines whether the TP applies to the entire team or individual members. Team-level TPs ensure collective performance, while member-level TPs guide individual contributions.

After processing data according to the defined TPA, Bluejay generates dynamic dashboards that display both team-wide and individual compliance over time. These dashboards help instructors and teams identify potential problems or areas for improvement.

The auditing system allows users to see results at both the team and individual levels. The generated graphs show whether the team is meeting the goals defined in the TPAs, and these visualizations can be customized according to the needs of the course or project. Audits are displayed in graphical format, with tables showing the specific values for each TP, marked green for compliance and red for violations.

3.2 Tools Monitored by Each Company

The monitoring tools selected varied between companies, but were identified as the most critical components of their workflows through discussions with their teams. Company A primarily used GitLab for code management and Jira for task tracking, while Companies B and C relied on GitLab for development and Redmine for project management.

After consulting with representatives of each company, it was agreed that these tools played a central role in their respective methodologies and workflows, making them the focus of the monitoring process.

To facilitate this, the Collector microservice in Bluejay was extended to gather and process data from the REST APIs of these three tools, enabling a comprehensive monitoring of their adherence to practices.

3.3 TPA Modeling

Each company had a different TPA tailored to its specific needs and development environment. These TPAs were designed to reflect the best practices and workflow preferences of each organization, ensuring relevant metrics for monitoring and improvement. In the following, the TPA for each company is described in detail.

Company A uses six different tools: Jira, Jenkins, GitLab, Nexus, SonarQube, TestLink, and Microsoft Teams. However, only Jira, for project management, and GitLab, for version control and code collaboration, were monitored. The TPA includes 6 distinct TPs, as shown in Table 1.

Table 1. TPA for Company A.

TP	Tool	Scope	Window	Metric	Condition
A.TP1	GitLab	Team	Weekly	Number of commits	≥ 10
A.TP2	GitLab	Team	Weekly	Number of new branches	≥ 1
A.TP3	GitLab	Team	Weekly	Number of new merge requests	≥ 1
A.TP4	GitLab	Team	Weekly	Number of updated branches	≥ 2
A.TP5	Jira	Team	Weekly	Number of new tasks	≥ 2
A.TP6	Jira	Team	Weekly	Number of updated tasks	≥ 2

Company B relies on just two tools: Redmine and GitLab. Metrics were established for both, Redmine supporting project management, and GitLab handling version control and code collaboration. The TPA covers a total of 15 distinct TPs, as shown in Table 2.

Company C makes use of two main tools: Redmine and GitLab. Metrics were established for both, Redmine supporting project management, and GitLab

Table 2. TPA for Company B.

TP	Tool	Scope	Window	Metric	Condition
B.TP1	GitLab	Team	Weekly	Number of commits	≥ 10
B.TP2	GitLab	Team	Weekly	Number of closed branches	≥ 1
B.TP3	GitLab	Team	Weekly	Number of updated branches	≥ 2
B.TP4	GitLab	Team	Monthly	Number of updated branches	≥ 13
B.TP5	GitLab	Team	Weekly	Number of new branches	≥ 1
B.TP6	GitLab	Team	Weekly	Number of new merge requests	≥ 1
B.TP7	GitLab	Team	Weekly	Number of releases	≥ 1
B.TP8	Redmine	Team	Weekly	Number of new tasks	≥ 1
B.TP9	Redmine	Team	Weekly	Number of updated tasks	≥ 1
B.TP10	Redmine	Team	Weekly	Number of closed tasks	≥ 1
B.TP11	Redmine	Team	Monthly	Number of updated user story tasks	≥ 5
B.TP12	Redmine	Team	Monthly	Number of updated tasks	≥ 5
B.TP13	Redmine	Team	Daily	Number of tasks updated within 24 h	≥ 1
B.TP14	GitLab and Redmine	Team	Weekly	Percentage correlation between new branches and tasks moved to In progress within less than a day	≥ 75%
B.TP15	GitLab	Team	Weekly	Percentage correlation between releases and closed branches within less than a day	≥ 75%

Table 3. TPA for Company C.

TP	Tool	Scope	Window	Metric	Condition
C.TP1	Redmine	Team	Weekly	Percentage of closed 1-point user stories	= 100%
C.TP2	Redmine	Team	Monthly	Percentage of closed user stories	≥ 90%
C.TP3	Redmine	Team	Weekly	Percentage of user stories in progress	≥ 90%
C.TP4	Redmine	Team	Weekly	Percentage of closed user stories	≥ 50%
C.TP5	Redmine	Team	Weekly	Percentage of user stories closed in less than 7 days	≥ 50%
C.TP6	Redmine	Member	Weekly	Number of user stories in progress	< 3
C.TP7	GitLab and Redmine	Team	Weekly	Percentage correlation between new branches and tasks moved to In progress within less than 1 day	≥ 50%
C.TP8	GitLab and Redmine	Team	Weekly	Percentage correlation between user stories closed and branches closed within less than 3 days	≥ 75%
C.TP9	GitLab	Team	Monthly	Ratio of commits in the develop branch	≥ 1

Table 4. Metrics of task management tools used by companies.

Metric	Comp. A	Comp. B	Comp. C
Number of new issues/tasks/user stories	X	X	
Number of updated issues/tasks/user stories	X	X	X
Number of closed issues/tasks/user stories		X	X
Number of 'in progress' issues/tasks/user stories		X	X
Number of members in 'in progress' user stories			X
Number of closed 1-point user stories			X

handling version control and code collaboration. The TPA includes 9 distinct TPs, as shown in Table 3.

3.4 Commonality

When trying to bridge the proposed TPAs for companies, several common points emerge that unify the best practices of different organizations through shared metrics, demonstrating that despite the differences in tools and approaches, there are common standards in the application of best practices. Typically, the best practices that differ between organizations are those that are more distinctive and inherent to the organization's specific work methodology. Tables 4 and 5 show that, regardless of the methodologies and tools used, there are common metrics that connect all three organizations.

Table 5. Metrics of code management tools used by companies.

Metric	Comp. A	Comp. B	Comp. C
Number of commits	X	X	
Number of new branches	X	X	X
Number of merge requests	X	X	
Number of updated branches	X	X	
Number of releases		X	X
Number of closed branches		X	
Number of releases after closing a branch			X
Commits to develop ratio			X
Number of branches created after a user story is moved to 'in progress'			X
Number of closed branches after closing an issue			X

This alignment in metrics suggests that certain key performance indicators are universally recognized as essential for effective project management and software development. For example, tracking the number of new issues, updates, and closures in task management systems, as well as monitoring code activities such as commits, branch creation, and merge requests, reflects a shared understanding of what drives productivity and progress in technical teams.

The bridge of metrics across the companies enables the creation of a unifying TPA that incorporates the shared best practices of the three organizations, establishing a framework to follow for effective collaboration and development.

4 Applicability Discussions

After monitoring three software development companies and analyzing data on TPA adherence, valuable insights were obtained. The results will be examined from both quantitative and qualitative viewpoints. In addition, the challenges and limitations faced during the implementation and data collection processes will be addressed, as well as the definition and validation of a unifying TPA. This unifying TPA aims to integrate the best practices from the three companies, providing a standardized framework that can be applied across diverse methodologies, fostering greater alignment in multi-organizational collaborations.

4.1 Study Design and Evaluation Methodology

This study follows a case study research methodology, with the aim of evaluating the applicability and impact of Bluejay in a real-world setting. Data were collected from three companies over a seven-month period, with monitoring conducted between 30 June 2022 and 19 February 2023, using the Bluejay framework. The evaluation focused on two key aspects: (1) the level of adherence to the defined team practices within each organization's customized team practice agreement and (2) the feasibility and perceived value of establishing a unifying TPA to support future collaboration between organizations.

The causal link explored in this study is between the use of monitoring and visualization tools (via Bluejay) and improved alignment and coordination between organizations. The case study methodology was selected due to the exploratory nature of the research and the need to understand how Bluejay performs in a real-world industrial context involving heterogeneous environments.

The unit of analysis was the development activity tracked in the companies project management and code repositories. Data were collected using the Bluejay framework and complemented with interviews and survey responses from key stakeholders. Quantitative metrics were extracted directly from the tools, while qualitative data helped contextualize the observed behavior.

4.2 Quantitative Analysis

The data collected from each company varied significantly in terms of volume and completeness. Below is an overview of the data collected for each company and which TPs had measurable results.

Company A: data was collected for all six TPs, yielding 1,149 data points. However, compliance rates were very low across the board. TP1 (commitments per week) and TP3 (merge requests) had 0% compliance, indicating that no weeks met the required thresholds. TP2 (new branches) and TP4 (updated branches) showed sporadic compliance, with adherence rates below 20%. For Jira-based practices, TP5 (new tasks) and TP6 (updated tasks) achieved compliance only in 15% and 18% of the monitored weeks, respectively. Although complete data was collected for all TPs, overall adherence remained below 20%, highlighting significant challenges in adopting defined practices.

Company B: data was available for only 6 of the 15 TPs, totaling 7,516 data points. Among these, TP1 (commits) showed the highest compliance at 43.8%, reflecting consistent engagement. TP3 (updated branches) and TP6 (merge requests) had compliance rates of 25.3% and 34.2%, respectively, indicating moderate adoption of these practices. TP5 (new branches) showed adherence of 37%, slightly better but still underperforming. For Redmine-based TPs, TP8 (new tasks) and TP9 (updated tasks) reached compliance rates of 31.5% and 35%, respectively. However, no data was collected for the remaining 9 TPs.

Company C: only TP7 (correlation between new branches and tasks moved to "In progress") had data, with a compliance rate of 40.5%. The remaining TPs, including TP1 through TP6 and TP8 through TP9, did not have data recorded.

4.3 Qualitative Analysis

From a qualitative point of view, the data indicate that the teams are still in the early stages of adapting to the TPA, with the full integration of these practices into their workflows remaining a work in progress. Company A, despite having data for all defined practices, shows only sporadic engagement with the tools, with several practices showing zero compliance. Company B's better adherence can be attributed to more consistent engagement with some practices, although the lack of data for several practices suggests room for improvement. Company C, on the other hand, has insufficient data to assess adherence, indicating that they have not yet correctly adopted the proposed team practices in their methodology.

4.4 Challenges and Limitations

One of the main challenges encountered during the monitoring was the adherence of the teams to the proposed methodologies. For the data to be meaningful, it was essential that the teams follow the defined practices. In many cases, non-compliance led to incomplete data, making it difficult to draw valuable conclusions from the quantitative analysis. Without reliable information, it was not feasible to assess the effectiveness of the methodologies or make recommendations for improvement. This variability in adherence became a key limitation of the study, affecting data quality and hindering the evaluation process. To ensure future success, it is essential that the teams are properly aligned with the proposed practices.

4.5 Unifying TPA

After analyzing the TPAs designed for each organization and the metrics discussed in the Commonality section, the goal is to establish a unifying TPA. This unification combines key metrics on team management and tool effectiveness from companies, with the aim of creating an analytical framework to evaluate and optimize development practices in future implementations. The unifying TPA integrates four key areas, based on the TPs established.

Team practices on the frequency of commits, branches, and merge requests in code management tools (see Table 6) ensure continuous contributions to the shared code repository, promoting constant integration and minimizing the risks of development.

Table 6. Team practices for frequency of commits, branches and merge requests.

TP	Scope	Period	Metric	Condition
TP1	Team	Weekly	Number of commits	≥ 10
TP2	Team	Weekly	Number of new branches	≥ 1
TP3	Team	Weekly	Number of new merge requests	≥ 1
TP4	Team	Weekly	Number of updated branches	≥ 2

Table 7 describes the team practices in task management tools, which promote the consistent creation, updating, and completion of tasks, ensuring steady progress, visibility, and accountability throughout the project. These practices are closely related to how teams manage their coding process. Thus, in Table 8 we describe team practices in code management tools that focus on evaluating the team's ability to manage tasks across different stages of the workflow, ensuring tasks move through defined stages efficiently and progress is maintained throughout the project codebase.

Table 7. Team practices for constant flow of tasks.

TP	Scope	Window	Metric	Condition
TP5	Team	Weekly	Number of new tasks created	≥ 1
TP6	Team	Weekly	Number of tasks updated	≥ 1
TP7	Member	Weekly	Number of tasks completed	≥ 1

Table 8. Team practices for Structured task life cycle.

TP	Scope	Window	Metric	Condition
TP8	Team	Hourly	Number of new branches created for "in progress" tasks	≥ 1
TP9	Team	Hourly	Number of merge requests created for "in review" tasks	≥ 1
TP10	Team	Hourly	Number of merge requests integrated for "done" tasks	≥ 1

Finally, Table 9 represents team practices in task focus and completion, assessing the team's ability to concentrate on one task at a time and finish it efficiently. These practices promote task prioritization and help reduce inefficiencies that arise from multitasking.

The unifying TPA bridges the best practices of code and task management tools, creating a comprehensive evaluation framework for agile teams. This framework will serve as the foundation for the next Bluejay use case that

Table 9. Team practices for focus and completion of the task.

TP	Scope	Window	Metric	Condition
TP11	Member	Hourly	Number of tasks focused on	≤ 1

Table 10. Unifying TPA Validation.

Area/TP	Grade	Condition
Frequency of commits, branches, and merge requests	4	-
TP1	3.5	Realistic
TP2	4	Realistic
TP3	3.75	Realistic
TP4	4.25	Realistic
Constant flow of tasks	3.5	-
TP5	3.25	Realistic
TP6	3.75	Realistic
TP7	3.75	Must be below
Structured task life cycle	3.5	-
TP8	3	Realistic
TP9	2.75	Realistic
TP10	3	Realistic
Focus and completion of the task	3.5	-
TP11	2.75	Must be above
Importance of best practices	4.25	-

encompases the second phase of Alpha Project, providing a strong approach to measure and optimize team productivity over the next period of the project.

4.6 Unifying TPA Validation

Before implementing the unifying TPA in the next Bluejay case study mentioned before, we assessed the companies' satisfaction with the proposed areas and Team Practices through a survey. The participants rated the importance of applying best practices and implementing each TP (1–5). They were also asked if they considered the objectives realistic, below, or above the realistic value. Lastly, we inquired about the importance of best practices for the successful completion of a software project (1–5). On a scale from 1 to 5, 1 is considered irrelevant and 5 is very important. Representatives of each of the three companies participated in the study, representing the roles of project manager, analyst, technical director, and developer. The data collected are presented in the following Table 10.

The validation process confirms that the unifying TPA is realistic and aligns with industry needs, establishing a solid standard for best practices. However, slight adaptations are necessary to address specific organizational needs and ensure consistent adoption of the methodology used.

5 Related Work

To the best of our knowledge, no tools have been proposed in the literature to monitor best practices in organizational or multi-organizational contexts in a flexible way. In [6], we introduced a prototype system for auditing team practices, but it only supports isolated metrics and lacks cross-tool correlation (e.g., B.TP14). However, Vallon et al. [13] highlight the importance of detailed empirical applications to generalize how certain practices improve productivity in agile methodologies. Most studies on best practices in industrial contexts lack monitoring tools and rely on manual evaluation. For example, Connelly et al. [2], Treude et al. [12], and Meyer et al. [9], successfully implemented practices in both academic and industry contexts. In these studies, metrics such as version control, issue tracking, branching strategies, and pull request strategies, among others, were applied effectively, leading to improved productivity outcomes. Bluejay enables automated monitoring of these practices, adapting to specific contexts.

In a multi-organizational context, the distances between organizations and developers are critical factors. Bjarnason et al. in [1] define distance as *"a difference in position or level between entities that requires effort to traverse in order to accomplish a software development task."*. This includes any barriers that can hinder effective progress between two entities. They propose best practices, such as cross-artifact reviews, focused on the eight types of distance they identify like Geographical, Organizational, or Psychological, that could be implemented in Bluejay, establishing a framework to break down multi-organizational barriers.

There are monitoring alternatives for specific tools. Siddiqui et al. [11] use dashboards to monitor Jenkins, focusing on real-time metrics to track system performance and behavior. However, Bluejay offers a broader and more extensible framework that allows for monitoring multiple tools in various contexts, making it a flexible solution.

One of the most important challenges highlighted by studies such as those of Zarour et al. [14] and Huijgens et al. [8] is demonstrating the effectiveness of applying a set of best practices in various contexts. This study shows how Bluejay facilitates the application of different practices in diverse contexts, allowing the proposed TPA to be quantitatively validated.

6 Conclusions

This study emphasizes the importance of integrating methodologies in collaborative multi-organizational environments through explicit Team Practice Agreements (TPAs), especially when engineering complex Information Systems (IS). The results demonstrate that adopting a unified framework of metrics and practices, as represented by the TPA, has the potential to enhance alignment amongst teams with diverse methodologies and tools.

The experience with Bluejay has shown that while each organization has particular strengths in specific areas, it is possible to identify common metrics that act as bridges for collaboration. In addition, the low compliance rates observed in certain practices highlight the need to strengthen training and awareness about the importance of adhering to consistent standards.

Finally, the development of the unifying TPA presented in this work marks a step toward standardizing best practices in IS engineering of complex multiorganizational scenarios. This approach establishes a foundation for future implementations and studies, providing an adaptable framework that promotes the integration of tools, methodologies, and teams.

In future work, our goal is to validate the unifying TPA in the next stages of the Alpha project and other scenarios. In addition, further exploration of techniques to improve practice adoption and continuous monitoring is planned, consolidating Bluejay as a key tool to improve collaboration in multiorganizational projects.

Acknowledgments. This publication is part of the R&D projects PID2021-126227NB-C21, PID2021-126227NB-C22, and PDC2022-133521-I00 funded by MICIU/AEI/10.13039/501100011033/ERDF/EU and of project TED2021-1310 23B-C21 funded by MICIU/AEI/10.13039/501100011033/European Union Next GenerationEU/PRTR.

References

1. Bjarnason, E., Smolander, K., Engström, E., Runeson, P.: A theory of distances in software engineering. Inf. Softw. Technol. **70**, 204–219 (2016). https://doi.org/10.1016/j.infsof.2015.05.004
2. Connelly, L.T., Hammel, M.L., Lin, L.: Leveraging best industry practices to developing software for academic research. In: 2023 7th International Conference on Management Engineering, Software Engineering and Service Sciences (ICMSS), pp. 7–13 (2023). https://doi.org/10.1109/ICMSS56787.2023.10117776
3. Fresno-Aranda, R., Ojeda-Perez, J., Fernandez, P., Ruiz-Cortés, A.: Governify. An agreement-based service governance framework. Softw. Impacts **19**, 100629 (2024). https://doi.org/10.1016/J.SIMPA.2024.100629
4. García, C., et al.: Bluejay: a cross-tooling audit framework for agile software teams. In: 43rd IEEE/ACM International Conference on Software Engineering: Software Engineering Education and Training, ICSE (SEET) 2021, Madrid, Spain, May 25-28, 2021, pp. 283–288. IEEE (2021). https://doi.org/10.1109/ICSE-SEET52601.2021.00038
5. Garousi, V., Petersen, K., Ozkan, B.: Challenges and best practices in industry-academia collaborations in software engineering: a systematic literature review. Inf. Softw. Technol. **79**, 106–127 (2016). https://doi.org/10.1016/J.INFSOF.2016.07.006
6. Guerrero, A., et al.: Eagle: a team practices audit framework for agile software development. In: Proceedings of the 2019 27th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering, p. 11391143. ESEC/FSE 2019, Association for Computing Machinery, New York, NY, USA (2019). https://doi.org/10.1145/3338906.3341181
7. Hernández-Ledesma, G., Ramos, E.G., y Fernández, C.A.F., Aguilar-Cisneros, J.R., Rosas-Sumano, J.J., Morales-Ignacio, L.A.: Selection of best software engineering practices: a multi-criteria decision making approach. Res. Comput. Sci. **136**, 47–60 (2017)
8. Huijgens, H., Lamping, R., Stevens, D., Rothengatter, H., Gousios, G., Romano, D.: Strong agile metrics: mining log data to determine predictive power of software metrics for continuous delivery teams. In: Proceedings of the 2017 11th Joint Meeting on Foundations of Software Engineering, p. 866871. ESEC/FSE 2017, Association for Computing Machinery, New York, NY, USA (2017). https://doi.org/10.1145/3106237.3117779
9. Meyer, A.N., Fritz, T., Murphy, G.C., Zimmermann, T.: Software developers' perceptions of productivity. In: Proceedings of the 22nd ACM SIGSOFT International Symposium on Foundations of Software Engineering, p. 1929. FSE 2014, Association for Computing Machinery, New York, NY, USA (2014). https://doi.org/10.1145/2635868.2635892
10. Müller, C., Gutierrez, A.M., Fernandez, P., Martín-DíÂŋaz, O., Resinas, M., Ruiz-Cortés, A.: Automated validation of compensable SLAs. IEEE Trans. Serv. Comput. **14**(5), 1306–1319 (2021). https://doi.org/10.1109/TSC.2018.2885766
11. Siddiqui, I., Pandey, A., Jain, S., Kothadia, H., Agrawal, R., Chankhore, N.: Comprehensive monitoring and observability with Jenkins and Grafana: a review of integration strategies, best practices, and emerging trends. In: 2023 7th International Symposium on Multidisciplinary Studies and Innovative Technologies (ISMSIT), pp. 1–5 (2023). https://doi.org/10.1109/ISMSIT58785.2023.10304904

12. Treude, C., Figueira Filho, F., Kulesza, U.: Summarizing and measuring development activity. In: Proceedings of the 2015 10th Joint Meeting on Foundations of Software Engineering, p. 625636. ESEC/FSE 2015, Association for Computing Machinery (2015). https://doi.org/10.1145/2786805.2786827
13. Vallon, R., da Silva Estácio, B.J., Prikladnicki, R., Grechenig, T.: Systematic literature review on agile practices in global software development. Inf. Softw. Technol. **96**, 161–180 (2018). https://doi.org/10.1016/j.infsof.2017.12.004
14. Zarour, M., Abran, A., Desharnais, J.M., Alarifi, A.: An investigation into the best practices for the successful design and implementation of lightweight software process assessment methods: a systematic literature review. J. Syst. Softw. **101**, 180–192 (2015). https://doi.org/10.1016/j.jss.2014.11.041

Declarative Domain Testing: An Approach for Automatic and Integrated Test Data Generation

José Francisco Crespo, Martí Juanola, Xavier Oriol(✉), and Ernest Teniente

Universitat Politècnica de Catalunya, Barcelona, Spain
{jose.francisco.crespo-sanjusto,marti.juanola,xavier.oriol, ernest.teniente}@upc.edu

Abstract. Software testing is crucial to any information system to ensure code correctness and validity. Currently, test design and implementation are essentially manual, which makes them error-prone and time-consuming. Moreover, generating a valid system state required to initiate testing can be challenging when this state must satisfy a set of business rules. In this paper, we propose a new approach based on specifying the tests declaratively, and using an automatic generation of business-rule-compliant initial test states from this specification. In our approach, users declaratively specify the required system state, together with the domain business rules it must satisfy, and we delegate to an automated reasoner the creation of the valid initial state. We also show how the approach can be effectively implemented in Java, and we share experimental results showcasing its feasibility in a realistic project.

Keywords: declarative domain testing · test data generation · model-driven testing

1 Introduction

Testing is a crucial activity in software development, as it ensures the correct behaviour of the current system and, over time, prevents new versions and modifications of the system from introducing unexpected errors. Some engineers put so much emphasis on testing activities that even some current trending software development practices are based entirely on them, such as TDD (Test-Driven Development) [3], and BDD (Behavioural-Driven Development) [19].

To reduce costs, the industry has created several technologies to automate its testing activities. For example, all major object-oriented languages have libraries for defining and launching unit tests (JUnit, xUnit, gTests, etc.), libraries for asserting test conditions (JAssert, Hamcrest, FluentAssertions, etc.), not to mention the pipelines that can be created to automatically launch such tests when integrating and deploying code, which is a crucial part of DevOps practices [12].

However, today the design and preparation of such tests is still too manual and error-prone. In fact, when a software developer writes a test on some functionality, he/she has to manually create some data, known as the *test arrange*

J. Krogstie et al. (Eds.): CAiSE 2025, LNCS 15701, pp. 258–273, 2025.
https://doi.org/10.1007/978-3-031-94569-4_15

[10], which represents 1) a state of the system and 2) some parameters necessary to operate that functionality. We argue that creating such data is especially difficult when dealing with business rules, i.e. when the data must satisfy certain conditions to be considered valid states to run the test.

Consider, for instance, that we are implementing a book recommendation system as described in the UML diagram from Fig. 1. In this system, we store books, some of them bestsellers, their authors, and the people who have read them. Also, when a person has read a book, he/she can recommend it to another person. As for business rules, we have primary keys (like books are identified by title), we do not allow recommending a book to its author or to oneself, and we do not store that an author has read his own books (i.e. we only store the books that each person reads by other authors). In addition, the years must be consistent. That is, no one can read a book before it was released, nor before he/she was born, and no author can release a book before he/she was born.

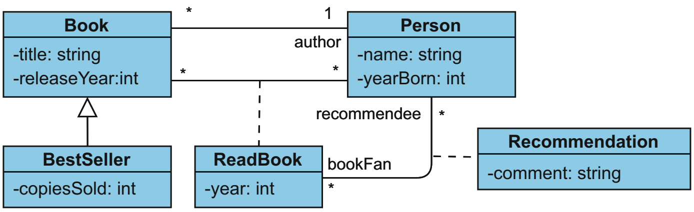


Fig. 1. Book recommendation system

Now, assume we want to test the functionality of removing a recommendation. To do so, a software engineer will need to create a valid system state containing one recommendation. Therefore, he/she will create a Book b_1 and a Person p_1 to be its author. Then, he/she will create another Person p_2 that has read the Book. At this point, he/she must realize that p_2 cannot be p_1 since we only store the books read by other authors. Also, he/she must be careful to create consistent years for the births and releases of the book, the author, and the reader. Furthermore, he/she will need to create a new Person p_3 to receive the recommendation, which again, cannot be either p_1 or p_2 due to business rules, and must have a consistent birth year.

The above example illustrates the difficulty of creating test data by hand. Note also that we are asking the engineers to do this for every test, when a real project may have thousands of them[1]. It is therefore not surprising to see some authors claiming that testing can take up most of a project's resources [2,15]. This problem can get worse if we take into account that the schema and business rules can change over time. For instance, adding to our example the date of the recommendation and stating that each person should have read at least one book

[1] The Spring framework, for instance, has nearly 30.000 tests (https://github.com/spring-projects/spring-framework).

would force us to reconsider all pre-existing data manually generated. In fact, all these difficulties arise from the NP-hardness of finding some data satisfying some constraints (e.g. satisfying a CSP problem). Furthermore, the problem becomes undecidable when the business rules are first-order logic constraints [7].

To address this situation, the literature has suggested using automated reasoners (e.g., SMT solvers) to facilitate test data creation [18,20]. However, these proposals involve developers working with abstractions of the system (e.g. UML diagrams), rather than with the system itself, which makes test data generation dependent on the outside of the test development pipeline (i.e. data is created externally, using external reasoners, and its results must be manually translated into the test system). In contrast, we argue that test data generation should be fully integrated into the test code, so that changes in business rules can be automatically updated across all data generated from all tests.

In essence, we propose an approach for automatic and integrated test data generation, based on the use of textual specification languages and reasoners to facilitate the creation and maintenance of test data according to (possibly evolving) business rules. We call such an approach *Declarative Domain Testing.*

In *Declarative Domain Testing* (DDT), users specify *Declarative Tests*, where a Declarative Test is a runnable test (e.g. a JUnit test) where the data of the test is declaratively specified through some specification language. When running the test, such specification is provided to an automated reasoner that creates the requested data as domain objects of the actual code (e.g., objects of classes *Books, BestSeller, Person*), and such objects are injected back into the test.

Hence, our approach enables the generation of business rule-compliant data, provided as a set of ready-to-use code domain objects for testing. Thus, it saves time and error-prone pitfalls not only based on the inherent difficulty of satisfying the rules, but also on the difficulty of finding the order in which to instantiate the data (e.g., one should create *ReadBooks* before *Recommendations*, and *Person* and *Book* before the *ReadBook*). Our approach is agnostic of the particular specification language and reasoning engine, hence, it might exploit different automated reasoning techniques from the literature [4,6,11].

In summary, the contributions of this paper are as follows:

- We propose *Declarative Domain Testing*, an approach for automated and integrated test data generation to reduce errors and save developers' time.
- We implement a tool in Java, *DArrange4J*, to facilitate the adoption of our approach. *DArrange4J* allows specifying tests in Datalog, and uses a Datalog reasoning engine to create valid data that is automatically integrated in a runnable JUnit test.
- We perform some experiments, using *DArrange4J* and a redesigned case study, showing that the amount of time required for our approach to generate test data in realistic scenarios takes only a few seconds, hence demonstrating the feasibility and practicality of our approach.

The paper is organized as follows: Sect. 2 introduces the approach in abstract terms (i.e., without coupling to any particular language). Section 3 discusses the implementation of *DArrange4J*. Section 4 provides some experiments performed with our implementation while Sect. 5 summarizes related work. Finally, Sect. 6 presents some conclusions and points out future work.

2 Declarative Domain Testing

Our proposal is based on the notion of *declarative tests*. A declarative test is a test where the software engineer specifies, declaratively, the initial data state and the parameters he/she wants to use to test its desired functionality. That is, the software engineer abstracts from the particular details of the system state for running the test (such as book titles or people's birth years, in our running example), to focus on the real *conceptual* state he/she wants to test (such as a valid recommendation with coherent years between author/reader/recomendee). In this sense, the software engineer only needs to specify the system state he/she wants and the business rules it must satisfy, and let an automated reasoner handle the low-level details.

Declarative testing is not bound to any particular specification language, nor a particular reasoner for creating the test data. In this sense, we can instantiate a declarative test with any of the already available specification languages (e.g., OCL, Datalog), and make them work with any automated reasoner (e.g., CQC [17], UMLToCSP [4], Alloy [6], or UseKodKod [11]). The unique requirement is that the reasoner should be able to deal with the expressivity of the business-rules allowed by the specification language. That is, if the reasoner is limited to first-order logic constraints, the specification language should also be limited to such language.

In the following, we first define what a declarative test is, in terms of a metamodel, and then, we show how to integrate them into the software testing.

2.1 Declarative Test

In Fig. 2, we show the metamodel of a declarative test. In essence, a declarative test contains a *declarative input* and is defined over a *domain context*, where such *domain context* is composed of a *domain model* and a set of *business rules*.

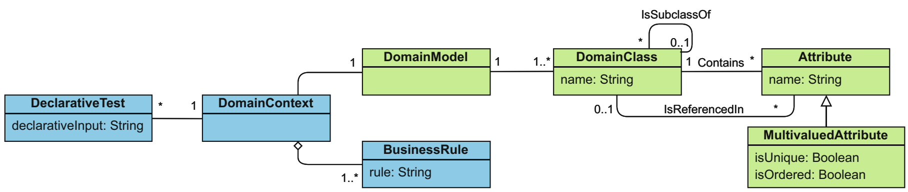


Fig. 2. Declarative test metamodel

- *Declarative input.* The declarative input is a statement, written in a specification language, expressing the objects of the domain that we need in order to run the test. More formally, it is a query, whose return represents the objects needed to run the test and whose body states the conditions satisfied by these objects.
- *Business Rule.* The business rules are a set of statements, written in the specification language, expressing the conditions that a data state must satisfy to be a valid instance of the system. The difference between the business rules and the declarative input is that, whereas the declarative input is specific for this test, the business rules are shared among all tests since they represent general conditions of the domain. Roughly speaking, textual constraints from UML class diagrams would be business rules.
- *Domain model.* The domain model is the set of classes, relationships, and hierarchies that comprise the domain code of the system under test. Such domain is essential in several terms. First, it determines the vocabulary to use in the business rules and declarative input (e.g., the business rules of our running example should speak about *Books*, *Person*, *Recommendations*, etc.). Second, it represents the classes and references that must be instantiated as a result of the reasoning (e.g., the reasoner should end up creating objects of the class *Book*, with a reference *author* to the class *Person*). Furthermore, the way the domain is structured also specifies some rules, which we call code-structure rules, that the reasoner should take into account. E.g., since *BestSellerBook* is implemented as a subclass of *Book*, we have that every *BestSellerBook* instance is also an instance of *Book*, hence, since every *Book* requires an author, a *BestSellerBook* requires an author too.

In the following, and due to its importance, we continue by detailing how we obtain such domain model, and then, the code-structure rules.

The Domain Model. In declarative domain testing, we propose that the domain model should be automatically obtained by analysing the code. Traditionally, the automatic approaches to generate data for a domain encompass the domain to be defined in UML [1,20] but, unfortunately, UML diagrams are usually not available in the project repository [8]. So, we advocate going the other way around, by generating the domain model from inspecting the code. This also helps to avoid maintenance and synchronization problems between the UML specification and the actual code.

We rely on a metamodel to represent the domain model. This metamodel, shown in green in Fig. 2, contains classes, attributes, their relationships, and their hierarchies. Note that, since the metamodel captures the domain as present in the user's code, there is no need to capture *n-ary* associations, nor other UML conceptual components. This simplifies the approach, facilitating its implementation and adoption. Furthermore, this metamodel is only coupled to the typical features of (strongly typed) object-oriented programming, which enables our approach to be adopted by most object-oriented programming languages (such as Java, C++, or C#).

Obtaining the Code-Structure Rules. From the domain model, we can derive what we refer to as the code-structure rules (CSR). These rules are implicitly encoded in the structure of the domain code (such as a hierarchy) and need to be provided to the reasoner so that the reasoner generates data that conforms to the rules of the object-orientation paradigm. Intuitively, these rules correspond to the UML implicit rules. For ease of understanding, we make them explicit in the metamodel of Fig. 3.

In essence, the code-structure rules implied by the domain model are: 1) hierarchy constraints (such as a *BestSeller* object is also a *Book* object), 2) abstract class constraints (i.e., any instance of an abstract class must be instance of some concrete subclass), 3) integrity reference constraints (e.g., the *author* attribute from *Book* references a *Person* object), 4) unique object identifier rules (e.g., a *Book* object cannot have the same OID as another object of the domain). Figure 3 contains a metamodel showing the information that we require to capture these rules, where green classes are taken from the metamodel of Fig 2.

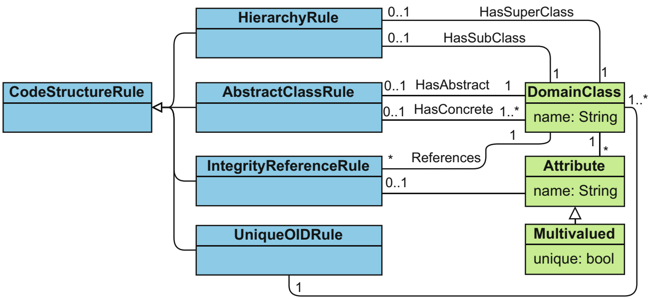


Fig. 3. Code-structure rule metamodel

2.2 Integrating Declarative Tests

We now show how to integrate the declarative tests into a software development environment. To do so, we assume that tests can receive objects as parameters, similar to the approach used in well-established testing libraries like JUnit.

We show in Fig. 4 the whole picture of how we propose to integrate declarative tests into software development. We begin by explaining the steps a user would follow to define the domain context, and then, the steps required to define the declarative tests. Note that the former are applied only once, while the latter are applied for each declarative test. For each of these steps, we also explain which computations are done automatically by our approach.

Defining the Domain Context. First, a software engineer should define the domain context of the information system. That is, he/she must state which part of the code represents the domain and he/she must define the business rules for

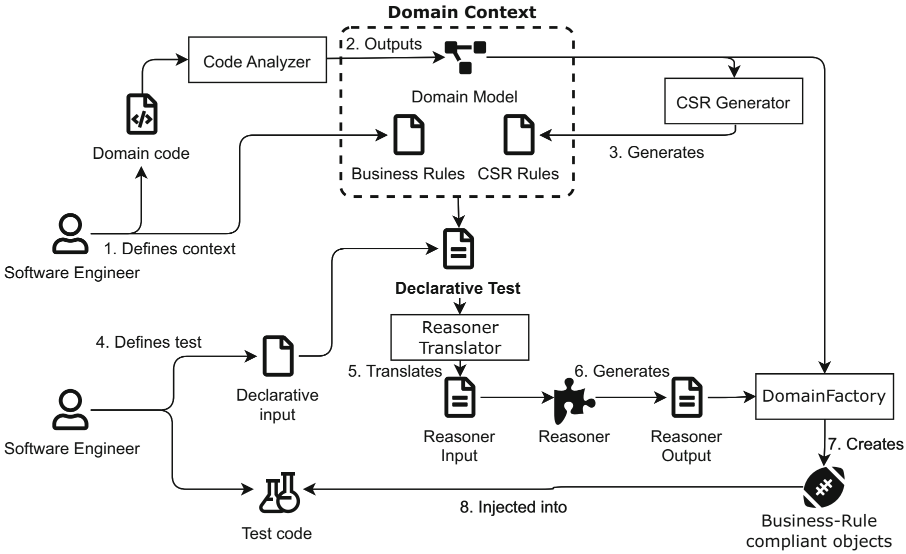


Fig. 4. Integration of Declarative Tests in a software project

this domain. This is a manual step since it encompasses defining the knowledge from the real world. From now on, the rest of steps are automatic.

Then, a code analyser should inspect the domain code of the project to obtain its domain model. Furthermore, such domain model is then analysed to obtain its corresponding code-structure rules.

This context can then be shared among all the declarative tests of the system.

Defining the Declarative Tests. When a software engineer wants to define a declarative test, he/she must first specify the *declarative input*, that is, the objects he/she needs for the test, together with its *domain context* (i.e., its business rules and domain model to use). Again, this step is manual since it encompasses describing the real-world conditions that the engineer requires for the test. From now on, the rest of steps are automatic.

Then, the declarative input, the business-rules, and the code-structure rules are translated into the language of the reasoner. Note that this translator allows keeping a low coupling between the specification language and the reasoner selected. Therefore, changing the specification language implies only modifying this translator.

Afterwards, the reasoner generates the data that satisfies all the given rules. Since the output is given in the language of the reasoner, we must translate it into objects of our programming language. This is precisely the mission of the DomainFactory, which accesses the domain model of the context. In particular, the DomainFactory accesses the domain model to know which classes reference which other classes (e.g., if the reasoner determines that a *Book b* should be authored by a *Person p*, the DomainFactory, apart from creating the *Book b* and the *Person p*, it must set *p* as the attribute value of *b.author*), and which classes are inheriting from which other classes (e.g., if the reasoner determines that we must have a *Book b* which should be also a *BestSellerBook*, the DomainFactory must know that *BestSellerBook* is a subclass of *Book*, so that the DomainFactory creates *b* in its proper subclass).

Finally, we must inject the generated objects into the test code, which we do via object parameters. Again, this step is fully automatic and transparent to the software engineer.

3 Implementation

To show the feasibility of our approach, we have implemented *DArrange4J*, a Java library that permits specifying declarative tests in Datalog, automatically generating object-oriented data satisfying the specification, and integrating such data into runnable JUnit tests, one of the most well-known Java testing libraries [13].

In the following, we discuss 1) How a developer can use the *DArrange4J* to write declarative tests, 2) the Datalog specification language used to specify the tests, 3) the reasoning engine behind *DArrange4J*, and 3) The JUnit integration we offer to the user.

3.1 *DArrange4J* usage

Recall our Introduction's example, in which a developer wants to test the functionality of removing a recommendation from a book recommendation system.

In such a case, the developer can write a Declarative Test like the one illustrated in Fig. 5:

```
@Domain(path = "edu.upc.recommendations.domain")
@DeclarativeInput( "(r,rb) :- Recommendation(r, rb, p, c), ReadBook(rb, p2,
     bo, y, rs)")
@BusinessRules(file = "recommendations.impl")
@DeclarativeTest
void shouldRemoveRecommendation(Recommendation r,ReadBook b){
    b.removeRecommendation(r);
    //Assertion code...
}
```

Fig. 5. Java declarative test example using *DArrange4J*

First, the developer specifies the domain under test, i.e., the Java package containing the classes to test, with the *@Domain* annotation. Then, he/she specifies the input needed for the test with a *DeclarativeInput*. Such DeclarativeInput is written as a Datalog query. In this example, we are asking for a Recommendation *r* containing a ReadBook *rb*. Then, he/she specifies the file containing the business rules that the domain objects, such as r and rb, should satisfy. In Fig. 6 we show some of these rules, which are written in the form of denials, that is, conditions that should never happen to the data.

This specification can be executed like a regular JUnit test. In particular, when running it, our *DArrange4J* automatically parses and transforms this specification to bring it to a reasoning engine. Then, the engine creates the data that satisfies all the business rules. Afterwards, *DArrange4J* automatically translates the reasoning output into Java objects. In our case, we obtain a Recommendation object, and a ReadBook object, containing all the additional data required (book, author, reader, recomendee, etc.) with coherent attribute values (i.e., no repeated names, and coherent years). Both objects are injected, as parameters, to the test, where they are used in the way specified by the user (typically, the tested function will be executed and the expected behaviour will be checked).

```
@OnlyReadOtherAuthors :- ReadBook(rb,p,b,year,s), Book(b,title,releaseYear,p)
@CannotRecommendToAuthor :- ReadBook(rb,p,b,year,s),
     Book(b,title,releaseYear,author), Recommendation(r,rb,author,comment)
@CannotReadBeforeRelease :- Book(b,title,releaseYear,author),
     ReadBook(rb,p,b,readYear,s), readYear <= releaseYear
...
```

Fig. 6. Business rules fragment

3.2 Datalog Specification

In *DArrange4J*, we rely on Datalog to specify the required business rules. To maximize the expressivity of the language, we have extended it to deal with negation and built-in literals (aka arithmetic comparisons), hence covering any full first-order logic constraint.

To write these business rules, the user must take into account that each Class is encoded with a Datalog predicate containing as many terms as attributes contains the class, plus one term to encode the Object IDentifier (OID). Hence, the class *Book* in our example is encoded as *Book(oid, title, releaseYear, authorOID)* (assuming that the author relationship is encoded by a reference from *Book* to *Person*). Then, the developer can write business rules using denials, i.e., by specifying what should not happen. For instance, we can state that no book can be earlier than the year 1440:

```
:- Book(oid, title, releaseYear, authorOID), releaseYear < 1440
```

The OID term is used to *navigate* across the objects. So, a constraint stating that no book can be released before its author was born can be stated as follows:

```
:- Book(oid, title, releaseYear, authorOID),
   Person(authorOID, name, yearBorn), releaseYear < yearBorn
```

Note the repetition of the term *authorOID* in *Book* and *Person*.

If some of the attributes refer to a collection type (Set, or List), we can access its elements by means of the predicate *CollectionElement(oid, position, element)*. For instance, assuming that we implement the *Authorship* relation through a set of references from *Person* to *Book* (instead of from *Book* to *Person*), our encoding would be:

```
:- Person(oid, name, yearBorn, setOfBookOIDs),
   CollectionElement(setOfBookOIDs, position, bookOID),
   Book(bookOID, title, releaseYear), releaseYear < yearBorn
```

Note that, for simplification purposes, we encode all collections as if they had positions, although this is not true for the case of sets. We do it this way since then we have a unique way of accessing collections, which is generic enough to deal with most typical collection types. In particular, using this notation we can encode and access the *Set*, *Bag*, *OrderedSet*, and *Sequence* collections, which are the collection types defined in UML. It is worth saying that, currently, the positions can only hold variables (i.e., in the current version, we cannot deal with *CollectionElement(setOfBookOIDs, 2, bookOID)*) so, we can only deal with relative positions (e.g., *this book is previous to this book in the list*).

We have also extended Datalog to include *unnamed variables*. Then, we can specify denials by forgetting about unused terms (such as positions for sets):

```
:- Person(_, _, yearBorn, setOfBookOIDs),
   CollectionElement(setOfBookOIDs, _, bookOID),
   Book(bookOID, _, releaseYear), releaseYear < yearBorn
```

Note that this way of writing denials allows the software engineer to focus on the data conditions required for the test and forget about other unnecessary details, hence saving time, reducing errors and increasing the readability of the test. Further details on the implementation of the specification language (parser, and metamodel) can be found in [5].

3.3 Reasoning Engine

We have decided to directly use a Datalog reasoning engine to simplify the step of translating the specification language to the reasoning language. In fact, during the translation step, apart from bringing different names to all the unnamed variables, the only transformation we apply is to inject extra literals to have more information in the logics. For instance, given the declarative input from Fig. 2, we inject an extra literal *Output(r,rb)* in the body of the query to easily recall that we must return the objects r and rb.

For our purposes, we use a Java implementation of the CQC method [17], which is able to deal with Datalog extended with negation and built-in literals, which we also use in the specification.

In addition, since tests are executed multiple times during software development without changes to the business rules or code structure, we have introduced a cache for the generated test data. With this cache, if two tests use the same rules (e.g., executing the same test twice), the reasoner computes the data only once and reuses it for subsequent test executions. As demonstrated in our experiments, this significantly reduces the execution time of *DArrange4J* after an initial run.

3.4 JUnit Integration

Our implementation uses Java annotations to integrate the declarative tests into the JUnit framework, the most well-known Java testing framework. As seen in Fig. 2, we have implemented the annotations *Domain*, *BusinessRules*, *DeclarativeInput*, and *DeclarativeTest* to specify declarative tests that can be executed automatically using the JUnit runner.

Furthermore, to define the context of the declarative tests, *DArrange4J* offers two different ways of specifying which part of the code represents the domain. We can either specify it using annotations inside the code (i.e., including the annotation *@DomainClass* to each class that is part of the domain), or by configuring a *DomainModelProvider*. The first requires modifying the domain code, whereas the second consists in implementing a provider class, hence, avoiding any coupling between the domain code and our tool.

Once the user provides all this information for his/her particular domain, the method is fully automatic. That is, any test annotated with *@DeclarativeTest* can be executed as a JUnit test that automatically generates the requested data and injects it into the test using the test parameters. That is, under the hood, *DArrange4J* collects the business rules specified by the user, inspects the code that forms the domain context to automatically identify the implicit code-structure-rules, creates the reasoning engine input based on them, invokes the engine and translates back the result to Java objects through the domain factory.

As a result, software engineers can specify the objects they want for the test, and what these objects should satisfy, without worrying about how to construct them. We encourage the reader to check the implementation of the running example at (https://github.com/inLabFIB/declarative-domain-testing-poc) to see our *DArrange4J* in practice.

4 Experiments

We have conducted some experiments in a realistic scenario to demonstrate the practical viability of our approach. In particular, we aim to analyse the efficiency of our *DArrange4J* implementation.

Experiment Design. To perform our experiments we need: 1) the domain code of a software project, 2) some business rules over such domain, and 3) some operations to test.

For this reason, we have used a Java reimplementation of the DBLP system domain, the well-known system that records the publications of computer science researchers. This domain has the advantage of being well documented in a UML case study that includes, in addition to the classes that compose it, its business rules. Moreover, this case study has already been used in several experiments in the software engineering research community [16].

In particular, the DBLP domain contains 17 classes (e.g. Person, Publication, Conference, etc.), 75 business rules (11 primary keys, 31 min/max cardinalities, 6 disjoint/complete hierarchy constraints, 26 OCL constraints, and 1 exclusive association constraint). It also has 6 derived attributes and 4 derived associations with their corresponding derivation rules (e.g., the attribute *numPublications* of *Person* is the number of publications a Person has).

The derived information has been considered to be operations to be implemented in our system, and hence, the target of our experiments. That is, for each derived attribute A defined in a class C, we have created an operation *getA()* in C which computes its value. Similarly, each derived association has been considered an operation on one of its association ends. As a result, our case study had 10 different operations to test.

For each of these operations, we have defined a declarative test that essentially requires the creation of the object containing the derived information, along with several objects that make such derived information non-empty (e.g., to test *numPublications* for a given person, we require at least one publication for that person). Thus, we have defined 10 declarative tests where each declarative test corresponds to the OCL derivation rule written in the UML use case. In this way, we have obtained realistic non-trivial declarative inputs.

Table 1. Experiment results

Operation to test	**Input preparation**	**Reasoning**	**Object creation**	**Total time**	**Objects created**
BookChapter::getConferencePaper	63 ms	7,297 ms	< 1 ms	7,360 ms	5
BookSeriesIssue::getBookSection	312 ms	9,081ms	15 ms	9,408 ms	7
EditedBook::getBookSection	78 ms	3,828 ms	< 1 ms	3,906 ms	4
JournalPaper::getConferencePaper	62 ms	22,650 ms	< 1 ms	22,712 ms	9
JournalIssue::getJournalSection	63 ms	8,984ms	< 1 ms	9,047 ms	8
JournalIssue::getNumPages	63 ms	6,188 ms	< 1 ms	6,251 ms	7
Person::numPublications	62 ms	1,579 ms	< 1 ms	1,641 ms	4
Person::getPublications	63 ms	1,518 ms	< 1 ms	1,581 ms	4
Publication::getYear	62 ms	2,422 ms	< 1 ms	2,484 ms	5
Publication::getEdition	62 ms	1,860 ms	< 1 ms	1,922 ms	4
AVG	89 ms	6,540 ms	1.5 ms	6,631 ms	5.7

Experiment Results. For each of the 10 declarative tests, we have measured the time it takes to automatically execute the whole pipeline, but divided in several steps: 1) the time taken to prepare the input for the reasoner, 2) the execution time of the reasoner, 3) the time taken to translate the output of the reasoner into objects. In addition, we have also measured the final number of objects created. The results can be seen in Table 1. All experiments have been carried out on a Windows 10 operating system version 22H2 (x64). Intel(R) Core(TM) i7-12700 CPU at 2.10 GHz with 32.0 GB RAM. All the artifacts needed to run the experiments—including the DBLP implementation, the *DArrange4J* implementation, and the raw experimental data—are available in our repository: https://github.com/inLabFIB/declarative-domain-testing-poc..

To better understand the complexity of the data automatically generated by *DArrange4J*, we include in Fig. 7 the equivalent Java code needed to create a valid *BookSeriesIssue* with some *BookSection*, which represents an average case in this study (6 objects needed for its creation). Note the complexity of determining the objects we need, along with their relationships, for a single object. Also, note that the existence of mutual references between the objects makes it necessary to create the objects first (e.g., lines 1–6), before adding the remaining references (e.g., lines 7–10). All of this complexity is automatically managed by *DArrange4J*.

```
1  //Objects required by BookSeriesIssue to have a BookSeries
2  BookSeries bs = new BookSeries("0","0");
3  Person p = new Person("0", Set.of(), Set.of(), Set.of());
4  BookSection bSec = new BookSection("0",0,Set.of());
5  BookChapter bc = new BookChapter("0", Set.of(), List.of(p), 0, 0,
       Set.of(bSec), Set.of(), Set.of());
6  EditedBook eb = new EditedBook("0",Set.of(),Set.of(),Set.of(bc),List.of(p));
7  p.addAllAuthoredPublications(Set.of(bc));
8  p.addAllEditedBooks(Set.of(eb));
9  bSec.addAllBookChapters(Set.of(bc));
10 bc.addAllEditedBooks(Set.of(eb));
11
12 //Creation of BookSeriesIssue with a BookSeries
13 BookSeriesIssue bsi = new BookSeriesIssue(0, List.of(), "0", 0,
       "0",eb,0,Set.of(bc),bs,Set.of());
14 bc.addAllBookSeriesIssues(Set.of(bsi));
15 eb.addAllBooks(Set.of(bsi));
```

Fig. 7. Java code to create a book series issue with some book section.

Discussion. The execution time is dominated by the reasoning time (average of 6.5 s per test), as expected. We argue that this is substantially faster than the time an engineer would spend manually creating the data needed to satisfy the domain's business rules. That is, the time to manually identify the necessary objects (between 5 and 6, on average), to specify valid attribute values, and to write their creation in the correct order, as shown in Fig. 7.

Note also that the cache allows us to dramatically improve performance. Indeed, while on a first run each test takes about 7 s to execute *DArrange4J*, subsequent applications of the same test take about 0.1 s. Certainly, by using the cache we can skip the reasoning time, so we only need to prepare the input (89 ms) and create the objects (1.5 ms).

Thus, we see that: 1) in the long term (once the code structure rules and domain rules are fixed), *DArrange4J* only takes 0.1 s per test, 2) it allows us to create new data for new tests in seconds, beating the time to manually create the test, and 3) if the business rules change, it allows to update all data for all tests in minutes (sum of reasoning time for all tests), which is much less than the time it would take an engineer to manually inspect the data creation of each test, and manually apply the necessary changes to accommodate the changed rules, for each test.

5 Related Work

Our work is related to the field of model-based test case generation [14], i.e., automatically generating test data from a model representation of a system.

In [1], Ali et al. showed that we can use heuristic-search to find valid data instances satisfying OCL constraints to create test data. Such approach will be used in [21] to generate valid test data for use cases, and can be considered extended by Soltana et al., in the PLEDGE tool [20], which uses a combination of heuristic-search and SMT solvers to find UML data satisfying OCL constraints. In a similar fashion, Semeráth et al. [18] defined another approach combining partial model refinement with SMT solving to find valid data.

In summary, all these approaches always start from a UML/OCL specification and result in UML/OCL data. We argue that these approaches could exploit their full potential if, as we suggested in our approach, they get integrated into the project development (i.e., without requiring the programmer to write additional specifications, nor translate back the reasoner output to objects). Apart from that, the implementation of our approach, *DArrange4J*, guarantees completeness (if there exists some data satisfying the constraints, it is always found), which is not guaranteed in those approaches relying on heuristics.

It is also worth mentioning the work of Iqbal et al. [9] who start from an executable UML specification, and generate Java tests for it. However, this work does not seem to consider business rules as we do.

6 Conclusions

We have presented *Declarative Domain Testing*, an approach for automatic and integrated test data generation. We have defined the approach in abstract terms (what it is for, and how it can be implemented), and we have provided a Java implementation to show its feasibility. Furthermore, we have run some experiments to show that, in a realistic scenario, it takes a few seconds to generate valid test data, hence showing the practicality of our proposal.

As further work, we would like to increase the expressivity of the specification language (e.g., including arithmetic operations such as sum), add support for other data structures (e.g., dictionaries), and perform usability experiments w.r.t. the technology.

Acknowledgments. This work was partially funded by the SUDOQU project, PID2021-127181OB-I00 from MCIN/AEI, and the Grup de Recerca Consolidat IMP, 2021-SGR-01252. Also with the support of inLab FIB at UPC.

References

1. Ali, S., Iqbal, M.Z., Arcuri, A., Briand, L.C.: Generating test data from OCL constraints with search techniques. IEEE Trans. Softw. Eng. **39**(10), 1376–1402 (2013)
2. Anand, S., et al.: An orchestrated survey of methodologies for automated software test case generation. J. Syst. Softw. **86**(8), 1978–2001 (2013)
3. Beck, K.: Test Driven Development: By Example. Addison-Wesley Professional (2022)
4. Cabot, J., Clarisó, R., Riera, D.: On the verification of UML/OCL class diagrams using constraint programming. J. Syst. Softw. **93**, 1–23 (2014)
5. Crespo, J.F., Juanola, M., Oriol, X., Recalde, M., Teniente, E.: IMP-Logics: a metamodel for analysis and transformations of Datalog programs. In: Proceedings of the ACM/IEEE 27th International Conference on Model Driven Engineering Languages and Systems, pp. 51–55 (2024)
6. Cunha, A., Garis, A., Riesco, D.: Translating between alloy specifications and UML class diagrams annotated with OCL. Softw. Syst. Model. **14**, 5–25 (2015)
7. Fagin, R.: Finite-model theory-a personal perspective. Theoret. Comput. Sci. **116**(1), 3–31 (1993)
8. Hebig, R., Quang, T.H., Chaudron, M.R., Robles, G., Fernandez, M.A.: The quest for open source projects that use UML: mining GitHub. In: Proceedings of the ACM/IEEE 19th International Conference on Model Driven Engineering Languages and Systems, pp. 173–183 (2016)
9. Iqbal, J., Ashraf, A., Truscan, D., Porres, I.: Exhaustive simulation and test generation using fUML activity diagrams. In: Giorgini, P., Weber, B. (eds.) CAiSE 2019. LNCS, vol. 11483, pp. 96–110. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-21290-2_7
10. Khorikov, V.: Unit Testing Principles, Practices, and Patterns. Simon and Schuster (2020)
11. Kuhlmann, M., Gogolla, M.: From UML and OCL to relational logic and back. In: International Conference on Model Driven Engineering Languages and Systems, pp. 415–431. Springer (2012)
12. Leite, L., Rocha, C., Kon, F., Milojicic, D., Meirelles, P.: A survey of DevOps concepts and challenges. ACM CSUR **52**(6), 1–35 (2019)
13. Madeja, M., Porubän, J., Chodarev, S., Sulír, M., Gurbál', F.: Empirical study of test case and test framework presence in public projects on GitHub. Appl. Sci. **11**(16), 7250 (2021)
14. Mohd-Shafie, M.L., Kadir, W., Lichter, H., Khatibsyarbini, M., Isa, M.A.: Model-based test case generation and prioritization: a systematic literature review. Softw. Syst. Model. **21**(2), 717–753 (2021). https://doi.org/10.1007/s10270-021-00924-8
15. Myers, G.J., Sandler, C., Badgett, T.: The Art of Software Testing. Wiley (2011)
16. Planas, E., Olive, A.: The DBLP case study (2006)
17. Rull, G., Farré, C., Teniente, E., Urpí, T.: Providing explanations for database schema validation. In: International Conference on Database and Expert Systems Applications, pp. 660–667. Springer (2008)

18. Semeráth, O., Babikian, A.A., Li, A., Marussy, K., Varró, D.: Automated generation of consistent models with structural and attribute constraints. In: Proceedings of the 23rd ACM/IEEE International Conference on Model Driven Engineering Languages and Systems, pp. 187–199 (2020)
19. Smart, J.F., Molak, J.: BDD in Action: Behavior-Driven Development for the Whole Software Lifecycle. Simon and Schuster (2023)
20. Soltana, G., Sabetzadeh, M., Briand, L.C.: Practical constraint solving for generating system test data. ACM TOSEM **29**(2), 1–48 (2020)
21. Wang, C., Pastore, F., Goknil, A., Briand, L., Iqbal, Z.: Automatic generation of system test cases from use case specifications. In: Proceedings of the 2015 International Symposium on Software Testing and Analysis, pp. 385–396 (2015)

Pre-processing and Forecasting

Anchorlogy: An Ontology for Anchoring Bias Detection in Forecasting

Mateus Peixoto[1](✉), Fernanda Baião[1], Renata Guizzardi[2], and Giancarlo Guizzardi[2]

[1] Department of Industrial Engineering, Pontifical Catholic University of Rio de Janeiro (PUC-Rio), Rua Marquês são Vicente 225, Rio de Janeiro 22451-900, Brazil
mpeixoto@aluno.puc-rio.br, fbaiao@puc-rio.br

[2] University of Twente, Drienerlolaan 5, 7522 NB Enschede, The Netherlands
{r.guizzardi,g.guizzardi}@utwente.nl

Abstract. Anchoring bias is one of the most prevalent biases within forecasting. It distorts managers' estimations whenever context-driven intervention to the statistical model output is required. Consequences extend beyond a single organization since forecasting affects order quantity decisions and, therefore, the relations among suppliers, potentially generating a bullwhip effect throughout the supply chain. Anchoring bias can have a significant impact, and despite being related to a numerical value, its detection is very complex. Moreover, it tends to be recurrent when the context that caused the distortion is not explored and precisely understood. Current detection approaches are incomplete, as they do not make explicit the directional component of anchors or their meaning to the decision maker's mental heuristics. In this work, we present Anchorlogy, an ontology devised to explicitly provide the required context to detect and mitigate anchoring bias during a decision-making process, and a metrological approach to measure it while addressing the deficiencies found in other metrics in the current psychological literature. Our proposal was validated by applying it to two case studies in the forecasting domain, and the results show that it effectively prevents the bullwhip effect in real-world scenarios.

Keywords: Anchoring Bias · Ontology · Forecasting · Supply Chain · Cognitive Bias · Bullwhip Effect Mitigation

1 Introduction

Demand forecasting is crucial for procurement, production, inventory, logistics, and the overall decision-making process within organizations [20]. While numerous quantitative models have been proposed to improve the accuracy of product forecasts [2], human judgment (henceforth referred to as *judgment*), whether used independently or in conjunction with these models, is still a established practice in forecasting, and well-documented in the literature. Various judgment

J. Krogstie et al. (Eds.): CAiSE 2025, LNCS 15701, pp. 277–293, 2025.
https://doi.org/10.1007/978-3-031-94569-4_16

heuristics or mental shortcuts are employed in the judgmental forecasting process, with over 71% of companies relying on judgmental inputs [20].

Judgment and forecasting are inherently intertwined. Even in companies that predominantly use statistical forecasting, judgment remains vital in selecting appropriate models, methods, and predictors. Regardless of the sophistication of decision support systems, judgment cannot be completely excluded, as ongoing context-aware performance assessments are essential [24]. The primary justification for the prevalence of human interference in the forecasting process lies in the need to incorporate tacit knowledge from the expert, which is not captured by statistical models. These judgmental adjustments are typically made based on intuition, experience, domain- and context-specific knowledge [24].

Because of the way judgmental forecasts and adjustments affect agents' beliefs and decision-making processes, several documented cognitive biases are known. Individuals exhibit a persistent tendency to inappropriately anchor their forecast based on an available value and make insufficient adjustments from that anchor [12]. Sanders and Wood explain this phenomenon: "When generating a numerical forecast, humans often start with a reference point or anchor, even if this approach is irrational. Due to the *Anchoring Effect*, this initial anchor can unduly sway people's estimates" [29]. An *anchor* is defined in behavioral economics literature as the numerical value to which the decision maker gets disproportionally attached [29].

However, deliberately setting an informative anchor as a reference point is a typical practice within judgmental forecasting. Common examples are adjusting last month's demand considering the trend of the time series, or toward the mean for an untrended auto-correlated demand time series [12]. This facilitates the manifestation of the anchoring bias in the form of insufficient adjustments from the reference, distorting decisions and leading to errors [12].

Anchoring bias is certainly one of the most relevant biases that affect judgmental forecasting and adjustment. The internal belief distortions created by the anchoring bias can generate critical impacts not just for a single firm, but for an entire supply chain, since sub-optimal decisions taken by a local agent tend to contribute to the generation of a *bullwhip effect* [4,34]. Implementing a system to prevent the occurrence of anchoring bias is paramount for improving supply chain resilience. Among the reasons for that are: a) the widespread presence of the anchoring bias in the forecasting literature; b) its potential impact on inventory management; and c) the causal relationship between poor decision-making and the bullwhip effect.

Kahneman [18] considers judgment to be fundamentally fallible and defines cognitive bias as a systemic deviation from a rational alternative. Zeller et al. [35] demonstrate judgement fallibility in certain scenarios. In other words, computer-based algorithms (CBA) can be more reliable than human estimates in certain scenarios, justifying the use of these algorithms as rational alternatives. Managers can be tempted to rely on their first instincts. However, it is crucial to revisit initial impressions to improve reliability [23].

Anchoring bias must be addressed to conciliate the best aspects of statistical and human forecasting, aligning the precision of a statistical model (SM) with the domain knowledge and context-awareness of a human decision-maker [29]. Considering that Anchoring Bias is numeric by its nature, the aforementioned biases may be detected by an algorithm considered the method used to determine the forecast [29] .

This research work takes a step in the direction of providing *an anchor detection supporting system*. First, we propose an ontology for anchor definition, explanation and detection, named ***Anchorlogy***. This ontology extends existing decision-making ontologies [14] [27], and is represented using OntoUML, a well-known ontology modeling language grounded on the Unified Foundational Ontology (UFO) [16]. Anchorlogy defines what an anchor is, represents different types of anchors, and details how the anchoring bias interacts with the belief formation process. Second, this paper proposes a new measurement for anchoring bias to be used as a detection mechanism while explicitly considering the context. This mechanism allows for automatic detection, thus mitigating the effect of bias on the behavioral component of the decision-making process.

The text is structured as follows: Sect. 2 describes background information relevant for this work, Sects. 3 and 4 respectively describe the proposed ontology and propose a new anchoring bias detection mechanism based on it. Section 5 details the evaluation of applying our proposal in two scenarios, while Sect. 6 poses main conclusions and points to future research directions.

2 Background

Behavioral Economics literature presents some interesting findings. Kahneman [18] argues there exist two decision-making modes within the human mind: while **system 1** is fast, intuitive, and inexact, **system 2** is slower, requires deliberation and care before making a decision, and is thus considered rational. Although the occurrence of cognitive biases is typical of system 1, an anchor may occur in both system 1 and system 2.

Schwarz et al. [6] state that "The effect of anchors is one indication that many statements of value and belief are not directly retrieved from memory but, rather, are constructed online in response to a query. (...) Anchoring illustrates this construction because judgments of value and belief are influenced by irrelevant or uninformative starting points present only at the time of questioning" (p. 116). Therefore, the *Anchoring Effect* interferes with *Agents' Beliefs* in an intuitive level, as the so called *Priming effect*.

Algorithm aversion is described as the innate preference for human-generated forecasts instead of CBA, although the latter is often more accurate than the former [9]. *Overconfidence bias* is another common bias within the field, defined as the tendency of decision-makers to believe that the information they have is more accurate than it is and to overestimate their performance [31]. Both algorithm aversion and overconfidence are intertwined biases.

In this work, we assume the so-called *Intentional Stance* [8]. Hence, we interpret artificial systems that enact complex information processes as **cognitive agents**. In other words, we assume that artificial agents (e.g., an AI system) can bear *digital beliefs and intentions (goals).*

3 Anchor and Related Concepts

Anchorlogy is structured with two main concerns, namely **defining** and **detecting** an anchor. We start by addressing the impact of the anchoring effect upon the beliefs of an agent. Anchorlogy plays a crucial role in conciliating the concepts from forecasting and behavioral economics research since bias definitions can diverge, considering the context associated with the process. Figure 1 depicts Anchorlogy. For references to the OntoUML stereotypes used in this figure as well as their semantics, the reader is referred to [1,10,11,15].

The core phenomena represented in the model of Fig. 1 relate to the process of Belief Formation. Beliefs (as well as Desires and Intentions) are examples of Mental Aspects [13], i.e., certain types of Dispositions [16] inhering in Agents [13]. We also consider Intuition [7] to be a type of Mental Attitude (or mental disposition) and therefore also a type of mental aspect. Here, however, intuitions include more primitive dispositions (e.g., "gut feelings"), which do not necessarily have a Propositional Content.

Mental Aspect Formations are Mental Events [5] (in which agents participate) that give rise to new mental aspects, i.e., a mental aspect is always created by an event of Mental Aspect Formation. These events include Perception Events, in which mental aspects are created by stimuli originating in the external world (e.g., I perceive that it is not raining outside, which gives rise to my belief that it is not raining outside at this moment). In addition to perceptual events, mental aspects can be formed by events occurring within the agent's cognition. These include processes of deliberate reasoning (system 2 processes) as well as the process of intuitive association (system 1 processes).

Like all events, Mental Aspect Formation events are manifestations of interacting dispositions [16]. So, existing mental aspects inhering in an agent can *influence* the *manifestations* of a mental aspect formation. For example, via logical inference, I can form a new belief based on my existing beliefs.

We are here particularly interested in Value Beliefs, i.e. the belief that a certain quality [16] (e.g., the price of my car) has a particular quality value (e.g., 20.000 euros) in a particular time interval (e.g., this month). A Future Value Belief is a value belief whose propositional content refers to a *future time interval*. For example, I can believe that the price of my car will be of 15.000 euros one year from now. The process of forming future value beliefs is called Value Forecasting, which plays a role in decision-making since decisions are grounded in preference relations [25,27,28] created by Future Value Beliefs.

To be more precise, a Value Forecasting is a mental act performed by a Value Forecaster. A value forecaster can be an Artificial Agent as well as a Human Agent. Value Forecasting can be Rational Value

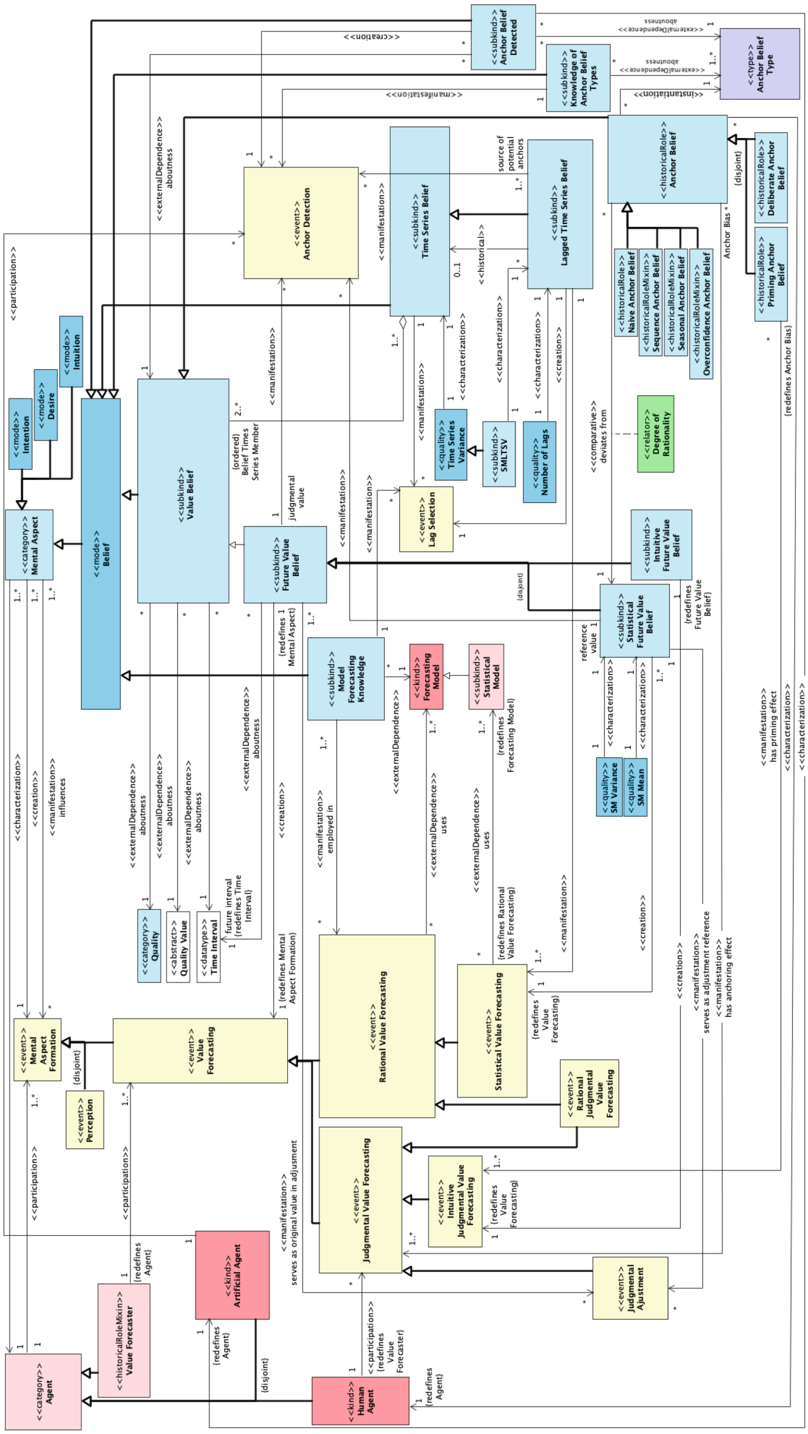


Fig. 1. Anchoring Bias Definition Model

Forecasting or Judgmental Value Forecasting. The latter can only be performed by human agents. In contrast, a rational value forecasting can be performed by human and artificial agents alike. A rational value forecasting employs the explicit knowledge of a forecasting model. So, a Model Forecasting Knowledge, like all knowledge, is itself a belief inhering in an agent, but a complex belief (composed of sub-beliefs) and whose aggregated propositional content refers to a forecasting model. A rational value forecasting event includes a *manifestation* of that model forecasting knowledge. A particular type of rational value forecasting event is *Statistical Value Forecasting*. In this case, the model knowledge employed is a knowledge of a Statistical Model. When a rational value forecasting event is performed by a human agent this is called a Rational Judgmental Value Forecasting. Although human agents can perform rational value forecasting processes, they also sometimes perform value forecasting in an intuitive manner. The latter type of event is called an Intuitive Judgmental Value Forecasting and can it include *manifestations* of intuitions.

To better illustrate the aforementioned concepts, consider the following example: Paul is a supply chain manager responsible for deciding how many of each product (SKU - stock keeping unit) should be ordered. To make such decision, Paul should forecast a value for each SKU conducting a Rational Value Forecasting; more specifically, he chooses a Statistical Value Forecasting, thus relying on a Statistical Model. Alternatively, Paul could also rely on Intuitive Judgmental Value Forecasting, thus relying on his Intuition (*gut feeling*).

A Time Series Belief (TSB) is a metrologically complex belief composed of a sequence of Value Beliefs *ordered* by their successive time intervals of reference. The parts of this entity are said to play the (formal) role of *Time Series Member*.

Further elaborating on the previous example, Paul is deciding how many items to order for an SKU. He has a TSB made of his beliefs of the number of items of this SKU that were sold in previous months (sales numbers). This TSB is composed of multiple sales numbers *Time Series Member*, ordered in a monthly interval.

A statistical value forecasting event employs a Lagged Time Series Belief (LTSB), which is a particular type of TSB. An LTSB is explicitly created by a Lag Selection event (hence, a mental aspect formation event - although this relation is not shown in the model). A Lag Selection event employs an original TSB and knowledge of statistical model (i.e., a model forecasting knowledge inhering in an agent) and creates an LTSB by choosing an appropriate Number of Lags. Typically, an LTSB is a proper part of the original TSB, i.e., the number of lags is typically less than the cardinality of the original sequence. An LTSB is hence *historically dependent on* [10] the TSB from which it was 'extracted'. The number of lags chosen in that selection process becomes a (derived) quality of the LTSB. Every TSB is characterized also by a Times Series Variance (TSV). If a TSV is a quality inhering in an

LTSB then it is called a Statistical Model Lag Time Series Variance (SMLTSC).

Going back to our example, suppose a Lag Selection event of 9 (i.e., Paul decided a Number of Lags equals 9). This means Paul uses the last 9 Time Series Members of the series in his forecasting. Moreover, a Statistical Model Lag Time Series Variance (SMLTSC) is applied. Paul is here making a statistical value forecasting.

A Future Value Belief can be: (a) an Intuitive Future Value Belief, i.e., a future belief created by an intuitive judgmental value forecasting event; (b) a Rational Future Value Belief (not in the figure), i.e., one that is created by a rational value forecasting event. A Statistical Future Value Belief (SFVB) is a rational future value belief created by a *Statistical Value Forecasting.* A SFVB is characterized by a Statistical Model Variance (SM Variance) but also by a Statistical Model Mean (SM Mean). A SFVB is a value belief and, hence, its propositional content refers to a quality value. In the case of a SFVB, that quality value is always the same as the quality value of the *SM Mean* inhering in that SFVB (not shown in the model).

Some illustrative examples of the aforementioned concepts would be the following: suppose Paul has an Intuition that the next month's demand will be 85. Paul is here making an Intuitive Judgmental Value Forecasting(IJVF) and 85 is an Intuitive Future Value Belief. Now, suppose a certain Statistical Model has as SM Mean of 70, as the result of aa *Statistical Value Forecasting.* We say that 70 is the Statistical Future Value Belief (SFVB.

Judgmental Value Forecasting (JVF) events can suffer a biased influence from another value belief. A value belief that: (1) *deviates from* a reference statistical future value belief (SFVB) (i.e., one for the same quality, quality value and time interval), and (2) actually exerts an influence in a JVF, playing the role of an Anchor Belief in that JVF. That influence relation between an anchor belief and a JVF is termed an *anchoring effect* (in whose relation the anchor belief is said to play the formal role of an *Anchor Bias*). The quality value referred by an anchor belief (i.e., the one that *deviates from* the quality value of the corresponding SFVB) is called an Anchor (not shown in the model). The degree to which these two values deviate from each other is called a Degree of Rationality .

In Paul's decision-making process, the value of 85 chosen by JVF (using gut feeling) may be the result of the influence of an Anchor Belief, with an Anchor equals to 90. Having 90 as an anchor interferes in Paul's decision, making it difficult to move away from 90, so even if the *Statistical Value Forecasting* results in 70, Paul chooses 85. This is a case of an Anchoring Bias.

A *value belief* can be deliberately selected by a *value forecaster* to play the role of anchor belief in a *judgmental value forecasting.* In this case, that anchor belief is called a Deliberate Anchor Belief. In contrast, an anchor belief that has an anchoring effect in an *intuitive judgmental value forecasting* (IJVF) event without being explicitly selected is called a Priming Anchor Belief. Moreover, IJVF is frequently unconscious. In this case, that anchor belief is said to have a *priming effect* on the IJVF event. In other words, the *has priming effect relation* is a specialization (technically, a redefinition) of the *has anchoring effect* relation for the case of priming anchor beliefs.

If Paul thinks the sales of the upcoming month of November will resemble the one in November of the previous year, Paul may choose these previous sales value as Anchor. Then we say Paul has a Deliberate Anchor Belief. Moreover, the Deliberate Anchor Belief would make it difficult for his estimate to distance itself from the sales value of the past months of November, thus incurring Anchoring Bias and distorting his forecast. On the other hand, suppose Paul has an Intuition about the next month's demand, but his estimation is distorted by a given Anchor Belief. This implies that Paul will be inclined to believe his intuition, attributing it an unreasonably high importance. In other words, Paul's estimation is distorted by an Anchor Belief.

Finally, a Judgmental Adjustment event is a JVF event in which a statistical future value belief serves as a reference for "updating" an existing (human) future value belief. Technically, since we consider the identity of beliefs to be defined by their propositional content, a judgmental adjustment event actually creates a new belief taking into consideration a SFVB (said to *serve as adjustment reference*) and an existing belief (said to serve as *original value*). The interaction between these two beliefs will create a new belief as the human agent incorporates the context surrounding the initial IJVF to the model. This context is used to adjust from the SFVB reference. Effectively, the access to the SFVB updates the Quality of the original belief, while the context that constructed the human belief alters the Perception of the SFVB, thus motivating an adjustment from the SFVB as the formation of a new Belief.

Now let us move to the concepts related to detecting anchor biases (see Fig. 1). The central concept in this model is the Anchor Detection event. This is an event performed by an artificial agent. The goal of this event is to detect whether a JVF event has been biased by another Value Belief. In other words, this event aims at detecting whether a value belief has played the role of an Anchor Belief w.r.t. to that JVF event. In addition, it aims at finding out what type of anchor belief belongs to (see discussion below). To do this, the Artificial Agent performing Anchor Detection takes into consideration the following items:

- the potentially biased Future Value Belief (termed here a *judgmental value*);
- a Statistical Future Value Belief (having a particular SM Variance - Statistical Model Variance) as *reference value*. This reference is produced by a Statistical Value Forecasting event aimed to predict a quality value to the same quality and for the same time interval;
- a Lagged Time Series Belief (LTSB) (having a particular SMLTSC), which the Value Forecaster took into consideration when forecasting that future value in a process of Statistical Judgmental Value Forecasting. As previously discussed, an LTSB is a sequence of future value beliefs

(*belief time series members*) ordered by their associated intervals. The members of this structure are *potential anchors*;
- the KNOWLEDGE OF ANCHOR BELIEF TYPES. Anchor beliefs instantiate different ANCHOR BELIEF TYPES (see discussion below).

In other words, an anchor detection event is a *manifestation* of these beliefs. Moreover, it is a MENTAL ASPECT FORMATION event, i.e., it creates a new belief inhering in the artificial agent performing that detection event. We call this belief ANCHOR BELIEF DETECTED (ABD). An ABD is the belief an artificial agent bears about a value belief is an ANCHOR BELIEF of a given ANCHOR BELIEF TYPE.

We consider here four different types (technically, roles) that a value belief acting as an anchor bias can instantiate. The instances of all these types must, of course, also satisfy all the general conditions for being an anchor belief, i.e., having an *anchoring effect* on a JVF event, and significantly deviating from a reference value. In the sequel, we discuss these different anchor belief types:

- NAIVE ANCHOR BELIEF: to play this role, a value belief has to be an anchor belief, and it has to be the last (more recent) *time series member* of the LTSB under consideration. Moreover, this LTSB must be composed of value beliefs referring to the same quality. We consider here that the VALUE BELIEFS in a TIME SERIES BELIEF refer to the same QUALITY which, despite changing qualitatively in time, preserves its identity. For example, the *amount of laptops* in my warehouse this month is taken to refer to a quality inhering in a collective entity (*my laptop collection*). This quality can take different values in time, but it stays numerically the same. When considering the *number of laptop sales* both in January and February, we are technically talking about two different qualities that inhere in two different complex events; namely, the aggregated sales of January; the aggregated sales of February. So, in this case, we should talk about qualities of the same kind. For the sake of simplicity, we here neglect this distinctions but without any lost of generality. For example, when predicting the number of pens I should buy next month, I could be biased by the number of pens I bought last month;
- SEQUENCE ANCHOR: to play this role, a value belief has to be an anchor belief, and it has to be the last (more recent) time series member of the LTSB under consideration. However, this must be an LTSB composed of qualities of a different type. For example, when predicting the number of pens I should buy next month, I could be biased by the number of pencils I just decided to buy for this month;
- SEASONAL VALUE BELIEF: in other to play this role, a value belief has to be an anchor belief and it has to a value belief for the same quality and for a *qualitatively equivalent* time interval in the past. For example, when predicting the number of pens I should buy next July, I could be biased by the number of pens I bought last July;
- OVERCONFIDENCE ANCHOR BELIEF: in a JUDGMENTAL ADJUSTMENT event, a value forecaster can take into consideration a RATIONAL FUTURE VALUE

BELIEF as a reference in order to consider updating their belief about a future value belief they have. We have an overconfidence anchor belief when that future value belief - despite significantly deviating from a reference value - causes that forecaster to override that belief updating event, thus, maintaining their own original prediction. In other words, that future value belief works as an overconfidence anchor belief in the judgmental adjustment event that correct its own associated quality value (i.e., its own prediction).

4 Anchoring Bias Detection Mechanism

Anchoring bias is typically measured with control groups. In other words, the group of references is split in two. Then, a group of experts are consulted before the stimuli, and *the median* of their decisions and *variance* serve as a reference for the inference of bias by the other group. However, this presents a *fragile perspective to neutrality* since the biases of the control group participants are never taken into account [6]. The measurements and notion of a biased response derive solely from the deviation of the rational alternative [33] this is insufficient considering the forecasting context. The most accurate SM should be used as the reference for rational decision-making since it is the CBA most likely to outperform the judgmental forecaster. The limitations of the previous measurement [21] are manifested in two main deficiencies: a) the definition of bias is only the deviation from the rational point, not taking into account the directional components of the anchor, and b) the lack of a clear threshold for bias detection. As aforementioned, it is important to use the SMLTSV[1] and SMV to ensure a fair comparison for model performance, that is, its ability to reduce the SMLTSV variance is relevant when determining the importance that should be given to it. We created a new measurement as shown in (1), where PA is the potential anchor; SMM is the SM's mean; and I is the input of the decision support system used by the decision maker.

$$\frac{|\mathrm{I}-\mathrm{PA}|}{2\sqrt{\mathrm{SMLTSV}}}-\frac{|\mathrm{I}-\mathrm{SMM}|}{2\sqrt{\mathrm{SMV}}}<0 \quad (1)$$

As can be seen, the relevance of any adjustment toward the anchor is mediated by the natural variance of the time series proportional to the number of points needed to construct the performance of the SM. This way, the directional component of the anchor is respected, and a clear threshold is established by giving more relative preference to the ANCHOR than the SM. It is important to note that a test should be performed for each potential anchor considering the SMM. Random external values may distort decision-making while not being part of any of the main anchor types documented by the literature. An arbitrary threshold is established, warning if there is a distortion greater than 20% from the SM MEAN.

[1] In the remainder of the paper, we use the name of qualities such SMLTSV, SMV or SMM as a shortcut to refer to their measured values.

The anchor detection test (ADT) is formulated to receive the following inputs: The SM mean (SMM) and variance, the SM lag time series variance (SMLTSV), a list of potential anchors (LPA) and another list of their respective anchor types (LAT), and the suggestion of decision made by the user, denoted by the input(I).

List of variables: I= Input of the user, LPA = List of Potential Anchors, LAT = List of Anchor Types, SMM = Statistical Model Mean, SMV= Statistical Model Variance, SMLTSV = Statistical Lag Time Series Model Variance.
Input Data [I, LPA, LAT, SMM, SMV, SMLTSV];
Output Data[Anchor List(AL) = [Anchor, Anchor type]];
for *element(el) in LPA* **do**
  if $|el - I| < SMV$ **then**
    $AL \leftarrow [el, LAT(el)]$;
  else
    if $|el - I| < |SMM - I|$ **then**
      If $\frac{|el-I|}{\sqrt{SMLTSV}} - \frac{|SMM-I|}{\sqrt{SMV}} \leq 0$;
      $AL \leftarrow [el, LAT(el)]$
    end
  end
  if $|I - SMM| > \frac{SMV}{5}$ **then**
    $AL \leftarrow$ "Considerable deviation detected"
  end
end

Algorithm 1: Anchor Detection Mechanism

As illustrated in Algorithm 1, the first step is to verify if the distance between each potential anchor and the decision is smaller than the SM variance (SMV). This is relevant because the SMV represents the expected error of the SM, and therefore, if the distance between the input and the potential anchor is smaller than the variance of the SM, it would imply the proximity to the potential anchor is more precise as a forecast than the output of the SM. In other words, this confirms that the potential anchor is in fact, an anchor.

The algorithm continues with the potential anchors that have a greater distance from the input than the SM variance. The second check is to verify if the absolute distances between the potential anchor and the input are smaller than the distance between the input and the SM mean. This is justified by the fact that the greater proximity to the potential anchor indicates a disproportional influence of such a potential anchor. If so, this triggers the third step, with the relative relevance check, verifying if the distance between the input and potential anchor divided by the standard deviation of the time series (the square root of SMLTSV) is lower than the distance between the input and the SM mean divided by the standard deviation of the SM. This third step is required to obtain an importance proxy of the potential anchor to the input while considering the precision of the model in the calculation of its importance. In other words, if the model cannot properly capture the time series's structure, it should receive less importance than a more accurate model. The test must be replicated for every potential anchor. Since the anchoring bias is moderated by knowledge, as time goes on and the agent becomes aware of the biases associated with her decisions and acquires domain knowledge, the anchoring bias tends to be mitigated [30].

5 Proposal Validation

To validate our proposed ontology, two case studies were conducted, and the methods employed consisted of the application of the anchoring detection mechanism in two real-world scenarios: 1) simulating the scenario of the COVID-19 toilet paper hoarding effect on Walmart [19] and 2) reproducing the experiment conducted by Haag et al. [17] that evaluated the effectiveness of an explainable artificial intelligence (XAI) information system (IS) as a mitigation mechanism for the anchoring bias.

5.1 Case Study: Hoarding & Walmart's Bullwhip Effect

The value of the tool was assessed by how its implementation can prevent real-life problems, such as the toilet paper demand surge during the COVID-19 pandemic. Toilet paper is a product with one of the most stable demands in existence. However, the panic buying and hoarding during the COVID pandemic led to an increase of 845% in demand [19]. The variation of the order quantity over time can be interpreted as the definition of the bullwhip effect [22]. In fact, suboptimal decision-making contributes to the generation of the bullwhip effect [32].

Therefore, despite not having access to the precise data values, the impact can be understood through the percentage movements that were disclosed, knowing the demand suffered a major crash after the hoarding period. A hypothetical scenario reflecting the behaviour of the real phenomena was constructed to analyze the effect of hoarding on order quantity time series, assuming an average of 500.

The selected potential anchor was the Naive anchor, which means that the decision maker replicates the demand of the previous instance. This seems likely, given the performance of the supply chain [26]. Considering the expected stability of the demand, a potentially good forecasting algorithm is Holt, achieved simply by removing the outlier of the demand surge and forecasting two steps ahead [3].

The time series and SM variance can be extracted from the simulated data before the outlier. After the demand surge, there is a noticeable falloff, and the increase in variance for the supplier is directly related to the proximity to the real demand at the falloff point, which later on stabilized over time.

Therefore, the ADT is conducted in the following way: TSV = 27, SMV = 23, SMM = 515 (e.g. Holt model prediction two steps ahead), Potential Anchor = 4,459, and anchor type = Naive. After conducting the anchor detection test, the *naive anchor* is thus identified. The effectiveness of the system in preventing the bullwhip effect is made explicit when comparing the variances of the order time series TSV with intervention = 2,572 and TSV without the intervention = 1,309,503. Therefore, there is a reduction of 508 times. Note that despite following the system, there is still a major increase in variance; this is due to the atypical nature of the period analyzed. The numerical experiment and pseudo-code 1 showed the detection mechanism's effectiveness in addressing the anchoring bias. Thus, if the decision maker follows the feedback of our system, a crisis of this type may be prevented.

5.2 Anchorlogy x XAI

The experiment conducted by Haag et al. [17] evaluates the relevance of price anchors and unrelated anchors in the price estimation of vehicles. They implement an AI system that warns the decision-maker if the suggested price (when available) is classified as fair, moderate, or unfair. The classifications were given by a difference of 5%, 10%, and over 20% concerning market price. The XAI component uses Shapley-based explanations for the machine learning (ML) algorithm that calculates this price classification, hence informing the decision-maker of the expected value of each relevant characteristic of the vehicle.

The experiment is conducted with 390 participants and takes the control group (i.e., the mean of the price estimations without the anchors and their variance) as a rational reference.

The authors conclude that the integration between the Shapley values regarding the vehicle characteristics and the ML price classification is effective in detecting and mitigating the anchoring bias in unfair scenarios but not in moderate and fair ones. When applying our proposal in the same scenarios specified in their work, we obtain better results. Their results are reproduced in Table 1, in which we added the last column with the results from our proposal.

Table 1. Comparing the performance of our anchoring bias detection (Anchorlogy) with the results from [17] (XAI IS)

Task	Listing price	Fair price	Price fairness	XAI IS	Anchorlogy
1	7500	5193	unfair	Detected	Detected
2	5990	5220	moderate	Not detected	Detected
3	8480	8256	fair	Not detected	Detected
4	22690	16652	unfair	Detected	Detected
5	28900	25502	moderate	Not detected	Detected
6	24999	22981	fair	Not detected	Detected

As Table 1 shows, the Anchorlogy metrological approach is effective in the detection of the anchoring bias for all categories because of the relative importance assessment that considers the directional component and differences in variance between groups (rational reference and biased).

Since the paper [17] described the standard deviation of the control group, the experiments described in Table 1 can be simulated using our methodology, like the following example: The "Price fairness" column indicated threshold of the distance from the input of the second group to the fair price. Take task 5, for instance; the fair price is 25,502, and its label is moderate, meaning that the input was between 5% and 10% above the fair price. The listing price is the anchor in question, and the variance of the control group and main group are displayed. Therefore, if we equate the control group to the statistical model, since it is the rational reference used by the authors, we have: SMM = Fair price = 25,502; PA = Listing price = 28,900; I = 25,502*107.5% = 27,415; SMV = Control group variance = 6,984; and SMLTSV = Experiment group variance = 10,925. When the ADT is conducted the anchor is detected, something that was not possible via the AI warning system.

Note that in the original experiment, the participants judgmentally estimated the price of a vehicle and had access to a listing price that played the role of anchor. The price estimation is unduly swayed by the listing price, even if the system informed the participants of the distortions in price fairness of moderate scale. Therefore, it is implied that the perceived relevance of the listing price is more relevant than the AI recommendation system. Thus, an *algorithm aversion bias* is automatically detected by our approach.

Furthermore, the anchoring effect manifested in the choice of maintaining the original decision is a consequence of the overconfidence of the participant's abilities to estimate the correct price of the asset. Thus, the detection of the anchoring bias automatically implies the detection of an *overconfidence bias* related to numerical estimation processes.

Due to our precise definitions of these subtypes of cognitive biases, our proposed approach is capable of automatically detecting not only anchoring but also overconfidence and algorithm aversion biases.

6 Conclusion

Considering the relevance that judgmental forecasting and adjustments play in decision-making, any distortion can bring disastrous consequences and even supply chain disruptions. Therefore, mitigating cognitive biases is paramount. Yet, its detection and measurement still pose challenges in the literature.

Anchoring is the most prevalent cognitive bias within the judgmental forecasting literature. In this work, we propose an ontology to precisely define the subtle notions that give rise to the Anchoring Effect and apply the ontology to help detect and mitigate the anchoring bias. The proposed mechanism uses a new metrological approach constructed for forecasting, allowing for the context-dependent nature of the anchoring bias to be dealt with. The system was effective when applied to prevent a real-world scenario during the COVID-19 pandemic and improved the detection of the Anchoring Bias when compared with an existing XAI IS solution.

Our proposal presents an opportunity to advance the *humachine paradigm*, defined by Sanders and Wood [29] as the perfect integration of computational mechanisms and humans in a system that can mitigate the deficiencies of both.

This integration is only possible because the belief formation process is made explicit by the ontology, thus, biased decisions, when detected by the proposed mechanism, indicate the belief distortion and the most likely anchor candidates for said bias. This proposal also makes explicit that all artificial agents, and by extension statistical models, have beliefs that the data they possess is correct and representative of the phenomena of interest, i.e. Time Series Beliefs.

Future work will address distinct types of cognitive biases while also integrating and testing it in real organizations, in a behavioral operations management case study. The explicit perspective of beliefs of artificial agents presents an interesting avenue of research to formulate an algorithmic bias ontology.

References

1. Almeida, J., Falbo, R.A., Guizzardi, G.: Events as entities in ontology-driven conceptual modeling. In: Laender, A., Pernici, B., Lim, E.-P., de Oliveira, J. (eds.) ER 2019. LNCS, vol. 11788, pp. 469–483. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-33223-5_39
2. Arvan, M., Fahimnia, B., Reisi, M., Siemsen, E.: Integrating human judgement into quantitative forecasting methods: a review. Omega **86**, 237–252 (2019)
3. Bermúdez, J.D., Corberán-Vallet, A., Vercher, E.: Forecasting time series with missing data using Holt's model. J. Stat. Plann. Infer. **139**(8), 2791–2799 (2009)
4. Brauch, M., Mohaghegh, M., Gröler, A.: Causes of the bullwhip effect: a systematic review and categorization of its causes. Manag. Res. Rev. (2024)
5. Bunnin, N., Yu, J.: The Blackwell Dictionary of Western Philosophy. Wiley (2008)
6. Chapman, G.B., Johnson, E.J.: Anchoring, activation, and the construction of values. Organ. Behav. Hum. Decis. Process. **79**(2), 115–153 (1999)
7. Chudnoff, E.: What intuitions are like. Philos. Phenomenol. Res. **82**(3), 625–654 (2011)
8. Dennett, D.C.: The Intentional Stance. MIT press (1989)
9. Dietvorst, B.J., Simmons, J.P., Massey, C.: Algorithm aversion: people erroneously avoid algorithms after seeing them err. J. Exp. Psychol. Gen. **144**(1), 114–126 (2015)
10. Fonseca, C.M., Porello, D., Guizzardi, G., Almeida, J., Guarino, N.: Relations in ontology-driven conceptual modeling. In: Laender, A., Pernici, B., Lim, E.-P., de Oliveira, J. (eds.) ER 2019. LNCS, vol. 11788, pp. 28–42. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-33223-5_4
11. Fonseca, C.M., et al.: Incorporating types of types in ontology-driven conceptual modeling. In: International Conference on Conceptual Modeling, pp. 18–34. Springer (2022)
12. Goodwin, P., Moritz, B., Siemsen, E.: Forecast decisions. In: Donohue, K., Katok, E., Leider, S. (eds.) The Handbook of Behavioral Operations, 1 edn, pp. 433–458. Wiley. (2018). https://doi.org/10.1002/9781119138341.ch12
13. Guizzardi, G., de Almeida Falbo, R., Guizzardi, R.S.: Grounding software domain ontologies in the unified foundational ontology (UFO): the case of the ode software process ontology. In: CIbSE, pp. 127–140. Citeseer (2008)

14. Guizzardi, R., Carneiro, B.G., Porello, D., Guizzardi, G.: A core ontology on decision making. In: ONTOBRAS (2020). https://api.semanticscholar.org/CorpusID:226301332
15. Guizzardi, G., et al.: Types and taxonomic structures in conceptual modeling: a novel ontological theory and engineering support. Data Knowl. Eng. **134**, 101891 (2021)
16. Guizzardi, G., et al.: UFO: unified foundational ontology. Appl. Ontol. **17**(1), 167–210 (2022)
17. Haag, F., Stingl, C., Zerfass, K., Hopf, K., Staake, T.: Overcoming anchoring bias: the potential of AI and XAI-based decision support. arXiv preprint arXiv:2405.04972 (2024)
18. Kahneman, D.: Thinking, fast and slow. Penguin Books, London (2012)
19. Kirk, C.P., Rifkin, L.S.: I'll trade you diamonds for toilet paper: consumer reacting, coping and adapting behaviors in the COVID-19 pandemic. J. Bus. Res. **117**, 124–131 (2020)
20. Kolassa, S., Rostami-Tabar, B., Siemsen, E.: Demand Forecasting for Executives and Professionals, 1st edn. CRC Press, Boca Raton, FL (2024)
21. Lieder, F., Griffiths, T.L., Huys, Q., Goodman, N.D.: The anchoring bias reflects rational use of cognitive resources. Psychon. Bull. Rev. **25**(1), 322–349 (2018). https://doi.org/10.3758/s13423-017-1286-8
22. Monroe, R.W., Sanders, N.R.: Supply chain management: a global perspective. Wiley, Hoboken, NJ. Transp. J. **51**(4), 506–508 (2012). https://doi.org/10.5325/transportationj.51.4.0506
23. Moritz, B., Siemsen, E., Kremer, M.: Judgmental forecasting: cognitive reflection and decision speed. Prod. Oper. Manag. **23**(7), 1146–1160 (2014)
24. Perera, H.N., Hurley, J., Fahimnia, B., Reisi, M.: The human factor in supply chain forecasting: a systematic review. Eur. J. Oper. Res. **274**(2), 574–600 (2019)
25. Porello, D., Guizzardi, G.: Towards an ontological modelling of preference relations. In: AI* IA 2018–Advances in Artificial Intelligence: XVIIth International Conference of the Italian Association for Artificial Intelligence, Trento, Italy, November 20–23, 2018, Proceedings 17, pp. 152–165. Springer (2018)
26. Rahman, T., Paul, S.K., Shukla, N., Agarwal, R., Taghikhah, F.: Managing panic buying-related instabilities in supply chains: a COVID-19 pandemic perspective. IFAC-PapersOnLine **55**(10), 305–310 (2022)
27. Ramos, E.d.C., Campos, M.L.M., Baião, F.: ABI approach: automatic bias identification in decision-making under risk based in an ontology of behavioral economics. arXiv preprint arXiv:2405.14067 (2024)
28. Sales, T.P., Guarino, N., Guizzardi, G., Mylopoulos, J.: An ontological analysis of value propositions. In: 2017 IEEE 21st International Enterprise Distributed Object Computing Conference (EDOC), pp. 184–193. IEEE, Quebec City, QC (2017). https://doi.org/10.1109/EDOC.2017.32
29. Sanders, N.R., Wood, J.D.: The Humachine: Humankind, Machines, and the Future of Enterprise, 1 edn. Routledge (2019). https://doi.org/10.4324/9780429001178
30. Smith, A.R., Windschitl, P.D., Bruchmann, K.: Knowledge matters: anchoring effects are moderated by knowledge level. Eur. J. Soc. Psychol. **43**(1), 97–108 (2013)
31. Van Oudenhoven, B., Van De Calseyde, P., Basten, R., Demerouti, E.: Predictive maintenance for industry 5.0: behavioural inquiries from a work system perspective. Int. J. Prod. Res. **61**(22), 7846–7865 (2023). https://doi.org/10.1080/00207543.2022.2154403

32. Yang, Y., Lin, J., Liu, G., Zhou, L.: The behavioural causes of bullwhip effect in supply chains: a systematic literature review. Int. J. Prod. Econ. **236**, 108120 (2021)
33. Yasseri, T., Reher, J.: Fooled by facts: quantifying anchoring bias through a large-scale experiment. J. Comput. Soc. Sci. **5**(1), 1001–1021 (2022)
34. Zanddizari, M., Tavakkoli-Moghaddam, R., Azaron, A.: Modeling stock-out loss and overstocking loss generated by bullwhip effect. Sci. Iranica **26**(3), 1913–1924 (2019). https://doi.org/10.24200/sci.2018.20199
35. Zellner, M., Abbas, A.E., Budescu, D.V., Galstyan, A.: A survey of human judgement and quantitative forecasting methods. Roy. Soc. Open Sci. **8**(2), 201187 (2021). https://doi.org/10.1098/rsos.201187

Process Model Forecasting Using Deep Temporal Learning

Wenjun Zhou(✉), Artem Polyvyanyy, and James Bailey

The University of Melbourne, Victoria, VIC 3010, Australia
{wenjun.zhou,artem.polyvyanyy,baileyj}@unimelb.edu.au

Abstract. Process discovery studies ways to construct process models from event logs of historical executions of a system. While discovered models aim to describe the system, process model forecasting aims to construct models that faithfully describe the executions the system will perform in a given period in the future, informing timely system improvements. Existing approaches tackle the problem of process model forecasting by decomposing it into multiple univariate time series forecasting problems. They forecast each directly-follows constraint over a pair of process activities separately and then aggregate these individual forecasts into the resulting process model. In this paper, we propose a deep learning-based approach that leverages multivariate time series forecasting to solve the process model forecasting problem. Our method learns dependencies across all activity constraints simultaneously, generating an integrated forecast of the entire model at once. Through evaluation over industrial event logs, we demonstrate that this approach significantly outperforms existing baselines and statistical multivariate methods in accuracy. Additionally, we introduce a new measure to evaluate the structural correctness of the forecasted models. In the context of information systems engineering, our work addresses the challenge of predicting process models to support future process planning and optimization.

Keywords: Process mining · process model forecasting · deep learning

1 Introduction

Business Process Management (BPM) is a key area within information systems engineering, focusing on the design, execution, and optimization of business processes to improve organizational efficiency and effectiveness [28]. Process mining is a subarea in BPM that studies ways to use event logs recorded by information systems to understand and improve these systems [25]. An *event log* is a collection of traces, each recorded as a sequence of executed activities by a system, for instance, during an execution of a business process. Within the process mining discipline, *process discovery* addresses the problem of constructing a process

This research was supported by The University of Melbourne's Research Computing Services and the Petascale Campus Initiative.

J. Krogstie et al. (Eds.): CAiSE 2025, LNCS 15701, pp. 294–312, 2025.
https://doi.org/10.1007/978-3-031-94569-4_17

model from an event log of a system that *describes the executions the system can support*, where a process model is a conceptual model composed of activities, routing decisions, and control flow that captures the ordering constraints over the activities and decisions. A *directly-follows graph* (DFG) is a process model often constructed by process discovery algorithms. It is a directed graph with nodes capturing activities, directed arcs specifying possible orders in which the activities can be executed, and numbers on nodes and arcs suggesting the frequencies with which the corresponding concepts can be executed. DFGs are discovered by most of the commercial process mining tools [26].

Recently, the problem of process model forecasting has been introduced [21]. Given an event log of a system and a time interval in the future, *process model forecasting* studies ways to construct a process model that *describes the executions the system will perform in the given time interval* [29]. Such a forecast, if accurate, can enhance organizations' understanding of their future business processes, allowing targeted planning for redesign and support initiatives. The state-of-the-art technique for process model forecasting tackles this problem by dividing it into multiple univariate time series forecasting sub-problems, one for each directly-follows (DF) constraint (an arc in a DFG), solving these sub-problems, and aggregating the results into the final forecasted DFG [8].

In this paper, we present a Deep Learning (DL) approach —*DeePMF* for process model forecasting that leverages multivariate time series forecasting. Instead of forecasting each DF constraint separately, *DeePMF* learns dependencies across all constraints simultaneously and then generates the forecasted process model at once. We demonstrate that this approach significantly outperforms existing baselines and statistical multivariate time series methods in accuracy. We also introduced a new measure of structural correctness of forecasted models and confirm that *DeePMF* constructs models of good structural characteristics.

Specifically, this paper makes these contributions:

1. A *sparsity test* for event logs that helps determine if an event log could be used to forecast accurate process models;
2. A measure of the level of *consistency* of a DFG that quantifies by how much the sum of incoming and outgoing arc frequencies differs for its activity nodes;
3. The *DeePMF approach to process model forecasting* grounded in DL multivariate time series forecasting techniques that delivers the state-of-the-art forecasting accuracy across multiple real-life datasets;
4. A comprehensive *evaluation* of *DeePMF* over a wide range of industrial event logs that confirms the effectiveness of our approach, suggesting that the transformer architecture often leads to better forecasts.

The next section discusses related work. Section 3 provides the concepts and background knowledge that supports the understanding of the subsequent sections. Section 4 presents our process model forecasting approach, while Sect. 5 presents evaluation setup and results. Section 6 discusses limitations and ideas for future work before Sect. 7 draws final conclusions.

2 Related Work

Predictive Process Monitoring (PPM) studies ways to predict future states, outcomes, and key performance indicators of business processes based on data from event logs [10]. PPM techniques learn historical patterns and then extrapolate the learned principles beyond a given event log. As part of the next process state prediction, PPM techniques can predict the next activity, or groups of activities, that will be performed in a given incomplete business process execution. Existing techniques that tackle this problem achieve high prediction accuracy using conventional statistical and process analysis [18,24,27], and DL [11,12,15,23] methods. Rather than predicting aspects of a currently running business process, process model forecasting (PMF) aims to construct a model that describes future executions from a requested period [29].This fundamental difference between PPM and PMF makes the artifacts they produce not directly comparable.

Our process model forecasting work is inspired by the work by De Smedt et al. [7]. They compared the effectiveness of the statistical time series forecasting techniques for forecasting DF constraints. The forecasting technique proceeds by splitting the event log into equitemporal or equisized periods, calculating frequencies of observed DF constraints for each period, and forecasting each DF constraint using univariate time series forecasting for the series of its frequencies stemming from the different periods of the event log. They evaluated five statistical time series forecasting techniques, namely naïve average (Naïve), auto-regressive integrated moving average (ARIMA) with the order of $(2, 1, 2)$, auto-regressive (AR) with the order of (2), Holt-Winters' model (HW), and generalized auto-regressive conditional heteroskedasticity (GARCH). They then evaluated the mean percentage error in terms of entropic relevance [2] between the ground truth future DFGs and the DFGs assembled from the forecasted constraints. In a follow-up work, they evaluate the vectorized auto-regressive (VAR) model with the order of (1) [8]. In our work, we forecast all DF constraints, and thus the DFG that describes the requested future executions, at once using multivariate time series forecasting and demonstrate that this approach leads to forecasted DFGs of superior accuracy. De Smedt et al. [8] report that techniques that perform the best on average are HW, AR, and the naïve average, while VAR performs worse than other techniques except for one dataset. In our experiments, we replicated ARIMA with the working order of $(1, 1, 1)$, AR with the order of (2), Naïve, and HW as univariate baselines, included VAR with the order of (1) as a multivariate baseline, and further introduced identity function (Identity) as another baseline.

While our techniques demonstrate improved forecasting of DF constraints, their interpretability is an ongoing research. Wil. van der Aalst [26] highlights potential inconsistencies that can arise after filtering DFGs, leading to misinterpretations by analysts. Leemans et al. [19] adapts and discusses the soundness property for DFGs, aiming to ensure their correctness. We propose *consistency* as a new measure of DFGs correctness that evaluates how well the forecasted DF constraints align with human interpretability, focusing on the balanced in-flow and out-flow of arc frequencies across nodes.

3 Preliminaries

This section introduces notions used in the discussions in the subsequent sections.

An *event* is a collection of attribute-value pairs comprising at least three elements storing values of case ID, activity, and timestamp attributes [25]. The case ID, activity, and timestamp attributes of an event specify the instance, or *case*, identifier of the business process that triggered the event, the activity that triggered the event, and the timestamp at which the event was recorded. An *event log*, or a *log*, is a collection of events recorded during the execution of multiple instances, or *cases*, of a business process. An example event log L is shown in Table 1, where each row specifies one event with the attribute values specified in the corresponding columns. The activities that triggered all the events with the same case identifier ordered by the timestamps of the corresponding events constitute a *trace*. Event log L contains two traces: $\langle a, b, c\rangle$ and $\langle a, b, b, d\rangle$.

Table 1. Event log L.

Case ID	*Activity*	*Timestamp*
1	a	1
2	a	2
1	b	3
2	b	4
2	b	5
2	d	6
1	c	7

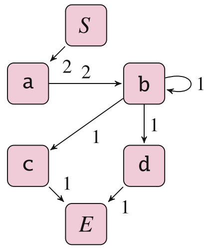


Fig. 1. DFG of L.

	a	b	c	d	E
a	0	2	0	0	0
b	0	1	1	1	0
c	0	0	0	0	1
d	0	0	0	0	1
S	2	0	0	0	0

Fig. 2. DFM of L.

A *Directly-Follows Graph (DFG)* is a process model often constructed from an event log to describe the process that generated the event log [26]. A DFG is also a weighted directed graph with two special nodes denoting the start and end of the process and other nodes annotated with activities. The arcs of a DFG are defined by the DF constraints in the event log, which comprise all pairs of consecutively followed activities in the traces of the event log. For instance, the DF constraints of event log L are defined by the set $\{(S,a),(a,b),(b,b),(b,c),(b,d),(c,E),(d,E)\}$. The start node has no incoming arcs, while the end node has no outgoing arcs. The outgoing arcs of the start node target nodes that represent activities that appear at the start of the traces. The incoming arcs of the end node originate from nodes that represent activities that appear at the end of the traces. In addition, arcs of a DFG are annotated with weights that reflect the frequencies with which the corresponding DF constraints, that is, subsequent executions of the activities, appear in the traces. For instance, the arc (S,a) has a weight of two, as both traces in the event log start with activity a, while arc (a,b) has a weight of two because activity b follows immediately activity a two times in the traces of L. Figure 1 shows the DFG constructed from the traces of event log L.

A *Directly-Follows Matrix (DFM)* is an adjacency matrix used to provide an alternative representation of a DFG. In a DFM, rows represent all the DFG nodes except the end node, while columns represent all the DFG nodes except the start node. Each entry in the matrix specifies the weight of the DFG arc from the corresponding row's node to the corresponding column's node. As the start node has no incoming arcs and the end node has no outgoing arcs, the corresponding column and row are omitted in the DFM. The DFM in Fig. 2 is an alternative representation of the DFG in Fig. 1.

Time Series Forecasting (TSF) studies techniques to predict future values of the time series given the historical data [6]. A time series is a series of discrete data measurements often ordered in regular time intervals. A univariate TSF uses a single time series data to perform forecasts, while a multivariate TSF analyses the dependencies between multiple time series to generate future values in these series simultaneously. Some established statistical univariate TSF approaches include AR and ARIMA. Popular multivariate TSF techniques include VAR and Vectorized ARIMA (VARIMA) [14,20,22].

4 Approach

This section presents the problem of process model forecasting, describes our data selection principle that aims to ensure accurate forecasting, summarizes our forecasting approach, and presents the way we evaluate the results of our forecasts.

4.1 Problem Definition

Let $\mathcal{L}$ be the universe of event logs, let $\mathcal{T}$ be the universe of timestamps, and let $\mathcal{M}$ be the universe of process models. By $\mathcal{P} = \{(s, e) \in \mathcal{T} \times \mathcal{T} \mid e > s\}$ we denote the universe of time periods, where period (s, e) starts at timestamp s and completes at timestamp e.

Given an event log $L \in \mathcal{L}$ and a period $P \in \mathcal{P}$ in the future relevant to L, the *process model forecasting* problem consists in constructing a process model that describes the executions the system that generated L will perform during period P. That is, a solution to process model forecasting can be given as a function $f : \mathcal{L} \times \mathcal{P} \to \mathcal{M}$, such that for each $(L, (s, e)) \in f$ it holds that $\mathit{latest}(L) \leq s$; by $\mathit{latest}(L)$, we refer to the latest timestamp in L, that is, the maximum timestamp value among timestamps of all the events in L. This is different from the classical process discovery problem studied in process mining that aims to construct a process model that describes all the executions of the system that generated the event log can perform in the period $(-\infty, +\infty)$. In this work, we use process discovery outputs (process models) as inputs to forecasting, and thus the quality of the discovery algorithm is crucial to the success of forecasting.

In this work, we study a restricted version of the process model forecasting problem $\hat{f}$ that aims to construct models that describe executions the system will perform in periods immediately after the latest timestamp in the event log, that is, $\hat{f} : \mathcal{L} \times \mathcal{P} \to \mathcal{M}$, such that for each $(L, (s, e)) \in \hat{f}$ it holds that $\mathit{latest}(L) = s$.

4.2 Data Selection

It is unrealistic to assume that every event log can support accurate process model forecasting. Event logs may suffer from issues such as insufficient data, poor quality, inconsistency, or sparsity. For example, forecasts based on empty or minimal logs are likely no better than random guesses. In contrast, large event logs collected over extended periods are more likely to capture critical process features—such as trends, recurring irregularities, and seasonality—thus enabling more reliable and meaningful forecasting.

Over the past two years of iterative experimentation and refinement, guided by the design science methodology, we developed the following three criteria for event logs. These criteria are designed to balance the simplicity of imposed requirements with their strong relationship to forecasting accuracy.

1. *Correctness.* An event log must conform to the designated format (e.g., XES).
2. *Completeness.* Every event in an event log must have valid values for three mandatory attributes: case ID, activity, and timestamp.
3. *Density.* An event log should include a sufficient number of occurrences for events representing different activities across its duration.

We operationalize the correctness and completeness checks for event logs using the Disco tool [13]. Specifically, we check if there are no errors or warnings reported when loading an event log in Disco. To ensure validity, the values of the mandatory event attributes were examined manually. To check the data is sufficient to yield meaningful forecasts, we designed a sparsity test described in Algorithm 1. This test evaluates whether the chosen time window is coarse enough to minimize empty time series DFMs, thereby ensuring the forecasts remain both meaningful and necessary.

Algorithm 1: SparsityTest

Input: M – a non-empty list M of DFMs, each of size $n \times n$, $n \in \mathbb{N}$; it – individual matrix sparsity threshold; tt – total sparsity threshold

Output: Result of the sparsity test for DFMs M

$passed \leftarrow$ false; `/* initialize test result to false */`
$count \leftarrow 0$; `/* number of sparse DFMs */`
for $k \in [1 .. |M|]$ **do** `/* for each position k in M */`
  $sparsity \leftarrow \sum_{i=1}^{n} \sum_{j=1}^{n} \mathbb{I}(M(k)_{ij}) / n^2$; `/* share of zero entries in M(k) */`
  $value \leftarrow \sum_{i=1}^{n} \sum_{j=1}^{n} M(k)_{ij} / n^2$; `/* average value in M(k) */`
  if $sparsity > it \wedge value < it$ **then** `/* if M(k) is sparse ... */`
    $count \leftarrow count + 1$; `/* ... then increment count */`
if $count / |M| < tt$ **then** `/* if share of sparse DFMs below threshold ... */`
  $passed \leftarrow$ true; `/* ... then test passed */`
return $passed$; `/* return test result */`

In the algorithm, $\mathbb{I}(.)$ is the indicator function, such that $\mathbb{I}(x = 0) = 1$; otherwise $\mathbb{I}(x) = 0$. Also, if M is a sequence, then $M(i)$ is the element at position i in M. The algorithm takes a list of DFMs as input. A *time window*, or a *lag*, represents a period of time. We assume that the duration of an event log (the period between the earliest and the latest timestamps of all its events) is split into a number of consecutive time windows, denoted by $\#lag$, each of the same duration. Given an event log and a lag size, we compute a sequence of DFMs, one DFM for each time window, in which DFMs are ordered according to the order of the corresponding time windows. This procedure is sketched in Fig. 3 for a sample event log and $\#lag = 3$; each day defines a time window. The obtained sequence of DFMs is then subject to the sparsity test. The test checks if the number of sparse DFMs in the input sequence is below the total sparsity threshold (tt). A DFM is defined as sparse if the share of its entries that are zeros is above the individual matrix sparsity threshold (it) and the average entry in the DFM is below this threshold. Empirically, we established that DFM sequences that pass the sparsity test for thresholds $it = 0.98$ and $tt = 0.2$ lead to meaningful forecasts.

4.3 Forecasting

We follow a similar approach to process model forecasting as De Smedt et al. [7]. An event log is prepared for training a process forecasting model in the same way as during data selection. Referring to the example in Fig. 3, the event log lasts for three days, and to simplify the example, to split the event log, we use $\#lag = 3$. Hence, each lag contains all the events recorded on a particular day. We then use PM4Py [4] to discover DFGs from (fragments of) traces from each lag and represent these DFGs as DFMs.

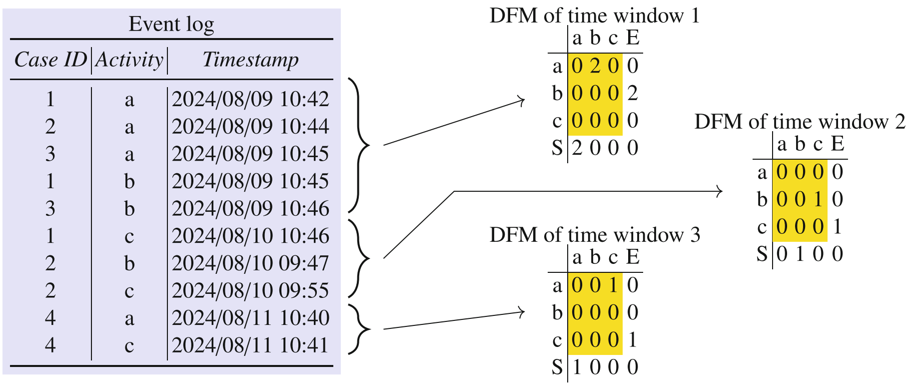


Fig. 3. Event log prepared as DFM time series ($\#lag = 3$).

We use DFM sequences to train DL multivariate time series forecasting models. We identified six DL models used in time series forecasting that can be trained on DFM sequences: Vanilla, Convolutional Neural Network (CNN), Recurrent Neural Network (RNN), Gated Recurrent Unit (GRU), Long Short-

Term Memory (LSTM), and transformer [16,17]. The Vanilla model is a fully connected deep neural network (NN) with a sequence of two linear layers, each followed by a ReLU activation function, and it is concluded with an additional linear layer as the output layer, providing the final predictions without an activation function. The CNN model is similar to the Vanilla model and differs in that it uses 1D convolutional layers instead of linear layers, while it still encompasses an additional linear layer as the output layer. The RNN model is an Elman RNN architecture with two RNN layers and a final linear output layer. The GRU and LSTM models use the same architecture as the RNN model. The transformer model is the default PyTorch transformer model without any additional layers.

To ensure that the comparison between NN models and baselines is fair, in this work, we fixed all models' (both NN models and statistical models) horizon (*#horizon*) and look-back window (*#lookback*) to be equal to one; the same configuration was used by [8]. In time series forecasting, a *horizon* is the number of time windows the method forecasts, while a look-back window defines how many previous time windows are used by the forecasting model to come up with a forecast.

4.4 Testing

We further describe how we split the data into 10 folds to perform a 10-fold cross-validation of our forecasting approach. After constructing the series of DFMs over time, we use these matrices as inputs to train the selected NN models, as well as the baseline models. We first split the ordered time windows from the earliest to the latest into 10 equal chunks. Then, to construct the i-th fold, we take all the DFMs starting from the first DFM in the entire series of DFMs up to and including all the DFMs from the i-th chunk; this is a standard approach for splitting time series data for cross-fold validation. Figure 4 visualizes this splitting process. Since the forecasting horizon we use is set to one, we use the last DFM in each fold as the testing ground truth, and we set the DFM before the testing DFM as our validation DFM. All the other DFMs are used to train

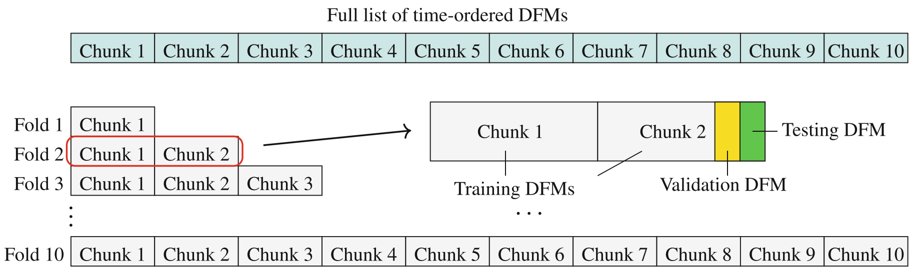


Fig. 4. Time series splitting and data used for training, validation, and testing.

the forecasting NN models. Specifically, for each fold, we train the NN models from scratch and use Optuna [1] to report on the hyperparameter combinations that have the lowest loss on the validation DFM. Since Optuna does not keep the model state, we retrain the model using the reported best hyperparameters after 50 trials of hyperparameter search.

To further improve the accuracy of our forecasts, we also experimented with two approaches: modifying every model architecture by adding a ReLU layer at the end of each model (the transformer model also introduced an additional linear layer for applying the ReLU activation) and post-processing the prediction by taking the maximum frequency between zero and the predicted values.

5 Evaluation

This section presents the datasets used in our evaluation, our implementation of the approach and experimental setup, quality measure used to assess the performance of our process model forecasting techniques, and reports the results of the conducted forecasts.

5.1 Datasets

We explained our data selection steps and criteria in Sect. 4.2 and applied them to all event logs made publicly available by the IEEE Task Force on Process Mining[1]. To increase the number of event logs suitable for analysis, we also manually truncated and filtered some logs that have a long idle period at the beginning or the end of the logging period. We use '_f' to indicate that the log has been truncated and filtered; for the events belonging to the cases outside the included period, we removed those events. During the data preparation stage, we read the event logs and retrieved the earliest and latest timestamps for each dataset. We further sliced the duration of the earliest to the latest timestamp into a number of equal time windows as described in Sect. 4.3. Then, for the events in each period, we use PM4Py [4] with default settings to discover the DFG of that period, and we further turn these DFGs into equivalent DFMs. The obtained lists of DFMs were used as inputs to our sparcity test (Algorithm 1). Ten event logs passed the test. These are Hospital Billing (hb), Road Traffic Fine Management Process (rtfmp), Sepsis Cases (sepsis), BPI Challenge 2017 (bpic17), the help desk log of an Italian company (helpdesk), BPI Challenge 2019 (bpic19), BPI Challenge 2013 Closed Problems (bpic13c), BPI Challenge 2012 (bpic12), NASA Crew Exploration Vehicle Software Event Log (nasacs), and BPI Challenge 2013 Open Problems (bpic13o) event logs. The characteristics of these event logs are summarized in Table 2. Note that not every event log passes the sparsity test for all chosen lag sizes.

[1] https://www.tf-pm.org/resources/logs.

Table 2. Event logs and their characteristics.

Log name	*Events*	*Traces*	*Activities*	*Earliest timestamp*	*Latest timestamp*
hb	451,359	100,000	18	2012-12-13 20:13:18	2016-01-19 18:58:56
rtfmp	561,470	150,370	11	2000-01-01 10:00:00	2013-06-18 08:00:00
sepsis_f	14,766	1,025	16	2013-11-07 18:18:29	2015-02-28 04:00:00
bpic17	1,160,405	31,509	26	2016-01-01 20:51:15	2017-02-02 01:11:03
helpdesk	21,348	4,580	14	2010-01-13 08:40:25	2014-01-03 13:20:58
sepsis	15,214	1,050	16	2013-11-07 18:18:29	2015-06-05 20:25:11
bpic19_f	1,588,420	251,478	11,879	2018-01-01 09:59:00	2019-01-19 00:34:00
bpic13c_f	6,483	1,456	7	2010-01-06 02:42:20	2012-06-01 07:49:06
bpic12	262,200	13,087	36	2011-10-01 08:38:45	2012-03-15 02:04:55
nasacs	36,819	2,566	47	2017-02-14 01:50:52	2017-02-14 01:50:56
bpic13o_f	2,319	812	5	2010-01-14 20:34:54	2012-06-15 20:19:56

5.2 Implementation and Experimental Setup

We split each event log into different numbers of time windows. Specifically, we use *#lag* of 100, 300, 500, 700, and 1,000. All the experiments were conducted on the University of Melbourne supercomputing platform—Spartan. Our experiments were implemented in Python 3.9.19. We configured the Pytorch DL framework in the Anaconda3 (2022.10) environment with CUDA (12.2.0). Table 3 summarizes the platform specification as well as other Python packages and version information.

We used Optuna [1] as a hyperparameter tuning tool. The hyperparameter values we used are listed in Table 3. We fix the other parameters including the number of layers, batch size, and kernel size (for the CNN model) to two, one, and one, respectively. For each NN model, we set Optuna to try 50 trials for each fold, and we picked the hyperparameters that returned the lowest loss on the validation dataset for training the final NN model. For each dataset, the NN models were optimized for 10 folds. The implementation of our experiments, including the data preparation, training, evaluation and result analysis, is publicly available.[2]

5.3 Quality Measures

To evaluate the accuracy of our forecasts, we rely on the commonly used mean absolute error (MAE) measure. Specifically, we measure the errors of the entries in the forecasted DFM ($Forecasted\_DFM$) with respect to the corresponding entries in the ground truth DFM ($Ground\_Truth\_DFM$). We calculate MAE for each fold and average the mean performance of each NN model for the 10 folds for each dataset.

[2] https://github.com/zhoudayun81/DeePMF.

Table 3. Platforms, packages and hyperparameters.

Processor: Intel(R) Xeon(R) Gold 6326 CPU @ 2.90GHz.	
Memory: 1,000GB (only utilised 16GB in our experiments).	**Cores:** 32.
GPU memory: 80GB GPU RAM per GPU.	**GPU type:** A100.

torchaudio: 2.4.0	torchvision: 0.19.0	pm4py: 2.7.11.13	scikit-learn: 1.5.1	optuna: 3.6.1
torch: 2.1.0.dev20230621+cu117		numpy: 1.26.4	pandas: 2.2.2	scipy: 1.13.1

Optimizer	Adam, SGD
Loss function	L1Loss, MSELoss, SmoothL1Loss
Hidden size	121, 196, 256, 324, 1296
Epochs	1000 to 2000
Learning rate	0.001 to 0.01
Dropout probability	0.1 to 0.3

Below, we detail the computation of MAE for an event log with n unique activities:

$$Ground\_Truth\_DFM \in \mathbb{N}^{n \times n}$$

$$Forecasted\_DFM \in \mathbb{Z}^{n \times n}$$

$$\mathrm{MAE} = \frac{1}{(n+1)^2} \sum_{i=1}^{n+1} \sum_{j=1}^{n+1} \left| Ground\_Truth\_DFM_{ij} - Forecasted\_DFM_{ij} \right|$$

We propose a consistency measure of DFG quality that quantifies how well the sum of the frequencies of the arcs entering and leaving its activity nodes match. For the *start* node, which only has outgoing arcs, the frequency should be the same as the frequency of arcs entering the *end* node. We require that a DFG has at least one arc. If a DFG has exactly one arc, this is the arc from its start node to its end node.

We compute *consistency* of a DFG with n activity nodes over its DFM as follows.

$$\mathrm{row\_sum}_i = \sum_{j=1}^{n+1} \max(DFM_{j\,i}, 0) \quad \text{(outgoing sum for node } i\text{)} \tag{1}$$

$$\mathrm{column\_sum}_i = \sum_{j=1}^{n+1} \max(DFM_{ij}, 0) \quad \text{(incoming sum for node } i\text{)} \tag{2}$$

$$\mathrm{Consistency} = \frac{1}{n+1} \sum_{i=1}^{n+1} \frac{\min(\mathrm{row\_sum}_i, \mathrm{column\_sum}_i)}{\max(\mathrm{row\_sum}_i, \mathrm{column\_sum}_i)} \tag{3}$$

Firstly, we calculate the sum of outgoing (Eq. 1) and incoming (Eq. 2) arcs for each node i of the DFG. As a forecasted DFM can, in general, contain negative entries, if a negative value is encountered when computing the sum for a row or column, it is replaced with the value of zero. If the outgoing, as well as the incoming arcs' frequencies, sum up to zeros, to avoid the division by zero problem, the ratio for the node in Eq. 3 is accepted to be equal to one. In the DFG, this is interpreted as the node does not exist in the graph. Consequently, it holds that consistency is a value between zero and one, with larger values signifying a higher degree of consistency.

There are several design options to evaluate the quality of the DFG. The reason for our choice of this measure is twofold. First, the measurement is bounded between zero and one. Second, the measurement values can be compared between different datasets.

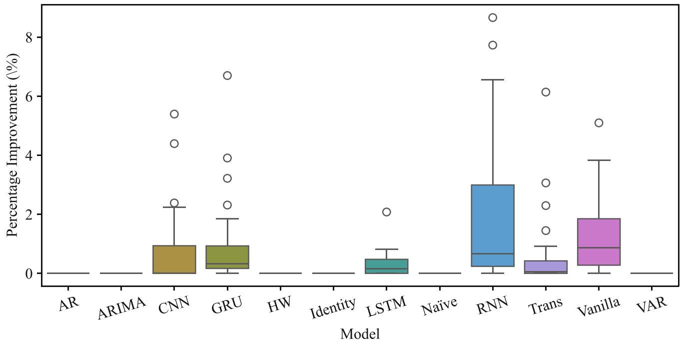


Fig. 5. Percentage improvement of MAE after applying normalization.

5.4 Results

Due to the page limit, we could not fully present the large-scale experiments conducted and the extensive data collected. The interested reader can access the complete experimental outputs and results in our GitHub repository, specifically in the *output* and *result* folders. Several statistical models we explored assume stationarity in the data, a requirement that the majority of the training DFMs do not meet. As a result, some models failed to converge during training. Consequently, we only report results for ARIMA with a modified order $(1, 1, 1)$, AR(2), HW, VAR(1), naïve average (Naïve), and the identity function (Identity). Although we attempted to apply a vectorized ARIMA model (VARIMA), none of the tested orders worked with our datasets.

In Table 4, we report on the mean MAE for each model over 10 folds for each dataset; the number of time windows used to split the dataset is annotated in the brackets next to the dataset name and we bold the best (lowest) values for each dataset. Table 4 provides several insights. First, transformer (Trans) has a superior lower MAE on the majority of the datasets comparing to the baselines, which are mostly more than 50% improvement in MAE comparing to the multivariate baseline VAR. Second, transformer almost consistently performs not good on hb and bpic17 datasets. Third, VAR model yields most of the poorest results and cannot even compete with the simple baselines (Identity and Naïve), hence we do not recommend using VAR for multivariate time series process model forecasting. Finally, when the lag size is small (e.g., $\#lag = 100$), the NN models have less advantage to win over the identity function. The reported results are the best after post-processing the predictions as described in Sect. 4.3. The percentage improvement of each model after post-processing is described in Fig. 5. It is clear that RNN benefits the most from this improvement, which also is an indication that the original RNN model quality can be poor. As we also explored adding a ReLU layer to ensure all the values in the forecasted DFM are positive, it is surprising that this approach deteriorates the forecasting accuracy.

To further analyze the impact of the lag on the forecasting results, we also plot the average rankings for all datasets in different lag groups in Fig 6. Figure 6

Table 4. Average MAE of All Models (rounded to 3rd place decimal).

	DeePMF						*Baselines*					
Dataset	*Trans*	*RNN*	*LSTM*	*GRU*	*CNN*	*Vanilla*	*Identity*	*Naïve*	*VAR*	*ARIMA*	*HW*	*AR*
hb(100)	5.344	4.996	4.153	4.253	15.712	5.788	**3.164**	7.584	10.653	3.188	3.062	3.357
rtfmp(100)	**47.098**	48.581	49.565	53.098	63.303	103.747	73.357	53.288	280.804	54.324	54.276	54.459
sepsis_f(100)	**0.274**	0.304	0.362	0.305	0.309	0.328	0.387	0.325	0.560	0.336	0.328	0.338
bpic17(100)	4.907	4.780	4.485	4.594	10.126	6.075	7.219	6.521	8.414	**4.331**	5.188	4.836
helpdesk(100)	0.683	0.656	0.568	0.612	0.564	0.656	**0.546**	0.784	1.828	0.630	0.613	0.588
sepsis(100)	**0.297**	0.326	0.376	0.352	0.311	0.316	0.359	0.439	0.503	0.335	0.326	0.382
bpic19_f(100)	6.077	**5.733**	6.383	6.328	11.569	10.606	8.761	6.761	12.835	6.265	6.615	5.952
bpic13c_f(100)	**0.602**	0.761	0.809	0.744	0.755	0.777	0.711	1.272	0.941	0.731	0.714	0.730
bpic12(100)	1.432	1.265	1.243	1.420	1.591	1.434	1.511	1.245	2.200	1.225	**1.193**	1.200
nasacs(100)	0.121	0.137	0.134	0.127	0.141	0.135	**0.120**	0.147	0.232	0.125	0.122	0.135
bpic13o_f(100)	**0.642**	0.656	0.664	0.658	1.144	0.847	0.894	1.289	1.041	0.651	0.689	0.770
hb(300)	2.010	1.651	1.580	1.650	1.673	1.683	1.557	2.944	4.237	1.379	**1.358**	1.433
rtfmp(300)	**15.475**	16.617	17.296	16.835	17.656	18.049	24.146	20.801	136.940	18.565	17.549	18.054
sepsis_f(300)	**0.102**	0.161	0.169	0.172	0.173	0.161	0.189	0.165	0.421	0.178	0.174	0.185
bpic17(300)	2.322	2.605	2.817	2.627	2.756	2.767	3.397	2.857	95.430	2.368	**2.292**	2.617
helpdesk(300)	**0.169**	0.312	0.318	0.328	0.262	0.324	0.316	0.339	0.395	0.301	0.297	0.305
sepsis(300)	**0.107**	0.160	0.157	0.158	0.145	0.145	0.178	0.186	0.385	0.164	0.163	0.188
bpic19_f(300)	3.993	**3.535**	3.932	3.590	4.277	4.453	4.913	3.957	1,440.695	3.907	3.994	3.990
bpic13c_f(300)	**0.172**	0.464	0.352	0.394	0.469	0.452	0.650	0.405	0.564	0.422	0.440	0.380
bpic12(300)	**0.454**	0.546	0.482	0.499	0.563	0.565	0.665	0.573	5.596	0.538	0.548	0.520
hb(500)	1.267	1.221	1.110	1.097	1.901	0.925	1.060	1.983	9.989	**0.820**	0.824	0.938
rtfmp(500)	**8.583**	10.252	9.751	10.678	10.367	12.209	17.915	14.020	13.468	11.927	11.387	11.758
sepsis_f(500)	**0.057**	0.110	0.123	0.111	0.101	0.120	0.127	0.115	0.256	0.129	0.127	0.132
bpic17(500)	**1.745**	1.928	1.836	1.929	2.309	2.215	1.958	2.439	54.097	1.831	2.031	2.090
helpdesk(500)	**0.095**	0.221	0.186	0.240	0.177	0.269	0.235	0.238	0.265	0.204	0.198	0.226
sepsis(500)	**0.065**	0.095	0.105	0.107	0.097	0.112	0.136	0.117	0.248	0.113	0.113	0.129
hb(700)	0.781	0.764	0.836	**0.699**	0.907	0.871	0.978	1.554	1.063	0.802	0.806	0.967
rtfmp(700)	**6.852**	8.246	8.838	8.324	8.624	11.401	16.510	11.636	9.358	9.844	9.583	9.870
sepsis_f(700)	**0.034**	0.082	0.088	0.103	0.091	0.096	0.108	0.092	0.171	0.102	0.102	0.106
bpic17(700)	**1.591**	1.631	1.492	1.616	1.941	1.792	1.759	2.261	5.785	1.779	2.065	1.943
helpdesk(700)	**0.063**	0.164	0.164	0.168	0.184	0.180	0.200	0.191	0.196	0.169	0.164	0.186
hb(1000)	0.528	**0.523**	0.708	0.570	0.986	0.633	0.698	1.092	1.035	0.610	0.605	0.641
rtfmp(1000)	**5.458**	6.730	8.117	7.832	7.348	7.109	12.595	9.485	8.221	7.872	8.008	7.680
sepsis_f(1000)	**0.019**	0.064	0.075	0.073	0.068	0.073	0.100	0.072	1.41E+09	0.082	0.083	0.079
bpic17(1000)	1.397	1.442	1.426	**1.276**	1.664	1.916	1.695	1.798	24.706	1.477	1.679	1.617

implies that with the greater number of time windows used for training, the transformer model has better performance, while the univariate baselines, as well as the identity function, lose their advantage with finer time windows. This makes sense and aligns with the results in Table 4, where the majority of the best results in the baselines are from the identity function for smaller lags. This infers that for a greater time span, it may not be suitable to apply time series techniques for process model forecasting, as the nuances are usually hidden by emergent global behaviors. Alternatively, a univariate forecasting model may be sufficient and cost-effective for a smaller lag DFM forecast.

We then ranked the models' performance, calculated the mean ranking and generated the critical difference diagram over all the datasets, refer to Fig. 7. The critical difference diagram was proposed by [9] and further refined by [3]. To compute statistics, we used the default value of alpha of 0.05. Figure 7 shows that transformer and RNN DL models (Elman RNN, LSTM, GRU) perform substantially better than the simple baselines and statistical models and significantly better than the VAR model. Although the critical differences between the *DeePMF* models and the univariate baselines are blended, the actual ranking difference between transformer (2.7) and the best baseline ARIMA (5.4) is significant. Looking back at Fig. 5 and 7, despite transformer receiving little

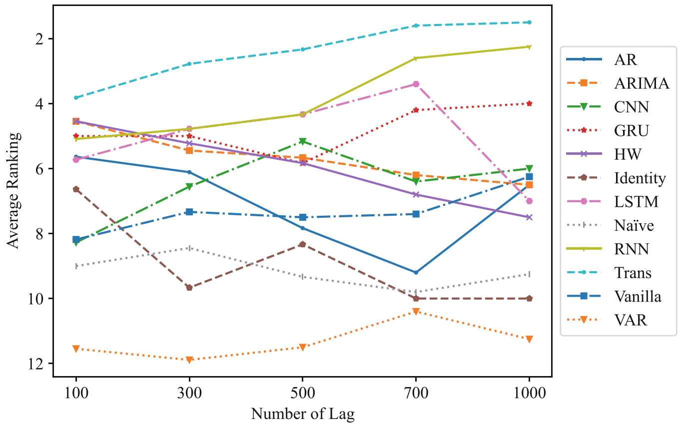


Fig. 6. Average ranking in terms of MAE for each time window size.

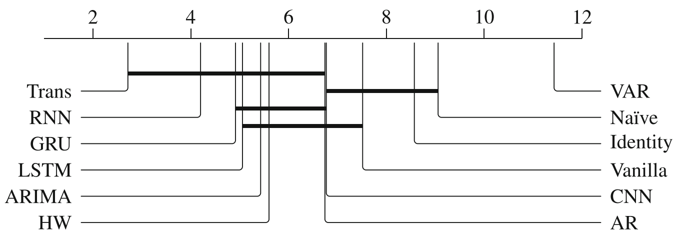


Fig. 7. Critical Difference (CD) diagram for all datasets.

improvement, it still ranks first among all models, which confirms its superiority in terms of the prediction accuracy and the DFM quality.

To evaluate the quality of the interpretation of the forecasted DFMs, we evaluated the consistency of the forecasts. The results are summarized in Table 5. We bolded the best results, underscored the second-best results, and italicized the worst. As Table 5 shows, the identity function has the highest average consistency, and ideally, this score should be exactly one, as every DFM discovered should be perfectly consistent. However, the DFGs discovered during training based on the bpic17 and nasacs datasets are not always consistent. Hence, they contribute to the reduction of the mean consistency in the results. The naïve average approach has mostly the second-best consistency, while the VAR model performs overall the worst again. It is interesting to see that most baselines (except VAR and ARIMA) can achieve better consistency compared to the *DeePMF* models. This could imply that there is a trade-off between accuracy and consistency for the forecasted process models.

6 Discussion

The results have demonstrated that *DeePMF*, mainly based on the transformer architecture, achieves superior accuracy of process model forecasts. The results may be biased as we did not fully explore all the DL model architectures. For example, we fixed the kernel size and stride for CNN and the number of layers for all DL models. It would be interesting to initiate research on each of the DL models and fully explore the architecture's potential for process model forecasting. Our research results can be used as a baseline for such endeavors. For suggestions on hyperparameter selection, unfortunately, we could not find any patterns of the optimal hyperparameters and make a recommendation for their use. As Optuna statistics shows, the best hyperparameter combinations always vary from dataset to fold, and the selected hyperparameters in our experiments seem to be reasonable. Yet, it remains open how much of the differences in the measurements result from choosing the proper configurations and parameters.

The *DeePMF* comes with several natural limitations. Firstly, for training, it requires a reliable process discovery algorithm that can best describe the system behavior for a certain period of time. Secondly, this approach is not able to cater unseen activities as constructed DFMs used for training also fix the number of the possible forecasted activities. Thirdly, it does not guarantee the quality of the forecasted models, where the best consistency score from NN models is around 0.9.

The improved ranking on finer time windows implies that process model forecasting on large numbers of time windows is promising. Due to the scope constraints, we did not fully explore the optimal time span for forecasting accuracy, which is an interesting direction for future work.

In terms of the training time, despite the DL model training being done with high-end GPUs, it can still take hours to days to train the most optimal NN model. With finer time windows, more training samples are available, and one

Table 5. Mean consistency of process model forecasts.

	DeePMF						*Baselines*					
	Trans	*RNN*	*LSTM*	*GRU*	*CNN*	*Vanilla*	*Identity*	*Naïve*	*VAR*	*ARIMA*	*HW*	*AR*
hb(100)	0.910	0.864	0.915	0.858	*0.814*	0.860	**1.000**	0.966	0.893	0.918	0.944	0.907
rtfmp(100)	0.876	0.845	0.866	0.854	0.804	0.841	**1.000**	0.982	*0.778*	0.909	0.896	0.903
sepsis_f(100)	0.885	0.883	0.926	0.921	0.899	0.916	**1.000**	0.884	*0.841*	0.903	0.932	0.890
bpic17(100)	0.944	0.945	0.934	0.947	*0.852*	0.914	**0.979**	0.970	0.898	0.934	0.957	0.939
helpdesk(100)	0.969	0.937	0.958	0.933	0.925	0.919	**1.000**	0.941	*0.879*	0.895	0.923	0.930
sepsis(100)	0.910	0.886	0.950	0.941	0.906	0.920	**1.000**	0.872	*0.835*	0.872	0.926	0.870
bpic19_f(100)	0.898	*0.797*	0.889	0.852	0.872	0.865	**1.000**	0.961	0.834	0.900	0.933	0.897
bpic13c_f(100)	0.913	0.884	0.900	0.928	0.884	0.949	**1.000**	0.920	0.893	*0.790*	0.827	0.885
bpic12(100)	0.897	0.902	0.912	0.891	0.917	0.898	**1.000**	0.946	*0.856*	0.950	0.971	0.945
nasacs(100)	0.747	0.764	0.886	0.846	0.789	0.825	**0.996**	*0.704*	0.732	0.794	0.780	0.770
bpic13o_f(100)	0.902	0.913	0.861	0.858	0.850	0.973	**1.000**	0.882	0.940	*0.845*	0.892	0.866
hb(300)	0.934	0.819	0.888	0.881	0.907	0.884	**1.000**	0.917	*0.813*	0.914	0.928	0.915
rtfmp(300)	0.857	0.857	0.874	0.853	0.866	*0.828*	**1.000**	0.955	0.927	0.922	0.911	0.934
sepsis_f(300)	0.887	0.890	0.911	0.915	0.913	0.921	**1.000**	0.924	*0.818*	0.916	0.959	0.927
bpic17(300)	0.918	0.898	0.910	0.898	0.911	0.928	**0.981**	0.964	*0.702*	0.951	0.942	0.940
helpdesk(300)	0.931	0.909	0.963	*0.901*	0.939	0.946	**1.000**	0.984	0.911	0.908	0.928	0.959
sepsis(300)	0.941	0.909	0.949	0.921	0.946	0.931	**1.000**	0.916	*0.788*	0.891	0.928	0.911
bpic19_f(300)	0.879	0.819	0.827	0.845	0.912	0.884	**1.000**	0.954	*0.688*	0.903	0.950	0.913
bpic13c_f(300)	0.949	0.885	0.860	0.878	0.864	0.896	**1.000**	0.932	0.910	*0.779*	0.857	0.912
bpic12(300)	0.903	0.874	0.918	0.906	0.941	0.940	**1.000**	0.936	*0.804*	0.900	0.967	0.955
hb(500)	0.922	*0.837*	0.923	0.877	0.933	0.860	**1.000**	0.945	0.880	0.918	0.940	0.920
rtfmp(500)	0.880	0.896	0.904	0.887	0.872	*0.836*	**1.000**	0.970	0.912	0.945	0.918	0.956
sepsis_f(500)	0.863	0.870	0.853	0.903	0.876	0.890	**1.000**	0.926	*0.840*	0.935	0.949	0.933
bpic17(500)	0.915	0.922	0.875	0.906	0.894	0.906	**0.980**	0.966	*0.730*	0.943	0.940	0.948
helpdesk(500)	0.946	0.967	0.973	0.969	0.943	0.967	**1.000**	0.993	0.939	*0.895*	0.928	0.956
sepsis(500)	0.912	0.886	0.911	0.940	0.933	0.944	**1.000**	0.919	*0.858*	0.898	0.926	0.923
hb(700)	0.909	0.822	0.904	0.914	0.927	0.852	**1.000**	0.939	*0.819*	0.913	0.943	0.917
rtfmp(700)	0.908	0.877	0.908	0.904	0.921	*0.836*	**1.000**	0.948	0.928	0.949	0.916	0.959
sepsis_f(700)	0.913	0.853	0.870	0.843	0.848	0.858	**1.000**	0.829	*0.817*	0.928	0.952	0.939
bpic17(700)	0.887	*0.869*	0.935	0.909	0.873	0.915	**0.980**	0.957	0.871	0.871	0.937	0.944
helpdesk(700)	0.934	0.951	0.934	0.954	0.963	0.954	**1.000**	0.983	0.980	*0.912*	0.931	0.968
hb(1000)	0.921	0.883	0.897	0.915	*0.841*	0.862	**1.000**	0.932	0.862	0.922	0.932	0.918
rtfmp(1000)	0.907	0.871	0.893	0.912	0.930	*0.837*	**1.000**	0.945	0.925	0.959	0.930	0.968
sepsis_f(1000)	0.876	0.849	0.892	0.864	0.833	0.827	**1.000**	*0.787*	0.867	0.909	0.963	0.929
bpic17(1000)	0.897	0.886	0.924	0.892	*0.881*	0.937	**0.982**	0.941	0.919	0.903	0.937	0.946

observes improvements in forecasting accuracy, but it could also take longer to train the model. It is a dilemma to trade off the forecasting accuracy and the time taken to train an NN model. Note that it can be impractical to use *DeePMF* if the time taken to train a forecasting DL model is the same or even longer as the forecasting horizon. However, if this training time dilemma is addressed, organizations and process analysts may use the prediction outcome to help organizational planning such as resource allocation or process model design in the next BPM cycle.

As another direction for future work, it is interesting to explore the impact of larger look-back windows and forecasting horizons on forecasting accuracy and consistency. The *DeePMF* can either be used recursively to forecast a longer period or adapted to use DL recurrent architectures for generating forecasts for longer horizons. Our approach only uses the event log for forecasting. Correlating the event log with other observations could be worthwhile for many applications, and we leave this as future work.

Finally, it is worth noting the benefits of process model forecasting and its potential applications. BPM lifecycle comprises five stages, namely, business process (re)design, implementation, monitoring, adjustment, and diagnosis [25]. It can take at least six months to over two years for a business model to be implemented and run from its initial design [5]. For such an extended period, the process behaviors may evolve, which, by the time of the implementation and monitoring phases of the lifecycle, could potentially make the following redesign phase obsolete. An accurate process model forecast can support resource allocation and planning. Additionally, it can provide analysts and stakeholders with valuable insights into potential process model changes over time. Finally, by incorporating an early process forecast into the redesign initiative, organizations can proactively account for anticipated process evolutions, ensuring that the redesigned process is implemented by the time the forecasted process changes materialize, ultimately facilitating a smoother transition into future business operations.

7 Conclusion

In this paper, we advanced process model forecasting techniques by leveraging deep neural networks. Our experiments demonstrate that deep neural networks offer greater potential than traditional statistical models, with the transformer architecture achieving the highest overall accuracy. To complement this higher accuracy, we introduced a quality measure to assess the consistency of the predicted process models. We explored two methods to further improve forecasting accuracy, finding that post-processing the forecasts can lead to further improvements. By improving forecasting accuracy and consistency, we aim to provide analysts with more reliable and interpretable results.

References

1. Akiba, T., Sano, S., Yanase, T., Ohta, T., Koyama, M.: Optuna: A next-generation hyperparameter optimization framework. In: KDD, pp. 2623–2631, ACM (2019)
2. Alkhammash, H., Polyvyanyy, A., Moffat, A., García-Bañuelos, L.: Entropic relevance: a mechanism for measuring stochastic process models discovered from event data. Inform. Syst. **107**, 101922 (2022)

3. Benavoli, A., Corani, G., Mangili, F.: Should we really use post-hoc tests based on mean-ranks? J. Mach. Learn. Res. **17**(1), 152161 (2016)
4. Berti, A., van Zelst, S., Schuster, D.: Pm4py: a process mining library for python. Softw. Impacts **17**, 100556 (2023)
5. Bingi, P., Sharma, M.K., Godla, J.K.: Critical issues affecting an ERP implementation. Inf. Syst. Manag. **16**(3), 7–14 (1999)
6. Box, G.E.P., Jenkins, G.M., Reinsel, G.C.: Time Series Analysis: Forecasting and Control. Wiley, 4th edn. (2008)
7. De Smedt, J., Yeshchenko, A., Polyvyanyy, A., De Weerdt, J., Mendling, J.: Process model forecasting using time series analysis of event sequence data. Conceptual Model. 47–61 (2021).
8. De Smedt, J., Yeshchenko, A., Polyvyanyy, A., De Weerdt, J., Mendling, J.: Process model forecasting and change exploration using time series analysis of event sequence data. Data Knowl. Eng. **145**, 102145 (2023)
9. Demšar, J.: Statistical comparisons of classifiers over multiple data sets. J. Mach. Learn. Res. **7**, 1-30 (2006)
10. Di Francescomarino, C., Ghidini, C., Maggi, F.M., Milani, F.: Predictive process monitoring methods: Which one suits me best? In: Business Process Management, pp. 462–479, Springer International Publishing (2018)
11. Di Francescomarino, C., Ghidini, C., Maggi, F.M., Petrucci, G., Yeshchenko, A.: An eye into the future: Leveraging a-priori knowledge in predictive business process monitoring. In: Business Process Management, pp. 252–268, Springer International Publishing (2017)
12. Evermann, J., Rehse, J., Fettke, P.: Predicting process behaviour using deep learning. Decis. Support Syst. **100**, 129–140 (2017)
13. Günther, C., Rozinat, A.: Disco: discover your processes. In: Proceedings of the Demonstration Track of the 10th International Conference on Business Process Management, pp. 40–44, CEUR Workshop Proceedings, CEUR-WS.org (2012)
14. Hamilton, J.D.: Time Series Analysis. Princeton University Press (1994)
15. Jalayer, A., Kahani, M., Beheshti, A., Pourmasoumi, A., Motahari-Nezhad, H.R.: Attention mechanism in predictive business process monitoring. In: 2020 IEEE 24th International Enterprise Distributed Object Computing Conference, pp. 181–186 (2020)
16. Längkvist, M., Karlsson, L., Loutfi, A.: A review of unsupervised feature learning and deep learning for time-series modeling. Pattern Recogn. Lett. **42** (2014)
17. Lara-Benítez, P., Carranza-García, M., Riquelme, J.C.: An experimental review on deep learning architectures for time series forecasting. Int. J. Neural Syst. **31**(03) (2021)
18. Le, M., Gabrys, B., Nauck, D.: A hybrid model for business process event prediction. In: Research and Development in Intelligent Systems XXIX, Springer London (2012)
19. Leemans, S.J., Poppe, E., Wynn, M.T.: Directly follows-based process mining: Exploration & a case study. In: 2019 International Conference on Process Mining, pp. 25–32 (2019)
20. Lütkepohl, H.: New Introduction to Multiple Time Series Analysis. Springer (2005)
21. Poll, R., Polyvyanyy, A., Rosemann, M., Röglinger, M., Rupprecht, L.: Process forecasting: Towards proactive business process management. In: Business Process Management, pp. 496–512, Springer International Publishing (2018)
22. Reinsel, G.C.: Elements of Multivariate Time Series Analysis. Springer, 2nd edn. (2003)

23. Tax, N., Verenich, I., La Rosa, M., Dumas, M.: Predictive Business Process Monitoring with LSTM Neural Networks. In: Advanced Information Systems Engineering, pp. 477–492 (2017).
24. Tschumitschew, K., Nauck, D., Klawonn, F.: A classification algorithm for process sequences based on markov chains and bayesian networks. In: Knowledge-Based and Intelligent Information and Engineering Systems, pp. 141–147, Springer Berlin Heidelberg (2010)
25. van der Aalst, W.: Process Mining—Data Science in Action. Springer, 2nd edn. (2016)
26. van der Aalst, W.: A practitioner's guide to process mining: Limitations of the directly-follows graph. Proc. Comput. Sci. **164**, 321–328 (2019)
27. van der Aalst, W., Schonenberg, M., Song, M.: Time prediction based on process mining. Inf. Syst. **36**(2), 450–475 (2011)
28. Weske, M.: Business Process Management: Concepts, Languages, Architectures. Springer, 3rd edn. (2019)
29. Zhou, W., Polyvyanyy, A., Bailey, J.: Event data and process model forecasting. In: CAiSE Forum, LNBIP, vol. 520, pp. 3–10, Springer (2024)

Author Index

J. Krogstie et al. (Eds.): CAiSE 2025, LNCS 15701, pp. 313–315, 2025.
https://doi.org/10.1007/978-3-031-94569-4