Johannes Manner
Daniel Lübke
Stephan Haarmann
Stefan Kolb
Nico Herzberg
Oliver Kopp

ZEUS 2022

14^{th} ZEUS Workshop, ZEUS 2022,
Bamberg, Germany, 24–25 February 2022
Proceedings

Volume Editors

Johannes Manner
University of Bamberg, Distributed Systems Group
An der Weberei 5, DE-96049 Bamberg
johannes.manner@uni-bamberg.de

Daniel Lübke
Digital Solution Architecture

Stephan Haarmann
Hasso Plattner Institute, Business Process Technology
Prof.-Dr.-Helmert-Str. 2-3, DE-14482 Potsdam
stephan.haarmann@hpi.de

Stefan Kolb
JabRef Research
stefan.kolb@jabref.org

Nico Herzberg
Campeleon GmbH
Gostritzer Str. 61, 01217 Dresden

Oliver Kopp
JabRef Research
oliver.kopp@jabref.org

Preface

In February 2022, we have the pleasure to organize the 14th edition of the ZEUS Workshop planned in Hannover, Germany. Due to the ongoing covid-19 pandemic, the workshop is held virtually, giving us the chance to also invite our PC members which are one of ZEUS' success factors. Thanks a lot for your reviewing work and the ongoing support.

This workshop series offers young researchers an opportunity to present and discuss early ideas and work in progress as well as to establish contacts among young researchers. For this year's edition, we selected all twelve submissions for presentation at the workshop. Each submission went through a thorough peer-review process and was assessed by at least four members of the program committee with regard to its relevance and scientific quality. The accepted contributions cover the areas of Business Process Management, Cloud Computing, Microservices, Software Design, and the Internet of Things.

The workshop was generously sponsored by Camunda Services GmbH.

Bamberg, February 2022

Johannes Manner
Daniel Lübke
Stephan Haarmann
Stefan Kolb
Nico Herzberg
Oliver Kopp

Organization

Steering Committee

Nico Herzberg	Campeleon GmbH
Oliver Kopp	JabRef Research
Stefan Kolb	JabRef Research
Stephan Haarmann	Hasso Plattner Institute, University of Potsdam
Johannes Manner	University of Bamberg

Local Organizer

Daniel Lübke	Digital Solution Architecture

Web Chair

Robin Lichtenthäler	University of Bamberg
Sebastian Böhm	University of Bamberg

Program Committee Chairs

Stephan Haarmann	Hasso Plattner Institute, University of Potsdam
Johannes Manner	University of Bamberg

Program Committee

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

Sponsoring Institutions

Camunda Services GmbH

Table of Contents

Towards an Experiment for Analyzing Subprocess Navigation in BPMN Tooling

Daniel Lübke[1,2][https://orcid.org/0000-0002-1557-8804] and Maike Ahrens[2][https://orcid.org/0000-0002-9577-0598]

[1] Digital Solution Architecture, Hanover, Germany
[2] Leibniz Universität Hannover, Germany
daniel.luebke@digital-solution-architecture.com
{daniel.luebke,maike.ahrens}@inf.uni-hannover.de
https://www.digital-solution-architecture.com

Abstract. Complex BPMN models can be decomposed vertically by using collapsed sub-processes and call activities. However, tool support to ease modelers and model readers with the task of following the links between such models is implemented differently in modeling tools and it is unclear which variant is the best. Thus, the primary objective of the planned study is to understand strengths and weaknesses of different modeling support in tools and its implications on model comprehensibility. We analyzed modeling tools for different navigation options and found three different ways of support for modeling users. Based on those findings we designed an experiment for an eyetracking study, which analyzes the usability of the different implementation variants.

Keywords: BPMN · Understandability · Subprocess Navigation · Experiment

1 Motivation

BPMN is the lingua franca for business process modeling. For serving as a communication medium best, models must be as comprehensible as possible for their stakeholders. Especially the larger models get, the less understandable they become. There has been much research into BPMN understandability lately. This includes two experiments in the area of BPMN layout that result in conflicting advice for designers of large BPMN models: A study that compares the use of diagrams with subprocesses vs. flat diagrams by Turetken et al. [6, 7] and an eye-tracking study comparing different layouts by Lübke et al. [4]:

While Turetken et al. found that subprocesses actually make it more difficult for model readers to work with diagrams and lower understandability of the models significantly, Lübke et al. found that diagrams that are too large to fit on a single page reduce understandability. To overcome this diagrams can be laid out to use more screen estate (e.g., multiline and snake layouts). However, this strategy takes only so far – larger diagrams must be partitioned horizontally by using link elements or vertically by using collapsed subprocesses.

J. Manner, D. Lübke, S. Haarmann, S. Kolb, N. Herzberg, O. Kopp (Eds.): 14th ZEUS Workshop, ZEUS 2022, Bamberg, held virtually due to Covid-19 pandemic, Germany, 24–25 February 2022, published at http://ceur-ws.org

2 Daniel Lübke and Maike Ahrens

Thus, the question arises how to structure large diagrams and which existing modeling and layout options come with the least penalties. Before this question can be answered, it needs to be established how to most efficiently navigate subprocesses in BPMN. While Turetken et al. used both paper and on-screen diagrams, we focus on screen-reading because – especially with executable BPMN models – modeling is done in a modeling tool and not on paper.

While there have been studies into BPMN tooling, e.g., regarding their standards compliance [3], and into the comprehensiblity of BPMN models in general (e.g., with modular process models [1, 8]), to the best of our knowledge the question of usability of subporcess navigation has not been researched yet. As such, we have analyzed different tools with regard to the navigation options offered to modelers for navigating process hierarchies with collapsed subprocesses. Based on the identified options we propose an eye-tracking experiment to better understand the impact of the different navigation options offered by modeling tools on BPMN understandability.

2 Experimental Design

Our goal of our proposed experiment, according to the GQM (Goal-Question-Metric) approach as adopted by Nick & Tautz for research [5], will be:

> For the purpose of *understanding the effect of different modeling tooling*
> with regard to the quality aspect of *understandability*
> of the object of *a large BPMN diagram decomposed with collapsed subprocesses*
> from the viewpoint of *a reader of that model.*

Prior to designing the expeirment we analyzed existing modeling tools for their implementation of navigating collapsed subprocesses. We could identify three navigation strategies, which we will use as different treatments in our experiment as shown in Fig. 1:

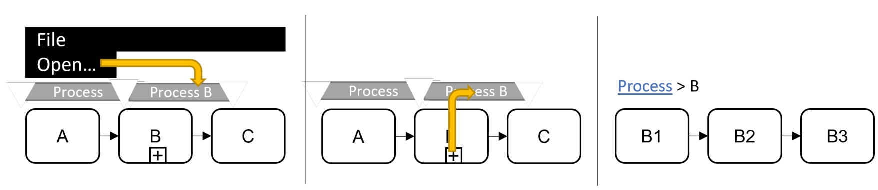


Fig. 1. Different Tool Implementation Choices: a) No linking of Subprocess, b) Link for Opening the Subprocess in a new Tab, and c) Bread Crumb Navigation

No Support (A): Users need to open a new model in a new tab, which requires them to know which model to open, where it is saved etc. We found this, for example, in Camunda Modeler and Enterprise Architect.

Subprocess Symbol Link (B): Users can link and jump to a subprocess model by clicking on the (+) icon in the collapsed subprocess. The new process is then opened in a new tab. We found this in Signavio Academic Edition.

Breadcrumb Navigation (C): Users can also click on the (+) icon but the opened model is shown in the same editor window. On top of the window a path is shown where the user is currently located. We found this in ActiveVOS Designer.

We decompose the overall research goal based on the identified navigation methods in modeling tools into the following research questions:

- **RQ1:** How does different tool suppport implementations influence speed of the users to navigate subprocess hierarchies?
- **RQ2:** How does different tool suppport implementations influence efficiency of the users to navigate subprocess hierarchies?
- **RQ2:** How does different tool suppport implementations influence cognitive load of the users to navigate subprocess hierarchies?

3 Planned Execution & Analysis

Our experiment will compare the three identified tool implementations of subprocess navigation. This is the only independent variable. To eliminate any other influencing factor, we will not use different existing tools in our experiment but will implement BPMN viewers each supporting one of the three navigation methods only.

As a process model we use a large process model of the industrial project Terravis, which is a Swiss large-scale process integration platform for end-to-end integration of land register processes [2]. The main process model is refined via several layers comprising 33 process models in the hierarchy.

We want to recruit both students and professional software developers & process designers. Students are recruited in our lectures. For recruiting professionals we want to offer a one day training in specialized BPMN topics, e.g., testing, understandability of models etc., in exchange for the participation in the experiment.

Each participant will be randomly assigned to one of three groups: one group for each tool implementation option. As part of the experiment each participant is asked to answer four questions regarding the business process model: Two questions can be answered by looking at a single BPMN diagram within the hierarchy while two other questions can only be answered with information contained in different parts of the process hierarchy. By recording the participant behavior with eye tracking, we can also determine differences in terms of visual effort and "gaze on target"ratios in addition to comprehensibility.

4 Daniel Lübke and Maike Ahrens

We want to measure and evaluate the following metrics for each experiment group, which serve as the dependent variables:

- Speed: Time to Answer Questions, Time for Navigating the Process Hierarchy, Number of Clicks used to navigate the process hierarchy
- Efficiency: Questions Correctly Answered, Questions Incorrectly Answered, Task Efficiency (Correct Answers in Time), Number of Correct Navigations in Process Hierarchy, Number of Incorrect Navigations in Process Hierarchy
- Cognitive Load: Average Fixation Duration, Number of Fixations, Dwell Time on BPMN Elements relevant to Question, Dwell Time on BPMN Elements irrelevant to Question, Pupil Diameter Size

These metrics will be tested for significant differences in means between the different implementation options. Because there are three levels for the independent variable, ANOVA will be used for testing differences in means of depedent variables.

For achieving the envisioned power of 0.8 for hypothesis testing, we require 3 groups of 37 participants each, to detect a difference in means with an effect size of 0.3 and, a significance level of 5%.

4 Conclusions & Outlook

Within this paper we have outlined an experiment for analyzing different tool implementation options in BPMN tools to support users navigating process hierarchies. The next steps will be to set up the experiment, which includes develop modeling tools to be used in the experiment, recruit participants, and develop the questions to be asked in the experiment) .When the COVID pandemic permits we will start recruiting both students in on-site lectures and reach out to software development companies for recruitment of professionals. If you are interested in participating in this experiment please contact us.

Bibliography

1. Amine Abbad Andaloussi, Pnina Soffer, Tijs Slaats, Andrea Burattin, and Barbara Weber. The impact of modularization on the understandability of declarative process models: a research model. In *NeuroIS Retreat*, pages 133–144. Springer, 2020.
2. Walter Berli, Daniel Lübke, and Werner Möckli. Terravis – large scale business process integration between public and private partners. In Erhard Plödereder, Lars Grunske, Eric Schneider, and Dominik Ull, editors, *Lecture Notes in Informatics (LNI), Proceedings INFORMATIK 2014*, volume P-232, pages 1075–1090. Gesellschaft für Informatik e.V., Gesellschaft für Informatik e.V., 2014.
3. Udo Kannengiesser. Evaluation of bpmn tools. *National Information Communications Technology Australia*, 2007.
4. Daniel Lübke, Maike Ahrens, and Kurt Schneider. Influence of diagram layout and scrolling on understandability of BPMN processes: an eye tracking experiment with BPMN diagrams. *Information Technology and Management*, 2021.
5. Markus Nick and Carsten Tautz. Practical evaluation of an organizational memory using the goal-question-metric technique. In *German Conference on Knowledge-Based Systems*, pages 138–147. Springer, 1999.
6. Oktay Turetken, Ahmet Dikici, Irene Vanderfeesten, Tessa Rompen, and Onur Demirors. The influence of using collapsed sub-processes and groups on the understandability of business process models. *Business & Information Systems Engineering*, pages 1–21, 2019.
7. Oktay Turetken, Tessa Rompen, Irene Vanderfeesten, Ahmet Dikici, and Jan van Moll. The effect of modularity representation and presentation medium on the understandability of business process models in bpmn. In *International Conference on Business Process Management*, pages 289–307. Springer, 2016.
8. Michael Winter, Rüdiger Pryss, Thomas Probst, Julia Baß, and Manfred Reichert. Measuring the cognitive complexity in the comprehension of modular process models. *IEEE Transactions on Cognitive and Developmental Systems*, 2021.

Towards a Framework for Business Process Sustainability Analysis

Finn Klessascheck

Hasso Plattner Institute (HPI), University of Potsdam, Potsdam, Germany
finn.klessascheck@student.hpi.de

Abstract. The assessment and improvement of business processes is an important driver of success in organizations. However, as opposed to established KPIs and other metrics, the sustainability of business processes is much less straightforward to measure and quantify, partly due to the term's ambiguity and an inherent difficulty to be measured. In order to facilitate sustainability-oriented process redesign beyond greenhouse gasses, existing methods can be enriched by considering additional information from methods such as Life Cycle Assessment. This enables a holistic and flexible analysis, and can serve as a measurable driver for process redesign.

Keywords: Sustainability · Business Process Management · KPI.

1 Introduction

It is undeniable that climate change, driven by human influence, has a severe impact on the world surrounding us. In order to avert even further alterations with potentially catastrophic and unforeseeable consequences, actions have to be taken. In its most recent report, the International Panel on Climate Change states that limiting the emission of CO_2 and other greenhouse gasses to at least net-zero would be required to curb the extent of climate change [8]. Furthermore, toxic substances introduced into the environment as a result of, for example, wasteful manufacturing, play a significant role in the endangerment of biodiversity and the promotion of risks to human health [12].

Industry and academia have reached a consensus that both a reduction of emission of greenhouse gasses and an overall promotion of sustainability hold the potential of mitigating or dampening the consequences of climate change. Business Process Management (c.f. [16]), dealing with analysis, design, and implementation of business processes, has led to various approaches for analysing and improving the sustainability of these business processes, but these approaches generally limit themselves to a few aspects which they assess (e.g., greenhouse gas emissions or energy consumption), and a holistic approach has not yet been established. In the following, related work and the concept of sustainability will briefly be discussed, and a method that aims at alleviating some existing drawbacks by combining additional perspectives with business process simulation will be outlined.

J. Manner, D. Lübke, S. Haarmann, S. Kolb, N. Herzberg, O. Kopp (Eds.): 14th ZEUS Workshop, ZEUS 2022, Bamberg, held virtually due to Covid-19 pandemic, Germany, 24–25 February 2022, published at http://ceur-ws.org

2 Related Work

In the recent years, three main ways of integrating aspects of sustainability into the toolset of BPM have emerged:

1. **Activity-based costing, ABC** — Methods which aim at measuring the impact of a business process based on the individual cost of activities in terms of, most commonly, greenhouse gasses, derived from a priori knowledge of what cost an activity incurs. Activity-based emission analysis (ABE) (c.f. [10, 11]) focusses on emissions according to the different scopes of the GHG Protocol[1], a set of standards for greenhouse gas accounting for businesses and governments.
2. **Structure-based costing** — Methods which aim at measuring a process-level degree of sustainability by checking whether specific patterns that contribute to sustainability in the domain of that process are followed [6, 7].
3. **Modelling concepts** — Methods which aim at enabling the expression and modelling of a processes carbon footprint and greenhouse gas emissions [11].

Generally, all these approaches operate under a shared definition of sustainability: $Sustainability = environmental + social + economic\ sustainability$ (c.f. [15, 18]). As per [1], the economic facet deals with controlled growth, the social facet with inter- and intra-generational justice, and the environmental facet with preservation of the natural basis of life and its lifecycle. Figure 1 illustrates the interrelationship between the three facets.

The focus here, however, generally lies on environmental sustainability, and in that, only on greenhouse gasses according to the GHG Protocol [6, 10, 17] or energy consumption [13]. Nonetheless, it should be noted that the aspect of environmental sustainability is concerned with the preservation of the natural basis of life and its lifecycles, and the security of the ecological conditions of human survival in general (c.f. [1, 5]). Here, it should be stressed that not just greenhouse gas emissions have a negative impact in that regard (c.f. [12], and therefore more factors should be considered in such an environmental sustainability analysis – be it the amount of toxic materials involved, or amount of waste generated, or the general detrimental impact the involved materials have on the environment. Furthermore, all three ways strongly constrain themselves to a specific domain, or even a specific process, in order to validly assess the sustainability. Additionally, activity-based costing is predominantly based on pre-determined costing measures, which assign costs to activities derived from

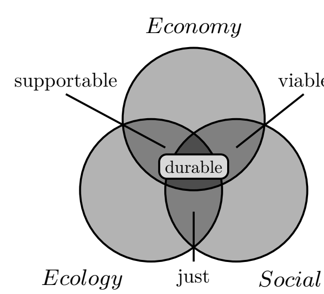


Fig. 1. Triad model of sustainability, adapted from [9]

[1] https://ghgprotocol.org/

certain measures, e.g., the amount of CO_2 produced per page of paper, and the average no. of sheets of paper involved in that activity. However, more than just paper might be involved in that activity in a greenhouse gas-causing fashion, and the degree of sustainability might be influenced more heavily by outliers, not allowing the aggregation using the average. Moreover, to what degree certain patterns contribute to the sustainability of a process highly depends on estimates as well, where the influence some factors have over others is never assessed but just estimated in order determine the more preferable patterns (c.f. [4, 6]).

3 Research Objective

The overall situation leads me to pose the following research questions:

1. How can the current understanding of sustainability be adapted to provide a holistic picture in conjunction with a processes-level view?
2. How can ABC/ABE methods be extended to include other factors beyond greenhouse gas emissions according to a holistic understanding of sustainability?
3. How can this extension be leveraged in a practical setting, e.g., in order to drive process redesign, and how can this be implemented?

In order to establish a holistic approach that enables a quantitative analysis of business processes w.r.t. sustainability, I aim to extend existing activity-based costing approaches by considering additional data, and leverage process simulation to arrive at a more accurate assessment.

In product design, the Life Cycle Assessment (LCA) method has been established in order to assess the environmental impact of different materials and products in a holistic fashion [2]. Using LCA, the impact of products or processes according to several measures can be determined and expressed in terms of numerical scores. Such an LCA methodology is well-suited to enrich existing sustainability analysis methods based on greenhouse gas-focussed activity-based costing, and provides a useful and actionable contribution when applied in a process-level setting. In detail, each activity can be assessed individually based on the LCA method and, for example, the relevant materials or resources involved in the execution. Additionally, based on the evaluation and scores of all activities, a measure indicating the overall impact of the process and its instances and variants can be determined. Here, the impact of activities and the process itself over the course of multiple process executions with different activity and process configurations can be assessed by using process simulation. Both individual and overall scores can then be leveraged to enable process redesign with a focus on improving sustainability and decreasing the environmental impact. It might also be feasible, in future work, to combine this approach with others that aim to optimize different metrics such as performance, service quality, or alignment with certain incentives (e.g., [3, 14]), to allow process redesign w.r.t. sustainability while maintaining other desirable properties of the process. An interesting question could also be how these different perspectives should be prioritized and reconciled with each other.

4 Conclusion

This position paper discusses the need for facilitating a holistic analysis of business process sustainability than allowed by the existing works. With a combination of activity-based costing methods and data elicited through Life Cycle Assessment methods, a clearer understanding can be reached. This understanding can then be used to re-design processes with sustainability as the primary motivator.

References

1. Bauer, S.: Leitbild der Nachhaltigen Entwicklung | bpb. Bundeszentrale für politische Bildung (2008-05-06), https://www.bpb.de/izpb/8983/leitbild-der-nachhaltigen-entwicklung?p=all
2. Finnveden, G., Hauschild, M.Z., Ekvall, T., Guinée, J., Heijungs, R., Hellweg, S., Koehler, A., Pennington, D., Suh, S.: Recent developments in Life Cycle Assessment. Journal of Environmental Management 91(1), 1–21 (2009)
3. Heindel, T., Weber, I.: Incentive Alignment of Business Processes. vol. 12168, pp. 93–110. Springer, Cham (2020), https://link.springer.com/chapter/10.1007/978-3-030-58666-9_6
4. Houy, C., Reiter, M., Fettke, P., Loos, P., Hoesch-Klohe, K., Ghose, A.: Advancing Business Process Technology for Humanity: Opportunities and Challenges of Green BPM for Sustainable Business Activities. In: Green Business Process Management, pp. 75–92. Springer (2012), https://link.springer.com/chapter/10.1007/978-3-642-27488-6_5
5. Johnston, P., Everard, M., Santillo, D., Robèrt, K.H.: Reclaiming the definition of sustainability. Environmental science and pollution research international 14(1), 60–66 (2007)
6. Lübbecke, P., Fettke, P., Loos, P.: Towards guidelines of modeling for ecology-aware process design. In: International Conference on Business Process Management. pp. 510–519 (2017)
7. Lübbecke, P., Goswami, A., Fettke, P.: A method for ecological process optimization based on compliance checking. In: 2018 IEEE 20th Conference on Business Informatics (CBI). vol. 1, pp. 119–128 (2018)
8. Masson-Delmotte, V., P. Zhai, A. Pirani, S. L. Connors, C. Péan, S. Berger, N. Caud, Y. Chen, L. Goldfarb, M. I. Gomis, M. Huang, K. Leitzell, E. Lonnoy, J. B. R. Matthews, T. K. Maycock, T. Waterfield, O. Yelekçi, R. Yu, B. Zhou: IPCC Climate Change 2021: The Physical Science Basis. Cambridge University Press (2021)
9. Pufé, I.: Was ist Nachhaltigkeit? Dimensionen und Chancen. Aus Politik und Zeitgeschichte, 64. Jg. pp. 15–20 (2014), https://bpb.de/system/files/dokument_pdf/APuZ_2014-31-32_online.pdf
10. Recker, J., Rosemann, M., Gohar, E.R.: Measuring the carbon footprint of business processes. In: International Conference on Business Process Management. pp. 511–520 (2010)
11. Recker, J., Rosemann, M., Hjalmarsson, A., Lind, M.: Modeling and analyzing the carbon footprint of business processes. In: Vom Brocke, J., Seidel, S., Recker, J. (eds.) Green Business Process Management, pp. 93–109. Springer (2012)
12. Romanelli, C., Cooper, D., Campbell-Lendrum, D., Maiero, M., Karesh, W.B., Hunter, D., Golden, C.D.: Connecting global priorities: biodiversity and human health: a state of knowledge review. World Health Organistion / Secretariat of the UN Convention on Biological Diversity (2015), https://cgspace.cgiar.org/handle/10568/67397

13. Roohy Gohar, S., Indulska, M.: Environmental Sustainability through Green Business Process Management. Australasian Journal of Information Systems 24 (2020)
14. Saeedi, K., Zhao, L., Sampaio, P.R.F.: Extending BPMN for Supporting Customer-Facing Service Quality Requirements. In: 2010 IEEE International Conference on Web Services. pp. 616–623 (2010)
15. Silvius, G.: Change the game: sustainability in projects and project management. In: Vom Brocke, J., Seidel, S., Recker, J. (eds.) Green Business Process Management, pp. 161–177. Springer (2012)
16. Weske, M.: Business Process Management - Concepts, Languages, Architectures, 2nd Edition. Springer (2012)
17. Wesumperuma, A., Ginige, A., Ginige, J.A., Hol, A.: Green activity based management (ABM) for organisations. Information Systems: Transforming the future: Proceedings of the 24th Australasian Conference on Information Systems, 4-6 December 2013, Melbourne, Vic., Australia (2013)
18. Zeise, N., Link, M., Ortner, E.: Measurement systems for sustainability. In: Vom Brocke, J., Seidel, S., Recker, J. (eds.) Green Business Process Management, pp. 131–146. Springer (2012)

All links were last followed on January 20, 2022.

Towards Retrograde Process Analysis in Running Legacy Applications

Marius Breitmayer, Lisa Arnold and Manfred Reichert

Institute of Databases and Information Systems, Ulm University, Germany
{marius.breitmayer,lisa.arnold,manfred.reichert}@uni-ulm.de

Abstract. Process mining algorithms are highly dependent on the existence and quality of event logs. In many cases, however, software systems (e.g., legacy systems) do not leverage workflow engines capable of producing high-quality event logs for process mining algorithms. As a result, the application of process mining algorithms is drastically hampered for such legacy systems. The generation of suitable event data from running legacy software systems, therefore, would foster approaches such as process mining, data-based process documentation, and process-oriented software migration of legacy systems. This paper discusses the need for dedicated event log generation approaches in this context.

Keywords: legacy systems, process mining, code analysis, event log

1 Introduction

Software applications are implemented to address the needs of users, use cases, and business processes. However, the majority of common software systems (e.g., legacy systems or individual software solutions) have not been designed with the goal to provide high-quality process-related event logs that allow for comprehensive process analyses and visualizations with modern process mining tools. Relevant questions emerging in legacy software modernization projects include, for example, how the process implemented by the legacy software system is structured (*Process Discovery*) or to what extent its execution deviates from a predefined to-be process (*Conformance Checking*). Currently, there exist three basic approaches to obtain process models:

1. **Log analysis** uses existing logs (e.g., event logs) to reconstruct the implemented process based on audit or workflow data. Consequently, the quality of the resulting process model is directly correlated with both the existence and quality of corresponding event logs [2,3]. However, a vast majority of individual applications and legacy systems are often unable to provide appropriate event logs. Moreover, even database-centric applications typically do not provide transaction-level audit data. Consequently, there has been no effective entry point for process mining yet.
2. **Interviews** may be conducted to discover the desired process model as perceived by key users and process owners [9]. Additionally, data models may be parsed to identify effects of processes on corresponding data. Analyzing such data models enables assumptions on the underlying processes.

J. Manner, D. Lübke, S. Haarmann, S. Kolb, N. Herzberg, O. Kopp (Eds.): 14^{th} ZEUS Workshop, ZEUS 2022, Bamberg, held virtually due to Covid-19 pandemic, Germany, 24–25 February 2022, published at http://ceur-ws.org

This approach, however, is very time consuming and paved with both misunderstandings and misconceptions. In addition, interviews do not ensure completeness of the relevant processes and their various aspects, as they often neglect exceptions or specific process perspectives (e.g., data, time).
3. **Pattern recognition** attempts to identify typical process patterns in various data pools using algorithms from the field of artificial intelligence [1]. The algorithms require a deep analysis and learning phase prior to their application to the raw data. This is a time-consuming, cost-intensive, and fuzzy approach, which is therefore hardly pursued.

In the context of legacy systems, however, none of the presented approaches is easily applicable. All three approaches have in common that the business processes (and event logs), implemented by the legacy software systems, need to be represented accurately. Since most individual software solutions do not necessarily use process engines capable of delivering suitable process data, alternative approaches are required. One approach to tackle this challenge is, to observe process participants during process execution and to record their interactions with the software system resulting in a fine-grained documentation.

Section 2 describes the proposed solution approach. Section 3 discusses related work. Finally, Section 4 provides a summary and outlook.

2 Solution Approach

A human-centered business process can be defined as a sequence of user interactions with a software application, where each interaction is subject-bound (i.e., part of the same transaction). In legacy systems, such processes can be initiated and terminated by suitable actions (e.g., pre-defined key combinations or menu items). Adding such actions to an event stream with the associated application object (e.g., an order identified by its unique order number), subsequently, process mining tools will have process related event logs as input. The collected event data may then constitute the basis for a plethora of use cases, such as process documentation, process mining, and process-oriented cost estimations for modernizing legacy software systems (i.e., software migration). We aim to create different logging variants for existing legacy production systems:

1. **Dedicated recording** documents existing processes by assigning related program components. Users may determine the start and end of the recording using predefined key combinations, thus precisely delimiting all activities that constitute the recorded process (or the considered process part).
2. **Silent recording** tracks the entire usage of the application from the first login until closing the application. A decision can be made as to whether this should be done for all sessions or only for selected user sessions (e.g., only sessions of users from a certain department). Furthermore, it may be configured, which information should be stored (e.g., to ensure compliance with data protection requirements).

To minimize the performance effects of these recording on running applications, we rely on existing logging mechanisms of the application infrastructure.

For Oracle applications using a WebLogic Server, for example, *Oracle Diagnostic Logging* (ODL) offers extensive possibilities to manage application information via the administration console. Among others, oracle logger classes (e.g., *Application Development Framework*) may use this information through ODL handlers [15]. In Single Page Applications (e.g., the Oracle JavaScript Extension Toolkit JET), the primary object is known, however, the context between multiple process steps may get lost due to the loose coupling of user sessions and services. Even applications based on Oracles Forms allow adding appropriate message calls for each PL/SQL unit.

Using existing system logging functionality, the recording quality is significantly increased compared to purely mining the data model, as user interactions can be unambiguously linked to the process, program code, and associated data.

Fig. 1 depicts the approach. In a first step we identify relevant objects using information from the database and the source code of the application. However, especially in databases of legacy systems, assumptions such as good normalization or even the existence of foreign key constraints are often not applicable. The reason for this is that in many cases the logic is represented in the source code of applications rather than the database. By combining knowledge from the database (e.g., create, read, update, and delete -operations) and corresponding source code (e.g., code fragments corresponding to such operations), we are able to tackle this issue. After having identified process-relevant objects in both source code and database, we correlate them and add code tracking capabilities to the legacy system using, for example, the possibilities mentioned previously. This does then enable the generation of event logs from either dedicated or silent recording. These event logs may then be used during analysis.

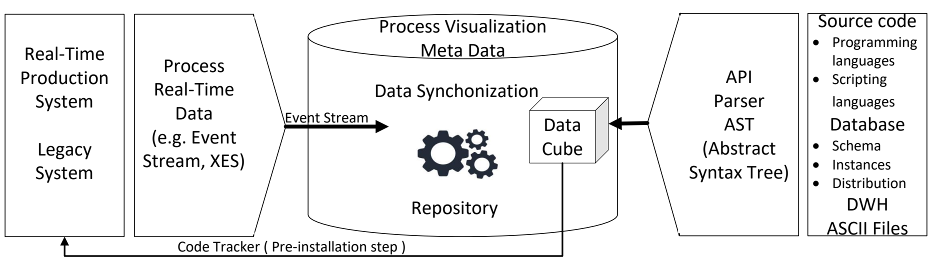


Fig. 1. General approach

When analyzing event logs generated from such legacy systems, a valuable effect can be achieved that the three approaches described in Section 1 are unable to provide: If certain entries in the event stream are missing when comparing the event stream with the source code, this indicates that the process steps involved, although implemented and present, have never been used. This information is essential when removing technical debts and modernizing legacy systems [8].

3 Related Work

This work is related to the research areas process mining, event log generation, and code analysis. Process mining [2] provides techniques to discover business process models from event logs [16,12], to evaluate conformance between process event logs and models [6], and to enhance processes [3]. Existing process discovery approaches mainly focus on the control flow perspective while the data perspective is mostly neglected [13]. The latter is of particular interest for meaningful process analysis and improvements (e.g., legacy system migration to new software architectures).

Event log generation is concerned with the generation of event log based on various sources. In [11,4], approaches to record user activities based on desktop actions (e.g., for robotic process automation) are presented. Our approach is also able to correlate such desktop actions with the corresponding source code fragments and database operations, allowing for a more detailed event log generation. The case study presented in [14] discusses the generation of event logs from a real-world data warehouse of a large U.S. health system. While some challenges (e.g., correlating events) may also arise in the context of legacy systems, we plan to minimize required domain expert interviews by automatically extracting domain knowledge from the source code.

Code analysis comprises traditional analysis (e.g., style checking or data flow analysis [10]) and profiling (e.g., CEGAR [7] and BMC [5]) which, combined with process knowledge, yield great potential for software improvement and migration.

4 Conclusion and Outlook

This paper emphasizes the need for spending research efforts on the recording of high quality event data in legacy systems. This not only enables the application of existing process mining algorithms, but also additional use cases such as, for example, data-driven process documentation, facilitation software migration projects or cost reduction through process-driven development. Note that corresponding work is also relevant in the context of robotic process automation [17].

Acknowledgments This work is part of the SoftProc project, funded by the KMU Innovativ Program of the Federal Ministry of Education and Research, Germany (F.No. 01IS20027A)

References

1. van der Aalst, W.M.P.: Process discovery: Capturing the invisible. IEEE Computational Intelligence Magazine **5**(1), 28–41 (2010)
2. van der Aalst, W.M.P.: Process Mining: Data Science in Action. Springer (2016)
3. van der Aalst, W.M.P., et al.: Process mining manifesto. In: Int'l Conf on BPM'11. pp. 169–194 (2011)

4. Agostinelli, S., Lupia, M., Marrella, A., Mecella, M.: Automated generation of executable rpa scripts from user interface logs. In: Business Process Management: Blockchain and Robotic Process Automation Forum. pp. 116–131. Springer International Publishing (2020)
5. Biere, A., Cimatti, A., Clarke, E.M., Strichman, O., Zhu, Y.: Bounded model checking. Carnegie Mellon University (2003)
6. Carmona, J., van Dongen, B., Solti, A., Weidlich, M.: Conformance Checking. Springer (2018)
7. Clarke, E., Grumberg, O., Jha, S., Lu, Y., Veith, H.: Counterexample-guided abstraction refinement. In: Emerson, E.A., Sistla, A.P. (eds.) Computer Aided Verification. pp. 154–169. Springer (2000)
8. Cunningham, W.: The wycash portfolio management system. SIGPLAN OOPS Mess. **4**(2) (1992)
9. Dumas, M., Rosa, M.L., Mendling, J., Reijers, H.A.: Fundamentals of Business Process Management. Springer, 2nd edn. (2018)
10. Khedker, U., Sanyal, A., Karkare, B.: Data Flow Analysis: Theory and Practice. CRC Press, Inc., USA, 1st edn. (2009)
11. Linn, C., Zimmermann, P., Werth, D.: Desktop activity mining - a new level of detail in mining business processes. In: Workshops der INFORMATIK 2018 - Architekturen, Prozesse, Sicherheit und Nachhaltigkeit. pp. 245–258. Köllen Druck+Verlag GmbH (2018)
12. Peña, M.R., Bayona-Oré, S.: Process mining and automatic process discovery. In: 2018 7th International Conference On Software Process Improvement (CIMPS). IEEE (2018)
13. Reichert, M.: Process and data: Two sides of the same coin? In: 20th Int'l Conf on Cooperative Information Systems (CoopIS'12). pp. 2–19. Springer (2012)
14. Remy, S., Pufahl, L., Sachs, J.P., Böttinger, E., Weske, M.: Event log generation in a health system: A case study. In: Business Process Management. pp. 505–522. Springer International Publishing (2020)
15. Vesterli, S.: Oracle ADF Survival Guide. Apress, Berkeley, CA, 1st edn. (2017)
16. Weerdt, J.D., Backer, M.D., Vanthienen, J., Baesens, B.: A multi-dimensional quality assessment of state-of-the-art process discovery algorithms using real-life event logs. Inf Sys **37**(7), 654 – 676 (2012)
17. Wewerka, J., Reichert, M.: Robotic process automation - a systematic mapping study and classification framework. Enterprise Information Systems (2022)

A Vision for Explainability of Coordinated and Conflicting Adaptions in Self-Adaptive Systems

Sandro Speth[1], Sarah Stieß[1], and Steffen Becker[1]

Institute of Software Engineering, University of Stuttgart, Stuttgart, Germany
{sandro.speth,sarah.stiess,steffen.becker}@iste.uni-stuttgart.de

Abstract. In the microservice domain, self-adaptive systems exist that reconfigure themselves to adhere to their guaranteed quality of service in the face of a changing environment. Constant changes in the environment enforce continuous adaptations of the system. Especially, different and potentially conflicting adaptations might interact, making it challenging to explain the decision and rationale behind the overall reconfiguration. In this paper, we discuss different approaches for the explainability of self-adaptive systems. Furthermore, we propose our approach to achieve a good trade-off between explainability and its performance impact for the mandatory data gathering. The approach encompasses eliciting requirements regarding explanations and their representations and experimenting on reference architectures for insights into the data required to fulfil the requirements.

Keywords: Microservices · Self-adaption · Explainability.

1 Introduction

Modern Cloud-native applications increasingly consist of self-adaptive microservice systems to better cope with constant changes in the environment and demands [10]. To achieve a better overall resilience, services adapt themselves through reconfigurations of their architecture, e.g., by scaling or by replacing entire failed services [1, 4]. Therefore, a self-adaptive system must monitor and analyze its current state, plan on which adaptions to take, if any required, and execute these actions without human intervention [6].

Due to the varying amount of users in the cloud, the workload changes constantly and enforces continuous adaptions, which may, either accidentally or on purpose, happen simultaneously and, therefore, influence or even conflict with each other. As an example, for two dependent services that are part of a larger architecture, scaling out the consuming service increases the incoming load of the consumed one, causing the consumed one to scale as well. Regarding conflicts, a service might define adaptation rules based on different metrics, e.g., response time and CPU load. In case the response time increases while the CPU load does not, e.g., if the increased response time is caused by waiting for another service, the response time rule triggers a scale out, followed by the CPU load rule trying to scale back in. In these examples, the behaviour deviates from the

J. Manner, D. Lübke, S. Haarmann, S. Kolb, N. Herzberg, O. Kopp (Eds.): 14th ZEUS Workshop, ZEUS 2022, Bamberg, held virtually due to Covid-19 pandemic, Germany, 24–25 February 2022, published at http://ceur-ws.org

expected or is sub-optimal. To comprehend the behaviour and to improve and fix the system, a DevOps engineer must understand the rationale behind the performed self-adaptations [12], especially if they are frequently recurring [13]. However, the interactions between potentially conflicting adaptations are challenging to understand, especially if the adaptation rules are more elaborated. Consequently, the need to explain the interactions between various adaptations arises. To create an explanation that DevOps engineers easily understand, we must identify which information the explanation should contain. Furthermore, for the sake of performance, we must find a reasonable trade-off between the amount of gathered data and the granularity of the explanation. This leads to our problem statements:

Problem 1. What is mandatory information to explain the coordination of and the interactions between multiple, perhaps conflicting reconfigurations and their impact on the system's overall adaptation behaviour?

Problem 2. How can we obtain the components of such an explanation while keeping a trade-off between the quality of the explanation and the performance impact of the data gathering?

2 Related Work

Explainability is becoming increasingly popular and essential in many research fields as it allows developers to understand systems more efficiently [5, 12]. In the context of cyber-physical systems, Bohlender et al. [3] characterise an explanation as a collection of information that has a target group and a subject and improves the target groups' understanding of the subject [3]. As the usefulness of the explanation depends on the target group, this endorses the importance of our first problem.

Klös et al. [8, 9] consider the explainability of self-learning self-adaptive systems. Their system adapts based on timed adaption rules and improves them with a genetic learning algorithm. It records various information, such as which condition in the system or environment triggered the adaption, the adaptation's expected effects and its actual effects, and feeds these information into a learning algorithm [9, 8]. Furthermore, they state that the collected information may serve as explanations of the system's adaptions or as a foundation to create further explanations for specific target groups [7]. In contrast to our problem, their initial focus is on explaining the self-learning aspect. In addition, they focus on single rules only instead of coordinated reconfigurations.

Blumreiter et al. [2] propose the reference framework MAB-EX for self-explaining systems. Their framework consists of four steps: (1) *Monitor*, (2) *Analyze*, (3) *Build* and (4) *EXplain*. *Monitor* and *Analyze* are analogous to the steps from the MAPE-K [6] loop. *Build* creates the explanation, and *Explain* transforms the explanation into a representation befitting the receiver and transmits it to the receiver [2]. The last step emphasises the importance of the target group. MAB-EX proposes two realisations for assisted driving systems [2]. In contrast to that, we focus on self-adaptive microservice systems.

3 Proposed Approach

In compliance with Bohlender et al. [3], we define DevOps engineers as the target group for explanations of self-adaptations. Furthermore, we identified three subjects: (1) (non-)application of a single reconfiguration, (2) coordination of reconfigurations, and (3) influences and relations between reconfigurations.

For our first problem statement, we already conducted an expert survey regarding reconfiguration on a Kubernetes cluster and found out that DevOps engineers consider Kubernetes' primarily textual representations and logs challenging to understand and, therefore, preprocessed cognitive effective representations are needed. Next, we plan to conduct an expert survey on DevOps engineers to identify requirements, mandatory information, and suitable representations, e.g. text or visual, interactive or static, which improve the DevOps engineers' understanding of the self-adaptations. We expect that an explanation requires at least information about (1) the components which were adapted, (2) the configuration of the components before the adaption, (3) the time of the adaption, and (4) the environmental change stimuli, e.g., the workload for the affected components triggering the adaption. Based on the elicited requirements, we decide on a fitting representation for the explanations. For example, explanations could be reported as cross-component issues [14] in Gropius [15], as issues are an already well-established natural platform to explain problems. This way, the explanations would be available in the developer's IDE to reduce context-switches [16].

For the second problem statement, we need a reference architecture for self-adaptive systems to evaluate our solution approach. The system is required to execute not only single reconfigurations but multiples in coordination while providing various metrics and data for the explanations. We plan to conduct a literature survey to identify suitable reference architectures, starting with the list provided by Taibi[1]. To monitor environmental change-stimuli to simulate and gain required information to explain adaptions, we plan to instrument OpenAPM [11] solutions. Especially, the monitoring solution should provide data and insights about the system's behaviour after a reconfiguration to assert the correct execution of adaption. However, deciding on the monitored metrics, their level of detail, and how long to preserve the data depends on the requirements collected in problem 1. Finally, we plan to evaluate explanations created from our reference architecture's adaptions for their comprehensibility by performing expert surveys with DevOps engineers as representatives of our target group.

4 Conclusion

Interactions between self-adaptations and potential conflicts between them are difficult to understand. Therefore, the need for explaining the rationale behind such adaptations arises. However, current approaches focus on explaining single adaptations only. Therefore, we propose our ideas of improving the DevOps engineers' understanding of a self-adaptive system by explaining single system

[1] https://github.com/davidetaibi/Microservices_Project_List

reconfiguration decisions as well as coordinated reconfiguration decisions and their influences on and relations with each other. Our ideas include (1) determining the requirements for explanations in self-adaptive systems and (2) how to create a suitable explanation.

References

1. Aderaldo, C.M., et al.: Kubow: An architecture-based self-adaptation service for cloud native applications. In: Proceedings of the 13th European Conference on Software Architecture - Volume 2. p. 42–45. ACM (2019)
2. Blumreiter, M., Greenyer, J., Chiyah Garcia, F.J., Klös, V., Schwammberger, M., Sommer, C., Vogelsang, A., Wortmann, A.: Towards self-explainable cyber-physical systems. In: 2019 ACM/IEEE 22nd International Conference on Model Driven Engineering Languages and Systems Companion (MODELS-C). pp. 543–548 (2019)
3. Bohlender, D., Köhl, M.A.: Towards a characterization of explainable systems (2019)
4. Garlan, D., Cheng, S.W., Huang, A.C., Schmerl, B., Steenkiste, P.: Rainbow: architecture-based self-adaptation with reusable infrastructure. Computer 37(10), 46–54 (2004)
5. Greenyer, J., Lochau, M., Vogel, T.: Explainable software for cyber-physical systems (ES4CPS): report from the GI dagstuhl seminar 19023, january 06-11 2019, schloss dagstuhl. CoRR abs/1904.11851 (2019), http://arxiv.org/abs/1904.11851
6. Kephart, J., Chess, D.: The vision of autonomic computing. Computer 36(1), 41–50 (2003)
7. Klös, V.: Explainable self-learning self-adaptive systems. In: Explainable Software for Cyber-Physical Systems (ES4CPS): Report from the GI Dagstuhl Seminar 19023, January 06-11 2019, Schloss Dagstuhl. pp. 46–47 (2019)
8. Klös, V., Göthel, T., Glesner, S.: Comprehensible and dependable self-learning self-adaptive systems. Journal of Systems Architecture 85-86, 28–42 (2018)
9. Klös, V., Göthel, T., Glesner, S.: Comprehensible decisions in complex self-adaptive systems. In: Software Engineering und Software Management 2018. pp. 215–216. Gesellschaft für Informatik, Bonn (2018)
10. Newman, S.: Building Microservices: Designing Fine-Grained Systems. O'Reilly, 2nd edn. (2021)
11. Novatec GmbH: Openapm: Landscape for apm tools, obervability tools and monitoring tools, https://openapm.io/landscape
12. Sadeghi, M., Klös, V., Vogelsang, A.: Cases for explainable software systems: Characteristics and examples. In: Proceedings of 2021 IEEE 29th International Requirements Engineering Conference Workshops (REW). pp. 181–187. IEEE (2021)
13. Speth, S.: Semi-automated cross-component issue management and impact analysis. In: Proceedings of 2021 36th IEEE/ACM International Conference on Automated Software Engineering (ASE). pp. 1090–1094. IEEE (2021)
14. Speth, S., Becker, S., Breitenbücher, U.: Cross-component issue metamodel and modelling language. In: Proceedings of the 11th International Conference on Cloud Computing and Services Science (CLOSER 2021). pp. 304–311. SciTePress (2021)
15. Speth, S., Breitenbücher, U., Becker, S.: Gropius — a tool for managing cross-component issues. In: Software Architecture. vol. 1269, pp. 82–94. Springer (2020)
16. Speth, S., Krieger, N., Breitenbücher, U., Becker, S.: Gropius-vsc: Ide support for cross-component issue management. In: Companion Proceedings of the 15th European Conference on Software Architecture. CEUR (2021)

Decision Support for Knowledge-Intensive Processes

Anjo Seidel and Stephan Haarmann

Hasso Plattner Institute, University of Potsdam,
Prof.-Dr.-Helmert-Str. 2-3, 14482 Potsdam, Germany
anjo.seidel@student.hpi.de, stephan.haarmann@hpi.de

Abstract. In knowledge-intensive processes, knowledge workers have to choose from many actions those that align best with their objectives. This is challenging since such a decision involves explicit and tacit knowledge and may affect the future of the process in intricate ways. In other words, they cause a high cognitive load. Using flexible case models, we present an automated recommender system that determines the best possible action for given key performance indicators. This supports knowledge workers to accomplish their goals efficiently.

Keywords: Case Management · Decision Support · Recommendations

1 Introduction

Knowledge-intensive business processes (KiPs) are characterized as multi-variant and unpredictable [2], calling for flexibility at design- and run-time [2]. Hence, new modeling approaches have emerged, which are more declarative [11, 16] and data-centric [3, 12, 13, 19] than traditional, imperative ones (e.g., such as BPMN).

With the help of an execution engine, modeled processes can be enacted [24]. At run-time, knowledge workers drive a case by deciding which of the possible next actions to execute. These decisions are interconnected and knowledge-intensive [22] and drive the process gradually towards its goal.

Due to the flexibility, knowledge workers may choose from numerous activities, and the effect of a particular activity on the process outcome is not necessarily apparent. This makes it difficult to plan the execution of KiPs, i.e., arranging actions in a sequence leading to a certain goal. Planning, however, is characteristic for knowledge work [17]. In KiPs, goals are typically defined by the knowledge workers at run-time. This is called *late goal modeling* [2].

Different approaches of providing recommendations to support planning exist, including predictive process monitoring techniques [4, 21] and decision support via process simulation [18, 25]. However, both approaches cannot be applied to KiPs, as these processes are unrepeatable and unpredictable [2].

Therefore, we propose a model-based approach for providing recommendations. In [6], we already presented a solution to allow knowledge workers to define objectives during run-time. Objectives describe desired case states. We aim to

J. Manner, D. Lübke, S. Haarmann, S. Kolb, N. Herzberg, O. Kopp (Eds.): 14$^{\text{th}}$ ZEUS Workshop, ZEUS 2022, Bamberg, held virtually due to Covid-19 pandemic, Germany, 24–25 February 2022, published at http://ceur-ws.org

analyze the model and the execution context to recommend how to reach such a state. Two research questions emerge:

RQ1 What are the requirements for recommendations in KiPs?
RQ2 How can such recommendations be derived?

Our approach is based on fragment-based Case Management [10]. We analyze the nature of KiPs and the requirements for late goal modeling. To provide recommendations, we query the state space of a case model and search for activities that most likely lead to desired states.

In Sect. 2, we present related work. The groundwork regarding fragment-based Case Management and modeling objectives is elaborated in Sect. 3, while our approach is elaborated in Sect. 4. We discuss the current state of work and future research and conclude the paper in Sect. 5.

2 Related Work

KiPs are highly flexible and driven by the decisions of knowledge workers [2, 20]. Various approaches for modeling knowledge-intensive processes have been proposed: some are declarative, like DECLARE [16] and Dynamic Condition Response Graphs [11]. Others are data-centric, such as Guard-Stage-Milestone [12], PHILharmonicFlows [13], and BAUML [3]. The survey papers by Di Ciccio et al. [2] and Steinau et al. [19] provide an overview of knowledge-intensive and data-centric approaches, respectively.

The limited support for data in declarative approaches and for activities in data-centric approaches, calls for *hybrid ones* [1], one of which is fragment-based Case Management [10]. This approach focuses on highly structured process fragments that can be combined dynamically during run-time. It allows combining imperative control flow and declarative data flow. Recent extensions define the modeling of data associations [7], multiplicity constraints [9], and colored Petri net semantics [5]. However, the models use implicit data flow to buy flexibility at the cost of comprehensibility, challenging knowledge workers in planning actions.

Planning is an important task in knowledge work [17]. Marella et al. proposed an approach for automating planning in business processes [14,15], which does not apply to the knowledge worker-centric nature of KiPs. Wynn et al. and Rozinat et al. provide decision support based on simulating business processes [18, 25]. As KiPs are unrepeatable and unpredictable [2], a non-repeatable simulation provides only limited support. Furthermore, predictive business process monitoring approaches aim at predicting the next actions to be executed [4, 21]. Those predictions are based on past executions, which, again, contradicts the unrepeatable and unpredictable nature of KiPs.

The challenge of assisting planning KiPs remains open. First steps have been made by providing a framework for knowledge workers to define objectives [6]. In this paper, We show how objectives can be used to derive recommendations.

3 Backgound

Our approach is based on the fragment-based case management (fCM) approach. Furthermore, this paper continues our work of allowing knowledge workers to define objectives during run-time [6]. In the following, we provide an overview of the fCM approach and our previous work regarding modeling objectives.

3.1 Fragment-Based Case Management

Fragment-based case management (fCM) combines imperative control flow and declarative data flow [10]. In fCM, the process is composed of multiple fragments, which are control flow graphs similar to BPMN models. Additionally, data flow defines data requirements and operations of activities. It constrains how fragments can be combined during run-time. An fCM case model furthermore includes a data model, object behaviors, and a termination condition. The data model consists of data classes, associations, and multiplicity constraints [5, 7, 9]. Each data class has a state transition system defining the behavior of corresponding objects. The termination condition specifies the goal of the process.

In the following, we introduce the exemplary case model for assessing and deciding on insurance claims. A more detailed explanation of the example can be found online[1].

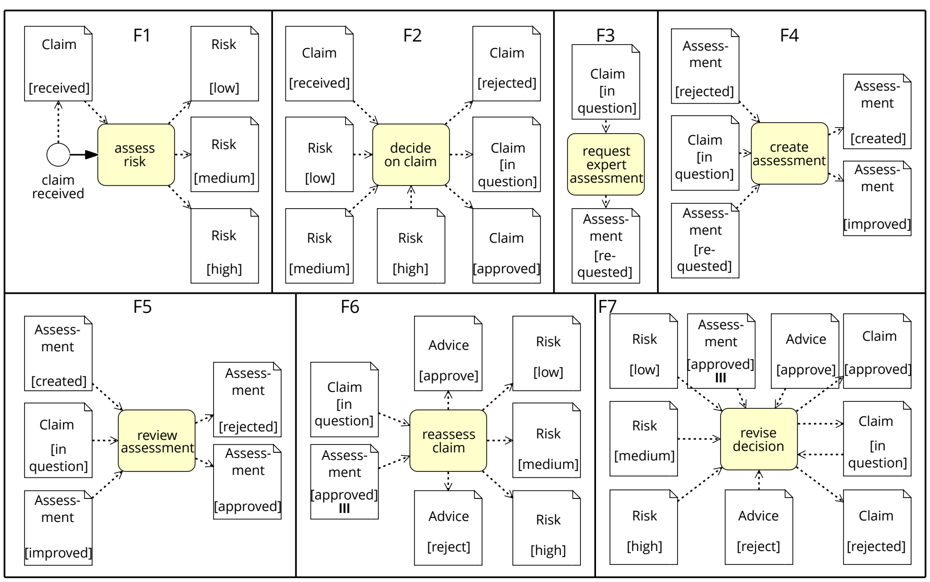


Fig. 1. Extract of fragments of the insurance claim handling process.

[1] The detailed example is available at `https://github.com/AnjoSs/DS4KiPs`

The process starts with receiving a claim. The first fragment *F1* is executed, and a risk is assessed. Given the risk, the knowledge worker can decide on the claim in *F2*. It can be accepted, rejected, or remain in question. A case in the state in question must be reassessed. During the reassessment, multiple expert assessments can be requested (*F3*), created (*F4)*, and reviewed (*F5)*. With the resulting assessments, the claim can be reassessed (*F6*), and the decision on the claim can be revised (*F7*).

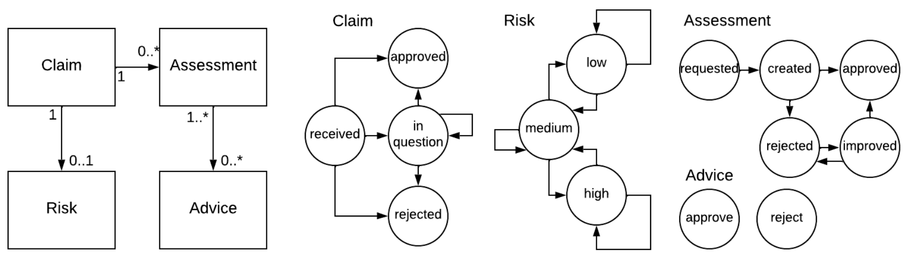


Fig. 2. The data model and object behaviors of the insurance claim handling process.

The data objects are instances of the classes *Claim*, *Risk*, *Assessment*, and *Advice* (see Fig. 2). Each claim can have one risk, and multiple expert assessments. From a number of assessments, an advice object can be retrieved. A claim can be in the states *received*, *approved*, *in question*, and *rejected*. A risk can be *low*, *medium*, or *high*. However, it cannot be changed from low to high or vice versa. An assessment can be *rejected*, *created*, then *approved* or *rejected* and *improved*. An advice can be either to *approve* or *reject* the claim

3.2 Modeling Objectives

In [6], we present a framework for specifying objectives based on an fCM model. Objectives are constraints on the state of a case. They can refer to data objects, their relationships, and to activities.

A case includes data, described by a set of data object O and a set of links L. Each object $o \in O$ belongs to a class $o.class$ and has an ID $o.id$ and a state $o.state$. A link $l \in L$ is an unordered pair of data objects.

Furthermore, each case has a set A of activity instances, henceforth called actions. An action $a \in A$ is an instance of an activity $a.activity$. It has a state $a.state$, which is either initial, control flow enabled, data flow enabled, enabled, running, or terminated [10]. Furthermore, an action reads a set of data objects $a.reads$ and writes a set of data objects $a.writes$. By executing an action, the state of the case (i.e., the sets O, L, and A) change. Using first-order logic, we can express knowledge workers' objectives using O, L, and A.

The objective g_1, for example, requires an enabled instance of activity revise decision reading an advice in state approve:

$$g_1 \equiv \exists a \in A, \exists o \in a.reads : a.activity = (revise\ decision) \wedge a.state = enabled$$
$$o.class = Advice \wedge o.state = approve$$

Multiple objectives can furthermore be composed by defining a partial order among them. It specifies the order in which the objectives need to be accomplished.

4 Recommendations for Knowledge Workers

With the opportunity to specify objectives at hand, the question is how to derive recommendations for the knowledge worker. Our approach focuses on analyzing the state space of the model itself. As the objectives are subject to the characteristics of late goal modeling, knowledge workers have special requirements for their recommendations. In the following, we elaborate on these requirements and explain how to derive suitable recommendations from a case.

4.1 Recommendation Requirements

KiPs are emergent [2]. Thus, it is impossible to plan far ahead. Instead, recommendations should focus on the immediate decision of choosing the next action. Yet, decisions still need to be made by knowledge workers, as they may have knowledge that is not part of the case state. To support workers, we calculate a score for all possible next actions. Purely based on the model, the action with the highest score aligns best with the objectives of the worker, i.e., it is recommended.

Objectives arise during run-time [9]. As the execution context may change, new objectives arise, and existing objectives change or become obsolete [2]. A knowledge worker must be able to update their objectives during run-time. Subsequently, recommendations can be calculated and actions can be (re)planned.

Weinzierl et al. [23] state that recommendations should be made w.r.t. to key performance indicators, which can be derived from data objects or past executions (i.e., event logs). In our approach, the key performance indicators are combined into a path cost function. Constant costs for all paths are equal to no cost function. Another simple implementation costs a path according to its length (number of activities). In summary, we require two user inputs:

1. A set of objectives that need to be fulfilled in the future.
2. A path cost function representing meaningful key performance indicators.

The expected results of recommendations and the described user inputs define the requirements of knowledge workers towards recommendations. **RQ1** is answered.

Consider our example from Sect. 3. The knowledge worker has specified the objective g_1 requiring *revise decision* reading an advice object in the state approve to be enabled. Assuming the case is in a state in which the claim has state *in* *question*, the risk is *medium*, two assessments are already *approved*, and no advice exists yet. The tasks *reassess claim* and *request expert assessment* are enabled. Now, a new objective g_2 emerges. It requires *revise decision* to be enabled for an advice object linked to at least three approved assignments.

Starting in the current state, the knowledge worker is interested in reaching the objectives g_1 and g_2. As a path cost function, the objectives should be reached with as few activities as possible. Therefore, we calculate a corresponding score for the next activities *reassess claim* and *request expert assessment*.

4.2 Deriving Recommendations

A business process model can be encoded into a planning domain [15], which can be used to derive recommendations. For this purpose, we reuse fCM's colored Petri net formalization [5,8]. It enables us to calculate and explore the model's state space, i.e., a directed graph consisting of all states and state transitions.

We calculate the scores for actions as follows (cf. Alg. 1): For each action, we start a breath-first search in the target state. We search for paths that result in a state satisfying the knowledge worker's objectives. For each such path, we calculate its costs. The inverse of the cost is added to the action's score. The rationale behind this scoring function is "if more cheap paths satisfying the objectives exist, the score of an action is higher." In other words, an action scores higher if it is likely to lead efficiently to a state, where all objectives are satisfied.

Algorithm 1 The score evaluation for next activities

```
function retrieve_recommendations(current_state, objectives, path_cost_function)
    action_scores ← [ ]
    Q ← queue(next(current_state))
    while Q is not empty do
        current_path ← Q.pop()
        if objectives hold in current_path[last] then
            action_scores at current_path[0] += 1 ÷ path_cost_function(current_path)
        else
            for next_action in next(current_path) do
                Q.push(current_path.append(next_action))
            end for
        end if
    end while
    return action_scores
end function
```

The presented algorithm provides a solution for deriving recommendations according to their requirements. It addresses and answers **RQ2**.

Considering the example, in the current state, *reassess claim* and *request expert assessment* are enabled. For both, a score is computed how likely they efficiently lead to a state, where g_1 and g_2 hold. All paths that start by executing *reassess claim* create an advice with only two assessments. This does not suffice to satisfy g_2. A new advice would need to be created with three or more assessments. On the other side, by executing *request expert assessment*, it is possible to create and review a new assessment, and to create the advice based on three assessments directly. There are shorter paths starting in *request expert assessment* than those starting in *reassess claim*. Therefore, Alg. 1 will rank *request expert assessment* higher than *reassess claim*.

5 Discussion and Conclusion

In our approach, we propose the use of a breadth-first search algorithm. The state space of a case grows exponentially and is possibly infinite. Search algorithms might not terminate. In combination with useful termination conditions, a breadth-first search can terminate early and lead to approximate results without querying the whole state space. The algorithm aims to find all reachable states where the objective holds, it derives optimal results for the specified path cost function. What especially suitable path cost functions look like, still needs to be evaluated.

For evaluation, we implemented a first prototype[2], which makes simple recommendations. It uses fCM's colored Petri net formalization and CPN-Tools[3] [5, 8]: By analyzing the model's state space, our prototype can verify for each possible next action whether the objectives can be satisfied eventually. This allows knowledge workers to assess whether an action complies with their objectives.

In future work, we plan to extend the prototype. First, knowledge workers need to be allowed to input the objectives and the cost function. Second, the prototype needs to calculate and return the scores of actions. Also, some technical challenges need to be addressed. Due to the flexibility of fCM, the state space is expected to grow exponentially. The algorithm for the state space search profits from optimization. The definition of fCM allows the state space even to be infinite, so the algorithm might not terminate at all. In practice, useful termination conditions for the search need to be found. Furthermore, a qualitative evaluation in the form of a user study can help to gain insights for the presented approach and prove it to work.

In this paper, we propose a framework allowing knowledge workers to state their requirements toward recommendations. These requirements consist of objectives and a path cost function, which encodes meaningful key performance indicators. The case model's state space is then analyzed in the search for paths towards states that satisfy the objectives. The more likely an action is to be part of such paths, and the cheaper the paths are, the higher the action is recommended.

With our work, we aim to support knowledge workers in making decisions. This support is a great asset for utilizing knowledge-intensive processes in practice.

[2] `https://github.com/bptlab/fCM-query-generator/tree/ZEUS_2022`

[3] `http://cpntools.org`

References

1. Andaloussi, A.A., Burattin, A., Slaats, T., Kindler, E., Weber, B.: On the declarative paradigm in hybrid business process representations: A conceptual framework and a systematic literature study (extended abstract). EMISA Forum **41**(1), 19–20 (2021)
2. Ciccio, C.D., Marrella, A., Russo, A.: Knowledge-intensive processes: Characteristics, requirements and analysis of contemporary approaches. J. Data Semant. **4**(1), 29–57 (2015), https://doi.org/10.1007/s13740-014-0038-4
3. Estañol, M., Queralt, A., Sancho, M., Teniente, E.: Artifact-centric business process models in UML. In: Rosa, M.L., Soffer, P. (eds.) Business Process Management Workshops - BPM International Workshops, Tallinn, Estonia, September 3, 2012. Revised Papers. Lecture Notes in Business Information Processing, vol. 132, pp. 292–303. Springer (2012)
4. Francescomarino, C.D., Ghidini, C., Maggi, F.M., Milani, F.: Predictive process monitoring methods: Which one suits me best? In: Weske, M., Montali, M., Weber, I., vom Brocke, J. (eds.) Business Process Management - 16th International Conference, BPM 2018, Sydney, NSW, Australia, September 9-14, 2018, Proceedings. Lecture Notes in Computer Science, vol. 11080, pp. 462–479. Springer (2018), https://doi.org/10.1007/978-3-319-98648-7_27
5. Haarmann, S., Montali, M., Weske, M.: Refining case models using cardinality constraints. In: Rosa, M.L., Sadiq, S.W., Teniente, E. (eds.) Advanced Information Systems Engineering - 33rd International Conference, CAiSE 2021, Melbourne, VIC, Australia, June 28 - July 2, 2021, Proceedings. Lecture Notes in Computer Science, vol. 12751, pp. 296–310. Springer (2021), https://doi.org/10.1007/978-3-030-79382-1_18
6. Haarmann, S., Seidel, A., Weske, M.: Modeling objectives of knowledge workers. In: Marrella, A., Weber, B. (eds.) Business Process Management Workshops - BPM 2021 International Workshops, Rome, Italy, September 6-10, 2021, Revised Selected Papers. Lecture Notes in Business Information Processing, vol. 436, pp. 337–348. Springer (2021), https://doi.org/10.1007/978-3-030-94343-1_26
7. Haarmann, S., Weske, M.: Correlating data objects in fragment-based case management. In: Abramowicz, W., Klein, G. (eds.) Business Information Systems - 23rd International Conference, BIS 2020, Colorado Springs, CO, USA, June 8-10, 2020, Proceedings. Lecture Notes in Business Information Processing, vol. 389, pp. 197–209. Springer (2020), https://doi.org/10.1007/978-3-030-53337-3_15
8. Haarmann, S., Weske, M.: Cross-case data objects in business processes: Semantics and analysis. In: Fahland, D., Ghidini, C., Becker, J., Dumas, M. (eds.) Business Process Management Forum - BPM Forum 2020, Seville, Spain, September 13-18, 2020, Proceedings. Lecture Notes in Business Information Processing, vol. 392, pp. 3–17. Springer (2020), https://doi.org/10.1007/978-3-030-58638-6_1
9. Haarmann, S., Weske, M.: Data object cardinalities in flexible business processes. In: del-Río-Ortega, A., Leopold, H., Santoro, F.M. (eds.) Business Process Management Workshops - BPM International Workshops, Seville, Spain, September 13-18, 2020, Revised Selected Papers. Lecture Notes in Business Information Processing, vol. 397, pp. 380–391. Springer (2020)
10. Hewelt, M., Weske, M.: A hybrid approach for flexible case modeling and execution. In: Rosa, M.L., Loos, P., Pastor, O. (eds.) Business Process Management Forum - BPM Forum 2016, Rio de Janeiro, Brazil, September 18-22, 2016, Proceedings. Lecture Notes in Business Information Processing, vol. 260, pp. 38–54. Springer (2016), https://doi.org/10.1007/978-3-319-45468-9_3

11. Hildebrandt, T.T., Mukkamala, R.R.: Declarative event-based workflow as distributed dynamic condition response graphs. In: Honda, K., Mycroft, A. (eds.) Proceedings Third Workshop on Programming Language Approaches to Concurrency and communication-cEntric Software, PLACES 2010, Paphos, Cyprus, 21st March 2010. EPTCS, vol. 69, pp. 59–73 (2010), https://doi.org/10.4204/EPTCS.69.5
12. Hull, R., Damaggio, E., Fournier, F., Gupta, M., III, F.F.T.H., Hobson, S., Linehan, M.H., Maradugu, S., Nigam, A., Sukaviriya, P., Vaculín, R.: Introducing the guard-stage-milestone approach for specifying business entity lifecycles. In: Bravetti, M., Bultan, T. (eds.) Web Services and Formal Methods - 7th International Workshop, WS-FM 2010, Hoboken, NJ, USA, September 16-17, 2010. Revised Selected Papers. Lecture Notes in Computer Science, vol. 6551, pp. 1–24. Springer (2010), https://doi.org/10.1007/978-3-642-19589-1_1
13. Künzle, V., Reichert, M.: Philharmonicflows: towards a framework for object-aware process management. J. Softw. Maintenance Res. Pract. **23**(4), 205–244 (2011), https://doi.org/10.1002/smr.524
14. Marrella, A.: What automated planning can do for business process management. In: Teniente, E., Weidlich, M. (eds.) Business Process Management Workshops - BPM 2017 International Workshops, Barcelona, Spain, September 10-11, 2017, Revised Papers. Lecture Notes in Business Information Processing, vol. 308, pp. 7–19. Springer (2017), https://doi.org/10.1007/978-3-319-74030-0_1
15. Marrella, A.: Automated planning for business process management. J. Data Semant. **8**(2), 79–98 (2019), https://doi.org/10.1007/s13740-018-0096-0
16. Pesic, M., Schonenberg, H., van der Aalst, W.M.P.: DECLARE: full support for loosely-structured processes. In: 11th IEEE International Enterprise Distributed Object Computing Conference (EDOC 2007), 15-19 October 2007, Annapolis, Maryland, USA. pp. 287–300. IEEE Computer Society (2007), https://doi.org/10.1109/EDOC.2007.14
17. Pyöriä, P.: The concept of knowledge work revisited. J. Knowl. Manag. **9**(3), 116–127 (2005), https://doi.org/10.1108/13673270510602818
18. Rozinat, A., Wynn, M.T., van der Aalst, W.M.P., ter Hofstede, A.H.M., Fidge, C.J.: Workflow simulation for operational decision support. Data Knowl. Eng. **68**(9), 834–850 (2009), https://doi.org/10.1016/j.datak.2009.02.014
19. Steinau, S., Marrella, A., Andrews, K., Leotta, F., Mecella, M., Reichert, M.: DALEC: a framework for the systematic evaluation of data-centric approaches to process management software. Softw. Syst. Model. **18**(4), 2679–2716 (2019), https://doi.org/10.1007/s10270-018-0695-0
20. Swenson, K.D.: Position: BPMN is incompatible with ACM. In: Rosa, M.L., Soffer, P. (eds.) Business Process Management Workshops - BPM 2012 International Workshops, Tallinn, Estonia, September 3, 2012. Revised Papers. Lecture Notes in Business Information Processing, vol. 132, pp. 55–58. Springer (2012), https://doi.org/10.1007/978-3-642-36285-9_7
21. Teinemaa, I., Dumas, M., Rosa, M.L., Maggi, F.M.: Outcome-oriented predictive process monitoring: Review and benchmark. ACM Trans. Knowl. Discov. Data **13**(2), 17:1–17:57 (2019), https://doi.org/10.1145/3301300
22. Vaculín, R., Hull, R., Heath, T., Cochran, C., Nigam, A., Sukaviriya, P.: Declarative business artifact centric modeling of decision and knowledge intensive business processes. In: Proceedings of the 15th IEEE International Enterprise Distributed Object Computing Conference, EDOC 2011, Helsinki, Finland, August 29 - September 2, 2011. pp. 151–160. IEEE Computer Society (2011), https://doi.org/10.1109/EDOC.2011.36

23. Weinzierl, S., Dunzer, S., Zilker, S., Matzner, M.: Prescriptive business process monitoring for recommending next best actions. In: Fahland, D., Ghidini, C., Becker, J., Dumas, M. (eds.) Business Process Management Forum - BPM Forum 2020, Seville, Spain, September 13-18, 2020, Proceedings. Lecture Notes in Business Information Processing, vol. 392, pp. 193–209. Springer (2020), https://doi.org/10.1007/978-3-030-58638-6_12
24. Weske, M.: Business Process Management - Concepts, Languages, Architectures, Third Edition. Springer (2019), https://doi.org/10.1007/978-3-662-59432-2
25. Wynn, M.T., Dumas, M., Fidge, C.J., ter Hofstede, A.H.M., van der Aalst, W.M.P.: Business process simulation for operational decision support. In: ter Hofstede, A.H.M., Benatallah, B., Paik, H. (eds.) Business Process Management Workshops, BPM 2007 International Workshops, BPI, BPD, CBP, ProHealth, RefMod, semantics4ws, Brisbane, Australia, September 24, 2007, Revised Selected Papers. Lecture Notes in Computer Science, vol. 4928, pp. 66–77. Springer (2007), https://doi.org/10.1007/978-3-540-78238-4_8

A Review of Approaches for Quality Model Validations in the Context of Cloud-native Applications

Robin Lichtenthäler and Guido Wirtz

Distributed Systems Group, University of Bamberg, Germany

Abstract. Quality models considering internal design characteristics of software should represent reality as accurately as possible. This can be ensured through a validation of relations between quality attributes. In this work we review validation approaches used in literature. We conclude that in an early design phase, surveys and expert interviews are suitable to validate quality attributes and their relations while for complete quality models quantitative validations through measures are advised.

Keywords: Quality model, Empirical validation, Cloud-native

1 Introduction

Quality models are used in software engineering to enable a structured assessment of the quality of a system according to quality attributes deemed important for it [12]. To do so, quality models typically include a theory of how certain software characteristics are related to higher level quality attributes and how such characteristics and attributes can be measured and combined to enable a quantitative quality assessment [1]. An example would be the theory used by Bansiya et al. [4] that in an object-oriented system the characteristic of coupling impacts extendability in the sense that high coupling has a negative impact on extendability. Coupling is stated to be quantitatively measurable by counting for each class to how many other classes it is directly related [4]. Such theories therefore constitute the inner basis of a quality model and the degree to which they are able to represent the reality ultimately determines the applicability and usefulness of a quality model. AL-Badareen et al. [1], however, also state that quality models are often formulated in a subjective manner and refer to a finding from Kitchenham and Pfleeger [25] that "*software quality models suffer from a lack of rationale for the relationships between quality characteristics and how the lowest levels properties are composed into an overall assessment of higher level quality characteristics*" [1]. The question of how quality models can be validated, that means how well they represent reality, is therefore relevant and important for providing quality models that are useful in practice.

In a recent study [33], we have formulated a quality model for cloud-native application architectures that is based on the Quamoco quality meta-model

J. Manner, D. Lübke, S. Haarmann, S. Kolb, N. Herzberg, O. Kopp (Eds.): 14$^{\text{th}}$ ZEUS Workshop, ZEUS 2022, Bamberg, held virtually due to Covid-19 pandemic, Germany, 24–25 February 2022, published at http://ceur-ws.org

[48]. Quality attributes are called *factors* in the Quamoco context and they can be either higher-level *quality aspects* or lower-level measurable *product factors*. To formulate *factors* and their interrelationships, called *impacts*, we relied on the ISO 25010 standard [19] in combination with suggestions from practitioner books. Nevertheless, the resulting quality model includes a subjective notion. A validation of the model and especially of the stated impacts would be beneficial for future work that uses the model. In this work, our aim therefore is to review existing approaches for validating quality models, especially in an empirical way. Our quality model for cloud-native applications [33] serves as a use case for which we want to derive implications for how a validation of such a newly created quality model could be performed and which aspects need to be considered. The contribution of this work is an overview of how and how often quality models proposed in literature are validated. In addition, we review approaches that exist to ensure the validity of quality models with implications for the formulation of new quality models. To summarize this, we aim to answer the following research questions:

RQ1: To what extent and how are quality models proposed in literature validated?

RQ2: Which implications can be derived for a proper validation of newly formulated quality models?

In the following, we provide some foundations on quality models in Sect. 2, discuss related work in Sect. 3 and present our methodology in Sect. 4. We describe our results and answers to our research questions in Sect. 5, before concluding our work with an outlook in Sect. 6.

2 Hierarchical Software Quality Models

The term quality model is to some extent used ambiguously and can for example also refer to a list of rules checked through static code analysis [34] where quality is measured directly based on the number of rule violations found in a software. In this work, however, the focus is on so-called hierarchical quality models [4] where a hierarchy exists from lower-level measures to higher-level quality aspects. Hierarchical quality models can integrate and interrelate multiple quality aspects and enable a more detailed evaluation of software quality. In turn, theories are needed to state the relationships between lower-level measures and higher-level quality aspects mediated by software characteristics.

In broad fields, such as software engineering, theories are difficult to generalize and often apply well only within certain contexts. Therefore, different quality models exist for different domains, for example object-oriented systems [4], embedded systems [35], Web services [42], or SOA architectures [16]. A contrast to these specializations are the emerged standards for quality in software: ISO 9126 [18] and its successor ISO 25010 [19]. The consequence, however, is that these standards mainly cover higher level quality attributes and advice on how to measure and evaluate software quality. Nevertheless, the standards provide a theoretical basis which has been validated through the structured definition and refinement process involving a group of experts. But for context-specific quality models including lower level quality attributes and measures the question of how well the underlying theory maps to reality remains. The methodological approach for ensuring the validity of a theory, and therefore of a quality model, is referred to as validation. We distinguish validation from evaluation in this work by considering evaluation as the approach of using a quality model for evaluating the quality of a software. Although in literature, these two terms are used inconsistently. Another distinction which is important for discussing different validation approaches, is that between *internal characteristics* of a software which are evaluated by analyzing the internal implementation of a system and *external characteristics* of a software which can only be evaluated at runtime when observing its behavior. This distinction is in line with the ISO 9126 [18] and ISO 25010 [19] standards which consider internal and external quality, or Kitchenham et al. [26] who differentiate between "externally visible properties" and "internally visible properties". For both types of characteristics impacts on quality aspects can be stated, but typically impacts of external characteristics are more intuitive. An example would be the externally visible uptime of a system for which it can be stated that a high uptime has a positive impact on the availability quality aspect. Especially considering internal characteristics, a clear rationale for the relationships between quality attributes is therefore important, as also stated by Wagner et al. [48] who developed Quamoco, a meta-model for quality models, in which relationships between quality attributes are defined as *impacts*. When formulating a quality model based on their meta-model such impacts need to be stated based on valid reasoning, which can for example rely on logical reasoning, previous literature, or empirical methods where empirical evidence is considered to be statements from people with experience in the domain of a quality model, collected through interviews or surveys. Empirical methods in specific are also found important for the acceptance of quality models in practice by Moody [39] who reviewed quality attributes of conceptual models (which is a superset of models used in software engineering and therefore also includes quality models). In conclusion, that means that also a validation of a hierarchical software quality model should put a focus on the validity of the stated impacts for a quality model which describe the relationships between the different factors, also considering their importance in relation to each other. On the basis of these properties of hierarchical quality models, we designed our approach for reviewing quality models and validation approaches applied to them.

3 Related Work

Our work is related to work considering the quality assurance of quality models themselves. In contrast to our perspective on the validity of the underlying theory, quality models can also be evaluated based on structural aspects: AL-Badareen et al. formulate a set of rules [1] for structural aspects of the impact graph of quality attributes. Furthermore, they formulate a set of rules for quality characteristics construction which, however, take into consideration a specific system to be evaluated. In addition, Moody [39] has done an evaluation of quality models for conceptual models and argues that empirical validation is important. He also discusses different methods to do so (e.g., laboratory experiments, action research, or surveys), but with a focus on the validation through applying a quality model, which is also important, but not practicable during the early design phase of a quality model as it is the case with our quality model for cloud-native applications. In addition, previous work has also systematically reviewed different quality models taking validation into consideration, but it is only addressed shortly. For example, Nistala et al. [41] only assess whether an evaluation has been done, but it is unclear if explicit validations of quality models are meant or evaluations of software systems using the quality model. Although Nistala et al. also report whether empirical approaches have been used, they are not discussed in detail. Yan et al. [49] shortly address validation methods used for quality models and find the categories *expert opinion*, *issue handling indicators*, and *industry validation*, but do not go into detail except from mentioning that validation is important for practical usage.

4 Methodology

To get an overview of quality models proposed in literature and have a basis for our investigation, we rely on review papers that have already searched the literature for software quality models in a structured way and that reported the quality models they have found as results. As recent review papers, we found the one by Galli et al. [13] (23 results) who aim to measure the relevance of quality models, as well as the systematic mapping studies by Nistala et al. [41] (40 results), focusing on types of model elements used, and by Yan et al. [49] (31 results), focusing on the scope and maturity of quality models. Additionally, we included the mapping study by Oriol et al. [42] (47 results), because of their thematically related focus on web services. Because these review papers present their results in different ways, we hereby introduce the term *entry* to consolidate the results of these review papers in a generic way. An *entry* refers to a research undertaking which may span one or several publications and may or may not explicitly report a quality model. This way we can consider a quality model that has been presented in one paper, but validated, applied, or evolved in additional papers, as a single entry. And we can also include papers that do not explicitly report a quality model, but for example methods for validation which have been proposed independently from a specific quality model. After merging the results from the review papers and removing duplicates, we had 121 entries as an initial set for our investigation. Next, we classified all these entries according to the following criteria:

- **Type of contribution**: Not all studies present hierarchical quality models with explicit factors and impacts, only those that do were classified as contributing a *model*. Others contribute a *meta-model* for quality models, just a *taxonomy* which cannot be used as a quality model, or any kind of *method* in the context of quality models (for example how to create a quality model or apply it within an organization). Studies that present a specific method for validating quality models were classified as *validation-method*.

– **Characteristics** considered: For each quality model we differentiate between the characteristics it considers, namely *internal characteristics* and *external characteristics* of a software as described in Sect. 2.
– **Rationale** for non-trivial relationships between quality attributes: This criterion covers the rationale which is used for stating impacts between factors and their relative strength, however considering only non-trivial relationships (In contrast, a trivial relationship would be that a lower latency positively impacts performance efficiency). A rationale can be argumentation simply based on *logical implication* or by relying on existing work (*literature-based*). Empirical evidence can be provided by relying on a small set of experts (*empirical-experts*), for example through interviews, or a structured survey among a larger set of participants (*empirical-survey*) can be used. For a quantification of the relative strengths of impacts, *algorithmic* evaluations are sometimes used. If no rationale is provided or it is not possible to determine it, we classified an entry as *none*.

It has to be noted that for each criterion multiple values could be assigned, for example when both internal and external characteristics are considered or a model together with a method is presented. To ensure that we do not miss validations of quality models published separately after the publication of a quality model, we performed a forward search by looking at the citations for each entry. However, we restricted the forward search to a filtered list of entries which only includes entries where the type of contribution includes a *model* or *evaluation-method* and the considered characteristics include *internal characteristics*. Our focus on internal characteristics is due to our use case of the quality model for cloud-native application architectures [33] which aims to evaluate software architectures at design time based on architectural models. Formulating impacts on quality aspects from internal characteristics is more difficult, because the actual behavior of a system can only be observed at runtime. Therefore validations for such stated impacts are especially important. For the forward search we used SemanticScholar[1] and searched the citations with the keywords **evaluation** or **validation**. The forward search lead to an additional set of 11 publications. Together with the filtered list of entries our final list which forms the basis of our investigation thus consists of 50 entries and can be found online[2]. Detailed information on the used literature, the search process, and classifications can also be found in the corresponding repository for this site[3]. To answer our research questions, we then quantitatively and qualitatively investigated these 50 entries to gain insights and provide implications for validations of quality models.

[1] `https://www.semanticscholar.org/`
[2] `https://r0light.github.io/qualitymodel-validations-review/`
[3] `https://github.com/r0light/qualitymodel-validations-review`

5 Results & Implications

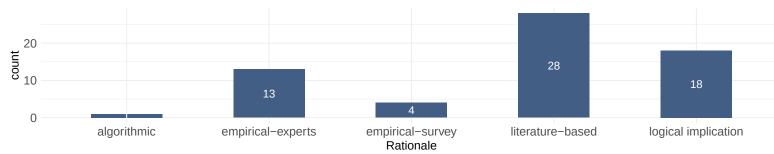


Fig. 1. Count of different types of rationale used

Overall, we were interested in the types of rationale on which the theoretical concepts of quality models are based. In Fig. 1 it can be seen that the majority of quality models relies on previously published literature or uses logical implication to infer conceptual relationships between factors. From a historical perspective it can be seen that early quality models for example from Boehm [5], McCall [37], or Dromey [11] have been formulated and described in comprehensive research reports using experience and logical implication. Later works then relied on them [4, 43] until the ISO standards [18, 19] became available and were again frequently used as a basis [3, 8–10, 14, 17, 21, 22, 24, 28, 32, 40, 44–47]. This shows a tendency to rely on existing literature for a sound theoretical foundation so that a further validation is less important. Empirical approaches are nevertheless also frequently used, especially for more recent domain-specific quality models [15, 35, 36, 38] which are more difficult to cover with the more general standards.

To answer **RQ1** we classified each entry according to whether an explicit approach has been used to validate the proposed quality model. Out of the 50 entries in our result set, 40 do present a quality model and from these we found 18 which included an explicit validation approach. The approach and scope of the validations however are diverse and we therefore further classified the validation approaches according to the scope in focus. For this classification of validations based on their scope we also considered the remaining 10 entries presenting validation methods, independently from a specific quality model An overview of the different validation approaches is provided in Fig. 2. On the left side of Fig. 2 the elements of a quality model in the sense of the Quamoco meta-model are shown while on the right side different perspectives on a software system are shown. The arrows represent relations and the stethoscopes (⚕) signify measures attached to the different perspectives on a system. So, for example, a measure at the source code level could be used to measure the degree to which a product factor is present and the product factor impacts a quality aspect which in turn might impact a higher level quality aspect. The numbered magnifiers (🔍) show the different scopes of validation approaches depending on which elements or relations are in focus. Generally, a differentiation can be done based on the amount of information needed for a validation and the point in time when a validation is suitable. In Table 1 additional details for the different validation approaches are provided, including which elements of a quality model are required. It can be seen that the validation of factors themselves (V1), the impacts between factors (V2) and relative weights of impacts (V3) can be done solely based on factors proposed for a quality model, and therefore also early in the design phase of a quality model (early in the sense that only factors are defined). Using interviews with experts, Gerpheide et al. [15] have defined and validated factors (V1) and Mayr et al. [35] early validated their model (V1, V2, V3) regarding comprehensibility, appropriate level of abstraction, and consistent classifications through conducting multiple workshops. Lampasona et al. [31] present an approach to rate the minimality and completeness of factors (V1) using interviews with experts. Surveys among practitioners have been used by Mehmood et al. [38] and Gerpheide et al. [15] to validate impacts (V2) and their weights (V3) from product factors on quality aspects by calculating the agreement of respondents regarding the existence and type of impacts. In a similar way, also Khomh et al. [23] validated impacts and their weights (V2, V3), although they investigated design patterns instead of product factors. In our opinion, however, product factors are just a more general construct through which also patterns can be expressed. An additional frequently used approach for assigning weights to impacts on quality aspects (V3) is the Analytical Hierarchical Process [2, 7, 29] in which experts compare impacts for a quality aspect pairwise and based on that weights are calculated.

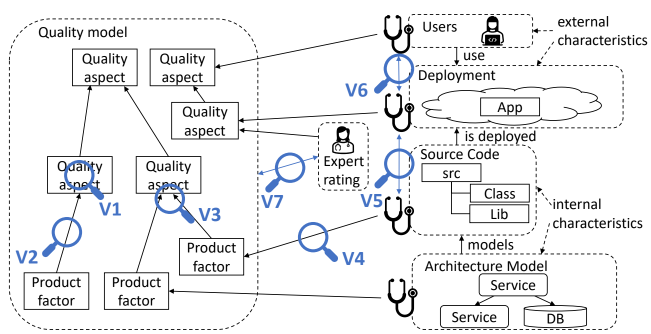


Fig. 2. Overview of possibilities for quality model validations

In contrast to that are validations for completely defined quality models, that means quality models with factors, measures, and relationships between them: Kläs et al. [27] validated factors based on diversification (V1) of measures and overall validity through a comparison with expert ratings (V7). A comparison with expert ratings (V7), also for relations specifically (V3, V4), has been done by Bansiya et al. [4] and Mayr et al. [35]. Braeuer et al. [6] validated measures by comparing them with previously gained measurements (V4). Finally, also considering external measures, Jung et al. [20] compared external measures with user measures (V6) while Kvam et al. [30] and Yu et al. [50] correlated internal measures with external measures (V5), such as productivity or performance.

Table 1. Validation approach scope details

	Validation target	Required elements	Examples	Early?
V1	Quality Aspect	Factors	[15, 27, 31, 35]	✓
V2	ProductFactor-QualityAspect	Factors	[15, 23, 35, 38]	✓
V3	ProductFactorImpactWeights	Factors	[2, 7, 15, 23, 29, 35, 38]	✓
V4	CodeMeasure-ProductFactor	Factors, Measures	[4, 6, 35]	
V5	DeploymentMeasure-CodeMeasure	Factors, Measures	[30, 50]	
V6	UserMeasure-DeploymentMeasure	Factors, Measures	[20]	
V7	ExpertRating-QualityModelResult	Factors, Measures, Expert Rating	[4, 27, 35]	

Regarding **RQ2** and the context of our quality model [33] we can therefore state that in an early phase of quality model formulation, where not all elements of a quality model are defined yet, surveys and interviews can be used for validation. This fits the context of cloud-native applications, because it is a comparatively new topic where less existing literature to rely on exists. Therefore, there is also a lack of measures focusing on the architectural level of service interactions and cloud deployment options [33] which makes validations where such measures are needed difficult. Nevertheless, when all elements of a newly formulated quality model are defined, the quality model should also be validated by comparisons of complete evaluations with earlier evaluations or independent evaluations by experts. In addition, it is common to rely on standards as a foundation which can therefore also be recommended for new quality models. A challenge that remains is the large number of factors [33] for which no implications can be derived from the literature, because the considered quality models contained less factors. Finally, an interesting observation is that we did not find any validations for quality models taking architectural models into consideration.

6 Conclusion and Outlook

Ensuring the validity of quality models regarding their internal conceptual basis is important for their applicability and usefulness in practice. During our investigation we found that creators of quality models mostly rely on existing literature for a validated foundation, but also explicit empirical methods are frequently used, especially for domain-specific quality models. A limitation of our work is that we relied on existing survey papers for our literature base, but we added a forward search based on the considered literature to also include more recent work. We plan to apply these results on our recently proposed quality model for cloud-native applications by performing a survey to validate its factors and impacts.

References

1. AL-Badareen, A.B., Desharnais, J.M., Abran, A.: A suite of rules for developing and evaluating software quality models. In: Software Measurement, pp. 1–13. Springer International Publishing (2015), https://doi.org/10.1007/978-3-319-24285-9_1
2. Alrawashdeh, T.A., Muhairat, M.I., Alqatawneh, S.M.: A quantitative evaluation of ERP systems quality model. In: 11th International Conference on Information Technology: New Generations. IEEE (2014), https://doi.org/10.1109/itng.2014.37
3. Alvaro, A., Almeida, E., Meira, R.: Towards a software component quality model. In: Submitted to the 5th International Conference on Quality Software. Citeseer (2005)
4. Bansiya, J., Davis, C.: A hierarchical model for object-oriented design quality assessment. IEEE Transactions on Software Engineering 28(1), 4–17 (2002), https://doi.org/10.1109/32.979986
5. Boehm, B.W., Brown, J.R., Lipow, M.: Quantitative evaluation of software quality. In: Proceedings of the 2nd International Conference on Software Engineering. p. 592–605. ICSE '76, IEEE, Washington, DC, USA (1976), https://doi.org/10.5555/800253.807736
6. Braeuer, J., Ploesch, R., Saft, M.: Measuring maintainability of OO-software - validating the IT-CISQ quality model. In: Advances in Intelligent Systems and Computing, pp. 283–301. Springer International Publishing (2016), https://doi.org/10.1007/978-3-319-46535-7_22
7. Correia, J.P., Kanellopoulos, Y., Visser, J.: A survey-based study of the mapping of system properties to ISO/IEC 9126 maintainability characteristics. In: IEEE International Conference on Software Maintenance. IEEE (2009), https://doi.org/10.1109/icsm.2009.5306346
8. Dixit, D.: Cbqm: Component based quality model. In: 4th International Conference on Reliability, Infocom Technologies and Optimization. pp. 1–5. Citeseer (2013)
9. Domínguez, K., Pérez, M., Grimán, A.C., Ortega, M., Mendoza, L.E.: Software quality model based on software development approaches. In: 11th IASTED International Conference on Software Engineering and Applications. p. 1–6. SEA '07, ACTA (2007), https://doi.org/10.5555/1647636.1647638
10. Dominguez-Mayo, F.J., Escalona, M.J., Mejias, M., Torres, A.H.: A quality model in a quality evaluation framework for MDWE methodologies. In: Fourth International Conference on Research Challenges in Information Science (RCIS). IEEE (2010), https://doi.org/10.1109/rcis.2010.5507323
11. Dromey, R.: A model for software product quality. IEEE Transactions on Software Engineering 21(2), 146–162 (1995), https://doi.org/10.1109/32.345830
12. Ferenc, R., Hegedűs, P., Gyimóthy, T.: Software product quality models. In: Evolving Software Systems, pp. 65–100. Springer Berlin Heidelberg (2013), https://doi.org/10.1007/978-3-642-45398-4_3
13. Galli, T., Chiclana, F., Siewe, F.: Software product quality models, developments, trends, and evaluation. SN Computer Science 1(3) (2020), https://doi.org/10.1007/s42979-020-00140-z
14. Gehlert, A., Metzger, A.: Quality Reference Model for SBA. S-CUBE consortium (2008)
15. Gerpheide, C.M., Schiffelers, R.R.H., Serebrenik, A.: Assessing and improving quality of QVTo model transformations. Software Quality Journal 24(3), 797–834 (2015), https://doi.org/10.1007/s11219-015-9280-8
16. Goeb, A., Lochmann, K.: A software quality model for SOA. In: Proceedings of the 8th international workshop on Software quality - WoSQ '11. ACM (2011), https://doi.org/10.1145/2024587.2024593

17. Hu, W., Loeffler, T., Wegener, J.: Quality model based on ISO/IEC 9126 for internal quality of MATLAB/simulink/stateflow models. In: IEEE International Conference on Industrial Technology. IEEE (2012), https://doi.org/10.1109/icit.2012.6209958
18. ISO/IEC: ISO/IEC 9126 software engineering— product quality. Online (2001), https://www.iso.org/standard/22749.html
19. ISO/IEC: ISO/IEC 25000 Systems and software engineering – Systems and software Quality Requirements and Evaluation (SQuaRE). Online (2014), https://www.iso.org/standard/64764.html
20. Jung, H.W.: Validating the external quality subcharacteristics of software products according to ISO/IEC 9126. Computer Standards & Interfaces 29(6), 653–661 (2007), https://doi.org/10.1016/j.csi.2007.03.004
21. Kalaimagal, S., Srinivasan, R.: Q'facto 12: an improved quality model for cots components. ACM SIGSOFT Software Engineering Notes 35(2), 1–4 (2010), https://doi.org/10.1145/1734103.1734116
22. Kanellopoulos, Y., Tjortjis, C., Heitlager, I., Visser, J.: Interpretation of source code clusters in terms of the ISO/IEC-9126 maintainability characteristics. In: 12th European Conference on Software Maintenance and Reengineering. IEEE (2008), https://doi.org/10.1109/csmr.2008.4493301
23. Khomh, F., Guéhéneuc, Y.G.: DEQUALITE: building design-based software quality models. In: 15th Conference on Pattern Languages of Programs. ACM (2008), https://doi.org/10.1145/1753196.1753199
24. Kim, C., Lee, K.: Software quality model for consumer electronics product. In: Ninth International Conference on Quality Software. IEEE (2009), https://doi.org/10.1109/qsic.2009.58
25. Kitchenham, B., Pfleeger, S.: Software quality: the elusive target [special issues section]. IEEE Software 13(1), 12–21 (1996), https://doi.org/10.1109/52.476281
26. Kitchenham, B., Linkman, S., Pasquini, A., Nanni, V.: The squid approach to defining a quality model. Software Quality Control 6(3), 211–233 (1997), https://doi.org/10.1023/a:1018516103435
27. Kläs, M., Lochmann, K., Heinemann, L.: Evaluating a quality model for software product assessments – a case study. In: Proc. of SQMB (2011)
28. Kumar, A., Grover, P.S., Kumar, R.: A quantitative evaluation of aspect-oriented software quality model (AOSQUAMO). ACM SIGSOFT Software Engineering Notes 34(5), 1–9 (2009), https://doi.org/10.1145/1598732.1598736
29. Kumar, P., Singh, S.K.: A comprehensive evaluation of aspect-oriented software quality (AOSQ) model using analytic hierarchy process (AHP) technique. In: 2nd International Conference on Advances in Computing, Communication, & Automation (ICACCA). IEEE (2016), https://doi.org/10.1109/icaccaf.2016.7748957
30. Kvam, K., Lie, R., Bakkelund, D.: Legacy system exorcism by pareto's principle. In: Companion to the 20th annual ACM SIGPLAN conference on Object-oriented programming, systems, languages, and applications - OOPSLA '05. ACM (2005), https://doi.org/10.1145/1094855.1094959
31. Lampasona, C., Mayr, A., Saft, M.: Early validation of software quality models with respect to minimality and completeness: An empirical analysis. In: Software Metrik Kongress (2013)
32. Lee, K., Lee, S.J.: A quantitative software quality evaluation model for the artifacts of component based development. In: Sixth International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing and First ACIS International Workshop on Self-Assembling Wireless Networks (SNPD/SAWN'05). IEEE (2005), https://doi.org/10.1109/snpd-sawn.2005.7

33. Lichtenthäler, R., Wirtz, G.: Towards a Quality Model for Cloud-native Applications. In: European Conference on Service-Oriented and Cloud Computing. ESOCC 2022. Springer (2022), (to appear)
34. Louridas, P.: Static code analysis. IEEE Software 23(4), 58–61 (jul 2006), https://doi.org/10.1109/ms.2006.114
35. Mayr, A., Plosch, R., Klas, M., Lampasona, C., Saft, M.: A comprehensive code-based quality model for embedded systems: Systematic development and validation by industrial projects. In: IEEE 23rd International Symposium on Software Reliability Engineering. IEEE (2012), https://doi.org/10.1109/issre.2012.4
36. Mayr, A., Plosch, R., Saft, M.: Objective measurement of safety in the context of IEC 61508-3. In: 39th Euromicro Conference on Software Engineering and Advanced Applications. IEEE (2013), https://doi.org/10.1109/seaa.2013.32
37. McCall, J.A., Paul K. Richards, G.F.W.: Factors in software quality. volume i. concepts and definitions of software quality. techreport ADA049014 (1977)
38. Mehmood, K., Si-Said Cherfi, S., Comyn-Wattiau, I.: Data Quality through Conceptual Model Quality - Reconciling Researchers and Practitioners through a Customizable Quality Model. In: ICIQ (International Conference on Information Quality), Hasso-Plattner-Institute Potsdam, Germany. pp. 61–74. France (2009)
39. Moody, D.L.: Theoretical and practical issues in evaluating the quality of conceptual models: current state and future directions. Data & Knowledge Engineering 55(3), 243–276 (2005), https://doi.org/10.1016/j.datak.2004.12.005
40. Mordal-Manet, K., Balmas, F., Denier, S., Ducasse, S., Wertz, H., Laval, J., Bellingard, F., Vaillergues, P.: The squale model: A practice-based industrial quality model. In: IEEE International Conference on Software Maintenance. IEEE (2009), https://doi.org/10.1109/icsm.2009.5306381
41. Nistala, P., Nori, K.V., Reddy, R.: Software quality models: A systematic mapping study. In: IEEE/ACM International Conference on Software and System Processes (ICSSP). IEEE (2019), https://doi.org/10.1109/icssp.2019.00025
42. Oriol, M., Marco, J., Franch, X.: Quality models for web services: A systematic mapping. Information and Software Technology 56(10), 1167–1182 (2014), https://doi.org/10.1016/j.infsof.2014.03.012
43. Pedrycz, W., Peters, J., Ramanna, S.: Software quality measurement: concepts and fuzzy neural relational model. In: IEEE International Conference on Fuzzy Systems Proceedings. IEEE World Congress on Computational Intelligence (Cat. No.98CH36228). IEEE (1998), https://doi.org/10.1109/fuzzy.1998.686259
44. Rawashdeh, A., Matalkah, B.: A new software quality model for evaluating cots components. Journal of Computer Science 2(4), 373–381 (2006)
45. Samoladas, I., Gousios, G., Spinellis, D., Stamelos, I.: The SQO-OSS quality model: Measurement based open source software evaluation. In: IFIP – The International Federation for Information Processing, pp. 237–248. Springer US (2008), https://doi.org/10.1007/978-0-387-09684-1_19
46. Satpathy, M., Harrison, R., Snook, C., Butler, M.: A generic model for assessing process quality. In: New Approaches in Software Measurement, pp. 94–110. Springer Berlin Heidelberg (2001), https://doi.org/10.1007/3-540-44704-0_8
47. Srivastava, S., Kumar, R.: Indirect method to measure software quality using CK-OO suite. In: International Conference on Intelligent Systems and Signal Processing (ISSP). IEEE (2013), https://doi.org/10.1109/issp.2013.6526872
48. Wagner, S., Lochmann, K., Winter, S., Deissenboeck, F., Juergens, E., Herrmannsdoerfer, M., Heinemann, L.: The quamoco quality meta-model. techreport TUM-I128, Technische Universität München, Institut für Informatik (2012), https://mediatum.ub.tum.de/doc/1110600/file.pdf

49. Yan, M., Xia, X., Zhang, X., Xu, L., Yang, D.: A systematic mapping study of quality assessment models for software products. In: International Conference on Software Analysis, Testing and Evolution (SATE). IEEE (2017), https://doi.org/10.1109/sate.2017.16
50. Yu, W.D., Radhakrishna, R.B., Pingali, S., Kolluri, V.: Modeling the measurements of QoS requirements in web service systems. SIMULATION 83(1), 75–91 (2007), https://doi.org/10.1177/0037549707079228

All links were last followed on January 17, 2022.

Algorithmic Classification of Layouts of BPMN Diagrams

Elias Baalmann[1] and Daniel Lübke[1[0000−0002−1557−8804]]

Leibniz Universität Hannover, Germany
e.baalmann@stud.uni-hannover.de,daniel.luebke@inf.uni-hannover.de

Abstract. Previous research is concerned with differences in BPMN diagram layout, e.g., with regards to understandability. However, layouts have neither been formally described nor their classification been automated. We aim at formalizing BPMN layouts and automating diagram layout classification for BPMN diagrams: We calculate sequence flow directions and encode them. By using regular expressions, these are clustered to diagram layouts. This results in a set of formally described BPMN layouts and a corresponding algorithm which we implemented in a tool. The results are very similar to previous work of manual layout classification on the GitHub process set. Researchers can use our definition when conducting BPMN diagram analysis and industry experts can use our tool for validating models against their layout guidelines.

Keywords: BPMN · Diagram Layout · Diagram Layout Formalization · Diagram Layout Detection · Flow Layout

1 Introduction

BPMN is the standard modeling language for business processes [1]. 2006 it was accepted as an OMG standard [4], the current version (BPMN 2.0) specifies multiple diagram types to model processes in different levels of detail [13]. Of the three specified types, only the process or collaboration diagram is considered in this paper. The BPMN is a documentation and communication tool which should allow readers to easily comprehend complex coherences. Thus, one key requirement for a model is understandability. Much research has been concerned with this topic recently, e.g., [5, 8, 7, 9–12].

One branch of BPMN understandability research is concerned with the layout of BPMN processes. The underlying hypothesis states that layout has a big impact on understandability. Besides small grained metrics like number of sequence flow crossings, the overall BPMN diagram layout has come into focus.

Up to now, layouts are only 'specified' by giving examples and appealing to the intuitive understanding of the reader ("top-down layout", "left-right layout"). This makes it hard to a) fully understand findings, b) replicate research and c) compare different research results. Furthermore, industry users cannot decide whether their diagrams are compliant to the latest research, thus preventing the implementation of scientists' recommendations for diagram layout.

J. Manner, D. Lübke, S. Haarmann, S. Kolb, N. Herzberg, O. Kopp (Eds.): 14$^{\text{th}}$ ZEUS Workshop, ZEUS 2022, Bamberg, held virtually due to Covid-19 pandemic, Germany, 24–25 February 2022, published at http://ceur-ws.org

In this paper, we want to outline a formal definition of prevalent flow layouts found in GitHub models and create an algorithm and followingly automated tool support for classifying BPMN diagram layouts. The term flow layout is chosen to clarify that the flow aspect of the Layout is considered. Other aspects, such as edge crossings or arrow lengths are not relevant here. Previous work, that investigated similar topics might use different terms such as “flow direction” [9] or “layout direction” [12]. But since ‘direction’ is to specific to describe the relatively complex layouts that are distinguished here, ‘flow layout’ seems to be a more adequate choice. By providing an implementation, that can classify diagrams based on formalized flow layouts we enable researchers to investigate statistics on large data sets such as “Does flow layout depend on the reading direction of the diagram author?” and practitioners to determine the most used layouts for example in their company and establish standards.

The paper is structured as follows: In the next section, we will introduce related work in the area of BPMN layouting, followed by a clear outline of our research questions in Sect. 3. In Sect. 4, we present the formalization and classification algorithm. The identified flow directions deducted from a large set of BPMN models found on GitHub are shown in Sect. 5 after which we conclude and provide an outlook.

2 Related Work

One of the first questions that arises in our context is how BPMN diagrams are laid out by practitioners. Effinger et al. [6, p. 400] state that “[i]n BPMN diagrams the flow direction is usually top-to-bottom or left-to right.” This statement is empirically validated by Lübke & Wutke [12, p. 52], who found that 79.52% of BPMN diagrams from their GitHub data set are laid out left-to-right. They also identified other layouts, like most prominently, top-down layouts and more complex layouts like multiline and snake layouts.

A more theoretical approach is taken by Figl & Strembeck. [10, p. 60] who state that “[b]asically, there are four main options for the overall direction: left-to-right, top-to-bottom, bottom-to-top, right-to-left.”, i.e., they take all four possible main directions as principal layout directions. However, they have also added that “zigzag models” should be subject to future research, thereby recognizing the use of more complex layouts in practice.

All modeling guidelines we found recommend left-to-right layouts, e.g., [2]. Even the BPMN specification itself favors left-to-right modeling [13, p. 42].

However, more recently, a study by Lübke et al. [11, p. 127] has shown that the understandability of large diagrams profits from more complex layouts like snake or multiline layouts in order to avoid the penalty of scrolling these diagrams on screen. For the case of smaller diagrams, this experiment found a slight advantage for left-to-right layouts in contrast to top-down layouts, affirming Figl & Strembeck’s earlier experiment. However, the findings are either minimal (some understandability metrics in the former experiment) or not significant (some metrics in the former experiment and all metrics in the latter experiment).

3 Research Questions

In this paper, we want to answer the following research questions.

RQ1: How can diagrams be classified automatically?
The automatic classification of flow layouts has many applications in research to answer questions such as "does the layout choice depend on the size of the diagram" and industry for example to enforce a style guideline.

RQ2: How can flow layouts be formalized objectively?
While formalizing all identified flow layouts is beyond the scope of this paper, we want to describe how such a formalization could be realized.

RQ3: Which are the most commonly used flow layouts, and are they worth formalizing?
By analyzing a large set of diagrams, we identify the most common layouts. Afterwards, we attempt to generalize the layouts to remove any biases that may be introduced by the data set.

4 Analyzing the Direction of Sequence Flow

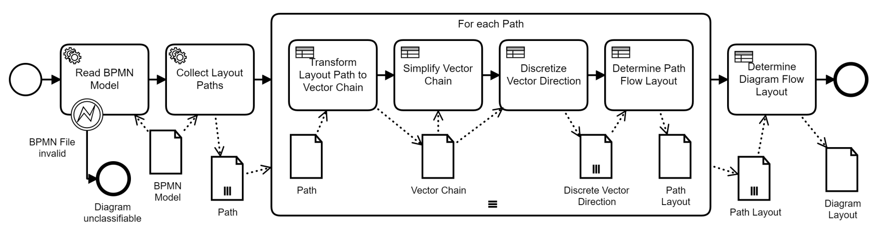


Fig. 1. Flow Layout Classification Algorithm

To classify BPMN diagrams automatically and thus answer the first research question, a modular algorithm is designed. Figure 1 illustrates the structure of the algorithm. First, the BPMN file is parsed and some sanity checks are performed to determine if the diagram can be classified at all. Since some BPMN editors do not serialize the diagrams in a standard way [1, p. 12], BPMN files exist, that are, e.g., missing layout data for the elements. For reference, an overview of the symbols and elements used in the BPMN is shown in the 'BPMN-Poster' by the BPM Offensive Berlin [3]. These diagrams cannot be classified with the current implementation. To determine the flow layout of the diagram, each path along sequence flows from any start element to any end element without loops is analyzed individually.

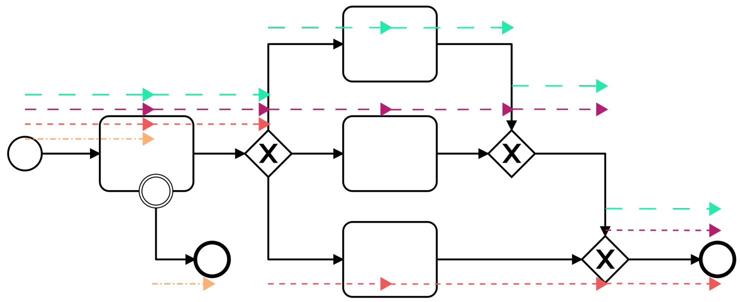


Fig. 2. BPMN Example with Vector Chain for each Layout Path in Colored Striped Arrows

The first of four tasks performed on the layout path is converting it to a vector chain. This is a list of the vectors between the centers of the flow elements on the layout path from the start to the end element. Some special cases need to be considered here. This is demonstrated by the example diagram shown in Fig. 2. Every path in the diagram is directed as straight as possible left to right. Gateways and boundary events do not allow for precisely straight layouts without overlapping the different paths.

To handle these cases, the (x or y) component of the vector between the centers of the elements (where the source element is a boundary event or a split, or where the target element is a join) that points in the orthogonal direction to the direction of the split, join or boundary event is set to zero. The direction of one element is determined by the following rules depending on the element type. **1. The element is a boundary event:** the direction is horizontal if it is connected to its parent at the top or bottom side; otherwise it is vertical. **2. The element is a split:** two cases are differentiated. Provided that the split element has an incoming sequence flow, the direction is horizontal, if the absolute value of the x component of the vector for the previous sequence flow is bigger than the absolute value of its y component. Otherwise it is vertical. The second case occurs if the split element is a start element. In this case, the direction is determined by constructing a vector that points into the average direction of the outgoing sequence flows of the split element (and comparing the x and y component as above). **3. The element is a join:** here the direction is calculated similar as for a split, just in opposite order. First, the next sequence flow is considered, and, if it does not exist (the join is an end element), the average direction of the incoming sequence flows is used. Join elements pose a problem, as the vector for the next sequence flow is not determined when the direction of the join is needed. To circumvent this issue, the direction of the first sequence flow, on the path from the join to the end element that is not entering another join, is used. If no such sequence flow exists, the average direction of the outgoing sequence flows of the join is used. The colored arrows in Fig. 2 showcase the vector chains which result from this step of the algorithm for each of the four layout paths. The vertical position of the vectors illustrates how each sequence flow is converted into a vector. In reality, only the vectors (x and y components) are relevant. Due to the rules explained above, all vertical (y) components of the vectors are set to zero, resulting in four straight vector chains from left to right.

In the next step, the vector chain is simplified by combining subsequent vectors with similar directions. The angle threshold is based on the number of discrete vector directions (NODVD) and calculated by the formula $\frac{360°}{\text{NODVD}}$. The vectors in the simplified chain which are a combination of at least two vectors get marked. Marked vectors are those that lay on a straight path in the diagram with at least one element between the start and the end of that path. This marking is important as it allows us to differentiate between otherwise indistinguishable flow layouts, e.g., Multiline and Snake (see Sect. 5). After that, each vector in the resulting simplified vector chain is mapped to a discrete direction. Currently, the NODVD used in the reference implementation is 16. This value was chosen because it felt natural, as a smaller NODVD like 8 would restrict the classification to much and a higher value like 32 would prevent many combinations of vectors and thus require diagrams to adhere very closely to a specific flow layout to be classified as that layout. The calculation of the angle threshold in the previous step guarantees that no two consecutive vectors have the same discrete direction.

To determine the flow layout for the path, the list of discrete vector directions is classified using regular expressions (see Sect. 5). In the end, the flow layout for the whole diagram is defined as the flow layout that occurs for most of the paths.

5 Classifying the Diagram Flow Layout

Rather than trying to describe every possible flow layout, our goal is to find commonly used layouts, formulate their distinguishing features, and build a classification algorithm that can detect these layouts and is extendable to possibly handle other layouts that are deemed worthy of classification in the future. Lübke and Wutke identified six flow layouts while manually classifying 5299 diagrams: Left-Right, Top-Down, Snake Horizontal, Snake Vertical, Multiline Horizontal and Multiline Vertical [12]. For this paper, a larger data set from GitHub (53984 diagrams) was used to identify possibly relevant flow layouts. The data set is a super set of the one used by the before mentioned authors. Because of the vast quantity, manual classification of all diagrams is unfeasible. Thus, the process shown in Fig. 3 was used. First, the algorithm described in Sect. 4 was applied

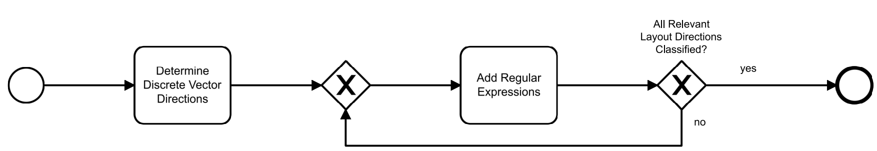


Fig. 3. Methodology

to all diagrams. The discrete vector directions determined by the algorithm were used to find common flow layouts. Though we established 16 distinct vector directions, only four distinct directions named north (N), east (E), south (S) and west (W) are used in the following examples to foster comprehensibility while keeping the regular expressions manageable. The marking of the vectors (see Sect. 4) is depicted by upper-case letters for marked vectors and lower-case for non-marked vectors. Grouping the diagrams by the discrete vector directions for each layout path showed that some sequences of discrete directions occurred in multiple diagrams. For instance, 55% of all diagrams had only layout paths with the direction E and 64 diagrams had only layout paths with the sequence EsW. Manual inspection of the grouped diagrams showed that multiple direction sequences exist for the same flow layout. E.g., the sequences EsW and EsWsE would both be considered Snake Horizontal. Thus, regular expressions were constructed to classify all variations of a particular flow layout. A simplification of the regular expression for Snake Horizontal would be **EsW(sEsW)*(sE)?(s)?**. This allows for an arbitrary number of lines. This way of formalizing flow layouts with regular expressions is our way to approach RQ2.

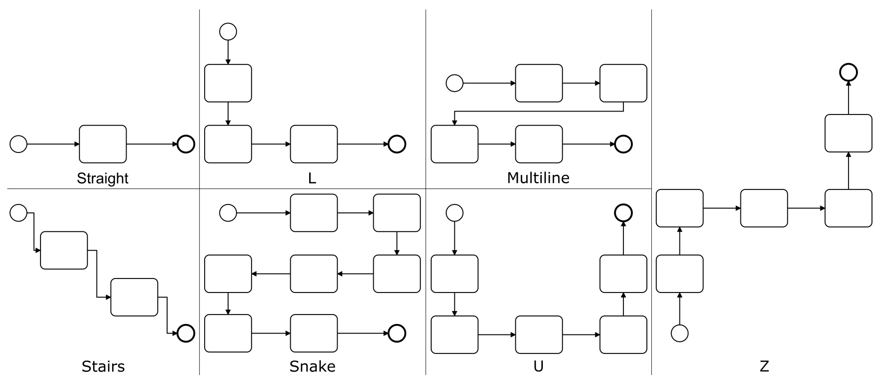


Fig. 4. The Seven Categories of Flow Layouts

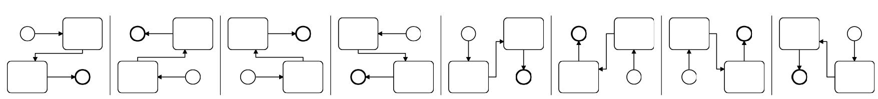


Fig. 5. Variants of Multiline Layouts

The seven categories of flow layouts that have been identified to answer RQ3, are: Straight, L, Multiline, Stairs, Snake, U and Z (Fig. 4). Multiple variants of flow layouts exist for each of these categories. Left-Right, Top-Down, Right-Left

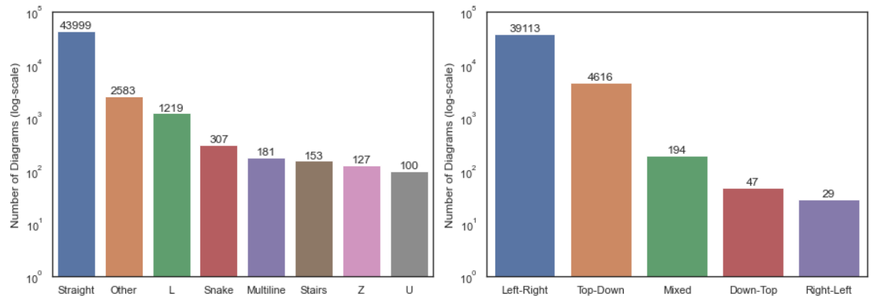


Fig. 6. Distribution of Flow Layouts in GitHub Data Set. Layout Categories (left) and Variants of Straight Flow Layout (right).

and Down-Top are the four variants of the straight category. Other categories can have more distinct flow layouts. One example is the multiline category. Eight variants can be differentiated as shown in Fig. 5. Even though not all these variants occur in the data set, they are all possible multiline layouts and should thus be identifiable. This extension allows us to generalize the usability of the classification by removing biases introduced by the data set as best as possible.

Figure 6 illustrates the distribution of the automatic classification for the large data set. The left diagram shows the seven flow layouts, the right diagram the four variants of the Straight flow layout. Diagrams that could be analyzed (no file reading error, no missing layout information,...) but not classified are shown as Other. The Mixed category in the right chart contains diagrams where two thirds of the layout paths where classified as Straight but no single variant of the Straight category occurred for this many paths. Not analyzable Diagrams are not shown, 5315 of the 53984 .bpmn files where not analyzable as they contained some syntactic error or missed layout information etc. The charts demonstrate how strongly the Straight flow layout is favored especially in the Left-Right and Top-Down variants (note the logarithmic scale).

6 Conclusion & Outlook

By analyzing a large data set of BPMN diagrams, we demonstrated that there are many flow layouts which are used for multiple diagrams. Subsequently, we identified seven categories of commonly used layouts. Formalizing flow layout with the use of regular expressions on discretized vector chains for each layout path is sensible. The formalization allowed us to create an algorithm and a tool implementation that showed promising results and can, e.g., be used by researchers to answer questions such as how diagram layouting differs between less experienced users and experts of BPMN. The tool can also be used by teams in the industry to validate models against their layout guidelines. This paper provides a concise overview of our work but fails to describe every detail of the complex subject that is layout detection. Examples of aspects that were considered but not explicitly reported in this paper are: the impact of swimlanes or subprocesses on flow layouts and how an accuracy score can be determined to indicate how exact a diagram is adhering to a particular flow layout. Furthermore some parts of the parameterization of the algorithm where chosen by feel and might appear arbitrary. For example determining the optimal NODVD based on more scientific metrics than 'does it feel natural' could be an interesting topic for future work to investigate.

References

1. Allweyer, T.: BPMN 2.0: Introduction to the standard for business process modeling. BOD - Books on Demand, Norderstedt, 2nd, updated and extended edition edn. (2016)
2. Birchler, A., Bosshart, E., Märki, M., Opitz, P., Pauli, J., Rigert, B., Sandoz, Y., Schaffroth, M., Spöcker, N., Tanner, C., Walser, K., Widmer, T.: eCH-0158 BPMN-Modellierungskonventionen für die öffentliche Verwaltung. WWW: https://www.ech.ch/dokument/fb5725cb-813f-47dc-8283-c04f9311a5b8 (September 2014)
3. BPM Offensive Berlin: BPMNPoster – www.bpmb.de (2011), http://www.bpmb.de/index.php/BPMNPoster
4. Chinosi, M., Trombetta, A.: Bpmn: An introduction to the standard. Computer Standards & Interfaces **34**(1), 124–134 (2012). https://doi.org/10.1016/j.csi.2011.06.002
5. Corradini, F., Ferrari, A., Fornari, F., Gnesi, S., Polini, A., Re, B., Spagnolo, G.O.: A guidelines framework for understandable bpmn models. Data & Knowledge Engineering **113**, 129–154 (2018). https://doi.org/10.1016/j.datak.2017.11.003
6. Effinger, P., Siebenhaller, M., Kaufmann, M.: An interactive layout tool for bpmn. In: 2009 IEEE Conference on Commerce and Enterprise Computing. pp. 399–406 (2009). https://doi.org/10.1109/CEC.2009.36
7. Effinger, P.: Layout patterns with bpmn semantics. In: Dijkman, R., Hofstetter, J., Koehler, J. (eds.) Business Process Model and Notation. pp. 130–135. Springer Berlin Heidelberg, Berlin, Heidelberg (2011)
8. Effinger, P., Jogsch, N., Seiz, S.: On a study of layout aesthetics for business process models using bpmn. In: Mendling, J., Weidlich, M., Weske, M. (eds.) Business Process Modeling Notation. pp. 31–45. Springer Berlin Heidelberg, Berlin, Heidelberg (2010)
9. Figl, K., Strembeck, M.: On the importance of flow direction in business process models. In: 2014 9th International Conference on Software Engineering and Applications (ICSOFT-EA). pp. 132–136. IEEE Computer Society, Los Alamitos, CA, USA (2014). https://doi.org/10.13140/2.1.3445.8247
10. Figl, K., Strembeck, M.: Findings from an experiment on flow direction of business process models. In: Kolb, J., Leopold, H., Mendling, J. (eds.) Enterprise modelling and information systems architectures. pp. 59–73. Gesellschaft für Informatik e.V, Bonn (2015)
11. Lübke, D., Ahrens, M., Schneider, K.: Influence of diagram layout and scrolling on understandability of bpmn processes: an eye tracking experiment with

bpmn diagrams. Information Technology and Management **22**(2), 99–131 (2021). https://doi.org/10.1007/s10799-021-00327-7

12. Lübke, D., Wutke, D.: Analysis of prevalent bpmn layout choices on github. In: Manner, J., Haarmann, S., Kolb, S., Herzberg, N., Kopp, O. (eds.) Proceedings of the 13th European Workshop on Services and their Composition (ZEUS 2021), Bamberg, Germany, February 25-26, 2021. CEUR Workshop Proceedings, vol. 2839, pp. 46–54. CEUR-WS.org (2021), http://ceur-ws.org/Vol-2839/paper9.pdf
13. Object Management Group: Business process model and notation (BPMN), version 2.0, https://www.omg.org/spec/BPMN/2.0/PDF

Preserving Data Consistency in Process Choreographies by Design

Tom Lichtenstein

Hasso Plattner Institute, University of Potsdam, Potsdam, Germany
Tom.Lichtenstein@hpi.de

Abstract. Data is essential for the execution of business processes. As today's organizations increasingly collaborate in process choreographies, data relevant to process execution is typically shared among participants. To avoid conflicts in the execution of process choreographies, the preservation of data consistency must be considered in the design of a choreography. However, current choreography modeling languages provide limited data modeling capabilities, thus potential conflicts arising from data inconsistencies at runtime may remain undetected during design time. Therefore, this paper motivates a framework allowing the design of data consistency-aware process choreographies.

Keywords: Process Choreography · Data Consistency · Design Time

1 Introduction and Motivating Example

In the current age of information, business processes and their outcomes rely on data and its manipulation during process execution [16]. As today's organizations increasingly collaborate in *process choreographies* [18], data relevant to business process execution is typically shared across choreography participants. To ensure the correct execution of the individual business processes involved in a process choreography, *data consistency*, i.e., a uniform view of data shared across multiple nodes [15], is a desirable property to be maintained among participants [12]. As the participants can only exchange data in the form of messages, maintaining data consistency is particularly challenging in process choreographies. Local changes to data shared with other participants can result in inconsistent views of the data, which may lead to conflicts such as unexpected interactions or data constraint violations on the receiving end. Since any deviation from the agreed interaction behavior can affect operations and incur costs, process choreographies require careful coordination [18]. Therefore, interaction behavior arising from data inconsistencies needs to be considered during process choreography design. The consideration of data consistency during design time is particularly relevant for process choreographies since in process orchestrations consistency can usually be enforced via central database management systems.

To support the design of process choreographies, graphical modeling languages such as *Business Process Model and Notation* (BPMN) [1] can be used. However, most contemporary choreography modeling languages provide only

J. Manner, D. Lübke, S. Haarmann, S. Kolb, N. Herzberg, O. Kopp (Eds.): 14th ZEUS Workshop, ZEUS 2022, Bamberg, held virtually due to Covid-19 pandemic, Germany, 24–25 February 2022, published at http://ceur-ws.org

limited data modeling capabilities. Therefore, potentially erroneous behavior resulting from data inconsistencies may remain undetected during design time. To illustrate the relevance of data management in choreography design, we consider a simple online ticket reservation choreography depicted in Figure 1. The choreography starts with a customer querying and selecting available seats for an event. Then, depending on the price offer, the customer decides whether to cancel the reservation or book the tickets. In the latter case, the ticket store confirms the reservation. While the interaction behavior is locally enforceable [19], data inconsistencies may arise with concurrent executions. Assuming that the ticket shop can sell only one ticket for each seat, if two customers select overlapping seats and the reservation of the first customer is confirmed, the second customer has an inconsistent view of the available seats. Furthermore, since the confirmation of the second reservation would lead to a constraint violation on the part of the ticket shop, compensation behavior is required to restore the consistency between the participants. Still, the need for compensation and the compensation behavior itself are both not evident from the given model. In the following, related work in the area of data consistency preservation is discussed and a framework for designing consistency-aware process choreographies is motivated.

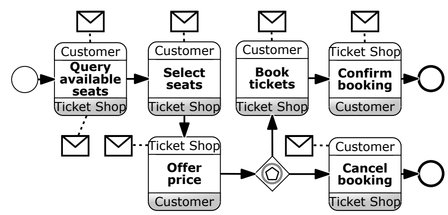


Fig. 1. Ticket reservation choreography between a customer and an online ticket shop

2 Related Work

Maintaining consistency in distributed environments is extensively studied in literature [3,6,10]. In particular, the preservation of data consistency is addressed by *consensus protocols* [11] which allow distributed nodes to agree on specific data values required for further computation, thus providing a consistent view of the data. The application of consensus protocols to process choreographies is discussed by Weber et al. [17]. The authors propose the use of blockchain technology [14] as an execution environment for process choreographies. To model and execute blockchain-driven choreographies, Ladleif et al. [9] refine BPMN 2.0 choreography diagrams [1] with blockchain-specific extensions. However, maintaining consensus in process choreographies may introduce synchronization overhead as not every change needs to be propagated to all participants. Moreover, in scenarios where inconsistencies rarely lead to conflicts, sustaining consensus among all participants may limit concurrent behavior and thus affect throughput. Incorporating data consistency management into the interaction design instead allows the choreography to be flexibly tailored to the use case.

Despite the existence of approaches that enrich choreography models with data management-specific information [5,13], the preservation of data consistency in process choreographies with concurrent instances has received little attention in research. Hahn et al. [5] introduce an approach that decouples the data flow between participants from the message flow by introducing a middleware that coordinates cross-partner data objects among corresponding participants. Yet, the handling of concurrent accesses to data shared by multiple instances is not further specified in their approach. Haarmann et al. [4] introduce a framework to analyze the use of shared data in process models by defining data access semantics. While the framework includes concepts for data consistency preservation, it does not address data exchange between individual participants. Finally, Kopp et al. [8] propose choreography spheres ensuring transactional behavior for all included activities that may belong to different processes. Still, deciding on an adequate scope of the spheres might prove challenging without data-related information, as overly large transactions could create avoidable overhead.

3 Towards Data Consistency in Process Choreographies

Since the integration of existing data consistency-preserving concepts into the design of process choreographies poses various challenges as outlined in the previous section, this paper proposes the concept of a framework supporting the collaborative design of data consistency-aware process choreographies based on the BPMN modeling language. The framework is supposed to enable the detection of potential data inconsistencies between participants at design time that may lead to conflicts at runtime. To achieve this, the framework will include functionality to specify and verify data consistency-related information in process choreography diagrams. Thus, participants should be enabled to define data consistency constraints at the interaction level (i.e., in the public process) that must be followed in the individual data management of each participant's local behavior (i.e., in the private process). The specifications will also allow participants to identify activities or data objects that may be affected by inconsistent data and therefore require careful attention in their design. In addition, data consistency criteria are introduced to formally verify the preservation of consistency throughout the choreography with regard to the specifications. The criteria should allow the detection of interaction behaviors that potentially lead to conflicts considering data inconsistencies that are not resolved in subsequent interactions. To design the criteria, existing formal definitions of consistency models [15] and data-aware choreographies [2,7] will be considered. Formal verification based on choreography models may also require extending the BPMN modeling language to include information about the individual management of the exchanged data. Eventually, the framework can be used to automatically derive enhanced process designs when potential conflicts are identified.

Based on this framework, a methodology will be developed to enable business engineers to iteratively redesign interaction behaviors prone to inconsistencies. The redesigned behavior is supposed to either ensure data consistency until the interactions affected by the data are completed or include appropriate compensation behavior to restore consistency. To realize the proposed concept, the following research questions need to be answered:

- Which information is required to reason about data consistency across multiple interactions while also considering concurrent behavior?
- How can data consistency-related information be integrated into the design of process choreographies?
- How can the behavior of process choreographies be enhanced to avoid conflicts caused by data inconsistencies?

4 Conclusion

This position paper discusses the need for considering the preservation of data consistency when designing process choreographies. Since current choreography modeling languages provide limited support for data modeling, potential conflicts due to data inconsistencies between participants may go undetected during design time. By integrating data consistency-related information into the design of process choreographies, corresponding conflicts can be detected and addressed to ensure a more reliable interaction behavior.

References

1. Business process model and notation (BPMN), version 2.0 (2011), http://www.omg.org/spec/BPMN/2.0
2. Corradini, F., Muzi, C., Re, B., Rossi, L., Tiezzi, F.: Animating multiple instances in BPMN collaborations: From formal semantics to tool support. In: Weske, M., Montali, M., Weber, I., vom Brocke, J. (eds.) Business Process Management, Lecture Notes in Computer Science, vol. 11080, pp. 83–101. Springer International Publishing, Cham (2018). https://doi.org/10.1007/978-3-319-98648-7_6
3. Gilbert, S., Lynch, N.: Brewer's conjecture and the feasibility of consistent, available, partition-tolerant web services. ACM SIGACT News **33**(2), 51–59 (2002). https://doi.org/10.1145/564585.564601
4. Haarmann, S., Weske, M.: Cross-case data objects in business processes: Semantics and analysis. In: Fahland, D., Ghidini, C., Becker, J., Dumas, M. (eds.) Business Process Management Forum, Lecture Notes in Business Information Processing, vol. 392, pp. 3–17. Springer International Publishing, Cham (2020). https://doi.org/10.1007/978-3-030-58638-6_1
5. Hahn, M., Breitenbücher, U., Kopp, O., Leymann, F.: Modeling and execution of data-aware choreographies: an overview. Computer Science - Research and Development **33**(3-4), 329–340 (2018). https://doi.org/10.1007/s00450-017-0387-y
6. Herlihy, M.P., Wing, J.M.: Linearizability: a correctness condition for concurrent objects. ACM Transactions on Programming Languages and Systems **12**(3), 463–492 (1990). https://doi.org/10.1145/78969.78972
7. Knuplesch, D., Pryss, R., Reichert, M.: Data-aware interaction in distributed and collaborative workflows: Modeling, semantics, correctness. In: Pu, C., Joshi, J., Nepal, S. (eds.) Proceedings of the 8th IEEE International Conference on Collaborative Computing: Networking, Applications and Worksharing. IEEE (2012). https://doi.org/10.4108/icst.collaboratecom.2012.250443

8. Kopp, O., Wieland, M., Leymann, F.: Towards choreography transactions. In: Oliver Kopp, Niels Lohmann (eds.) 1st Central-European Workshop on Services and their Composition, ZEUS 2009, Stuttgart, Germany, March 2-3, 2009. Proceedings. CEUR Workshop Proceedings, CEUR-WS.org (2009)
9. Ladleif, J., Weske, M., Weber, I.: Modeling and enforcing blockchain-based choreographies. In: Hildebrandt, T., van Dongen, B.F., Röglinger, M., Mendling, J. (eds.) Business Process Management, Lecture Notes in Computer Science, vol. 11675, pp. 69–85. Springer International Publishing, Cham (2019). https://doi.org/10.1007/978-3-030-26619-6_7
10. Lamport, L.: Time, clocks, and the ordering of events in a distributed system. Communications of the ACM **21**(7), 558–565 (1978). https://doi.org/10.1145/359545.359563
11. Lamport, L.: The part-time parliament. ACM Transactions on Computer Systems **16**(2), 133–169 (1998). https://doi.org/10.1145/279227.279229
12. Liu, X.: Data consistency enforcement on business process transactions. In: Service-Oriented Computing, pp. 394–399. Springer Berlin Heidelberg (2013). https://doi.org/10.1007/978-3-642-37804-1_40
13. Meyer, A., Pufahl, L., Batoulis, K., Fahland, D., Weske, M.: Automating data exchange in process choreographies. Information Systems **53**, 296–329 (2015). https://doi.org/10.1016/j.is.2015.03.008
14. Nakamoto, S.: Bitcoin: A peer-to-peer electronic cash system (2008)
15. Tanenbaum, A.S., van Steen, M.: Distributed Systems: Principles and Paradigms. Pearson Education (2008)
16. Tsoury, A., Soffer, P., Reinhartz-Berger, I.: Data impact analysis in business processes. Business & Information Systems Engineering **62**(1), 41–60 (2020). https://doi.org/10.1007/s12599-019-00611-5
17. Weber, I., Xu, X., Riveret, R., Governatori, G., Ponomarev, A., Mendling, J.: Untrusted business process monitoring and execution using blockchain. In: La Rosa, M., Loos, P., Pastor, O. (eds.) Business Process Management, Lecture Notes in Computer Science, vol. 9850, pp. 329–347. Springer International Publishing, Cham (2016). https://doi.org/10.1007/978-3-319-45348-4_19
18. Weske, M.: Business Process Management. Springer Berlin Heidelberg, Berlin, Heidelberg (2019). https://doi.org/10.1007/978-3-662-59432-2
19. Zaha, J., Dumas, M., Hofstede, A., Barros, A., Decker, G.: Service interaction modeling: Bridging global and local views. In: 2006 10th IEEE International Enterprise Distributed Object Computing Conference (EDOC'06). pp. 45–55. IEEE (10/16/2006 - 10/16/2006). https://doi.org/10.1109/EDOC.2006.50

Process Mining on Video Data*

Arvid Lepsien[1], Jan Bosselmann[1], Andreas Melfsen[2], and Agnes Koschmider[1]

[1] Group Process Analytics, Computer Science Department
Kiel University, Germany
{stu201368|stu206958}@mail.uni-kiel.de
ak@informatik.uni-kiel.de
[2] Institute of Agricultural Engineering
Kiel University, Germany
amelfsen@ilv.uni-kiel.de

Abstract. Disciplines like life and natural sciences could gain high benefits from process mining in terms of identifying anomalies in the process or supporting predictive analytics in what is being measured. These disciplines, however, mostly work with data at a much lower level of abstraction than business data. This paper discusses an approach for process mining on video data. As a use case, we applied our approach on video surveillance data of pigpens. Although, our process analytics pipeline from raw video data to a discovered process model has not yet been fully implemented, we are convinced that our approach is an essential contribution towards a (semi)automatic technique aiming to replace manual work.

Keywords: process mining · activity recognition · video labeling.

1 Introduction

Process mining is an established technique to give insights into data in terms of a structured order of activities (i.e., a process model). In this way, process mining allows identifying bottlenecks or compliance issues in business events. Mainly, process mining relies on business event data that is used as input to process mining algorithms and thus the data is expected to be on a high (business) abstraction level. Despite the success of process mining in the business context, process mining can provide an additional benefit to disciplines dealing with high volume and veracity of data. These disciplines like life or natural science have a high demand for a structured approach to answer process related questions like (1) what unknown processes are acting (i.e., did we find all processes that exist) and (2) whether the found processes actually work as thought.

Previously, we suggested approaches to discover process models from sensor event data [3] and "raw" time series data [12] with the purpose to give new insights into the data in terms of the identification of anomalies in the process flow

* Joined project with the Institute of Agricultural Engineering at Kiel University. Thank you for providing the data set and use case.

J. Manner, D. Lübke, S. Haarmann, S. Kolb, N. Herzberg, O. Kopp (Eds.): 14th ZEUS Workshop, ZEUS 2022, Bamberg, held virtually due to Covid-19 pandemic, Germany, 24–25 February 2022, published at http://ceur-ws.org

aiming to prevent unintended consequences. This paper presents our approach to discover process models from video data. As a use case, we applied our approach on video surveillance data of pigpens. So far, the behavior of pigs has been studied manually. Therefore, our approach aims to provide a (semi)automatic approach for pig behavior analysis in terms of health monitoring and understanding animal welfare. In this way, our approach makes a contribution to both questions (1) and (2) mentioned above.

The next section motivates why the use case is an appropriate starting point to develop techniques for process mining on video data.

2 Challenges

Compared to low-level raw data like sensor event data and time series used as input for activity recognition, video surveillance data of pigpens on the one hand eases the extraction of process activities, but on the other hand several challenges as mentioned below have to be overcome. Reasons facilitating the analysis are: (1) the behavior of pigs is limited to a few activities, which significantly simplifies activity detection compared to recognition of human activities in smart homes or smart factories. (2) A distinction between individual pigs is not necessary. This significantly simplifies the entity-centricity, which is challenging in smart homes where usually multiple objects are moving that need to be distinguished from each other.

To apply process mining on video data, however, requires bridging the following challenges: (1) no appropriate reference data set and labeled data exist. The freely accessible video-based data sets are mainly for object detection of other use cases like autonomous driving. Large computer vision libraries like Facebook AI Research's Detectron2 [10] grant access to trained neural networks, however, the detection of pigs is not covered by the commonly used COCO (Common Objects in Context) [5] and ImageNet [2] datasets, which provide a large amount of labeled images for object detection. We found two pig-specific data sets for detecting positions and orientation [7] and tracking [8], but these data sets do not suit process discovery purposes. Almost no process-specific data exist in the data set. Therefore a high manual effort is required since neither labeled data nor an appropriate data analysis pipeline exist for our use case. (2) Image quality significantly correlates with the analysis results. Image quality is affected by image resolution, camera angle and camera quality. We initially received a data set of very low quality. In addition, the data set was not representative (i.e., too short image sequences). Therefore, recording of a new data set was necessary with a camera installation from a different angle. (3) Image noise (e.g., due to randomly switching from day to night mode, camera pollution and distortion due to neighboring pigpens). Finally, we recorded a new representative data set of higher image quality and less image noise.

The next section presents our approach aiming to address the challenges mentioned above.

3 Approach

Figure 1 shows our approach to discover a process model from video data with the following sequential steps:

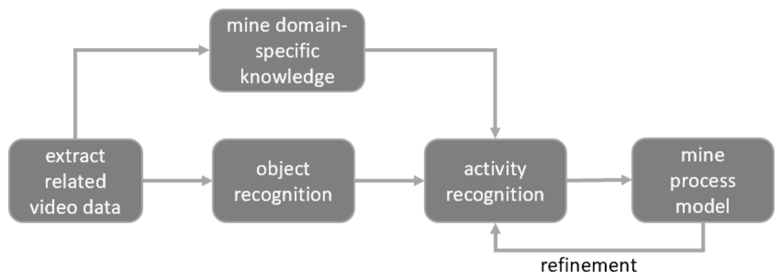


Fig. 1: The steps from video data to process discovery.

- *extract related video data from original data set*: we observed four pigpens, each with ten to twelve pigs, over a period of a few weeks. We recorded video material with a resolution of 1920x1080 pixels and 12.5 fps every day from 6:00 a.m. to 6:00 p.m. Mostly, the pig behavior does not change. Instead the pigs are in a kind of dormant phase. Many interesting actions only take place over a very short period of time, sometimes lasting just a few seconds. To detect related actions in our data set, we developed an algorithm measuring the movement intensity of a video sequence, which makes it easy to recognize the active phases of the pigs (see Figure 2). A spike in the chart indicates a new action.
- *mine domain-specific knowledge*: in this step we aim to identify context-related information that enhances action and object recognition. For instance, the location of the movement areas varies from groups of pigs. A group of young pigs would divide the pigpen differently than a group of older pigs. Thus, context information in terms of pig specificity is necessary in order to not distort the analysis results. Although, multiple data mining techniques have been used to mine domain-specific knowledge, again no specific technique exists for our use case. Therefore, the techniques have been tailored to our use case. First, we aimed to identify areas of high (visual) actions. The algorithm presented before has been enhanced to identify active movement areas. In general, a pigpen is divided into these three areas: sleeping/resting area, defecation area and feeding area. To automatically detect these areas, we used a slightly modified version of our motion intensity detection algorithm. We divided the images of a video into an area of 20x20 tiles and calculated the intensity of each tile over the entire video. Next, we converted the results into a 20x20 heatmap and easily identified the active

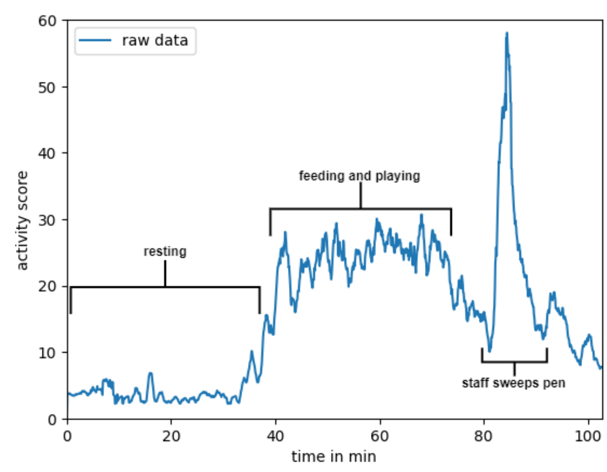
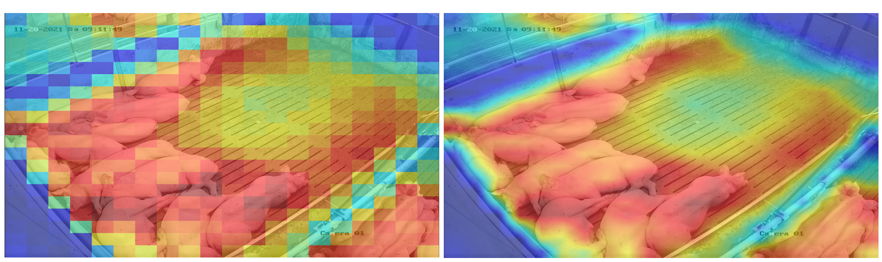


Fig. 2: Example output of our algorithm to determine movement intensity in a video sequence.

areas. Figure 3 shows an example. Then, knowledge of the positions of all pigs over time is used to create a heatmap of common pig positions. To do this, we calculate the midpoint of each bounding box detected on the video. The position heatmap is then constructed from the relative frequency of midpoints per heatmap bin (see Figure 4 for a log-normalized example output of this analysis).
Tracking traces have been clustered to find common movement patterns (and paths between common areas). Figure 5 shows an example of 150 movement trajectories extracted from one video. Different movement patterns can be observed, e.g. the pigs are mostly stationary in their resting area.



(a) Original movement intensity heatmap of a video divided into 20x20 rectangles.

(b) Smoothed movement intensity heatmap of the original version.

Fig. 3: Example of our algorithm to explore the movement intensity of areas in the video.

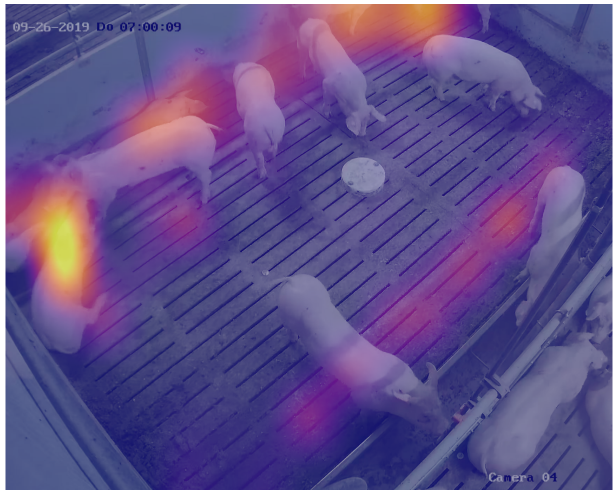


Fig. 4: Example output of our algorithm to determine common positions in a video sequence.

- *object recognition*: in this step, the video data is prepared for further analysis. First, an object detection is applied on the video. We chose YOLOv5 [4] due to its ease of use and ability to produce appropriate results with a relatively small amount of hand-labeled training data. Based on the object detection, (multi) object tracking has been used. This allows to analyze the same pig over multiple frames. We chose the DeepSORT (Deep Simple and Online Realtime Tracking) [9] algorithm to implement the tracking. DeepSORT is a well-established algorithm. The algorithm has been shown to work in a similar context to ours [1] and performs reasonably well on our data set without additional training. In the future, improved solutions for object detection and tracking could be applied to improve the quality of tracking results. However, many other solutions for the multiple object tracking problem require labeled tracking data for training. Since we aim to reduce manual labeling effort, the implementation of other tracking algorithms should be in proportion to manual effort. While a tracking dataset is available for pigs [8], it does not match our camera setup exactly. Also, there is not any labeled tracking data available when applying the analysis process in a different domain. If it was on purpose, the tracking results could be even used to localize individual pigs [1].
- *recognize activities in video*: The prepared video sequences and the associated position data from the tracking can be used as input for activity detection. In this step, also a model to learn visual features could be used. The learning process would have to be designed in a way where the features correspond to low-level events of the underlying process of the video. These low-level events can then be used to create event logs. While several techniques for

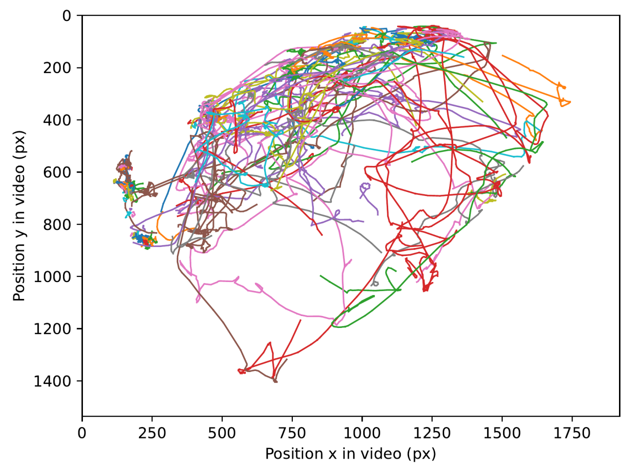


Fig. 5: Example output of our algorithm to track pig movements in a video sequence. In this example, 150 movement trajectories are shown.

pig activity recognition exist [11], they are either very specific to the unique properties of pigs or very specific to one type of activity (i.e. lying, standing, aggression). We choose not to use pig-specific techniques in this step to keep the approach generic.

- *discover process model*: the activities from the last step need to be aggregated/abstracted and enhanced with domain-specific knowledge (see step 2). Then, a case ID has to be created, e.g., according to the movement areas. A process model can then be mined from the event log.
- *refinement*: use the quality of the resulting process model to optimize the activity recognition and process model discovery.

4 Summary and Outlook

In studies of agricultural science alterations in behavior processes of pigs can be a helpful tool for analyzing and evaluating animal behavior, animal health and environmental impact. However, most approaches on identifying pig behavior based on video data only focus on single activities like e.g., lying, eating without analyzing the process. This paper suggested a process mining-pipeline to extract a process model from video data. As a use case, we applied our approach on video surveillance data of pigpens. Beside animal health, welfare and thermal comfort state, our approach can be used as a helpful indicator to evaluate and adjust climate conditions in mechanically ventilated barns. Likewise observations of activity and feed intake, which will vary depending on different climate conditions, supports the control of the above [6]. We see further use cases for our approach in medicine and material science that also handle large volume and veracity of data. Our approach of process mining on video data might be in medicine and material science for predictive analytics and outlier detection, which we believe to be more challenging than the current use case. Both assumptions that facilitate the analysis (i.e., low number of activities and entity-centricity) need to be bridged for an efficient solution.

References

1. Cowton, J., Kyriazakis, I., Bacardit, J.: Automated individual pig localisation, tracking and behaviour metric extraction using deep learning. IEEE Access **7**, 108049–108060 (2019)
2. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: Imagenet: A large-scale hierarchical image database. In: 2009 IEEE conference on computer vision and pattern recognition. pp. 248–255. Ieee (2009)
3. Janssen, D., Mannhardt, F., Koschmider, A., van Zelst, S.J.: Process model discovery from sensor event data. In: Process Mining Workshops - ICPM 2020 International Workshops, Padua, Italy, October 5-8, 2020. LNBIP, vol. 406, pp. 69–81. Springer (2020)
4. Jocher, G., Stoken, A., Chaurasia, A., Borovec, J., NanoCode012, TaoXie, et al.: ultralytics/yolov5: v6.0 - YOLOv5n 'Nano' models, Roboflow integration, TensorFlow export, OpenCV DNN support (Oct 2021). https://doi.org/10.5281/zenodo.5563715, https://doi.org/10.5281/zenodo.5563715
5. Lin, T., Maire, M., Belongie, S.J., Bourdev, L.D., Girshick, R.B., Hays, J., et al.: Microsoft COCO: common objects in context. CoRR **abs/1405.0312** (2014), http://arxiv.org/abs/1405.0312
6. Nasirahmadi, A., Hensel, O., Edwards, S., Sturm, B.: A new approach for categorizing pig lying behaviour based on a delaunay triangulation method. Animal **11**(1), 131–139 (2017). https://doi.org/https://doi.org/10.1017/S1751731116001208
7. Psota, E.T., Mittek, M., Pérez, L.C., Schmidt, T., Mote, B.: Multi-pig part detection and association with a fully-convolutional network. Sensors **19**(4), 852 (2019)
8. Psota, E.T., Schmidt, T., Mote, B., Pérez, L.C.: Long-term tracking of group-housed livestock using keypoint detection and map estimation for individual animal identification. Sensors **20**(13), 3670 (2020)
9. Wojke, N., Bewley, A., Paulus, D.: Simple online and realtime tracking with a deep association metric. In: 2017 IEEE international conference on image processing (ICIP). pp. 3645–3649. IEEE (2017)
10. Wu, Y., Kirillov, A., Massa, F., Lo, W.Y., Girshick, R.: Detectron2. https://github.com/facebookresearch/detectron2 (2019)
11. Yang, Q., Xiao, D.: A review of video-based pig behavior recognition. Applied Animal Behaviour Science
12. Ziolkowski, T., Koschmider, A., Schubert, R., Renz, M.: Process mining for time series data: Insights into ocean processes. Technical report (2022)

Visual Decision Modeling in IoT-Aware Processes

Yusuf Kirikkayis, Florian Gallik, and Manfred Reichert

Institute of Databases and Information Systems, Ulm University, Germany
{yusuf.kirikkayis,florian-1.gallik, manfred.reichert}@uni-ulm.de

Abstract. The enactment of real-world-aware business processes involves multiple interconnected devices. While the latter form the basis of the Internet of Things (IoT) and enable the exchange and collection of physical data via the Internet, Business Process Management (BPM) enables analyzing, modeling, implementing, executing, and monitoring business processes. In IoT-aware processes decision making may depend on the data provided by multiple IoT devices, which results in decision rules of complex structure. In this paper, we present two approaches for the visual modeling of decisions in IoT-aware processes. The first approach allows for the visual representation of complex decision rules by extending Business Process Model and Notation (BPMN) 2.0. The second approach separates decision logic from process logic using a drag&drop modeler. With both these approaches, IoT involvement in decision making becomes apparent and complex decisions can be represented in an intuitive and simple manner.

Keywords: BPM · Decision making · IoT-Aware Process

1 Introduction

The Internet of Things (IoT) is a network of physical objects that enable the collection and exchange of data via a network connection [1]. On one hand IoT devices allow for the acquisition and collection of data by sensors. On the other they support the response to an event by actuators. In particular, IoT enables bridging the gap between physical and digital world [6]. In turn, BPM enables optimizing, modeling, executing, and monitoring business processes [9]. By integrating BPM with IoT capabilities, process modeling, execution and monitoring can be enhanced. Furthermore, this integration enables the confirmation of manual steps through the use of IoT devices such as sensors and cameras [1]. In addition, low-level data generated by individual IoT devices (e.g., temperature, humidity or switch state) may be aggregated and combined, in high-level information that can be used for decision making. Moreover, high-level information enables BPM to understand the dynamic context of the physical world during process execution, which makes IoT-aware processes context-aware [6].

In contemporary approaches combining BPM and IoT, devices are added to process models as resources. Consequently, IoT data is used without linking it

J. Manner, D. Lübke, S. Haarmann, S. Kolb, N. Herzberg, O. Kopp (Eds.): 14th ZEUS Workshop, ZEUS 2022, Bamberg, held virtually due to Covid-19 pandemic, Germany, 24–25 February 2022, published at http://ceur-ws.org

to other contextual process data. The potential of gaining additional insights is not exploited [6]. As another challenge of using IoT devices for decision making, IoT-aware processes become decision-intensive [15]. In turn, this leads to an increased complexity due to a high number of decisions as opposed to IoT-unaware processes. In contemporary approaches, the involvement of IoT devices in decision making is not explicitly addressed. In addition, complex decision-making rules can only be represented in tables or gateways, which affects model readability and comprehensibility.

In this paper, we present two approaches for visually representing decisions in IoT-aware processes. One of these approaches is based on BPMN, whereas the other separates decision from process logic. The approaches aim to explicitly display the IoT devices involved. Furthermore, complex decision rules become more intuitive due to the chosen visualization. The remainder of this paper is organized as follows: Section 2 discusses related work. Section 3 presents the two solution approaches. Finally, Section 4 provides a summary and outlook.

2 Related Work

There exists a variety of approaches [2][4][7] that embed IoT into BPMN-based process models in terms of IoT tasks, physical entities, and resources. However, none of these approaches aggregates IoT data into higher-level information, which then can serve as input for decision making.

The Decision Model and Notation (DMN) standard provides a solution to separate decision from process logic [12]. In [14], the combination of BPMN and DMN is considered for IoT-aware processes. Note that the involvement of IoT devices actually neither becomes apparent in BPMN nor in DMN.

In [16], an approach for converting DMN models into DMN decision tables is presented. In particular, complex decision tables can be created automatically. Still, the problem remains that complex decision tables are difficult to read and comprehend. Furthermore, involved IoT devices are not explicitly represented or highlighted in [16].

3 IoT-aware visual decision modeling

This section presents the approaches for visual decision modeling in IoT-aware processes. Based on literature review and expert interviews we identified the challenge to model complex IoT decisions. To the lack of space we can not provide more details. First of all, a BPMN-based decision modeling approach is introduced, followed by an approach that separates decision modeling from the process logic using a drag&drop modeler.

3.1 Approach 1: Decision modeling in BPMN

To enable decision modeling directly in BPMN, the following extensions are introduced in BPMN (Figure 1); *IoT representative (1), IoT decision container*

(2,3&4), and *IoT decision table (5)*. Each *IoT decision container* contains an *IoT decision table*, which can be filled using the properties panel (7). The *IoT representatives* may be inserted into the *IoT decision container* via drag&drop. An *IoT representative* visualizes IoT sensors such as limit switches, temperature sensors, pressure sensors or light sensors.

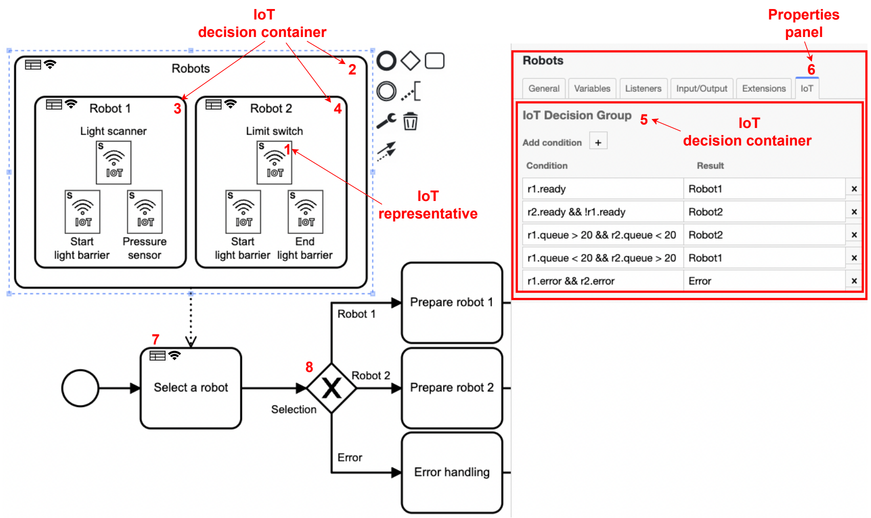


Fig. 1: IoT-aware decision modeling in BPMN

Figure 1 shows the modeling of the decision as well as its logic. First, the light scanner, start light barrier, and pressure sensor are queried (low-level information). The query results are then used for defining conditions in the decision table based on boolean algebra (high-level information). In turn, the results of Robot 1 (3) and Robot 2 (4) are used to define conditions for the root decision container Robots (2). The evaluation of this top-level decision, is then considered for controlling the flow of the corresponding process (8). When clicking on the *IoT decision* icon of the corresponding task (7), the root IoT decision container may be expanded/collapsed. Each IoT decision container may contain n IoT decision containers and likewise n IoT representatives, which results in a tree data structure. The latter is read bottom-up (i.e., the results of the nodes are passed from the bottom to the top) until the final decision is made by the root IoT decision container.

3.2 Approach 2: Separate Decision modeling

To separate the decision from the process logic, a drag&drop modeler is used. This allows the insertion and movement of elements on a modeling area (Figure 2). For modeling the decision logic the following elements are available; *IoT* *decision container (1), IoT representative and attached comparison (2), logical gates (4&5)* and *result module (6)*.

First, the IoT representative queries the physical sensor. Then, the attached condition is checked. Depending on the result, the IoT representative returns a corresponding boolean value. Each *IoT decision container* may contain any number of *IoT representatives*, *comparison modules*, and *decision containers (1)*. In turn, *IoT representatives* are connected to logical gates, which process multiple boolean input signals into a boolean output signal based on logical operators, such as conjunction (AND gate 4), disjunction (OR gate 5), or negation (3). The output signals of the logical blocks may either lead to a final result or be nested with other *logical blocks*. The final result may be of any data type and be represented by a result module (6). The transfer of the final decision from the drag&drop modeler to the BPMN model is done via an IoT decision task.

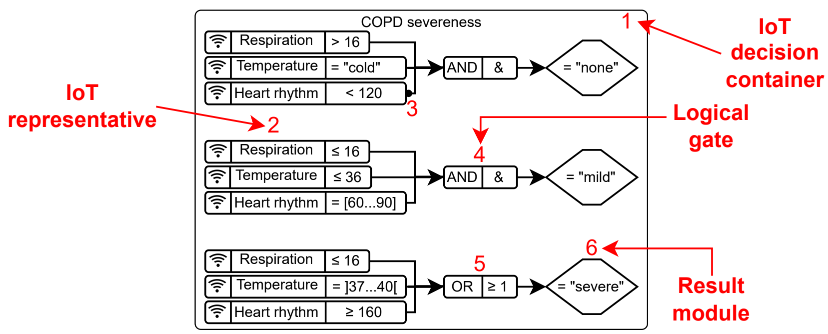


Fig. 2: IoT-aware decision modeling separated from BPMN

4 Summary and Outlook

We presented two approaches for visually modeling decisions in IoT-aware processes. The first one enables explicit decision modeling in BPMN using an IoT decision container with a corresponding IoT decision table. The IoT representatives are dragged and dropped into the IoT decision container. These representatives query the physical sensors and store the retrieved values in the decision table. As each IoT decision container has its own decision table, complexity is reduced. Each IoT decision container passes its final decision upwards until reaching the root IoT decision container. The latter makes a final decision, which then flows into the process and can be used in BPMN. As opposed to Approach 1, Approach 2 separates decision modeling from process modeling.

In future work, we will implement the two approaches and evaluate their usability and benefits in a case study. Moreover, we will develop an engine, for processing decision rules in IoT-aware processes. Finally, we will model IoT-aware processes from different domains to further verify the approaches.

Acknowledgments This work is part of the ZAFH Intralogistik, funded by the European Regional Development Fund and the Ministry of Science, Research and Arts of Baden-Württemberg, Germany (F.No. 32-7545.24-17/12/1)

References

1. Janiesch, C.: The Internet of Things Meets Business Process Management: A Manifesto. In: IEEE Systems, Man, and Cybernetics Magazine 6(4), pp. 34–44. (2020).
2. Suri, K., Gaaloul, W., Cuccuru, A., Gerard, S.: Semantic Framework for Internet of Things-Aware Business Process Development. In: 26th International Conference on Enabling Technologies: Infrastructure for Collaborative Enterprises, (2017).
3. Bazhenova, E., Zerbato, F., Oliboni, B., Wekse, M.: From BPMN process models to DMN decision models. In: Information Systems 83(12), (2019).
4. Cheng et al.: Modeling and Deploying IoT-Aware Business Process Applications in Sensor Networks ,(2019).
5. Combi, C., Oliboni, B., Zardiniy, A., Zerbato, F.: Seamless Design of Decision-Intensive Care Pathways. In: International Conference on Healthcare Informatics, (2016).
6. Song, R., Vanthienen, J., Cui, W., Wang, Y., Huang, L.:Context-Aware BPM Using IoT-Integrated Context Ontologies and IoT-Enhanced Decision Models. In: IEEE Conference on Commerce and Enterprise Computing, (2019).
7. Meyer, S., Ruppe, A., Hilty, L.: The Things of the Internet of Things in BPMN. In: Conference in Advanced Information Systems Engineering Workshops, (2015).
8. Hasić, F., De Smedt, J., Vanthienen, J.: Augmenting Processes with Decision Intelligence: Principles for Integrated Modelling. In: Decision Support Systems, (2018).
9. Dumas, M., La Rosa, M., Mendling, J., Reijers, H.: Fundamentals of Business Process Management 2nd edn. Springer-Verlag GmbH, Germany (2018).
10. Hasić, F., De Smedt, J., Vanthienen, J.: Developing a Modelling and Mining Framework for Integrated Processes and Decisions. In: On the Move to Meaningful Internet Systems, (2018).
11. Gardašević, G., Veletić, M., Maletić, N. et al.: The IoT Architectural Framework, Design Issues and Application Domains. In: Wireless Pers Commun 92, 127–148 (2017).
12. Object Management Group (OMG): Decision Model and Notation 1.3. (2021).
13. Koschmider, A., Mannhardt, F., Heuser, T.: On the Contextualization of Event-Activity Mapping. In: Business Process Management Workshops, (2018).
14. Hasić, F., Snoeck, M., Serral, E.: Comparing BPMN to BPMN + DMN for IoT process modelling: a case-based inquiry. In: 35th ACM/SIGAPP Symposium on Applied Computing, (2020).
15. Hasić, F., Asensio, E.: Executing IoT Processes in BPMN 2.0: Current Support and Remaining Challenges. In: 13th International Conference on Research Challenges in Information Science (RCIS), (2019).
16. Etinger, D., Simić, D., Buljubašić L.: Automated decision-making with DMN: from decision trees to decision tables. In: 42nd International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO), (2019).

BPMN in the Wild: A Reprise

Jasmin Türker[1,3], Michael Völske[2], and Thomas S. Heinze[3,4]

[1] Technische Universität Darmstadt
jasmin.tuerker@stud.tu-darmstadt.de
[2] Bauhaus-Universität Weimar
michael.voelske@uni-weimar.de
[3] Institute of Data Science, Jena
German Aerospace Center (DLR)
[4] Cooperative University Gera-Eisenach
thomas.heinze@dhge.de

Abstract. The Business Process Model and Notation (BPMN) language is the de facto standard for business process modeling and automation. While there exists a number of public model collections, there is still a need for ample datasets for empirical analysis on the usage and practice of BPMN. In this paper, we present our repository mining approach for generating a corpus of open source BPMN models by systematically analyzing software projects on GitHub.com. In contrast to our previous work, where we have limited the analysis to a 10% sample of GitHub.com, including 6.1 million projects, we here report the results for our new analysis of the whole 82.8 million projects hosted on GitHub.com at the time of conducting the research. We describe the resulting dataset, containing 79,713 distinct BPMN models from 18,534 open source projects.

1 Introduction

Empirical research on business process modeling can help gaining insight into the usage and practice of modeling languages like BPMN and thus answering questions about, e.g., frequently and rarely used language features or the recurrence of certain modeling styles and preferences. There is however still a lack of publicly available collections of realistic process models, which hinders empirical analysis [11,13]. While, traditionally, researchers have resorted to controlled experiments, surveys or case studies, with often smaller and homogeneous collections of process models, systematically searching for models in open source software projects provides a complimentary approach to tackle the lack of real-world data.

Recent work on mining BPMN model artifacts from open source software repositories hosted on Github.com has highlighted the utility of such data mining efforts to support BPMN tool development [9] and to investigate BPMN usage "in the wild" [1,8,14]. Nevertheless, previous efforts have been rather limited in scale. In this paper, we describe our effort to mine BPMN artifacts from as close to all public Github.com repositories as is reasonably possible, as represented by the repositories included in the most recent GHTorrent dump [5][1]. All told, we

[1] https://ghtorrent.org

J. Manner, D. Lübke, S. Haarmann, S. Kolb, N. Herzberg, O. Kopp (Eds.): 14th ZEUS Workshop, ZEUS 2022, Bamberg, held virtually due to Covid-19 pandemic, Germany, 24–25 February 2022, published at http://ceur-ws.org

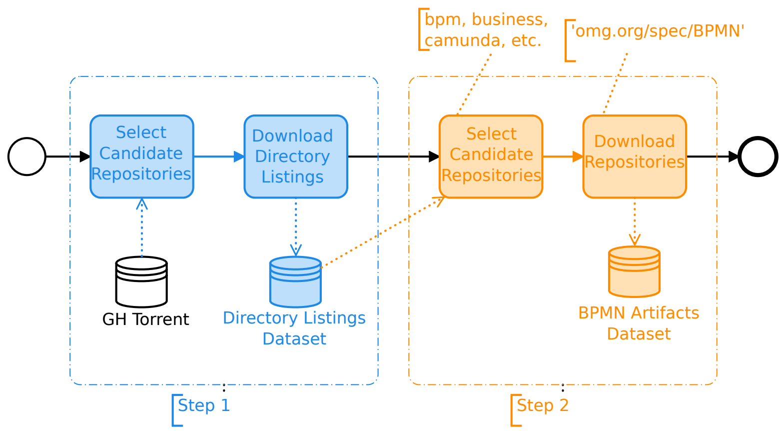


Fig. 1. The data mining pipeline

were able to identify 79,713 distinct BPMN artifacts and thereby can provide the to our knowledge most comprehensive collection of open source BPMN models.

In what follows, we describe our mining pipeline, as well as a preliminary analysis of the collected data. Our data mining pipeline follows a two-step approach, where we first identify interesting candidate repositories based on the names of the files and directories they contain, and inspect the files' contents only after this filtering. Since the initial collection of more than 82 million repositories' directory listings is the most time-consuming step, we make these listings available as a public resource for further research [18]. In the same spirit, we make links to the collected BPMN model artifacts available, as well [17].

2 Mining Approach for `Github.com`

Our data mining pipeline consists of two steps as shown in Fig. 1. In the first step, we retrieve all non-forked, non-deleted, public repositories from the latest *GHTorrent* database dump when conducting our research as of March 6th, 2021, and fetch a shallow and blob-less clone[2] of each repository from `Github.com`. We thus retrieve the file and directory names in the latest revision, and nearly nothing else, in less than 5 % of the time required to gather the same information via the REST API, which is limited to 5,000 requests per hour. The resulting *Directory Listings Dataset*[3] is collected over a span of about four weeks, and contains directory listings for the HEAD revisions of 82.8 million different software repositories. The corpus is available online [18].

To select candidate repositories for the second mining step, we queried the file and directory names in this dataset for the following keywords: *business, bpm,*

[2] `git clone --bare --single-branch --depth=1 --filter=blob:none`

[3] `https://zenodo.org/record/5856129` and `https://zenodo.org/record/5903352`

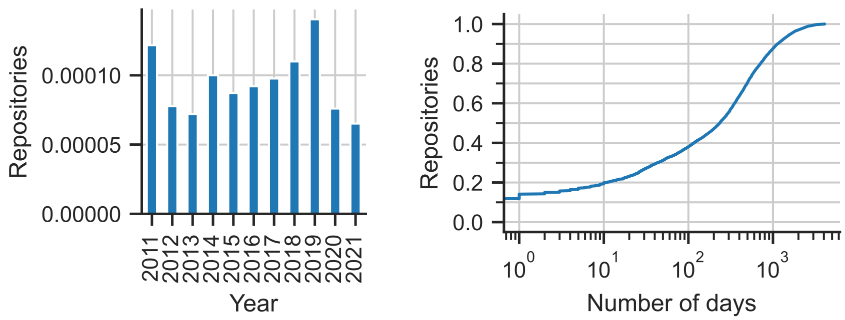


Fig. 2. Left: Fraction of GHTorrent repositories with BPMN artifacts (n=18,126); Right: Days between repository creation and time of latest update (n=12,235).

camunda, activiti, imixs, yaoqiang, modelio, signavio. The first two patterns are strongly connected with BPMN processes in general, and the latter six refer to popular modeling tools frequently used to design BPMN process models. This search matched at least one file or directory name in 1.1 million repositories. Of these, we again fetched a shallow clone, this time also including the blobs of the HEAD revision, i.e., the contents of its files. We filtered these repositories to contain at least one XML-serialized BPMN process model artifact, using the occurrence of the string `‘‘omg.org/sepc/BPMN’’` in at least one file as a heuristic. This left 18,534 repositories for inclusion in our *BPMN Artifacts Dataset*,[3] also available online [17]. Note that we currently only provide links to the identified BPMN process model artifacts and a Python code snippet for retrieving the models, similar to related research [6]. Due to the various licensing, copyright, and related constraints, further work will be required when the models are published on their own. The second mining step was completed in 3 days. A 135-node Kubernetes cluster was used for parallel processing in all steps.

3 Corpus and Preliminary Analysis

Fig. 2 (left) shows the fraction of repositories with BPMN artifacts relative to the total number of repositories in the GHTorrent database [5], broken down by repository creation year (number for 2021 based on the data available up to March). This differs notably from the overall number of repositories collected by GHTorrent: The latter had fewer repos in 2019 than in 2018, but more in 2020 than in either of the former years. Only 30 repositories in the dataset were created before 2011 (not shown in the plot), with the earliest creation date being June 2009. The empirical cumulative distribution for the number of days between repository creation and the most recent update is shown on the right of Fig. 2: While about 10 % of repositories were never updated again after a day past their creation, half of the repositories were still being updated after one year, and 20 %

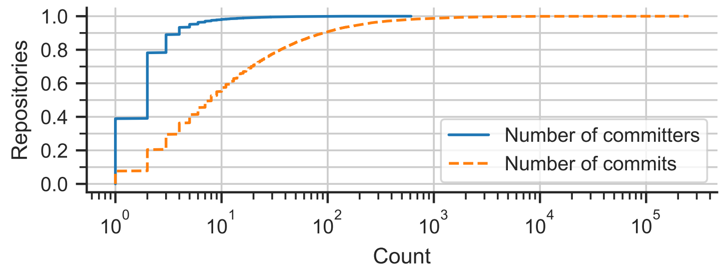


Fig. 3. Cumulative distributions for committers and commits per repository (n=15,773).

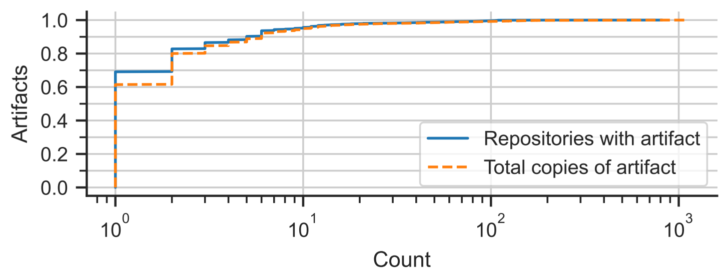


Fig. 4. Distributions for duplication among the 79,713 unique BPMN artifacts.

after two years. In the GHTorrent metadata, the creation date was unknown for 408, and the time of the last update for 6,299 of the repositories we collected; these missing data points are omitted from the above figures.

Fig. 3 shows the cumulative distributions for the number of committers and the number of commits per repository, for the subset of 15,773 repositories in our dataset for which this information is available in the GHTorrent database: Nearly 40 % of the repositories are single-committer projects, and 90 % have fewer than four contributors. Conversely, less than 10 % of the projects have only a single commit, and about 10 % have more than a hundred.

From the 18,534 collected repositories, we have identified 337,436 potential BPMN artifacts—XML files containing a URL for the BPMN 2.0 schema definition. Based on their SHA-1 hashes, we identify 79,713 unique BPMN artifacts; Fig. 4 shows the cumulative distributions for the number of copies of the artifacts and the number of repositories in which they appear. While SHA-1 file hashes provide a straightforward way to identify simple model duplicates, i.e., XML files with the same content, we acknowledge that finding similar though not exactly matching process models would require more laborious effort [7], which is however out of the scope of this paper. Similar to our previous findings in [9], we observe a large number of exact model duplicates. About 60 % of artifacts

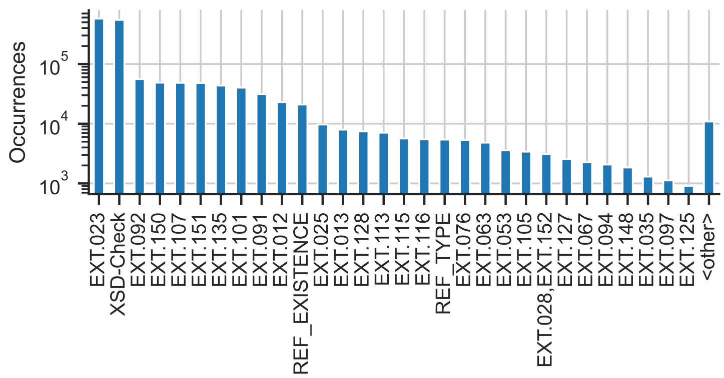


Fig. 5. Most common constraint violations detected by BPMNSpector among 60,779 unique, non-valid BPMN process models.

occur only once, and about 70 % in only one repository (even if multiple times); less than 5 % of all artifacts have ten or more copies across the dataset, meaning most of the copies are from a few thousand, frequently-copied artifacts. Based on several spot-checks, these are typically BPMN artifacts included as test cases with popular software libraries, which end up in many repositories, e.g., through accidentally-committed `node_modules` directories.

For a first look at the quality of the mined BPMN process models, we follow [9] and check all 79,713 distinct BPMN artifacts with the *BPMNSpector* [4] tool, which identifies 18,216 (23%) to be fully standards-compliant and 60,779 (76%) to contain at least one constraint violation (718 artifacts caused the tool to crash or hang); overall, 1,573,635 occurrences of 117 distinct constraint violations were found, the most common of which are shown in the histogram in Fig. 5. In terms of relative frequency, our results are rather similar to those reported by [9]: the most frequently violated constraints are Ext.023 (inconsistent definition of sequence flow; 36.5 % of violations / 38.6 % in [9]) and XsdCheck (violation of BPMN's normative XML schema; 34.8 % / 29.0 %) in both studies, and the next three most common violations, i.e., Ext.107, Ext.092, Ext.101 (inconsistent definition of sequence flow at start/end events, missing/ambiguous sources of data associations), all appear in our top eight, albeit in a slightly different order.

4 Related Work

Empirical research on business process modeling [13] often comes in the form of controlled experiments, surveys or case studies using a limited number of process models, usually with up to hundreds of process models, e.g., [3,12]. In recent years, though, various community initiatives have been started to increase the number of process model collections available for empirical analysis [11,10,2].

The *BPM Academic Initiative*[4], as a notable effort for business process modeling, provided a platform for creating and sharing business process models in academic teaching. In their 2011 publication [11], the initiators reported on 1,903 different process models, including BPMN, created by ca. 4,500 different users and covering various complexities and language features. While the recent number of models is 29,810[5], unfortunately, data collection is discontinued. Various similar platforms and datasets exist, including *GenMyModel*[6] with 12,575 BPMN models, *RePROSitory* [2] with 593 models, and *Camunda BPMN for research*[7] with 3,721 models, respectively. Another process collection has been created in the *BenchFlow* project [16], where companies donated process models for process engine benchmarking. The dataset described in [16] included overall 8,363 models, with a share of 64% of BPMN, but the collection is not publicly available. To the authors knowledge, the corpus presented in our paper with 79,713 distinct process models is the largest publicly available BPMN dataset. Not a dataset by itself, but an open-source business process analytics platform is offered by *Apromore*[8]. The platform has originally been conceived as process model repository [15] and now offers an extensible, service-based framework for a wide range of tools supporting the whole business process lifecycle, including process discovery, analysis, implementation, and monitoring.

In their previous work on repository mining for BPMN, the authors of this paper considered a sample of 10% of software projects on `GitHub.com` [8,9], resulting in a corpus of 8,904 BPMN model artifacts. An analysis proved the diversity of the corpus but also stressed the high number of model duplicates as well as model flaws as detected by BPMNspector [4]. While being a significant sample, the corpus presented and made publicly available in this paper provides ten times more models and thus a more complete and comprehensive picture about the practice of BPMN in open source software projects. The authors' prior dataset has been used in empirical studies on BPMN since then. In a recent study, BPMN models from various public process repositories, including the `GitHub.com` dataset [9] have been combined into a collection of 25,590 models and analyzed according to their complexity and usage frequency of BPMN language elements. Similarly, Lübke and Wutke investigate process layout choices in open source BPMN process models based on the corpus in [14].

5 Conclusion

In this paper, we introduce our approach to extract two hiqh-quality corpora from `Github.com`. Our *Directory Listings Dataset* is publicly available and may be an interesting starting point for research across various fields. Due to the amount of BPMN artifacts it contains, our *BPMN Artifacts Dataset* can contribute to

[4] `http://fundamentals-of-bpm.org/process-model-collections/`
[5] Numbers are reported for the time of writing this paper (25 January 2022).
[6] `https://www.genmymodel.com`
[7] `https://github.com/camunda/bpmn-for-research`
[8] `https://apromore.org`

the investigations on the usage and practice of BPMN process models. Subject of future work will be the exhaustive analysis of the retrieved corpus to further characterize the usage of BPMN in open soource software projects with respect to, e.g., complexity of process models, usage of different BPMN language constructs, utilized modeling tools, correlation of modeling tool with standards compliance, relation of repository metadata and BPMN artifacts, etc.

References

1. Compagnucci, I., Corradini, F., Fornari, F., Re, B.: Trends on the Usage of BPMN 2.0 from Publicly Available Repositories. In: BIR 2021. LNBIP, vol. 430, pp. 84–99. Springer (2021)
2. Corradini, F., Fornari, F., Polini, A., Re, B., Tiezzi, F.: RePROSitory: a Repository Platform for Sharing Business PROcess modelS. In: BPM PhD/Demos 2019. pp. 149–153. CEUR (2019)
3. Fahland, D., Favre, C., Jobstmann, B., Koehler, J., Lohmann, N., Völzer, H., Wolf, K.: Instantaneous Soundness Checking of Industrial Business Process Models. In: BPM 2009. pp. 278–293. Springer (2009)
4. Geiger, M., Neugebauer, P., Vorndran, A.: Automatic Standard Compliance Assessment of BPMN 2.0 Process Models. In: ZEUS 2017. CEUR Workshop Proceedings, vol. 1826, pp. 4–10. CEUR-WS.org (2017)
5. Gousios, G.: The GHTorent dataset and tool suite. In: MSR 2013. pp. 233–236. IEEE (2013)
6. Hebig, R., Quang, T.H., Chaudron, M., Robles, G., Fernandez, M.A.: The Quest for Open Source Projects that use UML: Mining GitHub. In: MODELS 2016. pp. 173–183. ACM (2016)
7. Heinze, T.S., Amme, W., Schäfer, A.: Detecting semantic business process model clones. In: ZEUS. CEUR Workshop Proceedings, vol. 2839, pp. 25–28. CEUR-WS.org (2021)
8. Heinze, T.S., Stefanko, V., Amme, W.: BPMN in the Wild: BPMN on GitHub.com. In: ZEUS 2020. CEUR Workshop Proceedings, vol. 2575, pp. 26–29. CEUR-WS.org (2020)
9. Heinze, T.S., Stefanko, V., Amme, W.: Mining BPMN Processes on GitHub for Tool Validation and Development. In: BPMDS/EMMSAD@CAiSE 2020. LNBIP, vol. 387, pp. 193–208. Springer (2020)
10. Ho-Quang, T., Chaudron, M.R.V., Robles, G., Herwanto, G.B.: Towards an Infrastructure for Empirical Research into Software Architecture: Challenges and Directions. In: ECASE@ICSE 2019. pp. 34–41. IEEE (2019)
11. Kunze, M., Luebbe, A., Weidlich, M., Weske, M.: Towards Understanding Process Modeling – The Case of the BPM Academic Initiative. In: BPMN 2011 Workshops. pp. 44–58. Springer (2011)
12. Leopold, H., Mendling, J., Günther, O.: Learning from Quality Issues of BPMN Models from Industry. IEEE Software **33**(4), 26–33 (2016)
13. Lübke, D., Pautasso, C.: Empirical Research in Executable Process Models. In: Empirical Studies on the Development of Executable Business Processes, pp. 3–12. Springer (2019)
14. Lübke, D., Wutke, D.: Analysis of Prevalent BPMN Layout Choices on GitHub. In: ZEUS 2021. CEUR Workshop Proceedings, vol. 2839, pp. 46–54. CEUR-WS.org (2021)

15. Rosa, M.L., Reijers, H.A., van der Aalst, W.M.P., Dijkman, R.M., Mendling, J., Dumas, M., García-Bañuelos, L.: APROMORE: an advanced process model repository. Expert Syst. Appl. **38**(6), 7029–7040 (2011)
16. Skouradaki, M., Roller, D., Leymann, F., Ferme, V., Pautasso, C.: On the Road to Benchmarking BPMN 2.0 Workflow Engines. In: ICPE 2015. pp. 301–304. ACM (2015)
17. Türker, J., Völske, M., Heinze, T.: Github BPMN Artifacts Dataset 2021 (Jan 2022). https://doi.org/10.5281/zenodo.5903352
18. Türker, J., Völske, M., Heinze, T.: Github Directory Listings Dataset 2021 (Jan 2022). https://doi.org/10.5281/zenodo.5856129

A Use Case-based Investigation of Low-Code Development Platforms

Robin Lichtenthäler, Sebastian Böhm,
Johannes Manner, and Stefan Winzinger

Distributed Systems Group, University of Bamberg, Germany

Abstract. Rapid application development without profound development skills are the stated advantages of the recent trend in Low-Code Application Development. In a time-constrained experiment we investigate these promises for three Low-Code platforms by implementing a practical use case. While this was in fact feasible in a short time for major parts of our use case, the platforms differ significantly and a technical understanding is still required for non-trivial applications.

Keywords: Low-Code, Process Automation, Use Case, Mob Programming

1 Introduction

Low-Code application development is a recent trend in the software industry. It is expected to become the technological basis for an increasing amount of newly developed applications, as predicted in the *2021 Gartner Magic Quadrant for Enterprise Low-Code Application Platforms* [9]. The promised advantages of Low-Code platforms are that on the one hand less skills are required for development and on the other hand applications can be developed much faster [6], because Low-Code platforms enable application development at a higher level of abstraction, often based on visual programming [6]. By integrating cloud computing, Low-Code platforms can furthermore support application deployment in an automated fashion on reliable and scalable cloud infrastructures [8]. A variety of platforms have emerged from both vendors specialized on Low-Code (e.g., OutSystems[1], Mendix[2], or Appian[3]) and established cloud providers (e.g., Microsoft[4]). Nowadays, the interest in Low-Code platforms also increases in academic research. Several studies focusing on single platforms in-depth have been published [2, 5, 7] and we are aware of one study [8] aiming to compare several Low-Code platforms to provide a broader overview of the existing platforms and their scope of features. Also evaluations of how well certain features facilitate application development have been performed. Henriques et al. [2] evaluated

[1] `https://www.outsystems.com`
[2] `https://www.mendix.com`
[3] `https://appian.com/`
[4] `https://powerapps.microsoft.com`

J. Manner, D. Lübke, S. Haarmann, S. Kolb, N. Herzberg, O. Kopp (Eds.): 14th ZEUS Workshop, ZEUS 2022, Bamberg, held virtually due to Covid-19 pandemic, Germany, 24–25 February 2022, published at `http://ceur-ws.org`

the process modeling language of OutSystems in a structured way and Sahay et al. [8] discussed their experiences as a side aspect. An investigation focusing specifically on the aspect of rapidness of development while considering several platforms in comparison, has however not been done yet. Therefore, we performed a qualitative investigation of several platforms by implementing a realistic use case in a constrained experimental setup. We aim to investigate the promises of easy, rapid application development and deployment by evaluating to which extent this is possible with a selected set of platforms. This is summarized by our research question:

RQ: To what extent do Low-Code Development platforms enable rapid application development despite low prior experience?

In the following, we describe foundations of Low-Code platforms and our use case in Sect. 2. Our approach is explained in Sect. 3. In Sect. 4 we state the outcomes of our investigation and discuss them with regard to our research question in Sect. 5, before concluding our work in Sect. 6.

2 Foundations

Low-Code Development Platforms are cloud environments in which applications can be created and hosted based on the technology stack provided by the platform vendor. The core characteristic is that vendors try to abstract as much as possible from the technical details (the coding aspect) of application development [3, 8] while still enabling developers, called *citizen developers* in this context [3, 8], to implement their specific use case. While a higher level of abstraction and a more constrained development environment simplify application development, the degree to which applications can be customized for a specific use case is limited. However, a more complex development environment is more difficult to use but would allow for a specific customization. Therefore, platform vendors apply a range of techniques to balance this trade-off, namely visual programming [4, 8], domain-specific languages [3], model-driven engineering [1, 3], or pre-built components and templates [3]. Internally, platform vendors make use of cloud computing to allocate resources for applications on-demand and in an automated and scalable way as described in more detail by Sahay et al. [8]. They also provide a comparison of current Low-Code Development Platforms [8]. Differences between the platforms originate from the diverse backgrounds of platform vendors. Regarding suitable use cases and applications, Luo et al. [6] found out that most developers, who discuss such platforms online, build mobile and web applications. A focus is on process automation [6] and business-centric applications [6], such as Customer Relationship Management (CRM) or Content Management Systems (CMS).

From our point of view, our use case therefore fits well into the context of Low-Code applications, as we want to automate the process of a student registering for a thesis at a university's chair. Similar to CRM, the student can be seen as a kind of customer managed by the chair and similar to CMS different documents are involved to describe the thesis topic and for the registration of a thesis. We have modeled the use case in its current manual form using Business Process Model and Notation (BPMN) to identify potentials for automation[5]. Our envisioned process includes a web-based application which manages and automates parts of the process. The beginning of this process is shown in Fig. 1. For a better understanding, the process can be divided into three Sub Processes (SP): Request Submission (SP1), Chair Notification (SP2) and Acceptance Decision (SP3).

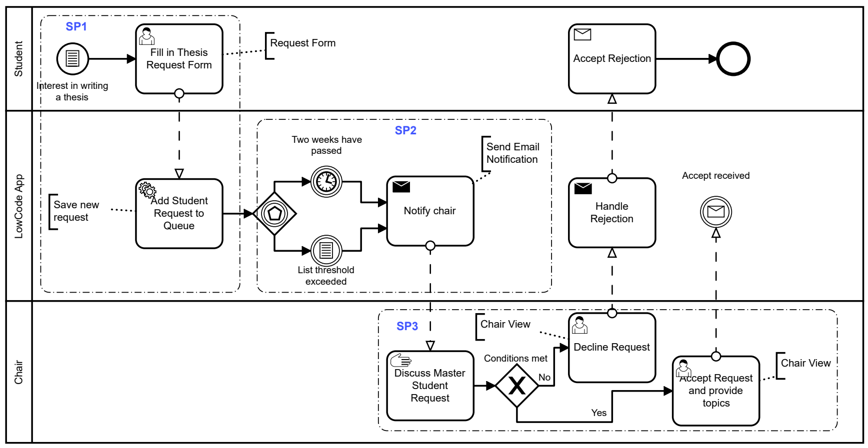


Fig. 1. Thesis Registration Process at the University of Bamberg (excerpt).

Although this excerpt is only the beginning of the whole thesis registration process, it is representative for the functional requirements of the whole process, covering all layers of a typical web application: User interfaces (UI), data storage (DS) and business logic (BL). Starting in SP1, at the UI level, students should be able to fill in a **Form** with their personal information, such as name, email, and course of studying. In addition, they should be able to upload a file that contains further information or ideas for potential thesis topics. The app should validate the submitted form and **Save** the **Data** in a storage layer, enriched with the date of submission and with a status "New". In SP2 at the BL layer, there are two possible events which represent **Triggers** for a notification. If either two weeks have passed since the submission of the oldest request or the list of new requests has exceeded a certain threshold, the app should detect this and thereupon send an **Email** to notify the members of the chair that a meeting should be held to discuss the new requests. Finally, in SP3, after a chair meeting, it should be possible for a chair member to **View** a list of all requests and to either decline or accept each request through the click of a button. The **Logic** layer behind it should validate that only new requests can be accepted or declined and change the status accordingly by performing an **Update** on the **Data** in the DS layer.

[5] `https://github.com/uniba-dsg/low-code-use-case`

3 Methodology

Our methodology mainly follows an experimental approach as we perform a qualitative study by implementing the previously introduced process model. However, for a general understanding and market overview of Low-Code platforms, we searched for empirical comparisons by using ACM, IEEE, and Scopus.

Our literature search yielded a comprehensive comparison of Low-Code platforms by Sahay et al. [8]. From their list of platforms, we selected those that are still available, offer *Process Automation* as a feature, and can be tested within a free tier offering. Hence, we selected OutSystems, Microsoft Power Apps, and Appian because only these fulfill our criteria. For the implementation, we strictly followed the process model. We documented the implementation process for all platforms, w.r.t. the perceived achievements and obstacles, structured by the defined subprocesses. We applied *Remote Mob Programming*[6] as a development technique. The authors of the study were also its participants. This technique offers distributed collaboration for a small team of developers (at least three). Hereby, the members of the team are working together at the same time remotely, e.g., via video conferencing. One by one shares the screen with the current development environment and works actively as a so-called Typist who is executing instructions given by the rest of the Mob. In regular intervals (we set 15 minutes instead of the recommended 10 minutes), the advance in the development is handed over to the next member who becomes the Typist. We followed this procedure for three days by spending 6h/day (in line with the recommendation) for each of the platforms. Remote Mob Programming as a development technique was selected because we believe that compared to an approach of working with a new platform alone, the Mob can prevent situations of a single developer getting stuck. In such situations others can contribute their ideas and discuss the next steps or potential solutions to overcome this. This can accelerate the development process and discharge the involved developers. For the assessment, we used a 3-step scaling system to rate the degree of fulfillment for our requirements. All Mob members had no prior experience with Low-Code platforms in general. Only preparations like account creation and software installation were performed beforehand.

4 Implementation Results

In this section, we summarize the results of the experiment which can be seen in Table 1. This includes the aspects of our use case implemented and the problems that occurred during implementation.

4.1 Appian

A unique characteristic of Appian is its custom scripting language. All building blocks (views, data, logic) are objects in the appian designer. The integration

[6] `https://www.remotemobprogramming.org/`

Table 1. Subprocesses from Figure 1 implemented: Fully implemented (●), partially implemented (◐), and not implemented (○).

	SP1 (Request)		**SP2 (Notification)**		**SP3 (Decision)**		
	Form	Save Data	Triggers	Email	View	Logic	Update Data
Appian	◐	◐	◐	●	◐	●	●
OutSystems	●	●	◐	●	●	●	●
PowerApps	●	●	◐	◐	◐	○	○

between these objects was hard to grasp at the beginning of our experiment. Some features were only realizable when implementing custom boolean expressions or configuring elements in a way where the graphical editor had no predefined option and we had to use the scripting language. This complicated the development process in our one-day-workshop. Reading the documentation and understanding the scripting language was often necessary to proceed. Appian integrates with public cloud providers and their systems as well as to other external sources like ERM/CRM systems. A deployment was implicitly done when storing the different elements.

The UI implementation was well supported via the graphical editor. Some aspects were not implementable with the editor, e.g., when more mature features like grouping input fields (SP1.Form) or a display of all students with their status buttons (SP3.View) were needed. The logic is defined in a process-like interface similar to the options in BPMN where tasks can be configured via the scripting language. Triggers can start these processes. For mail integration, we used the predefined Appian service. The timer trigger did not work in our experiment whereas the element size check of the already stored entities succeeded (SP2.Trigger). We stored primitives through our forms, but faced difficulties storing files (SP1.Save Data). With our gained knowledge, SP3 was implemented without issues.

4.2 OutSystems

In contrast to Appian and PowerApps, OutSystems requires locally installed client software which include self-explanatory hints, tooltips and autocompletion features. All study participants used the same account with the same log-in credentials. Only the desktop application was used since the synchronization worked properly via the cloud. Particularly noteworthy was the deployment which could be done by a single click without any further knowledge.

After some time, we found out, that forms can be created based on the data model. OutSystems' storage solution uses one of several predefined SQL servers where files can be stored as blob entries in a table. Therefore, we defined the data model first and created our views by attaching the data table to it. Additional fields of other tables were created via dragging and dropping the corresponding input fields from the tool bar. The input fields were automatically integrated in a submit form which already included the logic to persist the data. OutSystems was the only platform, where we managed the mentioned file upload and download (SP1.Save Data). The time based trigger did not work in our use case, whereas the *list threshold exceeded* trigger did (SP2.Trigger). The mail integration in SP2 was possible with an existing SMTP server. A consistent look and feel of the platform helped to implement SP3 without facing any challenges.

4.3 PowerApps

PowerApps is fully integrated with other Microsoft products and the Azure cloud. To make use of these integrations, code writing skills are required for different layers of the Low-Code platform (e.g., for validation or UI workflows). The documentation, however, is structured on an individual product level, making it difficult to find details and best practices on the integration with PowerApps. Another difficulty were unclear error messages, e.g., "Schemas do not match" or "HTTP request failed" in the context of integrating a Flow with a UI button.

For SP1, the form and data storage was realizable after some starting problems. Data storage (Dataverse) and the business logic execution (Flows engine) are separate, modular products which complicate building a full stack app in our experimental setting. As for the other platforms, business logic can be executed based on triggers which were difficult to integrate with the Dataverse tables. Triggering the mail worked partially, see Table 1, as we were able to send a mail based on a timer but not on the exact conditions of the list threshold. Mail integration was possible with an existing SMTP server. Due to the problems we faced, especially with the integration of the different products, there was not enough time left to implement SP3, apart from creating the *accept* and *decline* button in the view.

5 Discussion

As an initial answer to our research question we can state that we were able to rapidly realize large parts of our use case with all three platforms, despite no previous specific experience with them. Nevertheless, our general knowledge about software engineering helped us in finding the needed options in the platforms more quickly. Being familiar with the classic three-layer application design, for example, helped us to already have a general structure for the application in mind. From our point of view, someone without general software engineering knowledge had to follow more closely the guidelines, tutorials, and documentation provided by the platform to become productive but could, after some training, also profit from the features of a platform to build applications rapidly. However, as we could see from the more detailed aspects of our use case (e.g., data validation or checking custom conditions), a higher use case complexity usually requires more technical knowledge which is also in line with the overall assessment of Luo et al. [6]. Such technical knowledge might stem from a general background in computer science or from practical experience with a platform over a longer time. Detailed knowledge for one platform, however, might be difficult to apply elsewhere. That's why, from our point of view, a general background in computer science still provides a better foundation for application development. In addition to these general findings, the three platforms we investigated can be further differentiated. Overall, we found OutSystems to be a solution more focused on *citizen* developers. Its clean design via drag and drop elements, automatic integration with the storage layer and in particular its one-click-deployment contribute to the best Low-Code experience we faced in this experiment. The other two solutions could be described as *Less*-Code platforms. PowerApps is integrated with a lot of services of the Microsoft Azure's cloud environment. For enterprise users, this integration might be beneficial since already used services can be integrated into PowerApps, but programming is more technically compared to the application stack in OutSystems. Contrary to the other platforms, Appian offers a custom scripting language together with a custom component model. For feature-rich, rapid bootstrapped applications this platform design might support customers to build web applications. Without deeper technical knowledge in Appian and an understanding of the DSL, Appian is an expert system as the market for professional Appian developers shows.

Regarding features and techniques that supported rapidness in application development, we faced the best experience whenever the low code platforms provided intuitive interactions, suggestions, and direct feedback. One example are software wizards for generating UIs based on the defined data scheme which guide through the creation process in a focused interface. Additionally, these interfaces rely on visual programming by offering drag and drop features to alter the UI for displaying some fields respectively leaving out others, all while arranging these fields in a visually appealing layout as a default. In these cases, no extensive software engineering skills are needed to understand the layering of a system and progress can be made quickly, because the platform provides suitable defaults and templates. In other cases where custom DSL expressions need to be written the implementation flexibility increases, but comes with the difficulty to be familiar with the specific syntax. To take low coding literally, we refined the classification scheme for our selected platforms. We only rate OutSystems as a *low* code platform whereas Appian and PowerApps are *less* code ones.

Considering the feasibility of implementing our use case with the three platforms, we can state that from a functionality point of view, all platforms fulfill the functional requirements. In comparison to other implementation options, additional aspects like vendor lock-in or overall operation costs would need to be considered, but are out of the scope of this work. Finally, our experiment also has some threats to validity: (1) All study participants aka paper authors have a background in computer science. Therefore, the comparison to citizen developers is flawed and could be investigated in an additional experiment in future work. (2) The time limitation of six hours for each platform is one strength for a fair comparison but can introduce false conclusions due to the shallow investigation of the documentation and an absence of deeper knowledge about the platforms and their designs. (3) Some features like encryption of the data, security and role management with providers like LDAP were not tested due to time limitations.

6 Conclusion

In this paper, we made an experimental investigation of Low-Code platforms. We find that the investigated Low-Code platforms do enable rapid application development through easing the implementation effort. Nevertheless, coding is still required at some points, which is shown by the two Less-Code platforms in our investigation. Low-Code platforms are within this area of tension between ease of usage, enabling citizen developers to create apps, and the integration with other systems and a cloud provider's ecosystem. Due to this trade-off, some platforms are focused on a single application for development whereas others provide a feature-rich ecosystem.

References

1. France, R., Rumpe, B.: Model-driven development of complex software: A research roadmap. In: Proc. of FOSE (2007), https://doi.org/10.1109/FOSE.2007.14
2. Henriques, H., Lourenço, H., Amaral, V., Goulão, M.: Improving the developer experience with a low-code process modelling language. In: Proc. of MODELS (2018), https://doi.org/10.1145/3239372.3239387
3. Khorram, F., Mottu, J.M., Sunyé, G.: Challenges & opportunities in low-code testing. In: Proc. of MODELS Companion (2020), https://doi.org/10.1145/3417990.3420204
4. Kuhail, M.A., Farooq, S., Hammad, R., Bahja, M.: Characterizing visual programming approaches for end-user developers: A systematic review. IEEE Access 9, 14181–14202 (2021), https://doi.org/10.1109/ACCESS.2021.3051043
5. Lebens, M., Finnegan, R.: Using a low code development environment to teach the agile methodology. In: Agile Processes in Software Engineering and Extreme Programming, pp. 191–199. Springer International Publishing (2021), https://doi.org/10.1007/978-3-030-78098-2_12
6. Luo, Y., Liang, P., Wang, C., Shahin, M., Zhan, J.: Characteristics and challenges of low-code development. In: Proc. of ESEM (2021), https://doi.org/10.1145/3475716.3475782
7. Martins, R., Caldeira, F., Sa, F., Abbasi, M., Martins, P.: An overview on how to develop a low-code application using OutSystems. In: Proc. of ICSTCEE. IEEE (2020), https://doi.org/10.1109/ICSTCEE49637.2020.9277404
8. Sahay, A., Indamutsa, A., Ruscio, D.D., Pierantonio, A.: Supporting the understanding and comparison of low-code development platforms. In: Proc. of SEAA (2020), https://doi.org/10.1109/SEAA51224.2020.00036
9. Wong, J., Iijima, K., Leow, A., Jain, A., Vincent, P.: Magic quadrant for enterprise low-code application platforms. online (2021), https://www.gartner.com/doc/reprints?id=1-27IIPKYV&ct=210923&st=sb, published 20 September 2021

All links were last followed on January 18, 2022.

Index