Towards the Identification of Process Anti-Patterns in Enterprise Architecture Models

Barry-Detlef Lehmann[a], Peter Alexander[a], Horst Lichter[a] and Simon Hacks[b]

[a]RWTH Aachen University, Research Group Software Construction, Aachen, Germany
[b]KTH Royal Institute of Technology, Network and Systems Engineering, Stockholm, Sweden

Abstract

IT processes constitute the backbone of an integrated enterprise architecture (EA). The model thereof sustains the development and management of the EA. Nevertheless, the quality of such models tends to degrade over time due to, e.g. improper modeling practices or ineffective evaluation. In this regard, the knowledge of relevant modeling anti-patterns can help identify, mitigate, and prevent the occurrence of sub-optimal or adverse constructs in the model. In the field of business process modeling (BPM), a plethora of BPM anti-patterns has been defined and compiled in various taxonomies. However, these BPM anti-patterns mostly focus on technical issues, which thus are applicable for evaluating workflows but not EA-level processes. We strongly argue that the concept of process anti-pattern in EA domain can facilitate EA analyses on process-related issues. To address this gap, this paper presents a catalogue of 18 EA process modeling anti-patterns, which we derived from the existing BPM anti-patterns. Our result should serve as food for thought and motivation for future research in this context.

Keywords

enterprise architecture, process anti-pattern, model quality

1. Introduction

IT processes transform fragmented capabilities within an enterprise architecture (EA) into consolidated business assets. The models thereof often need to be consulted or even adapted in the efforts of managing and evolving the EA. Nevertheless, these process models are often developed with less consideration of quality due to, e.g. time pressure, little awareness of good modeling practices, or inadequate evaluation of the models. The uncontrolled development in this manner will eventually render the process models useless or even misleading [1]. This situation may hamper the sustainability of EA practices within the organization.

To avoid this, the development and evaluation of process models must be guided by the knowledge of relevant patterns and anti-patterns. In general, the *modeling pattern* is defined as a proven solution to a recurring modeling problem whereas the *modeling anti-pattern* is defined as a modeling solution that is known to pose risks [2]. The understanding of these concepts can help identify, mitigate, and prevent the occurrence of sub-optimal or adverse constructs within the models [1]. In this study, we focus on the concept of process anti-pattern in the context of EA modeling, specifically the application thereof in the evaluation of EA models.

In the field of business process modeling (BPM) research, a plethora of BPM anti-pattern taxonomies have been proposed [1, 3, 4]. However, these BPM anti-patterns mostly address rather technical aspects like the use of syntax or layout in the process model, which are very specific to the modeling notations in use. Moreover, discussions of BPM anti-patterns have mostly been presented in workflow modeling notations [5] (e.g. BPMN). These situations hinder the application of BPM anti-pattern to EA practices, in which processes are viewed from rather strategic perspectives and modeled in EA modeling notations. We strongly argue that transferring the existing BPM anti-patterns into process anti-patterns in the domain of EA can help improve the development of processes and their quality that underlie the EA.

The effort of transferring an existing concept into the domain of EA is not new. Salentin and Hacks introduced the concept of *EA smell*, which is defined as a hint to a bad habit that impairs the quality of the EA [6]. In their work, they transferred the concept of code smell into the context of EA through conceptual derivation and transformation methodology. Inspired by their work, the same methodology is applied in this study to answer the following research question (RQ):

RQ What process anti-patterns can be defined to support EA modeling activities through the analysis of published process anti-patterns?

The remainder of this paper is structured as follows: Section 2 gives an overview of previous studies on anti-pattern or other related concepts (e.g. smell) in the fields

QuASoQ 2020: 8th International Workshop on Quantitative Approaches to Software Quality, December 01, 2020, Singapore
barry.lehmann@rwth-aachen.de (B. Lehmann); alexander@swc.rwth-aachen.de (P. Alexander); lichter@swc.rwth-aachen.de (H. Lichter); shacks@kth.se (S. Hacks)
0000-0001-6534-278X (P. Alexander); 0000-0002-3440-1238 (H. Lichter); 0000-0003-0478-9347 (S. Hacks)

of BPM and EA modeling research; section 3 describes our methodology for obtaining process anti-patterns for EA modeling problems; section 4 elaborates our findings and the analysis thereof; section 6 demonstrates our results and discusses the implications as well as threats to the validity thereof; and section 7 motivates future research directions and concludes this paper.

2. Related Work

The concept of anti-pattern was coined in 1995 by Koenig [7] to describe a common solution to a recurring problem which poses risks of being counterproductive. Although an anti-pattern may serve as a practical short-term solution, the use of it sets a context in which certain changes may become more expensive or impossible. The (unintentional) use of anti-pattern is highly influenced by, e.g. time pressure, inadequate knowledge of best practices, or unforeseen changes.

The spectrum of studies about anti-patterns covers a wide-range of software engineering topics, such as software development and modeling. In the domain of BPM, a number of taxonomies of process anti-patterns have been proposed, each of the taxonomies addresses a specific area of concern. In 2019, a bibliography of all these taxonomies was published based on a literature review study [1]. Therein, the collected taxonomies are divided into seven categories based on the addressed modeling problems. Furthermore, this study suggests several rules of thumb in documenting process anti-patterns. The authors of this study advocate the use of this bibliography in the efforts to increase the quality of BP models.

Analogous with the concept of process anti-pattern, the concept of *process anomaly* is also known in BPM research. Vidacic and Strahonja present a literature review of this concept in which the collected process anomalies are divided into three categories: structural, semantical, and syntactical anomalies [8]. They also suggest approaches to the mitigation or prevention thereof. Suchenia et al. provide a brief overview of BPM anti-patterns, present these in BPMN models, and categorize these into three categories: syntactic, structural, and control flow anti-patterns [9]. Trcka et al. present data-flow anti-patterns and an approach to identifying these [10]. Further in this topic, Sadiq et al. identify seven common data-flow anti-patterns and provide the basic algorithm to address these [11]. Finally, Döhring and Heublein present a taxonomy of control-flow, rule-based, and data-flow anti-patterns; demonstrate examples of such anti-patterns in BPMN; and suggest detection as well as prevention mechanisms thereof [12].

In the domain of EA modeling, the concept of anti-pattern remained unknown until the recent suggestion of an EA smell taxonomy by Salentin and Hacks. Therein, the authors provide a catalog of 45 EA smells that originated from code smells. In their approach, they transform a catalog of well-known code smells into EA smells and categorize the EA smells based on the three concerns of EA: business, application, and technology. Furthermore, they present a tool that can detect 14 EA smells. As an extension to their work, this study explores the current knowledge about process anti-pattern to obtain a new understanding thereof in the EA domain.

Finally, the idea of looking at different abstraction levels in processes to address different stakeholders is not new. Several studies have been conducted to decompose processes into different abstraction levels. Giachetti proposes to divide the process hierarchy into functions, processes, sub-processes, activities, and tasks [5]. This study argues that the natural hierarchical attribute of the process should be used. Viljoen, on the other hand, decomposes it into enterprise model, macro, business process, sub-process, activity, and task [13]. Koschmider and Blanchard propose a semiautomatic detection for different process abstraction levels with processes modeled with Petri Nets [14]. Their goal is to detect a process hierarchy in a process model. All these studies commonly advocate that the levels of abstraction applied to processes should meet the goals of relevant stakeholders.

3. Methodology

To transfer the existing knowledge about process anti-pattern into the EA domain, this study follows the methodologies proposed by Peffers et al. and Hevner et al. for the Design Science Research (DSR) [15], [16]: a type of research which aims to devise an artifact that addresses a "heretofore unsolved and important business problem" by drawing on the existing knowledge. The resulting artifact must be rigorously evaluated in terms of its "utility, quality, and efficacy" and effectively communicated to the relevant audience.

This study is performed as follows: At first, we collected knowledge about the 336 already-published BPM anti-patterns in scientific literature, which have been compiled in [1] and are publicly accessible on [17]. Based on a mapping of modeling notations between BPMN and ArchiMate, we processed these BPM anti-patterns and finally derived 18 process anti-patterns of EA relevance, which are then documented in a structured template and exemplified in ArchiMate models. A closer look into this procedure is provided in the following subsections.

3.1. Notation Mapping

Most BPM anti-patterns have been analyzed and visualized in BPMN. While BPMN provides a full-fledged framework to create graphical business processes mod-

BPMN	ArchiMate	Source
Business Process Diagram, Pool, Lanes	Process	[21, 19]
Activities, Task, Sub-Process	Function	[21]
Collaboration Diagram	Interaction	[21]
Event	Event	[21, 18, 19]
Data Object	Object	[21, 19]
Lane	Business Role, Business Actor, Application Component	[21, 19]
Sequence flow	Triggering, Flow	[20]
Data association	Access	[20]
Inclusive and parallel gateways	And-Junction	[20, 18]
Exclusive and event-based gateways	Or-Junction	[20, 18]

Table 1
Mapping from BPMN elements to ArchiMate

els, BPMN is intended for rather detailed business process modeling, such as the modeling of conversation, choreography, and collaboration models. However, such business process models constitute only a small area within the broad view of EA. Therein, processes are analyzed from rather strategic perspectives, e.g. the connection of all high-level processes to the surrounding organization units to achieve strategic targets. The modeling of such perspectives has been facilitated by a number of EA modeling languages; the most popular one is ArchiMate. We strongly argue that method supports for analyzing EA processes must be built on top of an EA modeling language. Therefore, to transfer the existing BPM anti-patterns into the EA domain, we first need to create a mapping between BPMN and ArchiMate to figure out the possibilities of deriving something of EA relevance.

Previous studies have suggested several mappings between BPMN and ArchiMate [18, 19, 20, 21]. They show that both modeling languages share some conceptual similarities. Firstly, both provide similar notations for connecting process elements (e.g. the sequence, default, and conditional flows) and regulating these with gateways or junctions. Secondly, both support the modeling of similar relationships between processes and the related elements [19]. For example, BPMN's support for creating relationships between e.g. activities and data objects is similar to ArchiMate's support for creating relationships between e.g. business processes and business objects. In table 1, we list the mappings of notations which we use as a basis to conduct the next steps.

3.2. Transformation Design

From our analysis of related studies in BPM research, we found several taxonomies which collect in total of 336 BPM anti-patterns and classify these based on the characteristics thereof [1] [17]. From the perspective of our study, we argue that the depth of these taxonomies stretches beyond the context of EA analysis. For example, the various situations of deadlock are addressed by different BPM anti-patterns. While such specialized modeling problems are indeed identifiable within BPMN models, such detailed problems are not visible through the high-level notations of ArchiMate. Therefore, we pruned such detailed branches of classification within the selected taxonomies, thereby reducing the size of our analysis to 200 BPM anti-patterns.

Next, this study follows a defined procedure to derive process anti-patterns of EA relevance, which is as follows: Firstly, BPMN elements found in the names and descriptions of the BPM anti-patterns are translated into ArchiMate elements based on the notation mapping shown in table 1. Secondly, each translated description is evaluated to determine whether it is comprehensible within the context of EA. Through this step, we derived 18 process anti-patterns of EA relevance, while excluding the rest because the translated descriptions thereof do not describe relevant EA modeling problems (e.g. anti-patterns related to syntax errors such as the *sequence flow crosses process boundary* [3]). Among the derived process anti-patterns, some are directly applicable in the EA domain only after applying the notation mapping (e.g. the *Layout Deficit* [1]), while some other require broad modification in the description thereof to depict valid EA modeling problems (e.g. *Useless Test*).

4. An initial catalogue of process anti-patterns in EA

To support the usability of our contribution, we have organized the resulting 18 process anti-patterns in a catalogue, which is publicly accessible on our [22]. Since this catalogue is still in its initial stages, we encourage further extensions and improvements to it. A detailed discussion on future research directions is given in section 7.

In this section, we first introduce the categorization used in the catalogue. Next, we provide a deeper look into one process anti-pattern under each category. Lastly, we describe the template used to document some general attributes of the process anti-patterns.

4.1. Categorizing process anti-patterns

To present the proposed catalogue in an organized and systematic manner, we divide the contained process anti-patterns into the following five categories that we derived from some selected categories in [1].

The category of **semantic error** includes process anti-patterns that address the semantically error or inconsistent parts of the model under analysis. These problems are commonly caused by the improper or deficient use of modeling notations, which may distort the understanding of the model. It is worth to mention that such semantic errors should not be confused with syntax errors as the latter addresses the violation against the rules of assembling the notations, whereas the earlier addresses the false or ambiguous impression conveyed by the model [3]. Due to the gap between the syntactic rules of BPMN and ArchiMate, we derived no process anti-patterns of EA relevance under the syntax error category.

The category of **control-flow problem** includes process anti-patterns that address the flawed concurrency of process flows which raises unpredictability in the final outcome. Such issue occurs when the modeled processes are split or joined without properly considering the influence thereof to the final outcome.

The category of **understandability problem** includes process anti-patterns that address the excessive or lack of complexity within the model, thereby requiring excessive efforts to understand the process under analysis in its full context. Such an issue is commonly caused by the improper/inconsistent level of information granularity or inadequate/imprecise coverage of the actual process in reality.

The category of **rule-related defect** includes process anti-patterns that address the contradictions among the rules specified within the model. Such an issue may occur when e.g. the actual process is not well-defined, well-communicated, or well-understood among some contributors to the rules specification in the model.

The category of **data-flow related defect** includes process anti-patterns that address the proneness to conflicts when the same data object is concurrently used for different kinds of transactions. These issues are commonly caused by the overlapping data responsibilities among different processes or the centralization of too many data in a single data object.

4.2. Demonstrating process anti-patterns

Next, we name the 18 process anti-patterns and map these under the aforementioned categories, as listed in table 2.

Table 2
Categories of EA anti-pattern

Category	Anti-pattern
Semantic Error	End event missing Start event missing
Control-flow Problem	Dead Element Deadlock Infinite loop
	Lack of synchronization Undefined junction condition
Understand-ability Problem	Junction named as element Layout deficit Language deficit Missing negative case The word And in element name The word Or in element name Useless test
Rule-related Defect	Contradiction in input
Data-flow related Defect	Inconsistent data Mismatched data Missing data

Furthermore, to give a closer look into the catalogue, we elaborate one process anti-pattern under each category and provide an example thereof.

Under the category of *semantic error*, the **end event missing** occurs when the modeled process does not clearly specify the end events [3] [23], which then causes confusion or misinterpretation about the valid conditions to finalize or abort the process execution. An example of this problem is when multiple high-level processes of different organization bodies are integrated within the EA model without specifying the points when the collaborative outcomes have been achieved or any disruptions have to be handled. A solution to this anti-pattern would be to simply introduce *end event elements* that clearly signal all possible ways to end the process.

Under the category of *control-flow problem*, the **lack of synchronization** refers to the situation when the modeled process does not specify a proper synchronization among concurrent process flows, thereby showing no predictable outcome. An example of such a situation is when the modeled process is split by an *AND junction* and later joined by an *OR junction* [24, 25]. A solution to this anti-pattern is to ensure the highest test coverage of all points of synchronization specified in the model.

Under the category of *understandability problem*, the **useless test** is identified when the modeled process fulfills only some of all the possible cases in reality [26], thereby making it impossible to identify and test the real extent of the supported problem domain. An example thereof is when the handling of possible mistakes or disruption during the process execution is not specified within the model. As a solution, the model should be

Attribute	Meaning
Name	Gives the anti-pattern a meaningful designator
Problem	Describes why the anti-pattern leads to problems
Consequences	Describes what the consequences of the anti-pattern are
Solution	Describes a solution to the anti-pattern
Graphical Definition	Shows a graphical representation of the anti-pattern to reduce misinterpretations

Table 3
Documentation attributes of EA process anti-pattern (adopted from [1]).

incrementally and iteratively developed along with the continuous identification of relevant test cases until it reaches a reasonable level of complexity and covers the complete problem domain [27].

Under the category of *rule-related defect*, the **contradiction in input** occurs when some rules applied to the modeled process may contradict with each other, thereby hindering the process execution to continue as intended [12]. An example thereof is when a certain data object passes the input validations specified on a junction despite being actually invalid for the supplied process. A solution to this anti-pattern is to continuously perform a rigorous combinatorial testing of all possible input types and all the rules applied to the modeled process.

Under the category of *data-flow-related defect*, the **inconsistent data** describes the situation in which data objects (e.g. customer records or insurance claim) are accessed by concurrent process flows, thereby making it prone to data handling mistakes. An example thereof is when multiple processes work on duplicates of the same data object, and a (manual) synchronization procedure between the duplicates is required after every modification on one side. To mitigate this, the strategy of handling the data must be carefully defined and implemented.

4.3. Documenting process anti-patterns

In general, the documentation of *modeling anti-patterns* includes many attributes of *modeling patterns* [28] together with some other attributes like *cause* and *detection* [29]. To document the process anti-patterns identified in this study, we derive some attributes from the templates for documenting BPM anti-patterns introduced in [1], as shown in table 3. Please note that, at the time of this writing, not all attributes have been completed for each process anti-pattern due to the need of further information and analysis.

5. Applying process anti-patterns in EA

In order to illustrate the concept and the usage of process anti-pattern in the context of EA, we analyze a slightly modified ArchiMate EA model and annotate it with anti-pattern information. The model used, depicted in fig. 1, is contained in a publicly accessible collection of ArchiMate example EA models [30]. This model defines how new orders are processed. After a new order is received, planning the order and evaluating the customers' credibility are done in parallel. After an approved proposal is available, the customer signs the respective contract to accept the proposal.

When analysing the model, it can be observed that the processes Evaluate customer credit and Plan order do not wait for each completion before continuing to the Develop approved proposal process, which may lead to undesired results (e.g. the contract does not consider the customer's credibility). These are consequences of the *lack of synchronization* anti-pattern.

When analyzing the usage of the Order Data element, we identify the *inconsistent data* anti-pattern because this data can be changed without rerunning dependent processes (e.g. Evaluate customer credit).

Furthermore, the Evaluate customer credit process exhibits the *useless test* anti-pattern, as only the positive test result is modeled.

Next, we can identify the anti-pattern *end event missing* because no clear termination is defined for this EA process model. The process could end in Refuse proposal or in Accept proposal.

Finally, we detect the *contradiction in input* anti-pattern at the Or-Junction that splits the control flow after Develop approved proposal. There, the incorrect condition will never lead to the Refuse proposal process and therefore makes it a dead process.

6. Discussion

Some of the main goals of applying the EA discipline within an enterprise is to ensure the business-IT align-

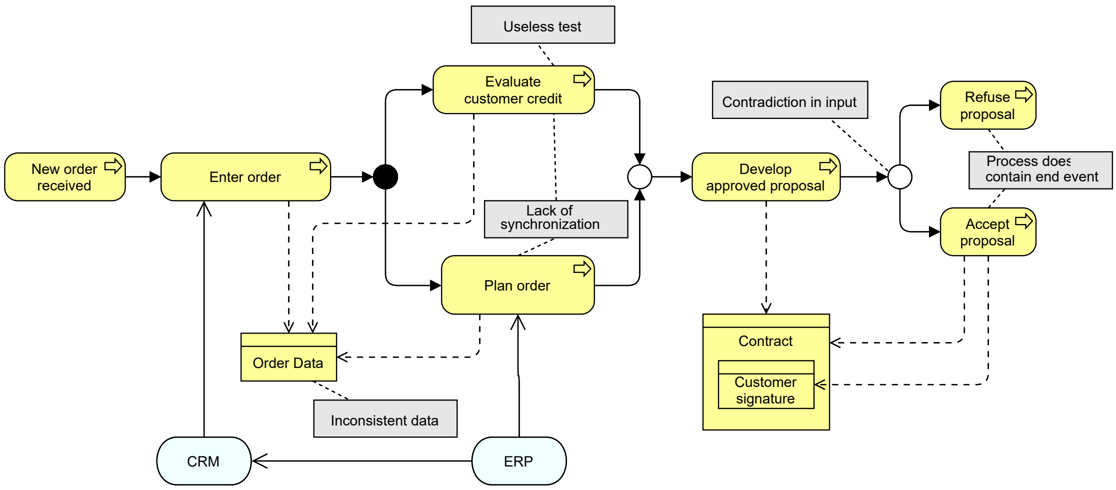

Figure 1: An example EA ArchiMate model with annotated process anti-patterns

ment [31] and to develop solid IT strategies that can help achieve strategic targets [32]. For this reason, this section seeks to answer the ultimate question of this study: "how can enterprise architects benefit from the contribution of this study?" In this section, we describe the use of the proposed process anti-patterns to support both research and practice of EA methods. Following to this, we discuss the threats to the validity of our results.

6.1. Implication for researchers & practitioners

The concept of *EA debt* has recently been introduced as the deviation between the current state and the hypothetical ideal state of the enterprise [33]. Factors to such deviation in EA (e.g. sub-optimal or adverse solution design) are likely to be identifiable within the EA models created or used during the planning, development, evaluation, or communication of the EA [6]. Therefore, the ability to recognize the existence of such deviation in EA models is needed, and the concept of process anti-pattern proposed in this study is intended to support such ability with sharp focus on processes. In this case, practitioners can use the process anti-patterns catalogue to scan the EA models for potential *EA Debt*. In any case, it is beneficial that the practitioners are aware of the potentially vulnerable parts of the EA Model as further development could be hindered if these remain ignored [6].

Furthermore, we also intend to impart food for thought into the EA research community and provide a basis for further research works in this topic. For example, the process anti-patterns identified in this study might help researchers to extend the automatic detection of EA anti-patterns in an EA model, as has been drafted in a program that currently detects 14 EA smells [6].

6.2. Threats to validity

The results of this study have to be seen in the light of some limitations. The limitations that affect the results of this paper are the lack of previous research and bias during the anti-pattern transformation.

Lack of previous research. There is little research on transforming low-level process methods to be applicable for high-level processes and even much less on bringing together the modeling notations for such processes. In addition, the already suggested mappings between the modeling notations for low-level and high-level processes are rather described as informal and lack of theoretical foundations, thereby leaving room for interpretation or different mapping solutions. This might reduce the validity of our results because our approach relies on the existing mapping between BPMN and ArchiMate.

Bias during the anti-pattern transformation. To reduce bias when selecting the relevant BPM anti-patterns to be transformed, we first need to establish objective selection criteria. Thus, we defined a mapping of notations and apply it on the collected BPM anti-patterns to prune the ones that do not fit in the new domain. Despite this, subjective assessment is still inevitably involved during the process, thereby leaving room for interpretation and may not produce unique results.

7. Conclusion & Future Works

The concept of anti-pattern has been long known to help recognize common solutions that are not sustainable for the future development. However, little emphasis has been put on studying this concept in the context of EA. The first step in this direction has recently been made to transfer the existing code smells into the EA domain, out of which an initial catalogue of 45 EA smells has been developed and proposed [6]. To pursue a meaningful extension to this result, this study focuses on transferring the existing BPM anti-patterns [1] into process anti-patterns for EA modeling problems, with reference to a mapping between the notations of BPMN and ArchiMate.

The process anti-patterns identified in this study are compiled in a catalogue that is publicly accessible on our [22]. Therein, the process anti-patterns are categorized and documented in a well-known template to ease the use or extension thereof by EA practitioners and researchers. The practical use of this catalogue covers a broad range of topics, starting from the identification of flaws in EA models to the identification of EA debt. Nevertheless, this catalogue is still in its initial stages. Much more information and analyses are needed before this catalogue can be evaluated in real industrial contexts. Therefore, to motivate further research in this topic, the rest of this section outlines some ideas of future research directions.

The future research directions in this topic can be divided into three main topics: to pursue different methods for defining new EA anti-patterns, to perform empirical studies for improving both conceptual and practical knowledge in this context, and to develop tool supports for the automatic detection of the anti-patterns in EA models. In terms of analyzing more methods to find EA anti-patterns, we suggest to investigate new domains (e.g. documentation, data, or requirement anti-patterns) to extend the catalogue with adapted or new EA anti-patterns. Also, the investigation of cause-and-effect relationships among the identified EA anti-patterns may provide insights on the possible propagation of impacts thereof. In terms of empirical studies, evaluations using real EA models from different business domains can help to verify and improve the quality of the catalogue and the proposed method supports. Finally, tool supports can be developed to support the continuous detection of EA anti-patterns or the (early) signs of their occurrences.

References

[1] A. Koschmider, R. Laue, M. Fellmann, Business process model anti-patterns: a bibliography and taxonomy of published work, in: J. vom Brocke, S. Gregor, O. Müller (Eds.), 27th European Conference on Information Systems - Information Systems for a Sharing Society, ECIS 2019, Stockholm and Uppsala, Sweden, June 8-14, 2019, 2019. URL: https://aisel.aisnet.org/ecis2019_rp/157.

[2] M. Fellmann, A. Koschmider, R. Laue, A. Schoknecht, A. Vetter, Business process model patterns: State-of-the-art, research classification and taxonomy, Business Process Management Journal (2018). doi:10.1108/BPMJ-01-2018-0021.

[3] T. Rozman, G. Polančič, R. V. Horvat, Analysis of most common process modelling mistakes in bpmn process models, in: EuroSPI 2007, Industrial Proceedings, 2007, pp. 1.21 – 1.31. URL: https://2020.eurospi.net/images/proceedings/EuroSPI2007-ISBN-978-3-9809145-6-7.pdf.

[4] S. Von Stackelberg, S. Putze, J. Mülle, K. Böhm, Detecting data-flow errors in bpmn 2.0, Open Journal of Information Systems (OJIS) 1 (2014) 1–19.

[5] R. E. Giachetti, Design of Enterprise Systems: Theory, Architecture, and Methods, 1st ed., CRC Press, Inc., USA, 2010.

[6] J. Salentin, S. Hacks, Towards a catalog of enterprise architecture smells, in: N. Gronau, M. Heine, H. Krasnova, K. Poustcchi (Eds.), Entwicklungen, Chancen und Herausforderungen der Digitalisierung: Proceedings der 15. Internationalen Tagung Wirtschaftsinformatik, WI 2020, Potsdam, Germany, March 9-11, 2020. Community Tracks, GITO Verlag, 2020, pp. 276–290. doi:10.30844/wi_2020_y1-salentin.

[7] A. Koenig, Patterns and antipatterns, J. Object Oriented Program. 8 (1995) 46–48.

[8] T. Vidacic, V. Strahonja, Taxonomy of Anomalies in Business Process Models, 2014, pp. 283–294. doi:10.1007/978-3-319-07215-9_23.

[9] A. Suchenia, T. Potempa, A. Ligęza, K. Jobczyk, K. Kluza, Selected Approaches Towards Taxonomy of Business Process Anomalies, volume 658, 2017, pp. 65–85. doi:10.1007/978-3-319-47208-9_5.

[10] N. Trcka, W. M. P. van der Aalst, N. Sidorova, Data-flow anti-patterns: Discovering data-flow errors in workflows, in: P. van Eck, J. Gordijn, R. J. Wieringa (Eds.), Advanced Information Systems Engineering, 21st International Conference, CAiSE 2009, Amsterdam, The Netherlands, June 8-12, 2009. Proceedings, volume 5565 of *Lecture Notes in Computer Science*, Springer, 2009, pp. 425–439. doi:10.1007/978-3-642-02144-2_34.

[11] S. W. Sadiq, M. E. Orlowska, W. Sadiq, C. Foulger, Data flow and validation in workflow modelling, in: K. Schewe, H. E. Williams (Eds.), Database Technologies 2004, Proceedings of the Fifteenth Australasian Database Conference, ADC 2004, Dunedin, New Zealand, 18-22 January 2004, volume 27 of *CRPIT*, Australian Computer Society, 2004, pp. 207–

214. URL: http://crpit.scem.westernsydney.edu.au/abstracts/CRPITV27Sadiq.html.

[12] M. Döhring, S. Heublein, Anomalies in rule-adapted workflows - a taxonomy and solutions for vbpmn, in: 2012 16th European Conference on Software Maintenance and Reengineering, 2012, pp. 117–126.

[13] S. Viljoen, Reflections on business process levelling, white paper, 2012. URL: https://realirm.com/sites/default/files/whitepapers/reflections_on_business_process_leveling_0.pdf.

[14] A. Koschmider, E. Blanchard, User assistance for business process model decomposition, in: Proceedings of the 1st IEEE International Conference on Research Challenges in Information Science, 2007, pp. 445–454.

[15] K. Peffers, T. Tuunanen, M. Rothenberger, S. Chatterjee, A design science research methodology for information systems research 24 (2007) 45–77. doi:10.2753/MIS0742-1222240302.

[16] A. R. Hevner, S. T. March, J. Park, S. Ram, Design science in information systems research, MIS Q. 28 (2004) 75–105. URL: http://misq.org/design-science-in-information-systems-research.html.

[17] Business process model patterns classification, 15.11.2020. URL: http://www.bpmpatterns.org/.

[18] Archimate® 3.1 specification, 27.06.2020. URL: https://pubs.opengroup.org/architecture/archimate3-doc/.

[19] D. Orlovskyi, A. Kopp, Enterprise architecture modeling support based on data extraction from business process models, in: S. Subbotin (Ed.), Proceedings of The Third International Workshop on Computer Modeling and Intelligent Systems (CMIS-2020), Zaporizhzhia, Ukraine, April 27-May 1, 2020, volume 2608 of *CEUR Workshop Proceedings*, CEUR-WS.org, 2020, pp. 499–513. URL: http://ceur-ws.org/Vol-2608/paper38.pdf.

[20] M. Lankhorst, Combining archimate® 3.0 with other standards – bpmn, 6.9.2016. URL: https://bizzdesign.com/blog/combining-archimate-3-0-with-other-standards-bpmn/.

[21] L. Penicina, Linking bpmn, archimate, and BWW: perfect match for complete and lawful business process models?, in: J. Grabis, M. Kirikova, J. Zdravkovic, J. Stirna (Eds.), Short Paper Proceedings of the 6th IFIP WG 8.1 Working Conference on the Practice of Enterprise Modeling (PoEM 2013), Riga, Latvia, November 6-7, 2013, volume 1023 of *CEUR Workshop Proceedings*, CEUR-WS.org, 2013, pp. 156–165. URL: http://ceur-ws.org/Vol-1023/paper15.pdf.

[22] B.-D. Lehmann, Enterprise process anti-pattern, 05.10.2020. URL: https://ea-anti-pattern.pages.rwth-aachen.de/process-anti-patterns/.

[23] C. Dias, V. Stein Dani, J. Mendling, L. Thom, Anti-patterns for Process Modeling Problems: An Analysis of BPMN 2.0-Based Tools Behavior, 2019, pp. 745–757. doi:10.1007/978-3-030-37453-2_59.

[24] Z. Han, P. Gong, L. Zhang, J. Ling, W. Huang, Definition and detection of control-flow anti-patterns in process models, in: 2013 IEEE 37th International Computer Software and Applications Conference Workshops (COMPSACW), IEEE Computer Society, Los Alamitos, CA, USA, 2013, pp. 433–438. doi:10.1109/COMPSACW.2013.111.

[25] R. Laue, A. Awad, Visualization of business process modeling anti patterns, ECEASST 25 (2010). doi:10.14279/tuj.eceasst.25.344.

[26] R. Laue, W. Koop, V. Gruhn, Indicators for open issues in business process models, in: International Working Conference on Requirements Engineering: Foundation for Software Quality, Springer, 2016, pp. 102–116.

[27] P. Desfray, G. Raymond, Chapter 5 - key modeling techniques, in: P. Desfray, G. Raymond (Eds.), Modeling Enterprise Architecture with TOGAF, The MK/OMG Press, Morgan Kaufmann, Boston, 2014, pp. 67 – 91. doi:https://doi.org/10.1016/B978-0-12-419984-2.00005-7.

[28] E. Gamma, R. Helm, R. Johnson, J. Vlissides, Design Patterns, volume 47 of *Addison Wesley Professional Computing Series*, 1995.

[29] J. Bogner, T. Boceck, M. Popp, D. Tschechlov, S. Wagner, A. Zimmermann, Towards a collaborative repository for the documentation of service-based antipatterns and bad smells, in: 2019 IEEE International Conference on Software Architecture Companion (ICSA-C), 2019, pp. 95–101. doi:10.1109/ICSA-C.2019.00025.

[30] P. Beauvoir, A collection of archimate® models., 27.06.2020. URL: https://github.com/archimatetool/ArchiModels.

[31] J. Gøtze, The changing role of the enterprise architect, in: 2013 17th IEEE International Enterprise Distributed Object Computing Conference Workshops, 2013, pp. 319–326. doi:10.1109/EDOCW.2013.42.

[32] C. Strano, Q. Rehmani, The role of the enterprise architect, Inf. Syst. E-Business Management 5 (2007) 379–396. doi:10.1007/s10257-007-0053-1.

[33] S. Hacks, H. Höfert, J. Salentin, Y. C. Yeong, H. Lichter, Towards the definition of enterprise architecture debts, in: 23rd IEEE International Enterprise Distributed Object Computing Workshop, EDOC Workshops 2019, Paris, France, October 28-31, 2019, 2019, pp. 9–16. doi:10.1109/EDOCW.2019.00016.