






# Towards a Knowledge Base of Terms on Enterprise Architecture Debt

Ada Slupczynski<sup>1</sup>  and Simon Hacks<sup>2</sup>  

<sup>1</sup> RWTH Aachen University, Aachen, Germany  
slupczynski@swc.rwth-aachen.de

<sup>2</sup> Stockholm University, Stockholm, Sweden  
simon.hacks@dsv.su.se

**Abstract.** The term Enterprise Architecture (EA) Debt has been coined to grasp the difference between the actual state of the EA and its hypothetical, optimal state. Since its first definition in 2019, different theses have been conducted on the topic, and different articles have been published working on and with the term EA Debt. Consequently, using different terms has evolved to describe different phenomena within the domain. Due to the different authors involved in this development, perceiving these terms might differ. To avoid misunderstandings and to ease common understanding of the domain, we propose an ontology for the domain of EA Debt. We rely on a lightweight methodology for rapid ontology engineering (UPON light) and the Unified Foundational Ontology (UFO) to engineer our ontology.

**Keywords:** Enterprise Architecture Debt · Ontology · Knowledge Base

## 1 Introduction

Digital transformation comes with opportunities and challenges, such as business-IT alignment [47] (BITA). A holistic view is required to achieve BITA that helps understand the impact of products, employees, and business models [39]. One solution that provides a holistic view is Enterprise Architecture (EA) [33], which provides methods and tools to align business with IT, operationalize the business strategy, and can drive innovations [34]. EA provides transparency utilizing business-related views, application landscapes, and information technology sketches [33, 53].

EA has often been established in many large organizations, and related research elaborates on various frameworks, methods, and tools [29, 32]. Organizations' EAs usually reflect this evolution through an organically grown architecture with many artifacts and systems implemented. Simultaneously, the general perception of EA is bureaucratic, document-centric, and hampering agility due to its focus on long-term effects [10, 52]. However, there might be significant discrepancies between long-term EA objectives and individual projects, causing

conflicts due to a misalignment between EA plans and business needs. These conflicts could be: (1) Complex application landscapes with legacy systems and redundancies; (2) Outdated or incomplete EA artifacts and documentation; or (3) Procedures and organizational units in EA management that hamper IT innovations.

These conflicts are caused by past decisions that might have been justified at the corresponding time. Still, due to organizational change, these changes might not be reflected in the application landscape. To cope with this challenge, Hacks et al. proposed the term EA Debt to describe those results from past decisions that hamper changes in the organization [22]. Like technical debt, EA Debt represents obstacles moving from the current EA (as-is) towards a desired to-be-landscape. In contrast to technical debt, EA Debt provides a more holistic view, not just encompassing technical systems but also processes, organizational units, and regulations.

Since the first definition of EA Debt in 2019 [22], different works have been conducted within the domain. Naturally, different terms have evolved to describe different phenomena within the domain, and due to the different authors involved, the perception of terms might differ. To avoid misunderstandings and to ease common understanding of the domain, this work has the Research Objective to **develop an ontology for the domain of EA Debt**.

The rest of this work is structured as follows: Next, we go into more detail about the concept of EA Debt and present the research already conducted in the field. Then, we explain the research method we followed to build our ontology, followed by a demonstration of the ontology on an example from previous EA Debt research. Before we conclude the work and discuss the ontology's implications, we present related work on ontologies in EA and technical debt.

## 2 Background

The digitalization of organizations is accompanied by agile methods, which is a challenge for EA, as the time to define proper target architectures becomes reduced [49]. This is caused by product owners preferring short-term business value over sustainable architectural solutions. At the same time, approaches that ease long-term architectural solutions are scarce [19, 50].

To address this challenge, Hacks et al. [22] extend the concept of Technical Debts, which describes past technical shortcuts that hamper IT developments [12, 37], to the EA domain by suggesting a more holistic view on the organization. Concretely, Hacks et al. [22] originally defined EA Debt as “a metric that depicts the deviation of the currently present state of an enterprise from a hypothetical ideal state”. Such a deviation can result from (1) decisions that are expedient in the short term but cause future changes to be more costly or (2) from a deviation in the actual EA that might have arisen due to changes in the valuation. The latter arises when an original decision aligns with the optimal EA, but a recent change in the strategy leads to another optimal EA, while the first hampers implementing better solutions immediately.

Research conducted in the field of EA Debt can be separated into two streams [2]: On the one hand, research related to the technical aspects of EA Debt. On the other hand, research elaborates on the socio-technical aspects of EA Debt.

Most of the research is related to technical aspects of EA Debts. Salentin and Hacks [41] defined the term EA Smell and published the first set of EA Smells together with a prototype that could identify some of the smells in ArchiMate models. Lehmann et al. [35] and Tieu and Hacks [48] continued in this line of research by finding other smells based on known anti-patterns and software architecture smells. Smajevic et al. [46] developed tool support to identify EA Smells in an automatized way in EA models.

Given a set of EA Debts, Yeong et al. [55] provide a method to prioritize which EA Debt to solve next. They adapt portfolio theory and utility functions to prioritize different EA Debts based on an organization's preferences. Having this prioritization, refactoring is necessary, presented by Liss et al. [38] to guide the removal of the respective debts. Slupczynski et al. [44] propose a process for evaluating the prudence and recklessness of enterprise architecture debts.

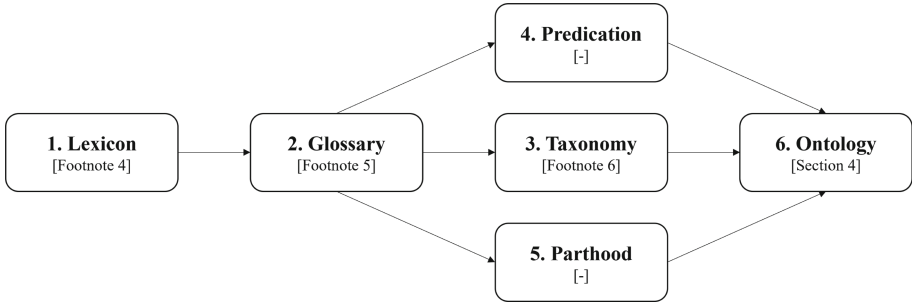
The technical proposals are framed by research on the socio-technical aspects. For instance, Alexander et al. propose a process to manage EA Debts [2]. It is suggested first to identify and collect EA Debts, assess and prioritize, and finally, remove or monitor actively. Here, the work of Jung et al. [28] provides a workshop format to identify EA Debts and EA Smells that cannot be detected solely relying on EA models. This was further developed by Daoudi et al. [13] to be more time efficient and also to consider when an EA Debt has a negative influence on an organization.

Finally, Hacks and Jung [23] conducted a first experiment to evaluate if the concept of EA Debt leads to a better EA. Therefore, they taught a group of students the concept. Afterward, the students were supposed to model a fictitious organization, and experts compared these models to models from student groups that did not know the concept. However, a positive effect could not be found in the experiment.

### 3 Method

To build our ontology to reason about the domain of EA Debt, we rely on the Unified Process for Ontology building (UPON) lite [14], which is a simplified version of UPON [15]. UPON lite is designed to enable domain experts with no deeper knowledge of ontology engineering to develop their ontology. As the authors of the article can be considered domain experts for EA Debt due to their involvement in a substantial amount of articles in the field and missing experience in the development of ontologies, UPON lite is deemed a well-suited solution to accomplish the overall goal of unifying the different terms in the domain.

UPON lite comprises six steps (cf. Fig. 1). In the following, we give a short recap on the single steps and how we realized them to build our ontology:



**Fig. 1.** Steps to Build an Ontology according to UPON lite [14]

- 1. Lexicon:** In the first step, a lexicon of all terms in the domain is created. This lexicon is a flat list of undifferentiated terms that also includes synonyms. To create such a lexicon, we considered all papers citing the original paper [22] suggesting the definition of EA Debt. We only excluded papers not in English or not available for download and added two papers in the publishing process [13, 23]. Next, we identified 1150 keywords in all papers via Yet Another Keyword Extractor (YAKE) [11]<sup>1</sup> and removed all duplicates and non-relevant keywords (e.g., author names). The resulting lexicon comprises 117 entries and can be found in github<sup>2</sup>.
- 2. Glossary:** The second step aims to unify the lexicon by identifying synonyms and providing a textual description of the single terms. Moreover, more semantics can be added to the terms, e.g., by relying on established foundations like the Unified Foundational Ontology (UFO) [21] or the Object, Process, Actor Modeling Language (OPAL) [16]. Accordingly, we identified synonyms and provided a textual description of 56 terms. Additionally, we first classified the terms according to UFO [21] as it is an established approach in our community and provides good tool support with its integration into Visual Paradigm. The glossary can be found in github<sup>3</sup>.
- 3. Taxonomy:** The third step focuses on defining a taxonomy of the terms within the glossary. I.e., one establishes a hierarchical “is a” order among the terms. Moreover, this step also evaluates the previous two steps in which unnecessary terms are removed, and missing terms can be added. Structuring the terms into a taxonomy, we identified 33 top-level concepts, 20 first-level specifications, and three second-level specifications. The resulting taxonomy can be found in github<sup>4</sup>.

<sup>1</sup> We used the following parameters:  $NGRAM = 3$ ,  $Keywords\_Number = 50$ ,  $Deduplication\_Threshold = 0.5$ .

<sup>2</sup> [https://github.com/simonhacks/ead\\_ontology/blob/main/Keywords.xlsx](https://github.com/simonhacks/ead_ontology/blob/main/Keywords.xlsx).

<sup>3</sup> [https://github.com/simonhacks/ead\\_ontology/blob/main/Definitions.xlsx](https://github.com/simonhacks/ead_ontology/blob/main/Definitions.xlsx).

<sup>4</sup> [https://github.com/simonhacks/ead\\_ontology/blob/main/Taxonomy.xlsx](https://github.com/simonhacks/ead_ontology/blob/main/Taxonomy.xlsx).

4. **Predication:** The fourth step determines the relevant attributes of the different concepts in an ontology. Generally, three types of properties are differentiated: atomic properties (AP), complex properties (CP), and reference properties (RP).

Due to our focus on defining terms and often missing information about the concrete properties of certain concepts, we decided to neglect this step for now.

5. **Parthood:** In the fifth step, the concepts are analyzed towards their architectural structure. In other words, PART-OF relations among these concepts are identified and documented.

We identified 22 PART-OF relations between the terms. We also added 24 terms, which we recognized were complementary to existing terms. Finally, we clustered the different terms into 5 clusters that refer to the main concepts in the domain of EA Debt.

6. **Ontology:** Finally, the ontology is constructed based on the findings from the previous steps. Additionally, domain-specific relations and constraints, such as cardinalities, are added, and the ontology is documented in a formal language.

We documented the ontology within Visual Paradigm following UFO [21]. The resulting ontology is presented in Sect. 4 along with its basic concepts. The ontology is available on github<sup>5</sup>.

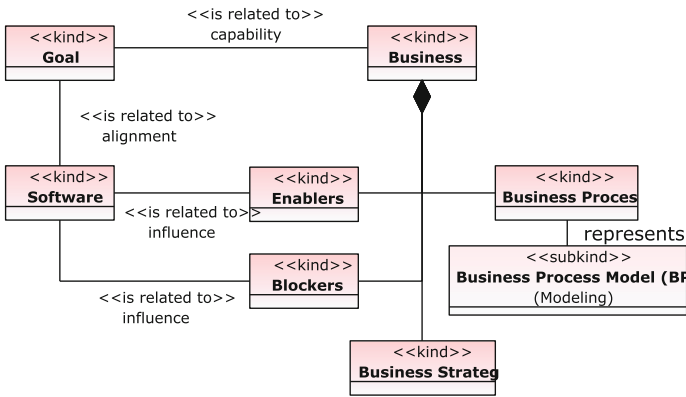


Fig. 2. Business domain ontology

## 4 Ontology

The ontology was split into five diagrams, representing five domains, namely Enterprise Architecture (EA), Technical Debt (TD), Software Engineering (SE),

<sup>5</sup> [https://github.com/simonhacks/ead\\_ontology/blob/main/ead-ontology.vpp](https://github.com/simonhacks/ead_ontology/blob/main/ead-ontology.vpp).





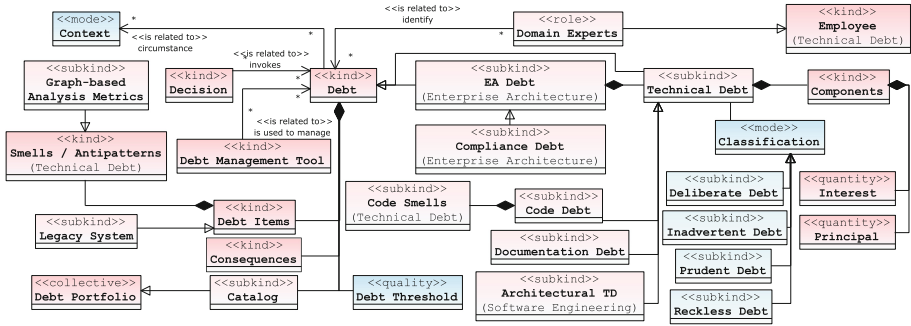


Fig. 6. TD domain ontology

Architecture is a big part of EA, in the identified keywords, this is not reflected. In the publications, there is no focus on architecture and projects, but instead on financing (as seen by the emphasis on debt [43] and risk aversion [55]), as well as conveying knowledge (as seen by the focus on representation [20] and stakeholders [31,45]).

Notably, we recognized a change in the used definition of EA Debt, no longer including the aspect of being a metric, thus defining it as: *“the deviation of the currently present state of an enterprise from a hypothetical ideal state.”* (e.g., [23]). This is mainly motivated to better differentiate between cause (debt) and symptom (smell), initially manifested in a single definition. As this eases the understanding of the concept, we also follow this differentiation. Furthermore, a discussion arose around the phrase “hypothetical ideal state” as it is impossible to determine. One suggestion is to use “planned future state” instead. However, we do not think this properly reflects the idea beyond EA Debt as a planned future state does not need to be flawless and thus still can incorporate EA Debts that are consciously taken but should be documented.

Given how young the field of EA Debts is, one can only expect the ontology to change and grow with time. This is rightfully so, as with each piece of information, there will be more clarity about the definitions and relations of the terms.

*Technical Debt.* The second domain (cf. Figure 6) concerns TD. TD is an inseparable part of EA Debts. It involves the debt committed on the Technical and Application layer. It is tightly related to the product and has been studied longer than EA Debts. Even though the two are inseparable, based on the identified keywords, there is a gap in their consideration in papers related to EA Debts. We observed that only one of two metrics is discussed, leaving the review of the principal insufficient. Similarly, considering the TD classification proposed



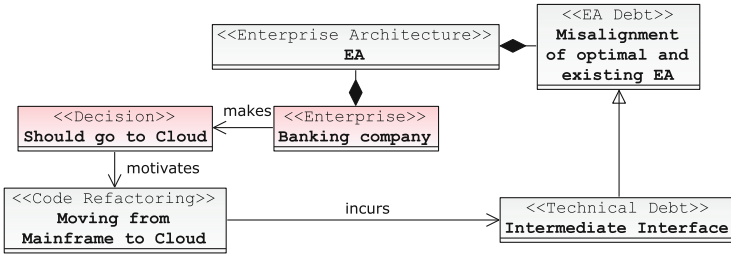


Fig. 7. Instantiation of the Ontology for an Intermediate Interface.

by Fowler [17], only inadvertent debt got enough attention to appear as one of the identified keywords. This might indicate a lack of research into other debt classes.

As the concepts of TD and EA Debt are related to Financial Debt, one can observe that many keywords on the diagram are from the financial sector.

*Software Engineering.* As the area of SE has been studied longer, it was easier to identify the underlying relations between the identified keywords. As seen in Fig. 5, the diagram presents a small part of a SE ontology composed of keywords often appearing in the context of EA Debts. However, it still provides valuable information. It is related to TD through Refactoring [18], which pays back the TD made visible through Code Smells. It is also associated with EA through the consideration of Architecture. The limited review of Architecture in the context of EA Debt indicates that SE is closer to TD than EA. This is further supported by the definition of EA, which focuses more on Business and Data, leaving the Technical aspects compared to TD.

*Business.* The Business domain (cf. Figure 2) focuses on the process and aspects enabling or blocking it. Relevant business information can be modeled to support stakeholders in evaluating the business processes [51]. However, business seems to be a supporting domain as it does not result in many relevant keywords.

*Modeling.* Modeling is a supporting domain for the entire ontology and is shown in Fig. 4. It represents the functionality needed to assess the state of other domains, especially business, EA, and SA.

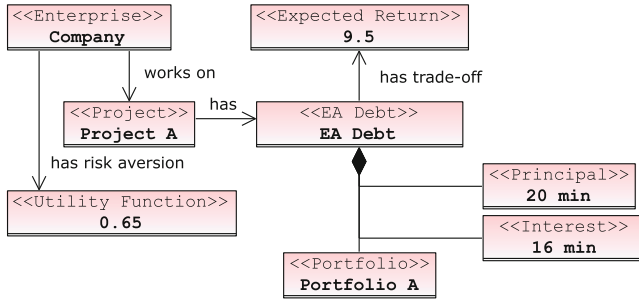


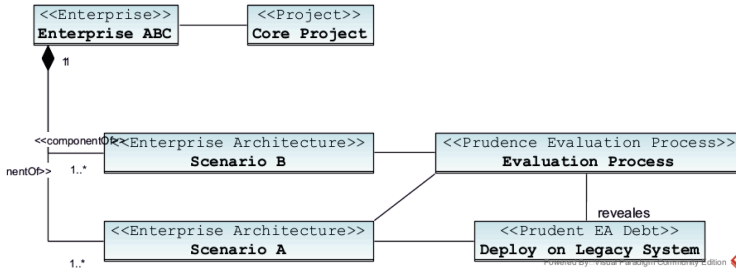
Fig. 8. Instantiation of the Ontology for Calculating the Optimal Portfolio.

## 5 Demonstration

We create concrete instances based on previously reported cases to showcase the developed ontology. More concretely, we illustrate one case from the first publication [22], one case elaborating on the prioritization of EA Debts [55], and one case introducing the concepts of prudence [44]. We opted for these cases as they cover many aspects captured in the ontology.

The first case [22] takes place in a company in the banking domain, which is step-by-step moving its applications from the mainframe to the cloud. Two applications that depend on each other should go to the cloud. However, due to delays, one of the applications cannot be moved to the cloud at the same time as the other application. Therefore, an intermediate interface between the cloud and the mainframe needs to be implemented that is not part of the envisioned optimal EA and, thus, causes an EA Debt. The instantiation of this example can be found in Fig. 7.

The second case [55] illustrates how a company can decide which project to conduct to find the optimal amount of EA Debt according to their risk aversion. Therefore, five project options are envisioned with their respective portfolio risk and expected return. To explain the computation of these parameters, one project is presented in more detail, with four different EA Debts. The principal, the interest, and the interest growth rate are given for each EA Debt. Additionally, the covariance between the EA Debts is provided to calculate the portfolio risk and the expected return. The instantiation of this example can be found in Fig. 8.



**Fig. 9.** Instantiation of the Ontology for Prudence in EA Debt.

The third case [44] presents a toy example where an enterprise faced with complex maintenance of a core functionality project decides to work on reducing TD through refactorings and enforcing architecture guidelines. The conflict between on the one hand time and budget constraints and on the other hand the importance of new functionality forces the enterprise to evaluate the prudence and recklessness of two potential scenarios using the Prudence Evaluation Process (PEP). In the first scenario, the guidelines are violated by deploying on the legacy system, saving time and resources. The second scenario is where policies are followed by deploying the modern system while spending more time and resources on integration with the core functionality. The instantiation of this example can be found in Fig. 9.

## 6 Related Work

One stream of research that links ontologies and EA uses ontologies to describe EA itself. As such, Kang et al. [30] describe an ontology that provides more detailed semantics about EA and facilitates communication around the different stakeholders. Therefore, they define three different ontologies. One ontology to explain business terms, one for the elements of the EA, and finally, the different concepts are linked via relationships. Al Hadidi and Baghdadi [1] propose an ontology to model the interaction between different organization types. They focus on loosely coupled enterprises that join forces to provide new services and temporary cooperations to access new markets. Similarly, Janulevičius et al. [27] suggest an ontology to reflect on the security properties of cloud computing in EA. Mainly, they focus on security controls for the essential documents and integrating the ontology into EA modeling.

Another direction of research elaborates on linking EA models with concrete ontologies. For example, Hinkelmann et al. [25,26] use EA models to have a graphical organization representation and integrate them with machine-readable enterprise ontologies. Therefore, they map the EA modeling notation to a respective ontology that provides additional information to make the graphical representation machine-readable. Another angle is taken by Bakhshandeh et al. [9] and Antunes et al. [5,6], who use ontologies to integrate ArchiMate with different other modeling languages that are better suited for specific domains. Moreover, their integration of ArchiMate and ontologies allows a better analysis of the model.

A third facet of related research uses ontologies to enrich existing EA modeling approaches with new concepts. Azevedo et al. [7,8] perform an ontological analysis on the concepts of resources, capabilities, and competencies and extend ArchiMate to be able to cope with these concepts in the domain of portfolio management. Other approaches enrich ArchiMate to allow risk analysis [40] based on the Common Ontology of Value and Risk, perform value modeling [42] illustrated on the case study of a low-cost airline, or incorporate trust [4] demonstrated on a COVID-19 data repository.

Finally, there is research that elaborates on ontologies for Technical Debt. However, research in this direction is scarce. We could only identify three works that use ontologies to structure knowledge around Technical Debt or use it for its analysis. Firstly, Alves et al. [3] propose a first step to an ontology of terms of Technical Debt. Secondly, an ontology has been proposed for a more concrete instance of Technical Debt, i.e., Requirements Debt [36]. And finally, Händler and Neumann [24] suggest an ontology for refactoring in game design, which is used in a teaching case.

## 7 Conclusion

In this work, we have presented a first step to an ontology describing the domain of EA Debts. To achieve this, we analyzed the existing publications in the field, identified the most relevant keywords, and arranged them in the ontology. Moreover, we added further concepts that were missing. Those concepts seem to be understudied in the domain as they are usually considered in the domain of Technical Debt but do not appear in research related to EA Debt. Thus, this draws an opportunity for future research.

From a methodological point of view, we recognized a lack of tool support for extracting keywords from articles. We were forced to perform significant manual work to enable the analysis, as the articles are often solely provided in semi-structured format (PDF).

Finally, this is the first attempt to structure the knowledge in the domain of EA Debt. This is a continuous effort; the ontology needs to be updated as the field develops. Moreover, we plan to perform other actions to improve the ontology, e.g., by interviewing researchers and practitioners to grasp concepts not documented in scientific articles.

## References

1. Al Hadidi, F., Baghdadi, Y.: Ontology for enterprise interactions: extended and virtual enterprises. In: Baghdadi, Y., Harfouche, A. (eds.) *ICT for a Better Life and a Better World*. LNISO, vol. 30, pp. 365–379. Springer, Cham (2019). [https://doi.org/10.1007/978-3-030-10737-6\\_24](https://doi.org/10.1007/978-3-030-10737-6_24)
2. Alexander, P., Hacks, S., Jung, J., Lichter, H., Steffens, U., Uludağ, Ö.: A framework for managing enterprise architecture debts - outline and research directions. In: *CEUR Workshop Proceedings*, vol. 2628 (2020)
3. Alves, N.S., Ribeiro, L.F., Caires, V., Mendes, T.S., Spínola, R.O.: Towards an ontology of terms on technical debt. In: *2014 Sixth International Workshop on Managing Technical Debt*, pp. 1–7 (2014). <https://doi.org/10.1109/MTD.2014.9>
4. Amaral, G., Sales, T.P., Guizzardi, G., Almeida, J.P.A., Porello, D.: Modeling trust in enterprise architecture: a pattern language for ArchiMate. In: Grabis, J., Bork, D. (eds.) *PoEM 2020*. LNBP, vol. 400, pp. 73–89. Springer, Cham (2020). [https://doi.org/10.1007/978-3-030-63479-7\\_6](https://doi.org/10.1007/978-3-030-63479-7_6)
5. Antunes, G., Bakhshandeh, M., Mayer, R., Borbinha, J., Caetano, A.: Using ontologies for enterprise architecture integration and analysis. *Complex Syst. Inform. Model. Q.* **1**, 1–23 (2014)
6. Antunes, G., Bakhshandeh, M., Mayer, R., Borbinha, J., Caetano, A.: Using ontologies for enterprise architecture analysis. In: *2013 17th IEEE International Enterprise Distributed Object Computing Conference Workshops*, pp. 361–368 (2013). <https://doi.org/10.1109/EDOCW.2013.47>
7. Azevedo, C.L., Iacob, M.E., Almeida, J.P.A., van Sinderen, M., Pires, L.F., Guizzardi, G.: An ontology-based well-founded proposal for modeling resources and capabilities in archimate. In: *2013 17th IEEE International Enterprise Distributed Object Computing Conference*, pp. 39–48 (2013). <https://doi.org/10.1109/EDOC.2013.14>
8. Azevedo, C.L., Iacob, M.E., Almeida, J.P.A., van Sinderen, M., Pires, L.F., Guizzardi, G.: Modeling resources and capabilities in enterprise architecture: a well-founded ontology-based proposal for ArchiMate. *Inf. Syst.* **54**, 235–262 (2015). <https://doi.org/10.1016/j.is.2015.04.008>

9. Bakhshandeh, M., Antunes, G., Mayer, R., Borbinha, J., Caetano, A.: A modular ontology for the enterprise architecture domain. In: 2013 17th IEEE International Enterprise Distributed Object Computing Conference Workshops, pp. 5–12 (2013). <https://doi.org/10.1109/EDOCW.2013.8>
10. Bente, S., Bombosch, U., Langade, S.: Collaborative Enterprise Architecture: Enriching EA with Lean, Agile, and Enterprise 2.0 Practices. Morgan Kaufmann (2012)
11. Campos, R., Mangaravite, V., Pasquali, A., Jorge, A.M., Nunes, C., Jatowt, A.: YAKE! Collection-independent automatic keyword extractor. In: Pasi, G., Piwowarski, B., Azzopardi, L., Hanbury, A. (eds.) ECIR 2018. LNCS, vol. 10772, pp. 806–810. Springer, Cham (2018). [https://doi.org/10.1007/978-3-319-76941-7\\_80](https://doi.org/10.1007/978-3-319-76941-7_80)
12. Cunningham, W.: The WyCash portfolio management system. ACM SIGPLAN OOPS Messenger **4**(2), 29–30 (1993). <https://doi.org/10.1145/157710.157715>
13. Daoudi, S., Larsson, M., Hacks, S., Jung, J.: Discovering and assessing enterprise architecture debts. *Complex Syst. Inform. Model. Q.* (2023)
14. De Nicola, A., Missikoff, M.: A lightweight methodology for rapid ontology engineering. *Commun. ACM* **59**(3), 79–86 (2016). <https://doi.org/10.1145/2818359>
15. De Nicola, A., Missikoff, M., Navigli, R.: A software engineering approach to ontology building. *Inf. Syst.* **34**(2), 258–275 (2009)
16. D’Antonio, F., Missikoff, M., Taglino, F.: Formalizing the OPAL eBusiness ontology design patterns with owl. In: Gonçalves, R.J., Müller, J.P., Mertins, K., Zelm, M. (eds.) Enterprise Interoperability II, pp. 345–356. Springer, London (2007). [https://doi.org/10.1007/978-1-84628-858-6\\_38](https://doi.org/10.1007/978-1-84628-858-6_38)
17. Fowler, M.: Technical debt quadrant (2009). <https://martinfowler.com/bliki/TechnicalDebtQuadrant.html>
18. Fowler, M.: Refactoring: improving the design of existing code. In: 11th European Conference, Jyväskylä, Finland (1997)
19. Gampfer, F., Jürgens, A., Müller, M., Buchkremer, R.: Past, current and future trends in enterprise architecture—a view beyond the horizon. *Comput. Ind.* **100**, 70–84 (2018)
20. Glaser, P.L., Ali, S.J., Sallinger, E., Bork, D.: Model-based construction of enterprise architecture knowledge graphs. In: Almeida, J.P.A., Karastoyanova, D., Guizzardi, G., Montali, M., Maggi, F.M., Fonseca, C.M. (eds.) EDOC 2022. LNCS, vol. 13585, pp. 57–73. Springer, Heidelberg (2022). [https://doi.org/10.1007/978-3-031-17604-3\\_4](https://doi.org/10.1007/978-3-031-17604-3_4)
21. Guizzardi, G., Botti Benevides, A., Fonseca, C.M., Porello, D., Almeida, J.P.A., Prince Sales, T.: UFO: unified foundational ontology. *Appl. Ontol.* **17**(1), 167–210 (2022)
22. Hacks, S., Hofert, H., Salentin, J., Yeong, Y.C., Lichter, H.: Towards the definition of enterprise architecture debts. In: 2019 IEEE 23rd EDOCW, pp. 9–16. IEEE (2019)
23. Hacks, S., Jung, J.: A first validation of the enterprise architecture debts concept. In: van der Aa, H., Bork, D., Proper, H.A., Schmidt, R. (eds.) BPMDS EMMSAD 2023. LNBIP, vol. 479, pp. 17–226. Springer, Cham (2023). [https://doi.org/10.1007/978-3-031-34241-7\\_15](https://doi.org/10.1007/978-3-031-34241-7_15)

24. Haendler, T., Neumann, G.: Ontology-based analysis of game designs for software refactoring. In: CSEDEU (1), pp. 24–35 (2019)
25. Hinkelmann, K., Gerber, A., Karagiannis, D., Thoenssen, B., van der Merwe, A., Woitsch, R.: A new paradigm for the continuous alignment of business and it: combining enterprise architecture modelling and enterprise ontology. *Comput. Ind.* **79**, 77–86 (2016). <https://doi.org/10.1016/j.compind.2015.07.009>. Special Issue on Future Perspectives On Next Generation Enterprise Information Systems
26. Hinkelmann, K., Maise, M., Thönssen, B.: Connecting enterprise architecture and information objects using an enterprise ontology. In: Proceedings of the First International Conference on Enterprise Systems: ES 2013, pp. 1–11 (2013). <https://doi.org/10.1109/ES.2013.6690088>
27. Janulevičius, J., Marozas, L., Čenys, A., Goranin, N., Ramanauskaitė, S.: Enterprise architecture modeling based on cloud computing security ontology as a reference model. In: 2017 Open Conference of Electrical, Electronic and Information Sciences (eStream), pp. 1–6 (2017). <https://doi.org/10.1109/eStream.2017.7950320>
28. Jung, J., Hacks, S., De Gooijer, T., Kinnunen, M., Rehring, K.: Revealing common enterprise architecture debts: conceptualization and critical reflection on a workshop format industry experience report. In: Proceedings - IEEE International Enterprise Distributed Object Computing Workshop, EDOCW, pp. 271–278 (2021). <https://doi.org/10.1109/EDOCW52865.2021.00058>
29. Kaisler, S., Armour, F.: 15 years of enterprise architecting at HICSS: revisiting the Critical Problems. In: Proceedings of the 50th Hawaii International Conference on System Sciences 2017, pp. 4807–4816 (2017)
30. Kang, D., Lee, J., Choi, S., Kim, K.: An ontology-based enterprise architecture. *Expert Syst. Appl.* **37**(2), 1456–1464 (2010). <https://doi.org/10.1016/j.eswa.2009.06.073>
31. Kanji, S., Alexander, P., Lichter, H.: Reporting framework for enterprise architecture debts. Master’s thesis (2022)
32. Kotusev, S.: Enterprise architecture: what did we study? *Int. J. Coop. Inf. Syst.* **26**(4) (2017)
33. Lankhorst, M.: *Enterprise Architecture at Work*. Springer, Heidelberg (2009)
34. Lapalme, J.: Three schools of thought on enterprise architecture. *IT Prof.* **14**(6), 37–43 (2012)
35. Lehmann, B.D., Alexander, P., Lichter, H., Hacks, S.: Towards the identification of process anti-patterns in enterprise architecture models. In: 8th International Workshop on Quantitative Approaches to Software Quality in conjunction with the 27th Asia-Pacific Software Engineering Conference (APSEC 2020), vol. 2767, pp. 47–54 (2020)
36. Lenarduzzi, V., Fucci, D.: Towards a holistic definition of requirements debt. In: 2019 ACM/IEEE International Symposium on Empirical Software Engineering and Measurement (ESEM), pp. 1–5 (2019). <https://doi.org/10.1109/ESEM.2019.8870159>
37. Li, Z., Avgeriou, P., Liang, P.: A systematic mapping study on technical debt and its management. *J. Syst. Softw.* **101**, 193–220 (2015). <https://doi.org/10.1016/j.jss.2014.12.027>

38. Liss, L., Kämmerling, H., Alexander, P., Lichter, H.: Towards a catalog of refactoring solutions for enterprise architecture smells. In: Joint Proceedings of SEED 2021 & QuASoQ 2021 Co-located with 28th Asia Pacific Software Engineering Conference 2021, Taipei [Virtual], 6 December 2021, vol. 3062, pp. 60–69 (2021)
39. Morakanyane, R., Grace, A., O'Reilly, P.: Conceptualizing digital transformation in business organizations: a systematic review of literature. In: 30th Bled eConference: Digital Transformation - From Connecting Things to Transforming our Lives, BLED 2017, pp. 427–444 (2017)
40. Prince Sales, T., Almeida, J.P.A., Santini, S., Baião, F., Guizzardi, G.: Ontological analysis and redesign of risk modeling in ArchiMate. In: 2018 IEEE 22nd International Enterprise Distributed Object Computing Conference (EDOC), pp. 154–163 (2018). <https://doi.org/10.1109/EDOC.2018.00028>
41. Salentin, J., Hacks, S.: Towards a catalog of enterprise architecture smells. In: WI2020 Community Tracks, pp. 276–290. GITO Verlag (2020)
42. Sales, T.P., Roelens, B., Poels, G., Guizzardi, G., Guarino, N., Mylopoulos, J.: A pattern language for value modeling in ArchiMate. In: Giorgini, P., Weber, B. (eds.) CAiSE 2019. LNCS, vol. 11483, pp. 230–245. Springer, Cham (2019). [https://doi.org/10.1007/978-3-030-21290-2\\_15](https://doi.org/10.1007/978-3-030-21290-2_15)
43. Schütz, J., Gómez, J.M.: Towards collaborative technical debt management in systems of systems. In: Proceedings of the 3rd International Conference on Technical Debt, TechDebt 2020, pp. 87–91. Association for Computing Machinery, New York (2020). <https://doi.org/10.1145/3387906.3388620>
44. Slupczynski, A., Alexander, P., Lichter, H.: A process for evaluating the prudence of enterprise architecture debts. In: Proceedings of the 25th International Conference on Enterprise Information Systems - Volume 2: ICEIS, pp. 623–630. INSTICC, SciTePress (2023). <https://doi.org/10.5220/0011971400003467>
45. Slupczynski, A.M.: Towards a framework for evaluating the prudence of enterprise architecture debts. Masterarbeit, RWTH Aachen University, Aachen (2022). <https://doi.org/10.18154/RWTH-2022-01253>, <https://publications.rwth-aachen.de/record/840789>. veröffentlicht auf dem Publikationsserver der RWTH Aachen University 2022; Masterarbeit, RWTH Aachen University, 2021
46. Smajevic, M., Hacks, S., Bork, D.: Using knowledge graphs to detect enterprise architecture smells. In: Serral, E., Stirna, J., Ralyté, J., Grabis, J. (eds.) PoEM 2021. LNBIP, vol. 432, pp. 48–63. Springer, Cham (2021). [https://doi.org/10.1007/978-3-030-91279-6\\_4](https://doi.org/10.1007/978-3-030-91279-6_4)
47. Tabrizi, B., Lam, E., Girard, K., Irvin, V.: Digital transformation is not about technology. *Harv. Bus. Rev.* 2–7 (2019)
48. Tieu, B., Hacks, S.: Determining enterprise architecture smells from software architecture smells. In: 2021 IEEE 23rd Conference on Business Informatics (CBI), vol. 02, pp. 134–142 (2021). <https://doi.org/10.1109/CBI52690.2021.10064>
49. Uludağ, Ö., Kleehaus, M., Xu, X., Matthes, F.: Investigating the role of architects in scaling agile frameworks. In: 2017 IEEE 21st International Enterprise Distributed Object Computing Conference (EDOC), pp. 123–132. IEEE (2017)
50. Uludağ, Ö., Reiter, N., Matthes, F.: What to expect from enterprise architects in large-scale agile development? A multiple-case study. In: 25th AMCIS (2019)



51. Weske, M.: Business Process Management-concepts, Languages, Architectures. Springer, Berlin (2007). <https://doi.org/10.1007/978-3-540-73522-9>
52. Wierda, G.: Chess and the Art of Enterprise Architecture. R&A (2015)
53. Wierda, G.: Mastering ArchiMate, 2 edn. R&A (2017)
54. Winter, R., Fischer, R.: Essential layers, artifacts, and dependencies of enterprise architecture. In: 2006 10th IEEE International Enterprise Distributed Object Computing Conference Workshops (EDOCW 2006), p. 30. IEEE (2006)
55. Yeong, Y., Hacks, S., Lichter, H.: Prioritization of EA debts facilitating portfolio theory. In: CEUR Workshop Proceedings, vol. 2511 (2019)