Learning Loops in the Age of AI

Learning Loops in the Age of AI

Jan Bosch[1,2][a] and Helena Holmström Olsson[3][b]
[1]*Department of Computer Science and Engineering, Chalmers University of Technology, Sweden*
[2]*Department of Computer Science and Engineering, Eindhoven University of Technology, The Netherlands*
[3]*Department of Computer Science and Media Technology, Malmö University, Sweden*

Keywords: Learning Loops, DevOps, Continuous Improvement, Software-Intensive Systems, Artificial Intelligence.

Abstract: Software-intensive systems companies face mounting pressures to accelerate time-to-market. This drives adoption of DevOps, frequent deployments and AI-enabled analytics. However, while traditional feedback loops channel usage data, logs and interactions into development cycles, this doesn't necessarily translate into learning. Although companies use DevOps, they still view product development as building products with a fixed scope. To address this, we conceptualize the notion of learning loops and how companies move towards continuous improvement of product performance. In our view, 'learning loops' distinguish themselves by translating data from products into actionable improvements executed by humans, by traditional ML, by Agentic AI or by a combination of these. This paper synthesizes longitudinal case study research and interviews, revealing that mechanisms like continuous integration and deployment, A/B testing, federated and reinforcement learning are unified instances of post-deployment learning loops. The contribution of this paper is two-fold. First, we provide empirical examples reflecting how R&D teams and systems learn and improve performance over time. Second, we present a conceptual model in which we detail the concept of learning loops that can be executed by humans, by traditional ML, by Agentic AI or by a combination of these.

1 INTRODUCTION

Software-intensive companies operate in an increasingly dynamic environment where time-to-market is critical. To realize this, short development and delivery cycles are key and during recent years the transition towards continuous practices has permeated most industry domains (López et al., 2021), (Nguyen-Duc et al., 2023), (Zhu et al., 2016) (Olsson and Bosch, 2020), (Bosch and Olsson, 2021), (Dakkak et al., 2023b). Release cadences that were once measured in months or years have been compressed into weeks, days or even multiple deployments per day. This acceleration is tightly coupled with the adoption of DevOps (Zhu et al., 2016), (Lwakatare et al., 2020), (Dakkak et al., 2023b), in which development and operations are integrated and value is delivered through frequent deployment of new software versions.

With continuous deployment of functionality there is the opportunity for continuous feedback from which R&D teams, as well as the system itself, can learn and improve. With effective feedback loops in place, data collected from products is continuously fed back into DevOps cycles as the basis for refining functionality, improving quality and informing new feature development (Zhu et al., 2016), (Bou Ghantous and Gill, 2017), (Perera et al., 2017). With the advent of Artificial Intelligence (AI), these feedback loops can operate at far greater scale and speed as machine learning techniques automatically extract meaningful patterns from high-volume and noisy data, detect anomalies and estimate the impact of alternative design choices. Consequently, AI has the potential to transform feedback into a continuous and data-driven learning mechanism that can used to iteratively optimize both products and processes in near real-time.

However, in our experience there is a difference between feedback loops and learning loops. While feedback represents data coming back from products in the field, learning loops represent the actions taken, and the improvements made, based on this data. What we have noticed is that most companies have ample feedback from products in the field. This, however, does not automatically mean that they learn from this data and that they take actions to improve their systems based on this data. In practice, companies have

[a] https://orcid.org/0000-0003-2854-722X
[b] https://orcid.org/0000-0002-7700-1816

Bosch, J. and Olsson, H. H.
Learning Loops in the Age of AI.
DOI: 10.5220/0015033400004015

In *Proceedings of the 21st International Conference on Evaluation of Novel Approaches to Software Engineering (ENASE 2026) - Volume 2*, pages 959-966
ISBN: 978-989-758-828-0; ISSN: 2184-4895

continuous integration (Ståhl and Mårtensson, 2018), and continuous deployment (Rodríguez et al., 2017), (Savor et al., 2016) infrastructures in place but often they still view product development as building products with a fixed scope. In addition, the traditional view is that R&D teams are the ones improving the systems. Although this is still a valid view, it is only partly true as the introduction of agentic AI is rapidly changing the nature of product development.

In this paper, we conceptualize the notion of learning loops. By using examples from our previous research as well as from a recent interview study, we present a holistic view on how systems learn over time and how this learning is amplified in AI-driven systems. To achieve this, we synthesize findings and present results that build on several years of close collaboration with companies in the software-intensive systems domain. Over the years, we have worked together with these companies to improve performance of their systems. The companies use a variety of different mechanisms such as DevOps, A/B testing, federated learning (FL) and reinforcement learning (RL). When reflecting back, we realize that although these mechanisms may look independent, and as if they serve different purposes, they are all instances of a common and higher-level concept that is what we refer to as *learning loops*. Our research focuses on post-deployment learning, i.e., learning that takes place when the product is deployed in the field and the conceptual model that we present is inductively derived based on empirical insights that we collected throughout our research.

The key contributions of this paper are the following. First, we provide empirical examples that reflect how R&D teams and systems learn and improve their performance over time. Second, we present a conceptual model in which we detail the concept of learning loops that can be executed by R&D teams (humans), by traditional ML, by Agentic AI or by a combination of these.

The remainder of the paper is organized as follows. In section 2, we outline the background and we review literature that is relevant for this study. In section 3, we describe the research method that we employed for this study. In section 4, we present the empirical findings. In section 5, we present a conceptual model that is inductively derived from our empirical findings. In section, 6, we discuss threats to validity and in section 7, we conclude the paper.

2 BACKGROUND

2.1 Continuous Value Delivery

Contemporary software-intensive companies operate in an environment where value delivery cycles are continuously shortening (López et al., 2021), (Nguyen-Duc et al., 2023), (Zhu et al., 2016). This acceleration is tightly coupled with the increasing adoption of DevOps practices (Zhu et al., 2016), (Lwakatare et al., 2020), (Dakkak et al., 2023b). DevOps denotes an organizational and technical approach that integrates software development and IT operations to enable fast, reliable and continuous delivery of value to users(Savor et al., 2016), (Rodríguez et al., 2017).

As the basis for continuous value delivery, and as input for DevOps cycles, data from products in the field is critical for closing the feedback loop between real-world usage and development (Olsson and Bosch, 2014), (Fabijan et al., 2015), (Fagerholm et al., 2017). For software intensive systems, this data is often the only reliable way to see how complex products behave at scale, under diverse conditions and in different usage situations. In software-intensive systems companies, the use of data has expanded to involve development of services for enhancing and extending product performance, as well as to develop entirely new data-driven and digital services (Kampker et al., 2018), (Bosch and Olsson, 2021).

2.2 Continuous Learning

In a dynamic context as described above, learning emerge as a critical organizational capability (Lakshmi and Ajay, 2025). With effective learning loops, companies can close the gap between action and insight by repeatedly collecting data on how the system behaves, analyzing the outcomes and feeding the results back into design, configuration and operation.

While the concept of learning loops is not new (Argyris, 1996), (Al-Baik and Miller, 2018), (Buckell and Macintyre, 2021), it has primarily been studied from an organizational perspective and particularly within management literature. In (Georges L. Romme and Van Witteloostuijn, 1999), the authors discuss how the development of a sustainable learning ability of the organization is a prerequisite to survive. The authors use the definitions by (Argyris, 1996) when distinguishing between single, double and triple loop learning. In (Löf, 2010), the author recognizes how the concept of learning loops display distinct similarities with adaptation and transformation and also here the concept of single, double and triple loop learning

is used.

In software engineering research, there is an increasing number of studies on the topic of self-adaptive systems e.g., (Weyns, 2019), (De Lemos et al., 2013), (Macías-Escrivá et al., 2013), (Gheibi et al., 2021). Although there exist several definitions, self-adaptivity is typically referred to as the capability of a system to adjust its behavior in response to the environment (Brun et al., 2009) and/or a system that evaluates its own behavior and changes its own performance when the evaluation indicates that it is not accomplishing what the software is intended to do, or when better functionality or performance is possible (Salehie and Tahvildari, 2012).

In our research, we recognize the importance of previous research. However, our research takes a different point of departure as we note that the traditional understanding of learning loops is changing. First, while organizational studies recognize learning as critical, the learning loops that are referred to are primarily concerned with process improvements based on historical data and collective (human) experiences. In contrast to this, our research focuses on teams that improve specific capabilities of a system and the learning loops we refer to are proactively initiated by utilizing techniques such as A/B testing, reinforcement learning and federated learning. In addition, we recognize the fact that learning is not only a human capability but a capability that can be executed by humans, by traditional ML, by Agentic AI or by a combination of these. Second, while research on self-adaptive systems has shown the importance of adaptation, the primary approach to achieve this is control loops. The focus is on different techniques for maintaining the desired behaviour of a system. In contrast, the learning loops that we refer to are not concerned with maintaining homeostasis but instead with how to continuously improve a system.

3 RESEARCH METHOD

The objective of this paper is to explore and conceptualize the concept of learning loops and how these are used as part of DevOps practices to improve performance of systems. To study this, we bring together two empirical components:

1. We synthesize findings from *longitudinal multi-case study research* in which we accumulated insights by engaging in close collaboration with companies in the software-intensive systems domain.
2. We present findings from an *expert interview study* conducted in 2025 - 2026 where we met with experts from multiple companies who have long-term experience from roles closely connected to continuous improvements of large-scale systems.

3.1 Longitudinal Multi-Case Study Research

In this paper, we synthesize insights that we gained during several years of multi-case study research (Runeson and Höst, 2009), (Maxwell, 2012). Based on these longitudinal and collaborative research engagements, we are able to reflect on the ways in which we see industry practices change.

3.2 Interview Study

We employed an expert interview approach due to its potential to generate in-depth and multi-perspective data that represents the experiences of experts in the field (Seaman, 1999), (Myers and Newman, 2007), (Schultze and Avital, 2011). All experts were senior people with long-term experience from the companies they represent. They work closely with teams responsible for data collection, analysis and processing and they are involved in AI/ML related initiatives to further enhance product capabilities and performance. All interviews lasted for one hour, they were conducted online using Teams and they were recorded and transcribed.

3.3 Data Collection and Analysis

For the synthesis of previous research, we use a set of selected studies in which we explored DevOps, A/B testing, federated learning and reinforcement learning initiatives in the companies. These studies were conducted as multi-case studies and they have been reported in previous publications. In this paper, we synthesize these findings as we realize that they are instances of a higher-level concept what we refer to as learning loops. Throughout our research, we collected data using a mix of data collection methods such as interviews, workshops, experiments and simulations (Runeson and Höst, 2009). The examples we have selected reflect research that was conducted between 2015 - 2025.

The expert interviews were conducted in two separate phases. The *first round of interviews* was conducted between January 2025 - April 2025 and involved semi-structured online interviews with 11 experts representing nine different companies (Table 1). The *second round of interviews* was conducted between October 2025 - January 2026 and involved in-

Table 1: Overview of interviewees involved in the first round of interviews.

Industry	Role
Telecommunications	CI/CD expert
Digital infrastructure and communication	Head of AI and data
Consulting	AI and technology advisor
Automotive	AI adoption manager
Consulting	AI & Society expert
Tobacco	AI transformation expert
Technology and services supplier	Head of Engineering
Security and surveillance	Engineering manager
Security and surveillance	Product owner
Technology and services supplier	Security & Software expert
Defense	Chief digital officer

Table 2: Overview of interviewees involved in the second round of interviews.

Industry	Role
Security and surveillance	Data and AI strategy
Telecommunications	Data scientist
Automotive	Data scientist
Automotive	Business transformation
Processing and packaging	Data and AI engineer
Defense	Product manager

terviews of a semi-structured format with six experts representing six different companies (Table 2).

For the analysis of our empirical data, we employed thematic analysis which is a well-established method for systematically identifying, analyzing and reporting patterns of meaning (themes) within qualitative datasets (Runeson and Höst, 2009), (Maxwell, 2012). By relating individual observations to higher-order themes the approach of thematic analysis provided a robust foundation for synthesizing our findings and for generalizing the concept of learning.

4 EMPIRICAL FINDINGS

In this section, we first present selected examples from several years of multi-case study research with companies in the software-intensive systems domain. Second, we present results from the interview study that add additional empirical insights from industry experts.

4.1 Multi-Case Study Research Examples

4.1.1 DevOps, DataOps and MLOps

In (Dakkak et al., 2023a), we develop a framework for product-service systems where services act as the glue between Dev (development) and Ops (operations). Based on longitudinal participant observations at a telecommunications company, we identify bidirectional flows, i.e., how software flows from 'Dev' to customers via 'Ops', while feedback (usage data, customer insights etc.) flows back via services to inform 'Dev'. In (Dakkak et al., 2024), we explore AIOps and we show how AI can automate anomaly detection, root-cause analysis and remediation in service delivery. Learning loops here involve AI models trained on operational data to predict and prevent issues and continuously refine service quality and system reliability. In (John et al., 2025), we develop and validate a five-dimensional MLOps framework encompassing a five-stage maturity model that illustrates the steps companies take to advance their MLOps practices.

4.1.2 A/B Testing

In (Fabijan et al., 2018b), we outline how mature organizations conduct experiments as an embedded part of their organization. Here, product teams continuously generate hypotheses, run controlled experiments, analyze results via shared platforms and use the learnings to shape subsequent ideas. In (Fabijan et al., 2018a), we extend this line of work by proposing a model that addresses the seven critical aspects of experimentation and can help companies to transform their organizations into learning laboratories. Finally, in (Liu et al., 2021) we study A/B testing practices in automotive where fleets are smaller and experiments are constrained. Here, learning loops are realized by repeatedly deploying instrumented variants to subsets of vehicles, thus enabling continuous improvement in conservative and hardware-constrained domains.

4.1.3 Federated Learning

In (Zhang et al., 2025), we focused on how federated learning techniques can support continuous data-driven improvements of complex and distributed systems. Our research presents federated learning as an infrastructure with edge nodes (vehicles, devices, base stations etc.) that repeatedly train local models on their latest data, send model updates to an aggregator, receive an improved model and immediately redeploy this model in the field. In another study (Zhang et al., 2024), we proposed real-time end-to-end federated learning with asynchronous aggregation protocols. This reduces training latency and allows models to be updated more frequently from whichever clients are currently available. Finally, in (Zhang et al., 2023a) we explored decentralized and low-effort setups where federated learning can be configured and operated with minimal manual intervention, emphasizing automation of client coordination, experiment

management and deployment to edge hardware.

4.1.4 Reinforcement Learning

In (Zhang et al., 2023b), we apply deep reinforcement learning to dynamic control problems in telecommunications. We frame RL as an iterative learning mechanism in which an agent interacts with a real or simulated environment, observes states and rewards, updates its policy and deploys the improved policy to perform better and hence, creating a self-reinforcing cycle of exploration, evaluation and adaptation. In a following paper (Zhang et al., 2023c), we extend our research to multi-agent RL where multiple drones learn independently without communication or central control. This decentralized setup allows the system to scale and improve continuously.

4.2 Interview Results

Our expert interviews reveal several patterns that the interviewees believe are changing the ways in which companies learn from data from products in the field to continuously improve performance of their systems. As one common pattern, customer feedback loops are evolving from being infrequent and manual activities to shorter, faster and more frequent mechanisms that directly influence product development. Our interviews reveal that this evolution is driven by three key enablers. First, ubiquitous telemetry and in-product signals from connected systems enable continuous and passive data collection on usage patterns, errors and engagement that by far surpasses the cadence of surveys or meetings. Second, DevOps and continuous delivery practices allow organizations to act on feedback within hours or days via small incremental deployments rather than waiting for quarterly reviews. Third, AI-powered analytics and experimentation, e.g., A/B testing, automate pattern detection and causal inference that helps surface actionable insights from massive feedback streams. As a result, customer feedback loops now operate as high-velocity closed systems that accelerate product evolution.

As a second pattern, the interviewees note how product and usage feedback loops, as well as development and test feedback loops, are shortening dramatically due to automation, continuous practices and richer data flows. According to the interviewees, this has profound effects on system performance improvements. In addition, AI-driven anomaly detection and automated alerting surface issues within minutes while techniques like A/B testing provide teams with insights on feature impact during live operation.

As a third pattern, business KPIs are shortening from monthly or quarterly reports to near real-time dashboards and automated alerts. This is transforming static metrics into dynamic signals that guide immediate learning and immediate action.

5 LEARNING LOOPS IN THE AGE OF AI

Traditional systems were built in such a way that the behavior was algorithmically defined through code developed by engineers. At that time, there was already a learning loop, but it was most often concerned with product generations. With DevOps, companies extended the traditional learning loop by implementing an additional, and continuous, learning loop between products deployed in the field and *R&D teams*. In such a learning loop, teams and product managers collect data from the field and use this data to determine the content for the next release (Ebert, 2007).

With the emergence of AI, we see that in addition to the R&D teams learning loop, multiple new learning loops are emerging. First, *reinforcement learning* and other machine learning approaches can operate on the deployed system itself. Here, the local learning loop allows the system to learn from its actions and the results that these actions generate (Wang et al., 2025), (Nouwou Mindom, 2025). Techniques realizing this may include periodic retraining locally on the system based on data collected during the previous period or reinforcement learning where the system operates based on a state space, an action space and a reward function and then trains itself to learn which action is the best to select in each state.

In the *reinforcement learning* case, the operating loop and the learning loop are typically located in the same location, i.e. the deployed system. However, that does not always have to be the case. One of the typical patterns is where a central server collects data from all the systems deployed in the field, periodically retrains a model and then distributes it to all deployed systems who then use the updated model in their operating loops. In this case, the learning loop runs between the centralized compute infrastructure and the systems in the field. In such a model, the deployed systems share their data with the central compute infrastructure which, in the case of many systems and frequent executions of the operating loop, can result in high data volumes. In addition, in the cases where the data that is executed upon in the operating loop has associated confidentiality and privacy concerns, there is an associated risk of data leakage and violations of privacy. In this case, the learning takes place

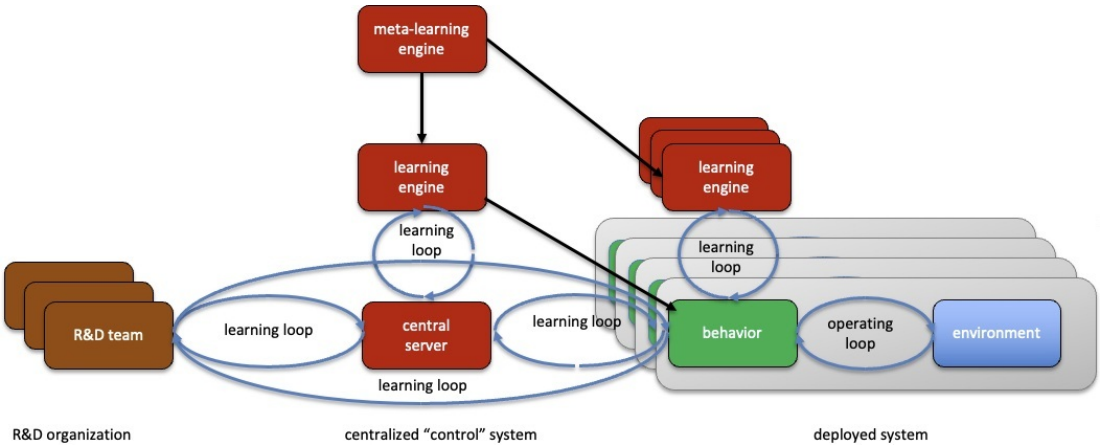

Figure 1: End-to-end learning loop system.

in the central server, but the operating loop still resides in the deployed systems.

A potential solution to the confidentiality and privacy challenge is where we adopt a hybrid approach. *Federated learning* is an approach where deployed systems train locally and share model updates with a central server that combines the model updates (Lo et al., 2021), (Goel, 2024), (Zhang et al., 2021). This central server then periodically distributes an updated global model that combines all the learnings from the deployed systems. This approach works well in the case where a global model is the optimal approach to all deployments. In some cases, however, deployed systems may learn to customize themselves to the local context in which the system finds itself. In that case, the performance of the global model need to be compared to the performance of the trained local model and each system may decide to adopt the global model or to stay with its local deployment.

We already discussed the learning loop between the *R&D teams and the deployed systems,* but R&D teams can also operate in a meta-learning context where the team optimizes the learning algorithms in the central server or on the deployed systems. For example, the team could develop an improved reinforcement learning algorithm for deployment on systems in the field or an improved federated learning model that is deployed in the central server.

In Figure 2, we illustrate our conceptualization of the entire learning loop infrastructure from an intra-company perspective. In the figure, we show multiple deployed systems that can have a local learning loop or that may rely on a centralized system for learning. In addition, the R&D teams can improve the software in the deployed systems and/or in the central server. Also, teams can improve the learning and meta-learning engines.

6 THREATS TO VALIDITY

Throughout our study, we carefully considered the validity and trustworthiness of our results as emphasized in e.g., (Runeson and Höst, 2009). To address and to mitigate *construct* validity, we made sure to share our definition of key concepts and terminology in the introduction of each workshop and/or interview to ensure a common understanding and to avoid potential misinterpretations. To address and to mitigate *internal* validity, we engaged with highly experienced and well-recognized experts from the different companies who were able to provide an in-depth and rich understanding of the complexity of the phenomenon. To address *external* validity, we used our empirical findings to derive the conceptual model we present with the intention to provide value for organizations with similar characteristics and challenges as the software-intensive systems companies we studied.

7 CONCLUSIONS

In this paper, we conceptualize the notion of learning loops and we show how these are critical for an organization to move towards continuous improvement of system performance. To achieve this, we synthesize findings and present results that build on several years of close collaboration with multiple companies in the software-intensive systems domain.

The key contributions of this paper are the following. First, we provide empirical examples that reflect how R&D teams and systems learn and improve their performance over time. We do this by summarizing our experiences from longitudinal case study research in multiple companies in the software-intensive systems domain. Second, we present a conceptual model

in which we detail the concept of learning loops that can be executed by *R&D teams (humans), by traditional ML, by Agentic AI or by a combination of these*.

ACKNOWLEDGEMENTS

We want to thank the Software Center companies for their support and engagement in this research.

REFERENCES

Al-Baik, O. and Miller, J. (2018). Integrative double kaizen loop (idkl): towards a culture of continuous learning and sustainable improvements for software organizations. *IEEE transactions on Software Engineering*, 45(12):1189–1210.

Argyris, C. (1996). Organizational learning ii. *Theory, method, and practice*.

Bosch, J. and Olsson, H. H. (2021). Digital for real: A multicase study on the digital transformation of companies in the embedded systems domain. *Journal of Software: Evolution and Process*, 33(5):e2333.

Bou Ghantous, G. and Gill, A. (2017). Devops: Concepts, practices, tools, benefits and challenges. *Pacis2017*.

Brun, Y., Di Marzo Serugendo, G., Gacek, C., Giese, H., Kienle, H., Litoiu, M., Müller, H., Pezzè, M., and Shaw, M. (2009). Engineering self-adaptive systems through feedback loops. In *Software engineering for self-adaptive systems*, pages 48–70. Springer.

Buckell, C. and Macintyre, M. (2021). Sustaining continuous improvement through double loop learning. In *European Lean Educator Conference*, pages 3–12. Springer.

Dakkak, A., Bosch, J., and Holmstrom Olsson, H. (2024). Towards aiops enabled services in continuously evolving software-intensive embedded systems. *Journal of Software: Evolution and Process*, 36(5):e2592.

Dakkak, A., Bosch, J., and Olsson, H. H. (2023a). Devservops: Devops for product-oriented product service systems. In *2023 49th Euromicro Conference on Software Engineering and Advanced Applications (SEAA)*, pages 328–331. IEEE.

Dakkak, A., Bosch, J., Olsson, H. H., and Mattos, D. I. (2023b). Continuous deployment in software-intensive system-of-systems. *Information and Software Technology*, 159:107200.

De Lemos, R., Giese, H., Müller, H. A., Shaw, M., Andersson, J., Litoiu, M., Schmerl, B., Tamura, G., Villegas, N. M., Vogel, T., et al. (2013). Software engineering for self-adaptive systems: A second research roadmap. In *Software Engineering for Self-Adaptive Systems II: International Seminar, Dagstuhl Castle, Germany, October 24-29, 2010 Revised Selected and Invited Papers*, pages 1–32. Springer.

Ebert, C. (2007). The impacts of software product management. *Journal of systems and software*, 80(6):850–861.

Fabijan, A., Dmitriev, P., McFarland, C., Vermeer, L., Holmström Olsson, H., and Bosch, J. (2018a). Experimentation growth: Evolving trustworthy a/b testing capabilities in online software companies. *Journal of Software: Evolution and Process*, 30(12):e2113.

Fabijan, A., Dmitriev, P., Olsson, H. H., and Bosch, J. (2018b). Online controlled experimentation at scale: an empirical survey on the current state of a/b testing. In *2018 44th Euromicro Conference on Software Engineering and Advanced Applications (SEAA)*, pages 68–72. IEEE.

Fabijan, A., Olsson, H. H., and Bosch, J. (2015). Data-driven decision-making in product r&d. Springer.

Fagerholm, F., Guinea, A. S., Mäenpää, H., and Münch, J. (2017). The right model for continuous experimentation. *Journal of Systems and Software*, 123:292–305.

Georges L. Romme, A. and Van Witteloostuijn, A. (1999). Circular organizing and triple loop learning. *Journal of organizational change management*, 12(5):439–454.

Gheibi, O., Weyns, D., and Quin, F. (2021). Applying machine learning in self-adaptive systems: A systematic literature review. *ACM Transactions on Autonomous and Adaptive Systems (TAAS)*, 15(3):1–37.

Goel, P. K. (2024). Introduction to ai, ml, federated learning, and llm in software engineering. In *Advancing Software Engineering Through AI, Federated Learning, and Large Language Models*, pages 1–16. IGI Global Scientific Publishing.

John, M. M., Olsson, H. H., and Bosch, J. (2025). An empirical guide to mlops adoption: Framework, maturity model and taxonomy. *Information and Software Technology*, 183:107725.

Kampker, A., Husmann, M., Harland, T., Jussen, P., and Steinbauer, M. (2018). Six principles for successful data-driven service innovation in industrial companies. In *2018 IEEE International Conference on Engineering, Technology and Innovation (ICE/ITMC)*, pages 1–8. IEEE.

Lakshmi, H. and Ajay, S. (2025). Learning loops and the process of continuous improvement. In *Unleashing Absorptive Capacity and Unlearning for Organizational Excellence*, pages 113–136. IGI Global Scientific Publishing.

Liu, Y., Mattos, D. I., Bosch, J., Olsson, H. H., and Lantz, J. (2021). Size matters? or not: A/b testing with limited sample in automotive embedded software. In *2021 47th Euromicro Conference on Software Engineering and Advanced Applications (SEAA)*, pages 300–307. IEEE.

Lo, S. K., Lu, Q., Wang, C., Paik, H.-Y., and Zhu, L. (2021). A systematic literature review on federated machine learning: From a software engineering perspective. *ACM computing surveys (CSUR)*, 54(5):1–39.

Löf, A. (2010). Exploring adaptability through learning layers and learning loops. *Environmental Education Research*, 16(5-6):529–543.

López, L., Bagnato, A., Ahberve, A., and Franch, X. (2021). Qfl: Data-driven feedback loop to manage quality in agile development. In *2021 IEEE/ACM 43rd International Conference on Software Engineering: Software Engineering in Society (ICSE-SEIS)*, pages 58–66. IEEE.

Lwakatare, L. E., Crnkovic, I., and Bosch, J. (2020). Devops for ai – challenges in development of ai-enabled applications. In *2020 International Conference on Software, Telecommunications and Computer Networks (SoftCOM)*, pages 1–6.

Macías-Escrivá, F. D., Haber, R., Del Toro, R., and Hernandez, V. (2013). Self-adaptive systems: A survey of current approaches, research challenges and applications. *Expert Systems with Applications*, 40(18):7267–7279.

Maxwell, J. A. (2012). *Qualitative research design: An interactive approach*. Sage publications.

Myers, M. D. and Newman, M. (2007). The qualitative interview in is research: Examining the craft. *Information and organization*, 17(1):2–26.

Nguyen-Duc, A., Cabrero-Daniel, B., Przybylek, A., Arora, C., Khanna, D., Herda, T., Rafiq, U., Melegati, J., Guerra, E., Kemell, K.-K., et al. (2023). Generative artificial intelligence for software engineering–a research agenda. *arXiv preprint arXiv:2310.18648*.

Nouwou Mindom, P. S. (2025). *Deep Reinforcement Learning in Software Engineering: Empirical Insights and Practical Guidelines*. PhD thesis, Polytechnique Montréal.

Olsson, H. H. and Bosch, J. (2014). The hypex model: from opinions to data-driven software development. In *Continuous software engineering*, pages 155–164. Springer.

Olsson, H. H. and Bosch, J. (2020). Going digital: Disruption and transformation in software-intensive embedded systems ecosystems. *Journal of Software: Evolution and Process*, 32(6):e2249.

Perera, P., Silva, R., and Perera, I. (2017). Improve software quality through practicing devops. In *2017 seventeenth international conference on advances in ICT for emerging regions (ICTer)*, pages 1–6. IEEE.

Rodríguez, P., Haghighatkhah, A., Lwakatare, L. E., Teppola, S., Suomalainen, T., Eskeli, J., Karvonen, T., Kuvaja, P., Verner, J. M., and Oivo, M. (2017). Continuous deployment of software intensive products and services: A systematic mapping study. *Journal of systems and software*, 123:263–291.

Runeson, P. and Höst, M. (2009). Guidelines for conducting and reporting case study research in software engineering. *Empirical software engineering*, 14:131–164.

Salehie, M. and Tahvildari, L. (2012). Towards a goal-driven approach to action selection in self-adaptive software. *Software: Practice and Experience*, 42(2):211–233.

Savor, T., Douglas, M., Gentili, M., Williams, L., Beck, K., and Stumm, M. (2016). Continuous deployment at facebook and oanda. In *Proceedings of the 38th International Conference on software engineering companion*, pages 21–30.

Schultze, U. and Avital, M. (2011). Designing interviews to generate rich data for information systems research. *Information and organization*, 21(1):1–16.

Seaman, C. B. (1999). Qualitative methods in empirical studies of software engineering. *IEEE Transactions on software engineering*, 25(4):557–572.

Ståhl, D. and Mårtensson, T. (2018). *Continuous practices: a strategic approach to accelerating the software production system*. Lulu. com.

Wang, D., You, H., Zhu, L., Lin, K., Chen, Z., Yang, C., Yu, J., Wang, Z., and Chen, J. (2025). A survey of reinforcement learning for software engineering. *arXiv preprint arXiv:2507.12483*.

Weyns, D. (2019). Software engineering of self-adaptive systems. In *Handbook of software engineering*, pages 399–443. Springer.

Zhang, H., Bosch, J., and Holmström Olsson, H. (2021). Engineering federated learning systems: A literature review. In *International Conference on Software Business*, pages 210–218. Springer.

Zhang, H., Bosch, J., and Olsson, H. H. (2023a). Quafedasync: Quality-based asynchronous federated learning for the embedded systems. In *2023 49th Euromicro Conference on Software Engineering and Advanced Applications (SEAA)*, pages 70–73. IEEE.

Zhang, H., Bosch, J., and Olsson, H. H. (2024). Edgefl: A lightweight decentralized federated learning framework. In *2024 IEEE 48th Annual Computers, Software, and Applications Conference (COMPSAC)*, pages 556–561. IEEE.

Zhang, H., Bosch, J., and Olsson, H. H. (2025). Enabling efficient and low-effort decentralized federated learning with the edgefl framework. *Information and Software Technology*, 178:107600.

Zhang, H., Li, J., Qi, Z., Aronsson, A., Bosch, J., and Olsson, H. H. (2023b). Deep reinforcement learning for multiple agents in a decentralized architecture: A case study in the telecommunication domain. In *2023 IEEE 20th International Conference on Software Architecture Companion (ICSA-C)*, pages 183–186. IEEE.

Zhang, H., Li, J., Qi, Z., Aronsson, A., Bosch, J., and Olsson, H. H. (2023c). Multi-agent reinforcement learning in dynamic industrial context. In *2023 IEEE 47th Annual Computers, Software, and Applications Conference (COMPSAC)*, pages 448–457. IEEE.

Zhu, L., Bass, L., and Champlin-Scharff, G. (2016). Devops and its practices. *IEEE Software*, 33(3):32–34.