e-Informatica Software Engineering Journal, Volume 18, Issue 1, 2024, pages: 240102, DOI: 10.37190/e-Inf240102

Continuous Software Engineering Practices in AI/ML Development Past the Narrow Lens of MLOps: Adoption Challenges

Sini Vänskä*, Kai-Kristian Kemell**, Tommi Mikkonen***, Pekka Abrahamsson****

*Deloitte, Finland
**Department of Computer Science, University of Helsinki, Finland
***Faculty of Information Technology, University of Jyväskylä, Finland
****Faculty of Information Technology and Communication Sciences, Tampere University, Finland

sini.vanska@deloitte.fi, kai-kristian.kemell@helsinki.fi,
tommi.j.mikkonen@jyu.fi, pekka.abrahamsson@tuni.fi

Abstract

Background: Continuous software engineering practices are currently considered state of the art in software engineering (SE). Recently, this interest in continuous SE has extended to ML system development as well, primarily through MLOps. However, little is known about continuous SE in ML development outside the specific continuous practices present in MLOps.
Aim: In this paper, we explored continuous SE in ML development more generally, outside the specific scope of MLOps. We sought to understand what challenges organizations face in adopting all the 13 continuous SE practices identified in existing literature.
Method: We conducted a multiple case study of organizations developing ML systems. Data from the cases was collected through thematic interviews. The interview instrument focused on different aspects of continuous SE, as well as the use of relevant tools and methods.
Results: We interviewed 8 ML experts from different organizations. Based on the data, we identified various challenges associated with the adoption of continuous SE practices in ML development. Our results are summarized through 7 key findings.
Conclusion: The largest challenges we identified seem to stem from communication issues. ML experts seem to continue to work in silos, detached from both the rest of the project and the customers.

Keywords: artificial intelligence, machine learning, continuous software engineering, continuous star, multiple case study

1. Introduction

Continuous Software Engineering is the current trend in Software Engineering (SE). In brief, continuous SE comprises various so-called continuous practices, which aim to eliminate discontinuities in SE in order to make it a more continuous process. For example, continuous integration focuses on closer collaboration between development and deployment.

Submitted: 09 Mar. 2023; Revised: 20 Aug. 2023; Accepted: 21 Aug. 2023; Available online: 28 Aug. 2023

Continuous SE is currently state of the art in SE, and companies even consider certain continuous practices vital for remaining competitive going forward [1].

In particular, bridging the gap between development and operations has received much attention following the advent of continuous SE. Indeed, one widely discussed topic related to continuous SE is DevOps. DevOps, a portmanteau of Development and Operations, focuses on the integration between development and deployment. It involves a number of continuous SE practices, which are also arguably some of the most high-profile ones: continuous integration, continuous delivery, and continuous deployment, as well as testing automation (or continuous testing), are typically considered to be necessary for DevOps [2]. To further highlight the current importance of continuous SE practices, we turn to Moreschini et al. [3] who state that "DevOps practices are the de facto standard when developing software".

By now, continuous SE practices have been explored in various contexts in SE research, including the field of Machine Learning (ML). In ML system development, continuous SE is still an emerging phenomenon. In fact, ML system development in general remains a novel topic from the point of view of SE [4, 5]. Much of the current discussion on continuous SE in the context of ML has been focused on the concept of MLOps. With some simplification, MLOps can be considered to be the application of DevOps into the context of ML systems [6].

Continuous SE, as Fitzgerald and Stol [7] conceptualize it, however, is a much larger phenomenon than DevOps (or MLOps), which only comprises a small set of continuous SE practices. Fitzgerald and Stol [7] posit that continuous SE comprises at least 13 different continuous practices, split between the areas of business, development, operations, and innovation. Many of these other continuous SE practices have received little attention compared to the ones included in DevOps. This is especially the case for ML development, where little research on continuous SE exists outside the topic of MLOps. Moreover, MLOps in and of itself is still an emerging phenomenon [6].

Overall, the development of ML systems presents novel challenges in SE, as ML components need to be incorporated into the software system being developed. These components require new know-how and are typically developed separately from the rest of the system by different personnel (e.g., data scientists). Incorporating the development of ML components into the general SE process (via tooling, methods, practices, etc.) is the core issue behind these challenges from the point of view of SE [4]. While ML has been widely studied across disciplines, much of this extant research has focused on technical challenges in ML instead of looking at the development of these systems through the lens of SE [4]. This is especially the case for continuous SE in ML, where little research outside the context of MLOps exists.

In this paper, we begin to address this perceived gap by studying ML development from a more general continuous SE point of view. In doing so, we look past the lens of MLOps (and DevOps) in order to explore the thus far largely unexplored (in ML development) continuous SE practices described by Fitzgerald and Stol [7]. We begin to explore this topic using an exploratory, qualitative multiple case study ($n = 8$) research approach. The specific research question we tackle is formulated as follows: *what are the challenges associated with the use of continuous SE practices in ML development?* Additionally, we are interested in understanding which continuous SE practices are currently used in ML development, which are not, and why.

2. Background

In this section, we discuss the theoretical background of this study. In Section 2.1, we discuss continuous SE in more detail. In Section 2.2, we discuss ML development and how it relates to conventional software development. In Section 2.3, we discuss related work on MLOps and challenges in ML development.

2.1. Continuous SE

Continuous SE aims to eliminate discontinuities in SE [7]. While previous lightweight methodologies have already stressed the importance of focusing on error detection and quick fixes in particular, continuous SE is more holistic. Indeed, in continuous SE, the entire software life cycle, including business strategy and operations, is considered to be a part of development process [7]. Continuous SE builds on agile SE, and agile has been argued to be an important requirement for adopting, e.g., DevOps [8].

Continuous SE, in practice, encompasses various different practices that Fitzgerald and Stol [7] refer to as continuous* (continuous star). These, they argue, can be split into three areas (as seen in Figure 1) that together comprise continuous SE: (1) Business Strategy and Planning; (2) Development, and (3) Operations. In addition (continuous) improvement encompasses all three. Whereas DevOps focuses on collaboration between development and operations, collaboration between business strategy and planning and development is referred to as BizDev, and collaboration between all three is referred to as BizDevOps [7].

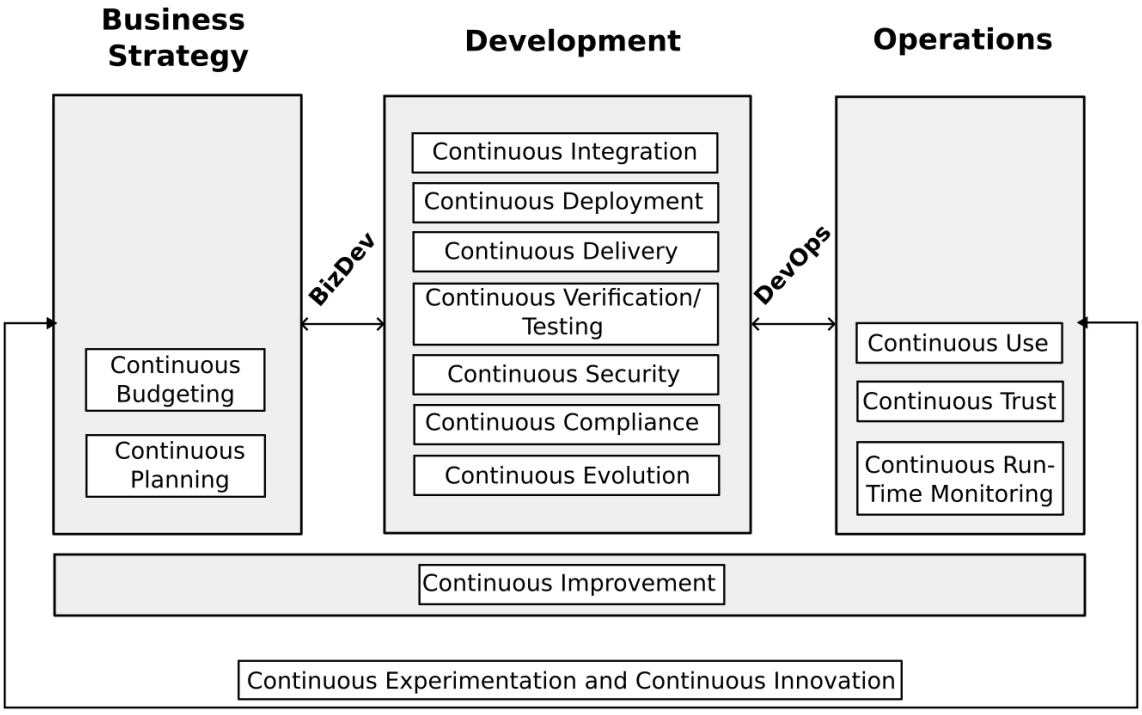

Figure 1. Continuous SE according to Fitzgerald and Stol [7]

Altogether, 13 different continuous* practices are identified by Fitzgerald and Stol [7]. These 13 practices are as follows. The first area (business strategy and planning)

includes continuous planning and continuous budgeting. The second area (development) includes continuous integration, continuous deployment, continuous delivery, continuous verification/testing, continuous security, continuous compliance, and continuous evolution. The third area (operations) includes continuous use, continuous trust, and continuous run-time monitoring. Finally, continuous improvement encompasses all the areas, and continuous experimentation and innovation drive organizations to perform better [7]. These are all illustrated in Figure 1.

Continuous SE further reinforces the need to break down silos inside organizations. DevOps focuses on breaking down the silos between development and operations [9]. However, even more comprehensive collaboration between departments may be needed to achieve continuous SE in some contexts [10]. For example, in the context of ML development, continuous SE faces new challenges as the continuity needs to be extended to also include the ML experts working on the ML components of the system [6], as we discuss in Section 2.3. Next, in Section 2.2, we discuss what makes ML development different from the point of view of SE.

2.2. ML Development from the point of view of software engineering

AI has largely become synonymous with ML today [5]. ML applications are useful for poorly understood problem domains, domains with valuable regularities in their databases waiting to be discovered, and domains in changing environments [11]. In terms of SE practice, ML systems differ from conventional software systems due to the addition of ML components. Developing these components requires new know-how and is handled by various ML experts (e.g., data scientists and ML developers), who are a new addition to the SE process [5]. Aside from various the technical challenges associated with developing ML components, integrating the ML experts into the larger SE workflow poses challenges for organizations developing ML [4].

ML systems are often divided into three classes. The most common is supervised learning where the training data and the "right answer" are accessible. In unsupervised learning, the systems learn by trying to find the common structure in the data on their own. The third, so-called reinforcement learning, refers to systems that evolve by learning in a sequence that leads it to a given goal [12]. The process requires a lot of testing and data sets created (or simply used) for training purposes, and the product can still fail to fulfill its requirements. Moreover, resource estimation in ML development is difficult [13].

Yet, as some existing papers argue, much of ML development is ultimately still software development. Mikkonen et al. [12] cite Google in stating that "even if ML is at the core of an ML system, only 5% or less of the overall code of that total ML production system" is code related to ML. Indeed, ML features are simply "embedded into a larger software system that hosts, provides access to, and monitors ML features" [10], while the rest of the development endeavor is conceptually speaking conventional SE.

Nonetheless, the addition of these ML components into the SE process poses challenges from the point of view of SE. Perhaps most importantly, as already touched upon, the ML development requires collaboration between ML experts and rest of the development team(s) [4, 5]. Two studies reviewing existing literature (one systematic literature review [4] and one systematic mapping study [5]) highlight various challenges in ML development discussed by existing literature. Martínez-Fernández et al., in their systematic mapping study [5], list various challenges associated with SE for ML systems, highlighting the wide variety of challenges associated with ML development from a SE point of view. These include

challenges such as end-to-end ML-based systems including components written in a wide variety of programming languages, reliance on third party components, ML presenting challenges from the point of view of changing requirements (whereas ML components create entanglement), etc.

Giray [4] conducted an SLR specifically focused on SE challenges in ML development, identifying various challenges unique to ML system development across the following six main categories: (1) requirements engineering (RE), (2) design, (3) software development and tools, (4) testing and quality, (5) maintenance and configuration management, and (6) SE process and management. First, in relation to RE, issues such as difficulties managing customer expectations due to lack of customer ML understanding, quantitative measurements being new to many stakeholders, and having to deal with new types of quality attributes (e.g., explainability, fairness) resulted in challenges. Secondly, design-wise, ML systems posed challenges from the point of view of monitoring performance degradation on production (concept drift, etc.), as well as in terms of system architecture, as the interplay of the different parts of an ML system can often result in issues. Thirdly, the vast number of tools and dealing with development environments and infrastructure, both in terms of understanding them and simply successfully utilizing them in practice (compatibility, etc.), were among the key challenges related to software development and tools, in addition to issues related to dealing with ML models and the data required for ML. Fourthly, testing ML components remains a challenge due to the non-deterministic nature of ML systems, including issues related to designing and evaluating test cases, preparing test data, executing tests, and evaluating test results, as well as actually fixing any bugs that are found (which may also be bugs in the ML libraries, frameworks, and platforms being used). Fifthly, dealing with a history of experiments, re-training and re-deployment, and the configuration management (CM) of data and ML models present new challenges for maintenance and CM. Finally, harmonizing the activities for developing ML components with the rest of the software development process, assessing the ML process, and estimating effort are new challenges related to the SE process and management in ML development.

2.3. Related work: MLOps and continuous SE in ML

Thus far, studies on continuous SE in ML have predominantly approached the topic through MLOps. MLOps can be largely considered DevOps for ML [6, 14]. John et al. [6] conceptualize MLOps in their study, highlighting three different pipelines (data, model, and release), each with 3–4 subprocesses, that together comprise the MLOps pipeline.

Lwakatare et al. [15] discuss SE challenges associated with DevOps in ML development contexts, focusing on technical aspects such as issues related to data and ML models. Symeonidis et al. [16] discuss tooling for MLOps and challenges related to it. Granlund et al. [17] discuss challenges related to implementing MLOps practices in a highly regulated application domain (medical). A number of other papers on MLOps are listed in the multi-vocal literature of John et al. [6], and in the systematic literature reviews of Kolltveit and Li [14] and Lima et al. [18]. Overall, much of the existing research on MLOps has focused on defining the concept and on challenges related to establishing an MLOps pipeline (e.g., challenges related to tooling). Fewer papers can be found on human aspects (e.g., project communication, etc.).

As our focus is on challenges, papers related to challenges in ML development could also be seen as related work. Indeed, we find connections between our results and such papers in Section 6. In this regard, a paper by Serban and van der Blom [19] looks at

SE best practices for ML development in general and discusses the adoption rates of the practices, pointing towards challenges in some areas. The secondary studies of Giray [4] and Martínez-Fernández et al. [5] also list various SE challenges related to ML development based on existing literature, as we have discussed in Section 2.2. Other primary studies discussing challenges include the study of de Souza Nascimento et al. [20] that looks at challenges in ML development overall, with the main challenges being: identifying the clients' business metrics, lack of a defined development process, and designing the database structure. Nahar et al. [21] specifically look at communication and collaboration challenges between data scientists and software developers in ML development in their empirical study.

On the other hand, we are unable to identify existing empirical papers discussing continuous* more generally for ML development. As established in Section 2.1, MLOps (and DevOps) only cover some of the 13 continuous* practices discussed by Fitzgerald and Stol [7]. Thus, in looking at continuous* more generally in this study, we believe this paper presents a novel contribution in the area with its point of view.

3. Research framework

This study is an empirical study utilizing qualitative, thematic interviews as the data collection method (see Section 4). Prior to the data collection, to aid in the creation of a suitable interview instrument, we constructed a research framework. This research framework is split into five elements. Fitzgerald and Stol [7] classify continuous* practices into categories as follows: business, development, operations, and innovation. These are elements (2), (3), (4), and (5) in our framework. In addition, we look at current tools and the use of methods (1) related to both continuous SE and, more broadly, agile in ML development, to better understand the development practices being utilized. These five elements are described in more detail below:

(1) Current tools and use of methods. The first part of the research framework is focused on understanding the tools and methods used in ML development. In particular, this part of the research framework is focused on collecting data on development tools and practices, and especially continuous or agile frameworks or methods used in ML development.

(2) Business strategy and planning. Continuous SE considers software development a continuous, holistic process that bridges different organizational units, and also consequently encompasses business strategy and planning [7]. This link between development and business is referred to as BizDev (which in turn is a part of BizDevOps). The purpose of this theme is to understand how ML experts see their organizations' business and strategic elements. This component is also related to understanding how ML experts deal with changing customer/business requirements.

(3) Development. This part of the research model aims to understand the ML development actions, and how they relate to the project as a whole, i.e., whether the ML related tasks are integrated into the rest of the development process, or more of an independent, separate, and siloed process.

(4) Operations. Continuous SE also includes bridging together development and operations (as in DevOps). This part of the research framework is focused on understanding what practices are used after the project goes into production and how the system is monitored past this point. Existing DevOps literature argues that agile transformation is essential for improving the efficiency of the company in optimizing the lifecycle delivery,

breaking the gaps, and creating a continuous feedback loop between the business users and the development teams [8].

(5) Innovation. Fitzgerald and Stol [7] consider continuous improvement and innovation a part of continuous SE. In continuous SE, a new planning phase starts when new opportunities are recognized. This part of the research model is focused on understanding how ML experts feel about new opportunities, innovation, and new technologies. E.g., are they motivated to suggest new ways of developing ML systems based on lessons learned, both in terms of technologies and project management? In this regard, the state of their current project(s) was discussed in relation to what they would do next.

4. Research method

In this section, we discuss the methodology of the empirical study we conducted. In Section 4.1, we discuss our data collection approach. In Section 4.2, we discuss our data analysis approach.

4.1. Data collection

The data for this study were collected through qualitative interviews. The interviews in question were thematic, semi-structured interviews. The themes for the interviews were based on the research framework discussed in the previous section, i.e., there were five themes that the questions were focused on: (1) current tools and use of methods, (2) business strategy and planning, (3) development, (4) operations, and (5) innovation. The interview instrument in its entirety can be found in the Appendix A.

Eight interviews were conducted with respondents from different organizations working on AI-related projects. The respondents (and thus cases) were selected through convenience sampling. The participants had varied job titles, ranging from research assistant to service manager (see Table 1). Similarly, their past job experience varied greatly, ranging from some months to decades of working with ML technologies. We preferred to have respondents from a variety of organizations, as opposed to more in-depth case studies of fewer organizations, due to the novelty of the topic. Similarly, we opted for a semi-structured and thematic interview format due to the novelty of the research topic.

The interviews were conducted digitally due to the COVID-19 pandemic situation that was on-going at the time of the interviews. The interviews were conducted either in Finnish or English, with the (pre-planned) questions being the same in both cases. Due to the semi-structured approach, each interview was nonetheless different, as additional questions were posed based on the responses of each respondent.

Table 1. Respondents

Respondent	Job title
Respondent 1	Research assistant
Respondent 2	Data scientist
Respondent 3	Research assistant
Respondent 4	Service manager
Respondent 5	Professor (SE)
Respondent 6	Senior lecturer (SE and applied AI)
Respondent 7	Regulation specialist
Respondent 8	Software developer

The average duration of the interviews (recorded data) was 35 minutes, i.e., not including introductory statements, instructions, and any post-interview discussion. The interview recordings were transcribed, and the resulting transcripts were analyzed for this paper. We discuss our analysis approach next.

4.2. Data analysis

To analyze the data, we utilized qualitative thematic analysis as the primary data analysis method. More specifically, we utilized deductive coding as our coding approach. In deductive coding, codes are pre-determined based on a framework and then applied to the data in the coding process.

We utilized deductive coding as the coding approach, using our research framework (see Section 3) as the basis for coding. In practice, the codes were the 13 continuous* practices discussed by Fitzgerald and Stol [7]. As we were interested in exploring how different continuous* practices are utilized in the context of ML development, we focused on determining which practices were utilized and how (if at all), resulting in a rather straightforward coding approach.

These codes were then organized in themes according to our research framework. Thus, these 13 codes were arranged into 5 themes for reporting as follows: (1) current tools and use of methods, (2) business strategy and planning, (3) development, (4) operations, and (5) innovation. These codes and their occurrences are detailed in Table 2. The results of this analysis are reported next in Section 5.

In addition to this coding process focusing on solely continuous* practices, the data was analyzed more generally through the five elements of the research framework. For example, we were interested in the use of tools, which were outside the scope of the continuous* practices used as a framework for deductive coding. This was also the case for agile in relation to continuous* in ML.

Table 2. Codes and their occurrences in the data

Code [Continuous…]	Occurrence(s)
Planning	4
Budgeting	2
Integration	1
Delivery	2
Deployment	3
Verification	2
Testing	2
Compliance	0
Security	0
Evolution	5
Use	1
Trust	0
Run-time monitoring	2
Improvement	1
Innovation	2
Experimentation	3

5. Results

In this section, we discuss the results of the empirical study. This section is split into five subsections, with each subsection covering one of the five themes of our research framework.

While reporting our results, we structure the discussion through the use of **PEC**s (Primary Empirical Contributions). These PECs are intended to communicate our key findings in a clear manner. Later, in Section 6, these PECs are also used to structure the discussion of our results.

Additionally, although we use direct citations from the interviews in the text, it should be noted that these PECs are not based solely on the few citations found in the text. While including all relevant citations would not be feasible in the interest of space, we nonetheless use some citations to liven up the text and to provide some transparency in terms of our use of our data.

First, before going into more detail in our analysis, we can already make one interesting observation based on the codes and their occurrences (Table 2): *continuous compliance* and *continuous security* were not present in our data. In addition *continuous trust* also did not appear in our data the way it is understood by Fitzgerald and Stol [7] (who consider it to mean trust developed over time based on the belief that customer expectations are fulfilled without exploiting their vulnerabilities). In our data, only one of the respondents discussed *continuous trust*, and did so on a more general level in relation to having a good relationship with the client.

PEC1: Continuous compliance, continuous trust, and continuous security were not present within the data.

5.1. Overview of used tools and methods

The first interview theme (out of the five discussed in Sections 3 and 4) was focused on the current work role of the respondent and the tools and methods used in the ML projects they were involved with. The tools of interest included programming languages and software tools used in ML development, alongside any other tools the respondents considered relevant enough to mention.

Python was by far the most common language, discussed by seven of the eight respondents. Java was discussed by two. In terms of software tools utilized, the answers of the respondents were more diverse, especially regarding database systems and cloud services. For example, Respondent 2 discussed Dataprix, which was not brought up by any other respondent:

[Development was] 99% or like fully Python based, but then some of the data preparation was done in a Dataprix, you know the kinda Spark service, so that was used. [...] *I would not say that I followed any framework with intent but rather trying to have like the mindset within many of the frameworks.* [R2]

For the most part, the respondents discussed choosing tools based on their own preferences. Some respondents mentioned having a background in ML as a hobby and, thus, opted to use the tools they were familiar with. Most respondents worked either alone or in a small team with other ML experts, and consequently they were the only ones who needed to understand the tools and the ML components and code. On the other hand, some organizations had processes in place which determined what tools should/could be used. Nonetheless, little consensus existed in terms of tooling.

Aside from tooling, we also explored the use of SE methods. The respondents mentioned Scrum, SAFe, DevOps, and MLOps when asked about the use of methods and other development processes. However, the respondents discussed their use critically. The respondents seemed to consider it more important that, as opposed to strictly following a method, practices that fit the current project context are used on a case-by-case basis. Moreover, in this regard, there seemed to be a disconnect between the rest of the organization and the ML experts. For example, Respondents 2 and 4 both mentioned that their organization had adopted an agile approach but that it had little impact on their ML development aside from it being centered around sprints as a result.

The work is planned in sprints. We have 4 planning periods per year. If we speak about doing AI then you can think that sprints always produce something that can be put into production, with the next sprint it will be improved and expanded. [R4]

Some respondents felt that they did not use any methodologies in their development, but rather, a mindset built on many different methods. This seemed common, with many respondents mentioning frameworks but also specifying that they did not use any of them strictly (e.g., ScrumBut), either on the level of the entire organization or just the ML experts.

PEC2: The SE methods used by the organization at large seem to often have little impact on the ML development processes.

5.2. Business strategy and planning

Questions related to this theme were focused on teamwork, requirements, and resources. In both continuous* and agile, collaboration between business and development teams is considered important.

Half (4) of the respondents said that the ML experts in their organization mainly worked independently. In these cases, only project planning activities (e.g., sprint planning) and result reviews were carried out with the rest of the project team, while most of their work was carried out independently. Nonetheless, the majority of the respondents (6) considered collaboration on a project-level important when discussing teamwork and planning. In particular, two respondents working in large corporations with highly regulated projects considered collaboration and communication a critical success factor for producing ML components that fulfilled their requirements. Yet, aside from one respondent who worked on a project with internal stakeholders, the respondents felt that their work was very independent with little collaboration with other employees. To this end, three respondents felt that they knew very little about the work of other people inside the organization as well.

Multiple respondents discussed issues related to collaboration, or from the point of view of continuous*, *continuous planning* (and budgeting, though hardly discussed). To some extent, this lack of collaboration was attributed to the remote work situation stemming from the COVID-19 pandemic situation that was on-going at the time, although not all respondents worked remotely. Respondent 3 highlighted the importance of informal communication for formulating new ideas and discussing problems with co-workers, while otherwise collaboration was minimal. In fact, three respondents felt that they did not even understand the point of the project they were working on and were simply executing tasks:

Personally, the point of the project is unclear to me. [R3]

Customer involvement was also highly varied between projects. In projects with external customers, customer involvement was not regular from the point of view of the ML experts. Most felt that they were working in a silo, especially as far as the customer was considered.

The respondents also discussed challenges related to *(continuous) budgeting*, as well as resource allocation more generally. Multiple respondents mentioned working on multiple projects at a time. Respondent 4 specified that their organization had difficulties finding ML development capabilities, leading to the existing ML experts being stretched thin.

Practically everyone is doing multiple projects. All people are caught in all cases, work needs to be prioritized. [...] *Resourcing is challenging.* [...] *Plus, there is less and less expertise. AI needs to have analysts and people who know algorithms, it is not that simple to have them on every branch.* [R4]

I do not know how they [resources] are decided. It feels like my time is being spent on everything else that is not related to my work. [R3]

As all except one respondent worked on externally commissioned projects, *continuous budgeting* was up to the customer(s). While the respondents had some control over suggesting the addition of new features (or requirements) into the system/project, the customer had the final say in such matters. Some of the projects had additional resources reserved for use in case there were changes in the project scope (e.g., due to added functionalities).

If we get a green light from a company, the project price already includes a lump sum of money either from the company or in the form of some collaboration. And we try to do our best with that or go as far we can go with that sum of money. [R5]

On the other hand, three respondents felt that the customers' indecisiveness caused issues and unpredictability in the development. This, they felt, was frustrating because the customers seemed to not have a sufficient grasp of the realities of ML development, leading to unreasonable demands at times.

For example, the clients might change their mind in every two weeks as it has happened in some projects or the approach to gain some insight to something has changed. [R1]

Overall, the relationship of the respondents and the rest of their organizations, as well as their clients, was vague. The ML experts were not actively involved in project planning and seldom interacted with the customer(s). This resulted in various issues, as discussed above.

PEC3: ML experts seem to often work in a silo. They do not often participate in business strategy related activities.

5.3. Development

The interview questions related to ML development included questions related to how functionalities are added, how the product is tested, and when the product is ready for production. The disconnects between the ML experts and the rest of the project participants were even more apparent in the respondents' responses when discussing practical development matters. When asked about integration, continuous approaches were not discussed by most respondents. Respondent 3 summarized their issues with the lack of collaboration in terms of development as follows:

I feel that it is up to you to decide [when to implement your work] because there is no teamwork, therefore others have nothing to say. I don't know if it's because of my own experience, but it's hard to trust that that product will work. [R3]

While testing was discussed with all respondents, only one respondent [R7] discussed *continuous testing* in the form of automated testing using an MLOps pipeline. Other respondents said that they tested ML functions manually for various reasons. One respondent specified that they felt that automated testing tools for ML were simply not available for their particular system context. Another respondent added that ML functionalities

were difficult to test due to the high level of technical know-how required to develop them, resulting in the tests being carried out by the same individual(s) who developed the functions.

It was not automated for sure. It was all manual. There was unit testing, then there was regression testing, also integration testing and all the things that you see. [R5]

You work with simple test cases at first and apply it into a bigger chunk of data. Then there's some peer review done by the other data scientist in the product team but it kinda depends how well that can work because if you are working with something that is pretty exotic then not even other data scientist might know that much about it. [R2]

PEC4: Automated testing in ML development remains a challenge for organizations developing ML systems.

Continuous verification was mostly discussed in relation to regulations. This was mostly in relation to data, with regulations such as the GDPR (the EU's General Data Protection Regulation) often affecting ML development due to their use of large amounts of data.

Continuity in general seemed to be less of a concern in situations where the organization simply developed a narrow ML functionality or a model that the customer then implemented and monitored on their own. In such situations where no full, functional system was developed, *continuous quality*, for example, was of little concern to the respondents. This also seemed to apply to *continuous verification* and *continuous compliance*.

5.4. Operations

The questions about operations focused on what happened to the product and project after initial deployment. This included usage and monitoring of the product, as well as customer relations. The questions dealt with user interaction, user expectations, monitoring, as well as the conclusion of the project.

Continuous use focuses on understanding whether user expectations are fulfilled. However, the respondents felt that the users were not well understood. The ML experts seldom had direct contact with the user(s) and only dealt with the representatives of the customer organization. In some cases the respondents had no contact with the customer at all and only communicated with their own organization's project staff. When the ML experts got feedback from the users, it was gathered by the customer organization and forwarded to them. No direct channels existed. One of the respondents specifically remarked that such feedback only reached them if something was "great or terribly wrong."

[Interaction with the users] is rare, a privilege. [R6]

With internal customers, interaction was more frequent (as discussed by Respondent 4). With external customers, much depended on who was the end-user and how the project was organized. On the other hand, some ML experts did not want to interact with the user or customer in the first place:

I guess some people like it and it can give some valuable feedback for the developers, but I did not appreciate it in the past. [R2]

PEC5: The lack of user and customer interaction makes it difficult for ML experts to ensure that the product can be continuously used.

Monitoring practices varied between projects. Monitoring was largely left to the customer. In some cases, the ML experts were in charge of monitoring and problem detection right after initial release, until the customer later took over. Active communication with the customer would typically end after the product was delivered. This also meant that operations was not of much concern in such projects from the point of view of the ML experts.

Well, there is a short price/crisis period during which I can still provide help if still needed but it is the customer's job to deal with the rest. [R1]

None of the respondents mentioned trust (*continuous trust*) as something that they tried to actively establish with the customer. The ML experts seemed to nonetheless recognize the importance of having a good relationship with the customer, e.g., in terms of securing future projects.

There are less opportunities to do something completely innovative once you've released the project or the product. But if new opportunities arise in the sense, if it is a long-term relationship with the client, and if you're continuously working on something bigger, then of course you have the possibility of improving or actually innovating or completely replacing something that you've already delivered a couple of years ago. [R7]

The lack of interaction between the ML experts and the users of the product is challenging for operations. Project contracts typically determined what kind of operational activities the ML experts had in that project. The ML experts themselves, however, seemed to not mind this situation. In fact, many of the respondents seemed satisfied about not having to interact with external stakeholders after deployment.

PEC6: The emphasis placed on operations varies greatly depending on project context. If the customer takes over entirely after release, there is little need for operations from the development organization.

5.5. Improvement and innovation

Due to the disconnect between development and operations resulting from externally commissioned projects and their contractual agreements, only R4, whose organization developed ML systems for internal use, engaged in *continuous improvement and innovation*. As the other organizations (four out of eight) largely moved on after deployment, in some cases following an initial phase where they continued to make sure the system worked as intended briefly after deployment, opportunities for *continuous improvement* were limited. Only if the client proposed further improvements, and provided the funding for them, would work on the system continue past bug fixes or minor improvements made shortly after initial deployment, based on contractual obligations. R4, on the other hand, who worked for internal customers, discussed how feedback gathered after deployment was used to continuously improve the system.

Because the customers were in charge of the projects, improvements and innovations were not automatically thought of, or even considered welcome in the project. As the contracts typically determined the project scope, any potential changes would have to be discussed with the customer. Three respondents described cases where they had proposed small improvements. However, larger innovations were often seen as risky by the customer and the ML experts, and were seldom suggested to the customer. Thus, *continuous innovation* on a project level was largely not considered relevant by the respondents.

On the other hand, half of the respondents felt that an innovative mindset was essential in their line of work on a personal level. As ML is a quickly evolving field, they felt that they needed to be open to learning new things and coming up with new ideas. Even if the current project was easier to carry out as planned earlier, new innovations could help with future ones.

PEC7: New ideas or innovations are not automatically added to on-going ML projects when an external customer is involved, as the customer sets the scope of the project.

6. Discussion

In this section, we discuss the implications of our results. In Table 3 we summarize the Primary Empirical Contributions (PECs) we highlighted during our analysis. We use these PECs to structure the discussion in this section. After covering all the individual PECs, towards the end of this section we return to the research question we outlined at the start of this paper and answer it based on our findings.

Table 3. Primary empirical contributions of the study

PEC	Description
1	Continuous compliance, continuous trust, and continuous security were not present within the data.
2	The SE methods used by the organization at large seem to often have little impact on the ML development processes.
3	ML experts seem to often work in a silo. They do not often participate in business strategy related activities.
4	Automated testing in ML development remains a challenge for organizations developing ML systems.
5	The lack of user and customer interaction makes it difficult for ML experts to ensure that the product can be continuously used.
6	The emphasis placed on operations varies greatly depending on project context. If the customer takes over entirely after release, there is little need for operations from the development organization.
7	New ideas or innovations are not automatically added to on-going ML projects when an external customer is involved, as the customer sets the scope of the project.

PEC1, on a more general level, highlights that some continuous* practices receive less attention than others, as seen in more detail in Table 2. This, to some extent, supports our original motivation behind the study: some continuous* practices are seldom studied compared to the most commonly discussed ones. However, as our data set is not large, given the exploratory nature of the study, we would not place much emphasis on this particular observation. It is also worth keeping in mind that the domain of the project may play a large role in how relevant various regulations are (e.g., medical domain) from the point of view of continuous compliance, and that issues such as cybersecurity may be delegated to specific experts within an organization. PEC1 may nonetheless be of interest from the point of view of future studies, however, as it may give an idea of which continuous* practices are common out on the field and which are not.

PEC2. Based on our data, ML development is seldom carried out using SE methods by the book. This is consistent with existing research where the lack of a defined development process in ML development is identified as an issue [20]. Moreover, some of the respondents discussed using some practices, such as sprints, as a part of the project in general, but that 1) the associated method (e.g., SCRUM) was not followed by the book in the project in general, and 2) they were at best erratically applied to the ML portion of the project. PEC2 in this fashion highlights one way in which ML experts continue to work in a silo (as they often seem to do [22] at present). As far as agile approaches are considered, our results support the observations of Serban and van der Blom [19] who posit, based on a survey, that traditional SE practices have a lower adoption rate than ML specific practices in ML development. Our results in this regard provide further insights on how this manifests in practice. Finally, the respondents of our study discussed a wide variety of tools (database systems, cloud services, etc.) used in ML development, which corresponds with existing

research where, e.g., Kim et al. [23] highlight that the vast number of available tools can be a challenge in and of itself.

PEC3 corresponds with extant research in that bridging the gap between the ML experts and the rest of the developers is a challenge in ML development [4, 21]. It seems common for ML experts to work in a silo [21, 22], although there are also examples of successfully integrating ML experts with the rest of the development team(s) found in existing papers [23]. This is also an issue MLOps aims to tackle as an approach to ML development, much like how DevOps focused on bridging the gap between development and operations. Our findings may help further illustrate what this means in practice and what kind of issues this results in.

Extant research argues that one of the key factors of success in implementing DevOps is communication [24]. This, thus, presents various problems for adoption of continuous* as well, as DevOps belongs under the umbrella of continuous*. Communication issues in ML development are also acknowledged in extent literature, where, indeed, communication issues between the ML experts and the rest of the development team and organization are considered a recurring challenge [4, 5]. Educating software engineers on ML and ML experts on SE could help in this regard by making it easier for the project participants to develop a mutual understanding of the project [21].

However, communication issues are not an issue unique to ML development or continuous SE in the context of ML. Indeed, organizations adopting DevOps often face issues related to communication in and between teams, due to, e.g., differences in the professional and personal backgrounds of the employees [25]. Issues with communication and siloing are also seen in relation to software security, where collaboration between security experts and software developers has been a challenge, and where more recently DevSecOps has looked to improve the situation in a similar manner to DevOps and MLOps [26]. Just as how ML experts may have problems communicating with software developers and vice versa due to their different areas of expertise [21], security experts and software developers face communication issues as well (e.g., developers may feel attacked when security experts point out security flaws in their code) [26]. In fact, interdisciplinary collaboration between team members with differing academic and professional backgrounds is seen as a challenge in teamwork overall [27].

PEC4 highlights another challenge in adopting continuous practices: lack of experience with testing among ML experts. Only one respondent discussed utilizing an MLOps pipeline for automated testing. The other respondents felt that tooling was still lacking in the area, or were simply not concerned with automated testing. One potential issue here could be the specific know-how required in ML development: the people otherwise in charge of testing in the project may not have the required skillset to carry out the testing on the ML components. This is consistent with existing literature where various challenges associated with testing ML systems are discussed [4]. Serban and van der Blom also report that testing ML artefacts overall (even outside doing so in a continuous fashion) remains a challenge in software organizations based on the low adoption rate of ML testing best practices [19].

PEC5 further points towards communication issues on a project-level (together with PEC3). Many of our respondents felt that they had very little interaction with the (end-)users, or even the customer organization. This further points towards ML experts working in a silo. ML experts seem to often be in a silo not just in relation to other project members within the same organization, but also in relation to the customers and users. Existing research considers identifying the relevant business metrics of the customer

a challenge in ML development, as the customer company itself may not understand what data or metrics could be important [4, 20]. Not having ML experts interact with the customer would seem to be a bad practice, unless there is someone acting as a bridge between the ML experts and the customer who is capable of bridging this communication gap. Multiple existing studies have highlighted challenges associated with requirements engineering in ML development, and many of these challenges are related to communication (e.g., lack of knowledge of ML on the part of the customer(s), dealing with quantitative measurements for requirements, etc.) [4].

PEC6 provides some additional insights into customer involvement in ML project contexts. While customer and user involvement more generally is a widely studied topic in SE, and is at the core of agile SE as well, ML development adds complexity to SE projects in this regard, too. Maintenance of ML systems results in novel challenges in SE [4] as, for example, an otherwise technically functioning system may still degrade in (ML) performance over time due to concept drift [28]. How this is handled in projects where the system is ultimately handed over to a customer is a practical challenge that the customer organizations need to be aware of.

Finally, PEC7 highlights a conflict between practical project matters and organizational/personal interest. The respondents felt that continuous innovation was important, especially on a personal level, in a field as topical and rapidly evolving as ML development. Yet, when ML systems or ML components were developed for an external customer, innovating during projects was not considered beneficial. New innovations would only serve to increase the scope of on-going projects and doing so was seen as counter-intuitive, or simply difficult, because it would necessitate having the customer okay the changes first.

6.1. Answers to research questions

Finally, to directly answer the research question we posed at the start of this paper (i.e., *what are the challenges associated with the continuous development of artificial intelligence?*), we summarize our results as follows. Our findings suggest that the adoption of continuous software engineering in the development of ML has many challenges caused by the addition of ML components into the SE process. ML development is carried out in a more rigid fashion than agile SE in the context of conventional software, and ML experts mainly work independently, i.e., in a silo. This results in difficulties adopting continuous* practices that require collaboration across teams. Communication issues caused by a lack of shared knowledge, lack of guiding frameworks, and issues related to the roles and responsibilities of ML experts meant that the project life cycle did not resemble a continuous cycle but a step-by-step heavyweight development model. Furthermore, the ML experts rarely interacted with the customer or the product users, as they felt that their work role did not include such actions.

Perhaps partially as a result of the types of projects the respondents were involved in, the respondents also discussed challenges adopting continuous* practices related to operations. Many of the respondents worked in projects commissioned by external customers where the customer was largely responsible for the ML system once it had been deployed (or once the ML component had been finished and delivered). Thus, the ML experts had little control over the operational life of the system or components past an initial grace period where it was jointly monitored after release to ensure it worked as intended. Such practical, project-specific challenges require deliberation from the project participants, as

the maintenance (or continuous development) of such systems is important in ML where, for example, concept drift can degrade the performance of a system over time.

While ML is currently coveted by organizations everywhere, few organizations possess ML development competencies themselves. This situation makes externally commissioned ML projects common. Such projects can pose challenges when it comes to utilizing continuous* practices if the monitoring and operations are left to the customer organization. This can lead to continuous* practices receiving less emphasis in organizations working on such ML projects.

6.2. Limitations

We utilized a qualitative thematic interview approach in this study. This study design poses limitations to the results of the study. First, the respondents held different roles in their organizations and these roles influenced their answers. For example, one of the respondents worked in a management role and as such could provide more insights into the business aspects of the project, but was less knowledgeable about the technical aspects, and vice versa in the case of some other respondents. Moreover, as we interviewed only one respondent per organization, we arguably gained a limited understanding of each organization through the lens of a single respondent. We selected this approach due to the novelty of the topic, as we wanted to explore a larger number of organizations to understand how (continuous) ML development is handled in different organizations.

Secondly, in spite of this, the number of the respondents (and organizations, where $n = 8$) is a potential limitation when it comes to generalizing the results of the study. To this end, it is a limitation as well that three of these organizations were research projects with industry collaboration, as opposed to purely industrial contexts. However, given the novelty of the topic, we argue that this is a sufficient number for an exploratory study into a novel topic, e.g., Eisenhardt [29] recommends case study research particularly for novel research topics. Thirdly, we highlight our utilization of convenience sampling as a further limitation for this study, both on the level of organizations and respondents.

Finally, due to the novelty of the topic, we also utilized a more general research approach. We did not focus, for example, only on certain technologies or project contexts. While this approach let us gather more diverse data, this also presents some further limitations for the study. This study ultimately provides a look at the current state of practice more generally, i.e., we studied how different types of organizations develop ML systems. This makes our findings less specific as well. Another approach to the topic could have been to study organizations that specifically (claim to) utilize MLOps, for example, and to discuss what they felt had been the largest challenges in adopting it in the past, or challenges that they still continued to face. In such a fashion, the scope of the study could have been very different. As it is, we have studied what aspects of continuous SE are used in different projects, and which ones are omitted, across a more diverse set of organizations. We chose this approach due to the fact that we specifically wanted to explore less commonly discussed continuous* practices as opposed to the ones present in DevOps and MLOps.

7. Conclusions and future research suggestions

In this paper, we have explored the utilization of continuous* practices (as conceptualized by Fitzgerald and Stol [7]) in the context of ML development. Though continuous SE is

currently state of the art in SE, it has been less studied in the context of ML development. In particular, continuous* practices outside those related to DevOps, and in this case MLOps, have received little attention in ML development.

Through qualitative, thematic interviews with 8 respondents from different organizations involved in ML development, we sought to understand current challenges organizations face in continuous SE in the ML development context. Our findings are summarized in Table 3 at the start of the preceding section. In brief, the largest issues across the board were related to a lack of collaboration and communication between the ML experts and the rest of the project team(s) and stakeholders. With ML experts largely working in a silo, utilizing continuous* practices in the development of the ML components is challenging.

This paper presents a starting point for studying continuous SE in the context of ML development, outside the specific context of MLOps. Much like how DevOps, which it is based on, MLOps only comprises some continuous SE practices. Many of the continuous SE practices discussed by Fitzgerald and Stol [7] are out of the scope of these processes. We hope to encourage further studies into these practices in the context of ML development, as well as in SE overall. Future studies should adopt more specific points of views to the topic in delving deeper into the phenomenon (e.g., by focusing on specific, but nonetheless less commonly studied continuous* practices). The main contribution of this paper is to provide an initial look at the current state of practice through challenges associated with the adoption of these practices.

Acknowledgments

This work was partly funded by local authorities ("Business Finland") under grant agreement ITEA-2020-20219-IML4E of the ITEA4 programme.

References

[1] C. Parnin, E. Helms, C. Atlee, H. Boughton, M. Ghattas et al., "The top 10 adages in continuous deployment," *IEEE Software*, Vol. 34, No. 3, 2017, pp. 86–95.

[2] L. Leite, C. Rocha, F. Kon, D. Milojicic, and P. Meirelles, "A survey of DevOps concepts and challenges," *ACM Computing Surveys (CSUR)*, Vol. 52, No. 6, 2019, pp. 1–35.

[3] S. Moreschini, F. Lomio, D. Hästbacka, and D. Taibi, "MLOps for evolvable AI intensive software systems," in *IEEE International Conference on Software Analysis, Evolution and Reengineering (SANER)*, 2022, pp. 1293–1294.

[4] G. Giray, "A software engineering perspective on engineering machine learning systems: State of the art and challenges," *Journal of Systems and Software*, Vol. 180, 2021, p. 111031. [Online]. https://www.sciencedirect.com/science/article/pii/S016412122100128X

[5] S. Martínez-Fernández, J. Bogner, X. Franch, M. Oriol, J. Siebert et al., "Software engineering for AI-Based systems: A survey," *ACM Transactions on Software Engineering and Methodology*, Vol. 31, No. 2, 2022.

[6] M.M. John, H.H. Olsson, and J. Bosch, "Towards MLOps: A framework and maturity model," in *2021 47th Euromicro Conference on Software Engineering and Advanced Applications (SEAA)*, 2021, pp. 1–8.

[7] B. Fitzgerald and K.J. Stol, "Continuous software engineering: A roadmap and agenda," *Journal of Systems and Software*, Vol. 123, 2017, pp. 176–189. [Online]. https://www.sciencedirect.com/science/article/pii/S0164121215001430

[8] I. Karamitsos, S. Albarhami, and C. Apostolopoulos, "Applying DevOps practices of continuous automation for machine learning," *Information*, Vol. 11, No. 7, 2020. [Online]. https://www.mdpi.com/2078-2489/11/7/363

[9] R.V. O'Connor, P. Elger, and P.M. Clarke, "Continuous software engineering – a microservices architecture perspective," *Journal of Software: Evolution and Process*, Vol. 29, No. 11, 2017, p. e1866. [Online]. https://onlinelibrary.wiley.com/doi/abs/10.1002/smr.1866

[10] S. Mäkinen, H. Skogström, E. Laaksonen, and T. Mikkonen, "Who needs MLOps: What data scientists seek to accomplish and how can MLOps help?" in *2021 IEEE/ACM 1st Workshop on AI Engineering – Software Engineering for AI (WAIN)*, 2021, pp. 109–112.

[11] D. Zhang and J.J. Tsai, "Machine learning and software engineering," *Software Quality Journal*, Vol. 11, 2003, pp. 87–119.

[12] T. Mikkonen, J.K. Nurminen, M. Raatikainen, I. Fronza, N. Mäkitalo et al., "Is machine learning software just software: A maintainability view," in *Software Quality: Future Perspectives on Software Engineering Quality*, D. Winkler, S. Biffl, D. Mendez, M. Wimmer, and J. Bergsmann, Eds. Cham: Springer International Publishing, 2021, pp. 94–105.

[13] K. Srinivasan and D. Fisher, "Machine learning approaches to estimating software development effort," *IEEE Transactions on Software Engineering*, Vol. 21, No. 2, 1995, pp. 126–137.

[14] A.B. Kolltveit and J. Li, "Operationalizing machine learning models: A systematic literature review," in *Proceedings of the 1st Workshop on Software Engineering for Responsible AI*, SE4RAI '22. New York, NY, USA: Association for Computing Machinery, 2023, p. 1–8.

[15] L.E. Lwakatare, I. Crnkovic, and J. Bosch, "DevOps for AI – Challenges in development of AI-enabled applications," in *2020 International Conference on Software, Telecommunications and Computer Networks (SoftCOM)*, 2020, pp. 1–6.

[16] G. Symeonidis, E. Nerantzis, A. Kazakis, and G.A. Papakostas, "MLOps – Definitions, tools and challenges," in *IEEE 12th Annual Computing and Communication Workshop and Conference (CCWC)*, 2022, pp. 0453–0460.

[17] T. Granlund, V. Stirbu, and T. Mikkonen, "Towards regulatory-compliant MLOps: Oravizio's journey from a machine learning experiment to a deployed certified medical product," *SN Computer Science*, Vol. 2, 2021.

[18] A. Lima, L. Monteiro, and A.P. Furtado, "MLOps: Practices, maturity models, roles, tools, and challenges – A systematic literature review," in *Proceedings of the 24th International Conference on Enterprise Information Systems*, 2022, pp. 308–320.

[19] A. Serban, K. van der Blom, H. Hoos, and J. Visser, "Adoption and effects of software engineering best practices in machine learning," in *Proceedings of the 14th ACM/IEEE International Symposium on Empirical Software Engineering and Measurement (ESEM)*, 2020, pp. 1–12.

[20] E. de Souza Nascimento, I. Ahmed, E. Oliveira, M.P. Palheta, I. Steinmacher et al., "Understanding development process of machine learning systems: Challenges and solutions," in *ACM/IEEE International Symposium on Empirical Software Engineering and Measurement (ESEM)*. IEEE, 2019, pp. 1–6.

[21] N. Nahar, S. Zhou, G. Lewis, and C. Kästner, "Collaboration challenges in building ml-enabled systems: Communication, documentation, engineering, and process," in *Proceedings of the 44th International Conference on Software Engineering*, 2022, pp. 413–425.

[22] D. Piorkowski, S. Park, A.Y. Wang, D. Wang, M. Muller et al., "How AI developers overcome communication challenges in a multidisciplinary team: A case study," *Proceedings of the ACM on Human-Computer Interaction*, Vol. 5, No. CSCW1, 2021.

[23] M. Kim, T. Zimmermann, R. DeLine, and A. Begel, "Data scientists in software teams: State of the art and challenges," *IEEE Transactions on Software Engineering*, Vol. 44, No. 11, 2017, pp. 1024–1038.

[24] L. Riungu-Kalliosaari, S. Mäkinen, L.E. Lwakatare, J. Tiihonen, and T. Männistö, "DevOps adoption benefits and challenges in practice: A case study," in *Product-Focused Software Process Improvement: 17th International Conference, PROFES*. Trondheim, Norway: Springer, 2016, pp. 590–597.

[25] M.S. Khan, A.W. Khan, F. Khan, M.A. Khan, and T.K. Whangbo, "Critical challenges to adopt devops culture in software organizations: A systematic review," *IEEE Access*, Vol. 10, 2022, pp. 14 339–14 349.

[26] N. Tomas, J. Li, and H. Huang, "An empirical study on culture, automation, measurement, and sharing of devsecops," in *International Conference on Cyber Security and Protection of Digital Services (Cyber Security)*. IEEE, 2019, pp. 1–8.
[27] S. Brandstädter and K. Sonntag, "Interdisciplinary collaboration: How to foster the dialogue across disciplinary borders?" in *Advances in Ergonomic Design of Systems, Products and Processes: Proceedings of the Annual Meeting of GfA 2015*. Springer, 2016, pp. 395–409.
[28] L. Baier, F. Jöhren, and S. Seebacher, "Challenges in the deployment and operation of machine learning in practice," in *ECIS*, Vol. 1, 2019. [Online]. https://aisel.aisnet.org/ecis2019_rp/163/
[29] K.M. Eisenhardt, "Building theories from case study research," *The Academy of Management Review*, Vol. 14, No. 4, 1989, pp. 532–550.

Appendix A. Interview instrument

Theme 1: Current job and challenges
1. What is your current work role and what does it include?
2. How is the work divided in your project group? How much do you collaborate with others?
3. What kind of tools did you use when developing AI?
4. Can you name any framework, model, or mindset that you used to guide the development process?

Theme 2: Business strategy
1. Can you work independently in the project, or does your work require collaboration with other project participants?
2. How are the requirements of the project decided? You can use a previous or current project as an example.
3. How are the resources planned at the beginning of the project?

Theme 3: Development
1. When is a new functionality or part of the code applied to the larger project context at hand?
2. Can you describe the testing process in some of your projects?
3. How do you decide that the system is ready to be released?
4. Do you know how your work affects the overall quality of the project? For example, the quality of a software project.

Theme 4: Operations
1. Do you interact with the product users after the release?
2. After the product release, do you know if the user expectations are fulfilled?
3. Is the product monitored after the release? If it is, how?

Theme 5: Improvement and innovation
1. Is the product quality improved after the release by you?
2. Are new innovations added to the product if new opportunities arise?
3. When does your involvement with the project end?