Developing Machine Learning Systems: Situational Factors Shaping the Development Process

Selin Aydin[1,*], Horst Lichter[1]

[1]Research Group Software Construction, RWTH Aachen University, Ahornstraße 55, 52072 Aachen

Abstract

Because Machine Learning (ML) system development has some unique characteristics that distinguish it from traditional software development, it requires specialized processes and tools. In order to be able to understand the specific elements of processes for developing ML systems, we conducted expert interviews.

Based on the analysis of the interviews we were able to identify key artifacts, activities, sub-processes, and decision points. Furthermore, we were able to identify some important SF that influence the concrete development process of an ML system. These and four process variants derived from them are presented and discussed.

These results provide valuable insights into the challenges and considerations associated with ML system development and can serve as a basis for developing new development process frameworks for ML systems.

Keywords

Machine Learning, Data Science, Development Process, Expert Interviews

1. Introduction & Related Work

Developing and maintaining a Machine Learning system (ML system for short) is difficult and expensive. ML development projects (ML projects for short) have several unique characteristics that distinguish them from traditional software development, which results in a need for specialized development processes.

A widely used process framework for structuring ML projects is *CRISP-DM*, which was originally designed for data mining projects [1]. It focuses on the six phases of a data mining project: Business Understanding, Data Understanding, Data Preparation, Modeling, Evaluation, and Deployment. Another well-known process framework is the *Team Data Science Process* (TDSP), developed by Microsoft to guide data science projects from start to finish [2]. Similar to CRISP-DM, the TDSP is divided into five main stages: Business Understanding, Data Acquisition and Understanding, Modeling, Deployment, and Acceptance. It not only addresses each stage, but also provides recommendations for project structure, analytics, and storage infrastructure.

Nascimento et al. [3] performed a case study to identify challenges in the development process of ML systems. The authors identified that inexperienced developers work in a rather

2nd International Workshop on Intelligent Software Engineering (ISE 2023), December 4, 2023, Seoul, South Korea
*Corresponding author.
aydin@swc.rwth-aachen.de (S. Aydin); lichter@swc.rwth-aachen.de (H. Lichter)
0009-0006-1764-8091 (S. Aydin); 0000-0002-3440-1238 (H. Lichter)

undefined process, lacking guidance in the tasks they have to perform. The ad-hoc character of the performed development process also became apparent in a meta-summary study published by Nahar et al. [4] covering experiences from more than 4000 practitioners.

Current process frameworks are rather generic and may not apply to all kinds of ML projects. A case study by Nahar et al [5] points to the need for at least three process variants, as the development process of ML systems can be classified as "model-first", "product-first", or "parallel development" depending on which artifact is the focus at the beginning of the project.

In general, each ML project takes place in a context that is determined by, e.g., the domain, the selected technologies, or the available resources. That is, why a single development process cannot be applied to all kinds of projects [6]. Thus, a given process framework has to be tailored and adapted to the given project and its context [7, 8, 9, 10, 11]. The lack of research on what development processes and practices work best in different development project contexts is also mentioned in [4]. The context of a development project can be characterized by project-specific characteristics and constraints. Clarke et al. introduced the notion of *situational factors* in [12] to name those factors. The authors further propose a reference framework for development processes defining 44 situational factors, each of which is divided into 8 classes. Example situational factors are *team size*, *changeability of requirements*, *quality of the application*, or *time to market*.

To our knowledge, no research has been published describing specific situational factors for the ML development process. Therefore, our research aims to answer this question and investigate how these situational factors influence the execution of the process. To this end, we conducted expert interviews to identify such situational factors and their impact on the process.

The paper is structured as follows: In Section 2 we will describe the empirical setup of the conducted expert interviews. Following this, in Section 3 we will explain the elements of the ML development process, which we identified based on the the evaluation of the interviews. Next, in Section 4 we will analyze the results of the interviews, i.e., extracted situational factors and their impact on the process. Finally, we will summarize the results and set out the work planned for the future.

2. Empirical Setup of Expert Interviews

Expert interviews is a qualitative research method mainly used to gather in-depth and specialized knowledge from individuals who possess extensive expertise in a particular subject, or area of interest [13]. Because experts are frequently unaware of the significance of their actions, it is not possible to directly access expert knowledge; instead, it must be reconstructed from the experts' statements.

We conducted expert interviews with experienced developers in the software industry to gain a more detailed understanding of the development process of ML systems and to analyze whether there are situational factors specific to ML system development that lead to different processes.

2.1. Participants

We selected nine participants from three German companies and one Swiss company based on their LinkedIn profiles and recommendations. All participants have at least two years of experience in developing ML systems. Eight of them work with Python exclusively, one participant uses both Python and R. Three of the participants hold high-level positions such as Head of ML or Data Science, and one is a team leader. All interviewees agreed to participate voluntarily.

2.2. Methodology

We conducted individual interviews with each participant via Zoom. The duration of the interviews varied, lasting between 30 minutes and one hour.

At the beginning of the interview, the participant was informed about the objectives and how the data would be collected and used. The interview was then conducted as a semi-structured interview, with two prepared questions but also allowing for deviations when interesting issues were raised by the participants.

Q1: Could you describe your process for developing an ML system?
Q2: Do you continue to perform experiments after deploying the ML system? If so, how do you go about doing this?

The first question aims to identify the activities and sub-processes involved in the development of an ML system. The second question aims at the potential round-trip character of a process.

2.3. Data Analysis

The audio of each interview was recorded and transcribed afterward. To understand and analyze the processes described, we created a visual representation by extracting activities from each interview and modeling them as a UML activity diagram. We then iteratively reviewed the diagrams and revised or split them as we continued with more interviews.

2.4. Traceability of Results

The questions and transcripts of each interview as well as an overview of all questions and answers are provided on Zenodo [1] as an Excel table. This table contains all answers either by participant (rows) or by question (columns).

2.5. Threads to Validity

As usual, we consider the internal and the external validity.

[1] URL: https://zenodo.org/records/10020777?token=eyJhbGciOiJIUzUxMiJ9.eyJpZCI6IjM5N2NiYjFmLWZiYzUtNGMxYS1hZjBmLTc0MjAwNjc
6KkWqe6Jgf3fZwTTnJlR4katX0yKxNvNq7XgPie61sFjFkm6Cly2PHwKS7Zg07iIGubVDfBdTXnLHwT6GLNCPg

- **Internal Validity**: Threats to internal validity refer to factors that may affect the reliability of the results obtained in a study. Each participant received the same introduction and questions from the same interviewer. Since the interviews were designed as semi-structured, the participants were sometimes asked to clarify or elaborate on a certain aspect. In addition, the participants could of course ask questions if they did not understand a question.
- **External Validity:** The external validity determines how widely the results can be generalized. One potential threat to the external validity is the small sample size. While we made efforts to recruit a diverse group of volunteers, the pool of potential participants was limited due to the time and openness required for participation. All participants are from a similar geographical region and are partially colleagues in the same company. However, three of these companies are IT service providers, whereby each participant worked or even led teams on different ML projects for various customers. A larger sample size would have increased the generalizability of our findings.

3. Identified Process Elements

Based on the evaluations of the expert interviews conducted, we name and describe in this section all artifacts that are created and used in the development of an ML system. Furthermore, we describe all identified activities and sub-processes and also the central decision points mentioned in the interviews.

3.1. Artifacts

The following major artifacts are created and used in ML projects.

- *ML System:* An ML system is a software system based on an ML model. When an ML system operates on live data providing decisions or predictions, it is called a *productive ML system.*
- *Experimental Code:* Given an ML problem, developers explore different data processing techniques and ML algorithms to find a suitable solution. Since the focus is on finding a solution, the developed usually code does not follow best practices or clean code guidelines. Usually, the experimental code is available in the form of a Jupyter Notebook, providing explanations for the selected solution along the rather monolithic code.
- *Initial Model:* An initial (ML) model is a rather naive model that is produced in experimentation and does not yet deliver sufficient model performance metrics or leaves room for improvisation.
- *Production-ready Code:* As the name suggests, production-ready code can be run in the productive ML system. Thus, it has to meet specific quality standards, e.g., being reliable, maintainable, and reusable. The production-ready code follows best practices and coding guidelines, is tested, and is well-documented.
- *Model Package:* A model package is a collection of files and resources used to construct an ML model and make it available for use in an ML system. The package may also

include documentation or instructions on how to use the model. Model packages may be provided in a variety of formats, such as Python packages.

- *Placeholder Model Package:* A placeholder model package does not serve as a basis for further model package development and contains a dummy model and scripts. An example of such a script is an ML model prediction script which just returns random values as predictions.
- *Model Serving Infrastructure (MSI):* An MSI provides the tools needed to develop and deploy an ML model to operate on live data. A central component of an MSI is the data infrastructure, e.g. using an ETL-pipeline, which extracts data from a source, transforms it, and loads the results to a destination. The MSI further includes the possibility to integrate and switch out ML models, e.g., by providing different REST endpoints for training or prediction services. Furthermore, the MSI may provide means to monitor the performance of the ML model and to build/deploy the model automatically by an ML pipeline if necessary.
- *Model Performance Metrics:* To evaluate how well an ML model is performing on data, different model performance metrics are continuously computed, such as accuracy, precision, or recall.

3.2. Sub-processes and Activities

The artifacts are created through many different activities during an ML project. We have grouped related activities into five sub-processes, that serve as building blocks to design a concrete ML development process.

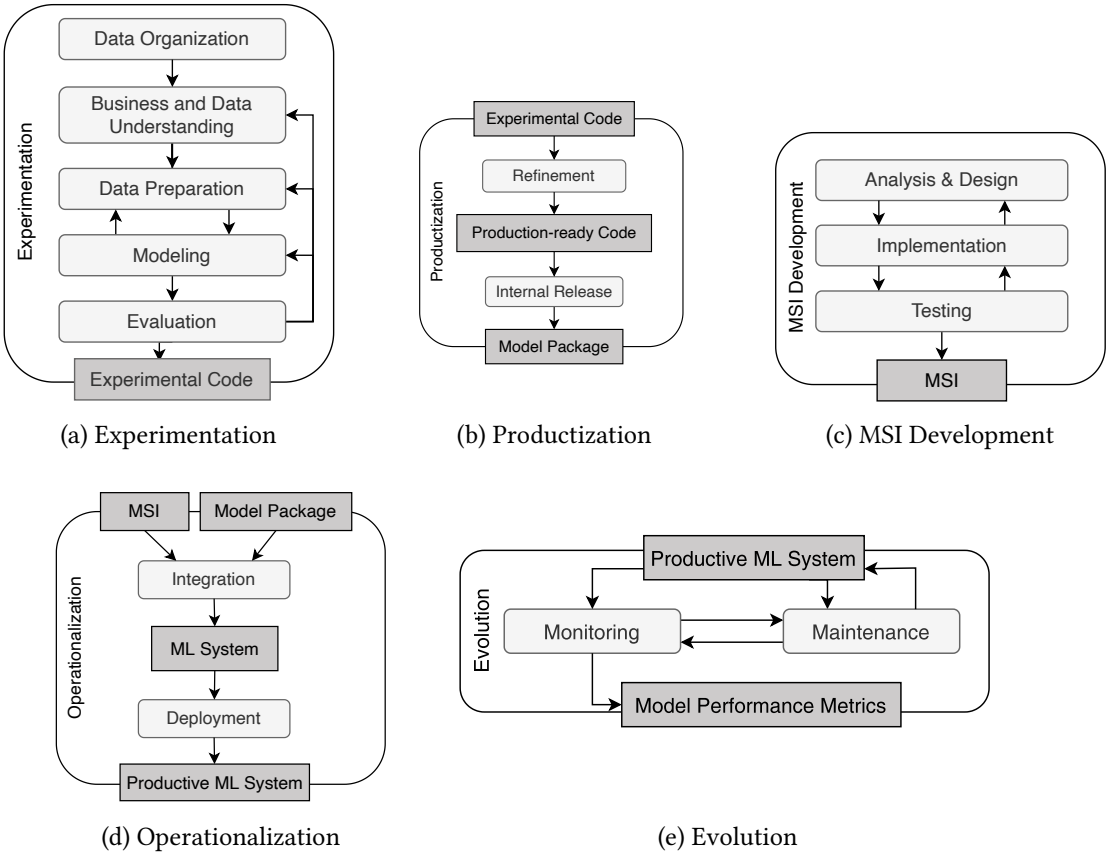
Experimentation

Our analysis clearly shows that the experimental code solving the ML problem is always developed by a similar set of activities, adhering to the CRISP-DM model. Thus, experimentation (see Fig. 1a) is the central sub-process of ML projects consisting of the activities Business and Data Understanding, Data Preparation, Modeling, and Evaluation.

Productization

To make an ML model usable, it has to be integrated into an MSI. However, the experimental code is not ready for integration, as it is usually developed as a Jupyter Notebook. It requires additional effort to make it suitable for integration into the MSI. This sub-process, called productization and depicted in Figure 1b contains the following two major activities:

- *Refinement:* To turn experimental code into production-ready code, the code must be cleaned and refactored into modular and reusable pieces. This is a prerequisite for making the code more maintainable and scalable. Additionally, the code must be prepared for its target environment. For example, if parallel processing is required, certain functions may need to be rewritten using a library that supports it. To ensure the code is of high quality, it is also important to add automated tests.



(a) Experimentation (b) Productization (c) MSI Development

(d) Operationalization (e) Evolution

Figure 1: Overview of sub-processes as process building blocks (UML activity diagrams)

- *Internal Release:* To facilitate the use of the ML model in various contexts, the production-ready code as well as explanations and usage instructions need to be provided in a way that the code is easily integrated into applications or pipelines. This can be done through scripts, packages, or containers. Furthermore, the crafted datasets and models need to be stored in appropriate storage solutions. We call the aggregated output of this activity a model package.

MSI Development

An MSI allows for the efficient and seamless deployment, hosting, and usage of an ML model. As this is an engineering task and requires no exploratory research, conventional software development methods and techniques are applied. This sub-process is depicted in Figure 1c with exemplary engineering activities.

Operationalization

To bring an ML model into operation the following activities must be performed (see Fig. 1d):

- *Integration:* The ML model is integrated into an MSI, e.g. an ML pipeline, web service, or model deployment platform, to create an ML system.
- *Deployment:* Next, the productive ML system is made available to end-users and set into a productive mode. To this end, infrastructure such as servers and cloud systems have to be provisioned.

Evolution

When a productive ML system is operating, it needs to be maintained like any other software (see Fig. 1e).

- *Monitoring:* The performance and behavior of the deployed ML model are monitored to detect ML model degradation early on. This includes tracking model performance metrics such as accuracy, precision, and recall.
- *Maintenance:* If errors are detected, the productive ML system must be corrected. At certain intervals, the model has to be retrained to improve its performance. Retraining can even be automatized and just requires expert supervision.

3.3. Decision Points

We identified three key decision points occurring in all analyzed ML projects that have a significant impact on the process structure:

- *ML Model Revision:* There are several reasons why developers may return to experimentation after the productive ML system is deployed.
  - New requirements may arise, as the business or project evolves
  - Weaknesses are detected, based on monitoring or feedback from stakeholders
  - Additional data sources become available
  - Changes in data properties, e.g. distribution, need to be considered
  - New ML algorithms/methods need to be explored

  For these reasons, developers may choose to make only minor adjustments to the ML model, such as fine-tuning some parameters, or they may choose to completely revise the ML model.
- *Feasibility:* Stakeholders must decide whether the ML problem can be solved feasibly based on the knowledge and code from the experimentation sub-process. In doing so, stakeholders must also consider the available resources and technical capabilities.
- *End of Evolution:* Monitoring and maintenance are activities that need to be performed. As a result, the ML system keeps evolving until it is no longer used.

4. ML Development Process Variants

Analyzing the information gained from the expert interviews, it became apparent that ML projects are organized based on the identified sub-processes but in different process variants which depend on some situational factors.

First, we will introduce the identified situational factors and their impact on the process. Then four example process variants mentioned by the interviewees will be presented.

4.1. Situational Factors

We extracted the following three situational factors with their expressions that influence the concrete performed ML development process (see Table 1).

- *Need for Proof of Concept (POC):* In some cases, it can be uncertain whether a given problem can be effectively solved using ML methods at the start of a project. To address this, a POC may be developed to confirm the feasibility of a particular ML solution or gather feedback from domain experts or customers. The POC can serve as the basis for determining whether to proceed with the project and which approach to take.
  On the other hand, when developers are faced with a familiar problem or have access to solutions for similar problems, they may be able to adapt an existing solution to the given problem without the need for a POC. In these cases, the developers can use their experience and knowledge to estimate feasibility, with varying levels of confidence depending on the specifics of the problem.
- *Product Development:* When starting an ML project, it is not always clear how the problem at hand can best be solved using ML methods especially if a potential solution provides business value. This leads to a decision point where all stakeholders of the project should continue.
- *Confidence in Feasibility:* If the confidence is high that a potential solution is technically realizable and will bring business value, it saves time to start product-related activities, such as developing the MSI, early on. Thus, the MSI can be developed in parallel to experimentation, or even as the first step.
- *Team Structure:* In the reported projects, developers were members of interdisciplinary teams of varying sizes, consisting of scientific and engineering personnel. In larger companies, teams may be divided. The scientific team develops experimental code which is then handed over to the engineering team for productization.

Table 1
Situational factors and expressions

Situational factor	Expressions		
Need for Proof of Concept	yes	no	
Product Development	yes	undecided	
Confidence in Feasibility	high	low	
Team Structure	small	large	divided

4.2. Implications on the Process

The primary situational factor affecting the process is the need for a POC. The development of the MSI and overall product-related activities only make sense if the feasibility is established.

The determining factor for developing in parallel is the team structure. If the team is small and includes both science and engineering skills, then the team can work on either the MSI or the ML model. In a larger team, both can be developed in parallel.

If it was planned to build and deliver a productive ML system, developers start refining experimental code early and are more likely to try to build the model package incrementally. This leads to frequent switching between the experimentation and productization sub-processes. Besides that, the possibility to continuously switch between these sub-processes depends on the need for a POC and having an interdisciplinary team. If the team is divided each handover creates an overhead, making frequent cycles very time-consuming.

4.3. Process Variants

Based on these situational factors and their expressions, we identified four different process variants which are described below. An overview of the variants regarding the combinations of situational factors is depicted in Table 2.

Table 2
Observed Process Variants

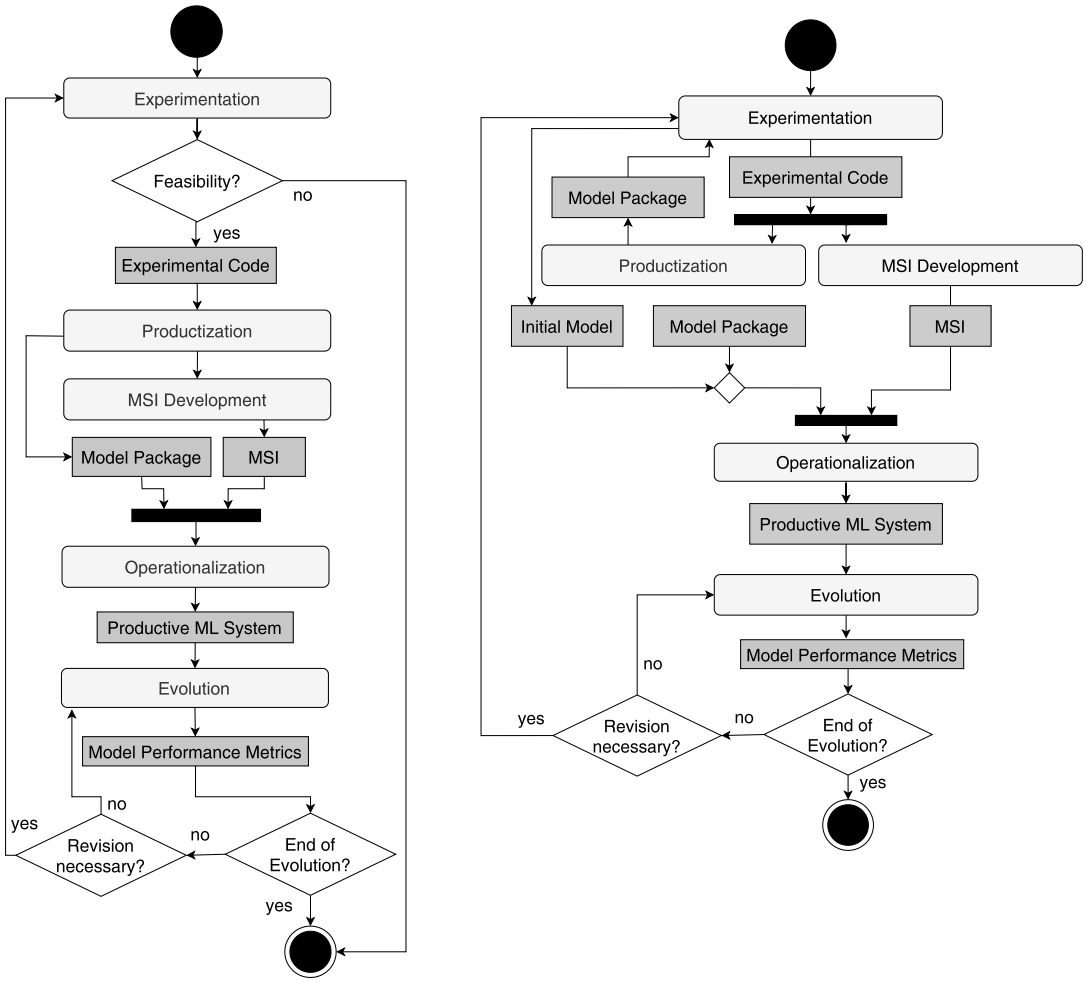
Need for Proof of Concept	yes	yes	no	no
Product Development	undecided	yes	yes	yes
Confidence in Feasibility	low	low	low	high
Team Structure	divided	large	large	small
Variant	**A**	**B**	**C**	**D**

Variant A

- *Situational Factors:* POC: yes | Product: undecided | Confidence: low | Team: divided
- *Description:* The data science team began experimenting with an ML solution in Jupyter Notebook. If no sufficient solution is found, the project is closed. Otherwise, the experimental code is transferred to scripts and refined in the productization sub-process. As a productive ML system can now be developed, the engineering team starts to develop the MSI. Next, the created model package and the MSI are integrated and deployed into production. If weaknesses are identified through monitoring, the model is refined. In some cases, retraining is sufficient. In other cases, experimentation continues in the initial Jupyter Notebook until an improvement is reached, with changes subsequently integrated into the productive ML system.

Variant B

- *Situational Factors:* POC: yes | Product: yes | Confidence: low | Team: large
- *Description:* First, the developers explore different approaches to establish a promising solution. An initial ML model can be shared once a promising approach is identified so that product-related activities can be carried out. With an initial model, the MSI development can begin, and integration with the initial ML model can be carried out.



(a) Process variant A

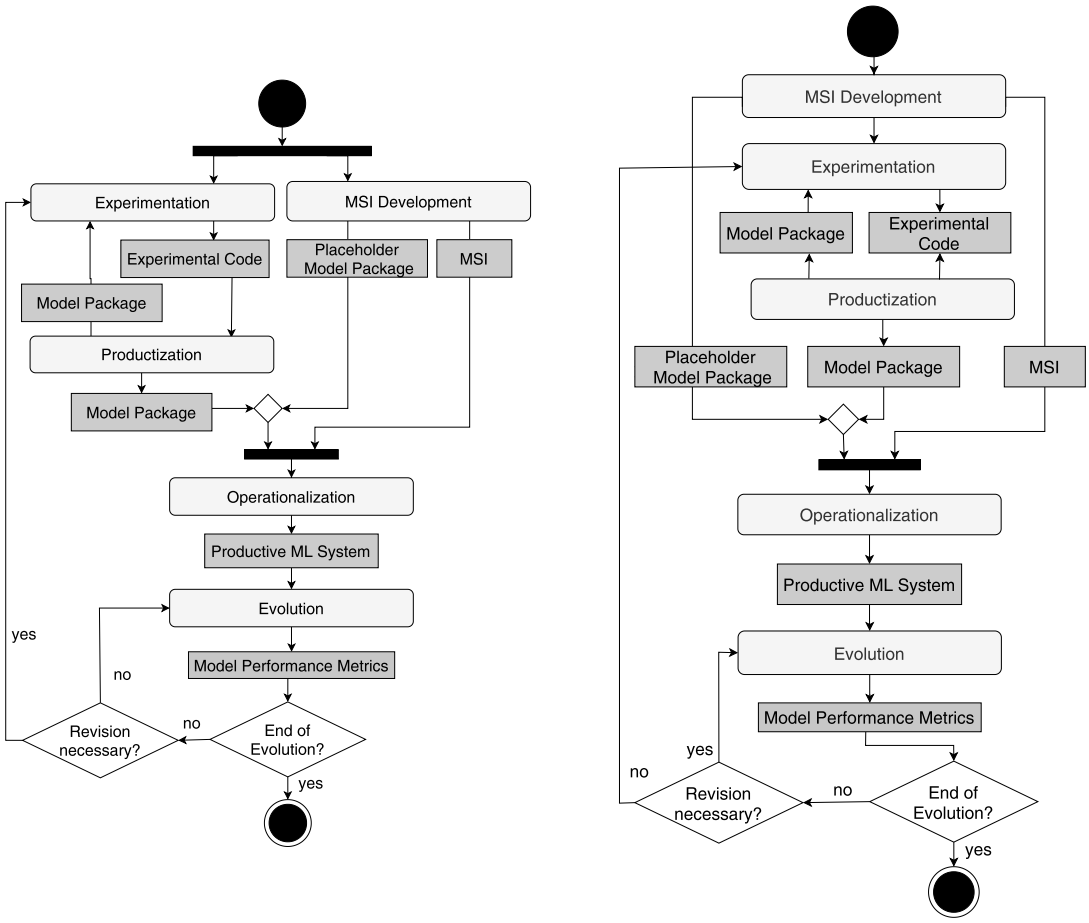
(b) Process variant B

Figure 2: Process variants A and B represented by UML activity diagrams

Meanwhile, developers alternate between experimentation and refinement activity, sharing refinements of newly acquired capabilities immediately. Overall, the model package is developed incrementally, and the initial model can be replaced in the system iteratively as the model is improved.

Variant C

- *Situational Factors:* POC: no | Product: yes | Confidence: low | Team: large
- *Description:* Experimentation and MSI development are executed in parallel from the start. If the MSI is completed before the model package is available, a placeholder package is used temporarily, so that the succeeding activities of operationalization and evolution can



(a) Process variant C

(b) Process variant D

Figure 3: Process variants C and D represented by UML activity diagrams

be performed early. The model package is developed incrementally, with each new version being integrated into the productive ML system. After the integration of a satisfactory model package, the system is monitored and the ML model is revised, if necessary.

Variant D

- *Situational Factors:* POC: no | Product: yes | Confidence: high | Team: small
- *Description:* The MSI and a placeholder model package are built first and are integrated to construct an initial productive ML system. The remaining time is dedicated to enhancing the ML model and optimizing its performance metrics. The ML model in the productive ML system is replaced iteratively as improvements are made. This process variant is especially beneficial when the schedule is heavily constrained.

4.4. Discussion

The extracted situational factors lead to different process variants. The placement of the sub-process *MSI Development* depends heavily on the necessity of a POC. Furthermore, the team structure determines the level of parallelization and iterations.

According to the interviews, the applied process depends on the personal experience of ML project managers, developers, or shared experience within the company. Often, the described situational factors are considered, but the process is essentially performed intuitively by experienced project managers.

This in no way corresponds to a systematic engineering approach. An inexperienced project manager lacks the experience to shape the process. Therefore, further research is needed to define guidelines and frameworks for ML development processes that clearly describe activities, roles, artifacts, and phases based on best practices. Such guidelines or frameworks can then be used to execute ML projects and reduce the risk of project failure.

5. Conclusion & Future Plans

In this paper, we investigate if there are situational factors that affect the software development process of ML systems in a way that leads to process variants. To this end, we conducted expert interviews that surveyed developers who have a lot of experience running or even leading ML projects. The analysis of the information collected enables us to identify artifact, activities, sub-processes, decision points, and several situational factors that lead to different process variants.

Project managers with several years of experience choose a process more or less intuitively. To guide ML project managers and teams without that experience, an ML development process framework is needed. Based on situational factors, it should support ML project managers in the design of their development processes and also provide best practices and guidelines. This is the only way to make ML projects plannable and avoid risks that can be traced back to an incorrect or inadequate process.

To investigate this further, we plan to conduct a systematic literature review to extract more situational factors affecting the software development process of ML systems. Based on that, a concept for a reference framework will be constructed.

Acknowledgment

We want to thank all the participants of our industry interviews who voluntarily gave us valuable insights into their day-to-day work.

References

[1] C. Shearer, The CRISP-DM model: the new blueprint for data mining, Journal of data warehousing 5 (2000) 13–22.

[2] Microsoft, What is the Team Data Science Process?, 2022. URL: https://learn.microsoft.com/en-us/azure/architecture/data-science-process/overview, accessed: 2023-10-05.
[3] E. d. S. Nascimento, I. Ahmed, E. Oliveira, M. P. Palheta, I. Steinmacher, T. Conte, Understanding Development Process of Machine Learning Systems: Challenges and Solutions, in: 2019 ACM/IEEE International Symposium on Empirical Software Engineering and Measurement (ESEM), 2019, pp. 1–6. doi:10.1109/ESEM.2019.8870157.
[4] N. Nahar, H. Zhang, G. Lewis, S. Zhou, C. Kastner, A meta-summary of challenges in building products with ml components – collecting experiences from 4758+ practitioners, in: 2023 IEEE/ACM 2nd International Conference on AI Engineering – Software Engineering for AI (CAIN), 2023, pp. 171–183. URL: https://doi.ieeecomputersociety.org/10.1109/CAIN58948.2023.00034. doi:10.1109/CAIN58948.2023.00034.
[5] N. Nahar, S. Zhou, G. Lewis, C. Kästner, Collaboration challenges in building ml-enabled systems: Communication, documentation, engineering, and process, 2022. arXiv:2110.10234.
[6] C. Jones, Development practices for small software applications, CrossTalk, the Journal of Defense Software Engineering 21 (2008) 9–13.
[7] P. H. Feiler, W. S. Humphrey, Software process development and enactment: Concepts and definitions, in: [1993] Proceedings of the Second International Conference on the Software Process-Continuous Software Process Improvement, IEEE, 1993, pp. 28–40.
[8] O. Benediktsson, D. Dalcher, H. Thorbergsson, Comparison of software development life cycles: a multiproject experiment, IEE Proceedings-Software 153 (2006) 87–101.
[9] A. MacCormack, R. Verganti, Managing the sources of uncertainty: Matching process and context in software development, Journal of Product Innovation Management 20 (2003) 217–232.
[10] G. H. Subramanian, G. Klein, J. J. Jiang, C.-L. Chan, Balancing four factors in system development projects, Communications of the ACM 52 (2009) 118–121.
[11] K. Kautz, Software process improvement in very small enterprises: does it pay off?, Software Process: Improvement and Practice 4 (1998) 209–226.
[12] Clarke, Paul and O'connor, Rory V, The situational factors that affect the software development process: Towards a comprehensive reference framework, Information and software technology 54 (2012) 433–447.
[13] S. Döringer, The problem-centred expert interview. combining qualitative interviewing approaches for investigating implicit expert knowledge, International Journal of Social Research Methodology 24 (2021) 265–278. URL: https://doi.org/10.1080/13645579.2020.1766777. doi:10.1080/13645579.2020.1766777.