On the Understandability of MLOps System Architectures

Stephen John Warnett, *Member, IEEE,* and Uwe Zdun, *Member, IEEE*

***Abstract*—Machine Learning Operations (MLOps) is the practice of streamlining and optimising the machine learning (ML) workflow, from development to deployment, using DevOps (software development and IT operations) principles and ML-specific activities. Architectural descriptions of MLOps systems often consist of informal textual descriptions and informal graphical system diagrams that vary considerably in consistency, quality, detail, and content. Such descriptions only sometimes follow standards or schemata and may be hard to understand.**

We aimed to investigate informal textual descriptions and informal graphical MLOps system architecture representations and compare them with semi-formal MLOps system diagrams for those systems. We report on a controlled experiment with sixty-three participants investigating the understandability of MLOps system architecture descriptions based on informal and semi-formal representations.

The results indicate that the understandability (quantified by task correctness) of MLOps system descriptions is significantly greater using supplementary semi-formal MLOps system diagrams, that using semi-formal MLOps system diagrams does not significantly increase task duration (and thus hinder understanding), and that task correctness is only significantly correlated with task duration when semi-formal MLOps system diagrams are provided.

***Index Terms*—controlled experiment, empirical software engineering, empirical study, distributed system modelling, distributed system architecture, understandability, machine learning, MLOps**

I. Introduction

A. Problem Statement

MACHINE Learning Operations (MLOps) is centred around rapid deployment and is similar in many respects to DevOps (software development and IT operations) but also introduces additional machine learning (ML)-specific activities [1]. It is a multidisciplinary engineering practice that intersects ML, software engineering, and data engineering. MLOps involves specific practices and concepts with respect to end-to-end activities such as conceptualisation, implementation, monitoring, deployment and scaling of ML products. It also utilises continuous integration/continuous delivery (CI/CD), workflow orchestration, reproducibility, data, model and code versioning, continuous training, and monitoring, among other practices [2]–[4]. While conducting previous research [3], [4], we noticed that MLOps system descriptions in grey literature are typically composed of informal textual descriptions and informal graphical system diagram representations. Based on our observations, the textual descriptions vary considerably in tone, detail, and content and do not always closely correspond to the accompanying informal MLOps system diagrams. We noted that the diagrams themselves do not usually follow consistent standards or schemata and may exhibit any of the following characteristics:

- Inconsistencies in, e.g. terminology, relationships between elements, and visual features.
- Lack of standardisation of, e.g. terminology, relationships between elements, and visual features.
- Lack of clarity.
- Varying levels of detail or abstraction.
- Omission or inclusion of relevant or irrelevant system aspects, technologies, and other details.

Indeed, in practice, system architecture descriptions are customarily of an informal nature [5]–[7], such as ad-hoc box-and-line or arrow diagrams, and this appears to be the case still, despite the prevalence of modelling languages such as the Unified Modeling Language (UML) [8] or domain-specific languages (DSLs) for specifying architectures [9].

B. Research Objectives

We conducted an empirical study on the understandability of MLOps system descriptions. Our study aimed to determine whether and to what extent the provision of additional, semi-formal MLOps system diagrams improves understanding of MLOps system descriptions. In software architecture, informal system diagrams are graphical representations of a software system or its components that are typically created with a focus on conveying high-level or conceptual information. These diagrams are often intended for quick communication and understanding among team members and stakeholders or initial design discussions. Informal diagrams may lack strict adherence to standardised notations, and they can be hand-drawn, sketched, or created using simplified drawing tools. Their primary purpose is to illustrate the system's structure and major components without going into exhaustive detail. Semi-formal system diagrams, on the other hand, are more rigorous and structured graphical representations of a software system. They follow standardised notations and conventions, and they are often used for detailed design, documentation, or analysis purposes. Semi-formal diagrams provide a comprehensive and unambiguous depiction of the system, including its architecture, components, relationships, and behaviours. These diagrams are typically expected to adhere to specific

The authors are with the Research Group Software Architecture, Faculty of Computer Science, University of Vienna, Währingerstraße 29, 1090 Vienna, Austria.

Stephen John Warnett is also with the UniVie Doctoral School Computer Science DoCS, Faculty of Computer Science, University of Vienna, Währingerstraße 29, 1090 Vienna, Austria.

E-mail: firstname.lastname@univie.ac.at

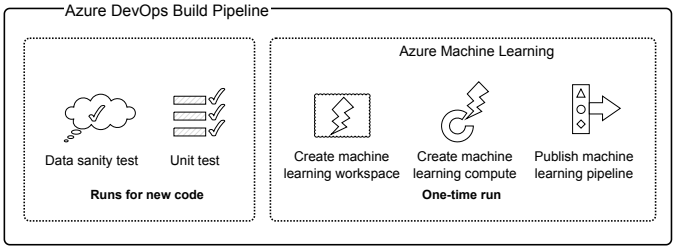

Fig. 1: Redrawn excerpt of a build pipeline from an informal MLOps system diagram.

modelling languages, such as UML [8], or specialised notations tailored to software architecture. They can be used for precise communication analysis and as a basis for further system development and implementation.

We deemed UML a good choice for the visual system representations in this study since it is familiar, widely understood, and has been utilised in the related work described in Section II. UML activity diagrams for the pipeline representations were ruled out because they were not considered precise or detailed enough for the study. Despite the variable nature of the formality of UML diagrams (see Rumpe and France [10], Whittle [11] and France et al. [12]), ours still create an improved commonality of communicating architecture decisions and intent since they are UML representations generated from coded models based on a precisely defined MLOps metamodel that ensures no variability or ambiguity in the models themselves.

Figure 1 depicts a redrawn excerpt of a build pipeline from an example MLOps system[1] and the figure in Appendix B shows the informal diagram for this pipeline excerpt. Please note that, like the UML-based diagram, the informal excerpt does not include any aspects unrelated to the build pipeline. In the original informal diagram, all pipelines and components are depicted in a single diagram, whereas our UML-based diagrams are separated into multiple component and pipeline diagrams. Aspects not present include the Azure Blob Storage Component, the Azure Machine Learning Pipeline endpoint, the Azure Machine Learning retraining pipeline, and the Azure DevOps build pipeline.

As in our previous work [13]–[15], we defined the experiment goal using the Goal Question Metric [16] template as follows: **analyse** MLOps system descriptions **for the purpose of** their evaluation **with respect to** their understandability **from the viewpoint of** both novice and moderately advanced software architects, designers or developers **in the context (environment) of** the Advanced Software Engineering (ASE), Distributed Systems Engineering (DSE) and Software Engineering 2 (SE2) courses offered by the Research Group Software Architecture[2] at the University of Vienna[3].

We conducted an empirical study (to our knowledge, the first of its kind) as a controlled experiment on the understandability of MLOps system description representations, incorporating our novel, semi-formal, UML-based MLOps system diagrams. Our main contribution is the result of the controlled experiment: that the understandability (quantified by task CORRECTNESS) of MLOps system descriptions is significantly greater using our semi-formal MLOps system diagrams, that using semi-formal MLOps system diagrams does not significantly increase task DURATION (and thus hinder understanding), and that task CORRECTNESS is only significantly correlated with task DURATION when semi-formal MLOps system diagrams are provided.

Our research makes significant contributions to the MLOps field by advocating the adoption of semi-formal diagrams as a best practice to enhance task correctness estimation. These findings not only catalyse further scientific investigations into the cognitive aspects of correctness estimation but also offer practical guidance to MLOps practitioners. They underscore the potential advantages of incorporating semi-formal diagrams and underscore the significance of realistic self assessment. Our results highlight the value of visual representations in improving correctness estimation.

Moreover, they have the potential to reshape the way software architecture is taught in university courses. Leveraging semi-formal MLOps system diagrams can empower educators to enhance learning outcomes, address prevalent misconceptions, accommodate diverse learning styles, and facilitate a deeper comprehension of intricate architectural concepts. Furthermore, our study encourages a more comprehensive and interdisciplinary approach to teaching software architecture, aligning it with the evolving needs of the industry.

Overall, our work contributes towards a solid foundation for further empirical work in the domain of architectural modelling of MLOps systems and represents a positive initial step towards determining how helpful semi-formal MLOps system diagrams may be in understanding MLOps system architecture.

We structured the rest of this paper as follows: Section II describes related work and compares it to this study. Section III covers the guidelines and design of our study, describes the participants, the materials and the experiment tasks, our dependent and independent variables, our hypotheses and the research question. Section IV documents the preparation and procedure of the experiment sessions. Section V describes the processing and analysis of the dataset resulting from the experiment sessions. We also discuss the participant demographics and descriptive statistics for the experiment data before testing our hypotheses and addressing our research question based on the experiment results. Section VI discusses our results, including their interpretation and the threats to the validity of our study. We conclude in Section VII by discussing the impact and relevance of our study, as well as suggestions for future work that could build on our study.

II. Background

A. Background and Motivation

Comprehending ML systems is notably more challenging than traditional software architecture due to the intricate inter-

[1]Source for the complete diagram from which we took the excerpt: https://tinyurl.com/mlops-system-ma

[2]https://swa.cs.univie.ac.at

[3]https://www.univie.ac.at

play of software development, data science, and data engineering [17]. Various authors have identified significant challenges in the software structure of ML applications, such as complex software modules and their dependencies [18], technical debt and anti-patterns [17], and design issues in ML models, such as model selection and reuse [19], [20]. These challenges highlight the intricacies of organising and decomposing ML systems, which have led to the introduction of Automated ML [21], [22] and MLOps [2], [23] as a sub-discipline of DevOps [24] and Continuous Delivery [25]. It has been shown in various studies of open source systems [26], scientific literature [27] and interviews with practitioners [28] that DevOps and Continuous Delivery lead to a complex deployment and delivery architecture that is hard to comprehend and needs to be understood in addition to the software architecture of the system to be deployed. This problem worsens for ML systems deployed with MLOps as, in addition to the complex model and system deployments, component architectures, and CI/CD pipeline behaviours, the ML pipeline behaviours and ML-specific components need to be understood [2], [23], [29]. For example, consider a model retraining step in a CI/CD pipeline that invokes a model retraining ML pipeline and must work harmoniously with model versioning and registry services. In a survey of case studies, Paleyes et al. identified significant practical challenges in deploying machine learning systems specific to ML architecture [30]. Some relate to tools, services, and architectures used in the deployment architecture; others relate to the potential disconnect between ML/data science experts and software engineering practices.

We investigate whether the provision of semi-formal, UML-based MLOps system diagrams can enhance the clarity and comprehensibility of these systems, potentially bridging the gap between these two domains and thereby enhancing overall MLOps efficiency. Our research motivation is based on observing inconsistencies in existing MLOps system descriptions and the complete lack of relevant studies, including controlled experiments, focusing on studying the understandability of semi-formal MLOps system models. We are unaware of any other empirical study that systematically investigates the understanding of system architectures based on informal textual descriptions and informal graphical system diagrams compared to semi-formal model diagrams in the context of MLOps. This study unifies these aspects and, to our knowledge, is the first study of its kind. While the work described below is relevant to various degrees, the studies differ in several respects from ours. By conducting this study, we wish to contribute towards building a scientific body of work in a new area of research.

Our study results confirm our suspicions and show that the understandability of informal system descriptions used in practice could be improved. In summary, our motivation to study whether semi-formal models can enhance MLOps system description understandability is that MLOps systems are significantly more complex than ordinary DevOps-based systems and need to be understood by software engineers and relatively untrained ML/data science experts.

B. Experimental Studies

Various studies have utilised controlled experiments, but we could not source any involving MLOps. The studies mentioned below are provided for comparison because they are similar to ours to varying degrees in describing user experiments involving software systems. However, they do not focus on MLOps, do not compare semi-formal and informal MLOps system models and descriptions, or do not focus on quantifying user understanding of system properties and behaviour.

Allodi, Cremonini et al. [31] compare how accurately security professionals and students with further technical education could assess the severity of software vulnerabilities based on various attributes, totalling seventy-three participants. Their focus was not on comparing different system description methods but on participants' background knowledge and education. One major difference in methodology in this study is that participants were divided into three groups - students with a BSc in information security enrolled on an MSc in information security degree course, students enrolled on an MSc in computer science course and security practitioners. Another difference is that they specifically recruited students with no professional expertise.

Allodi, Biagioni et al. [32] also conducted a controlled experiment with twenty-nine MSc students to determine how difficult it is for participants to assess system vulnerabilities when security requirements change. Thus, their comparisons were based on system requirement variations rather than modelling variations. Unlike our study, they opted for a within-subject design, but similarly, they formulated hypotheses, which were then tested using statistical methods.

Labunets et al. [33] report on a controlled experiment with twenty-nine MSc students to compare participants' perceptions of visual versus textual methods for security risk assessment in the context of effectiveness. This experiment is relevant to our study since it compares textual and visual representations. Still, it does not focus on the differences between semi-formal and informal representations, nor does it focus on system understanding. Also, it is concerned with security aspects rather than MLOps. Regarding methodology, the authors chose a within-subject design, unlike our study. They used Grounded Theory during a qualitative analysis phase of their process to explain the possible difference between the two methods of being considered. This is interesting because we also used Grounded Theory in our previous work upon which this study was partially based (see [3], [4] and our research process in Section 2). They translated their research questions into hypotheses that were statistically tested, which is another methodological similarity to our approach.

A more closely related study is described by Sharafi et al. [34], who conducted an experiment with twenty-eight participants and studied the impact of structured textual versus graphical representations on efficiency while completing requirement comprehension tasks. This study has some similarities with ours. It featured graphical system representations and measured CORRECTNESS and task DURATION. Still, it differed because it emphasised visual effort, the impact of native language, education and gender. It also utilised eye-tracking

and focused on requirements rather than MLOps systems. A difference in methodology was the use of a within-subject design, but similarities were the definition of hypotheses and the statistical testing of the hypotheses using R.

A similar study to ours involving a controlled experiment is that of Heijstek et al. [35]. They conducted a controlled experiment with forty-seven participants from industry and academia to study whether visual or textual artefacts are more effective at communicating architectural software design decisions to software developers. They used UML representations for the diagrammatic representations and informal textual descriptions. They found that neither diagrams nor textual descriptions proved to be significantly more efficient in terms of communicating software architecture design decisions, that diagrams are not more suited to convey design decisions of a topological nature, and that participants who predominantly used text scored significantly better than their counterparts. They also found that diagrams were not able to alleviate the difficulties non-native English speaker participants had in extracting information from the documentation. Some differences with this study compared to ours are that all participants assessed both representations (i.e. a within-subject approach was followed), the focus was not MLOps, and data was collected by filming participants and encouraging them to think aloud. A similarity to our approach included the definition of hypotheses that were statistically tested and the use of questionnaires.

Another similar study is the controlled experiment involving student participants by Jolak et al. [36]. Similarly, they compared textual and graphical (UML) representations. They focused not just on understanding but also on explainability, recall and communication. Their work was in the context of a mobile application, which contrasts with the focus of our study, which is MLOps. Like our study, they followed a between-subject design to minimise learning and transfer effects, gathered descriptive statistics to understand the data, and statistically tested their hypothesis. Their results spoke in favour of the graphical design decisions as opposed to the textual representations.

C. Studies on System Properties and Behaviour

There is much related work focusing on quantifying system properties and behaviour or providing alternatives or enhancements to UML, which bear similarities to our study, but again lack a focus on MLOps. Some notable examples are described below. These studies focus on quantifying architectural properties rather than user understanding of system descriptions, even if the properties were related to understanding. In contrast, we are interested in comparing semi-formal and informal MLOps system descriptions.

The selected studies described below propose various methods to model system architectures, provide alternatives or enhancements to UML, or derive improved architectures. Although some of the artefacts resulting from these contributions may be generally applicable, we are explicitly focused on MLOps system architecture. Nevertheless, these studies are still related to our work since our study involves the development of a Python-based metamodel for modelling MLOps systems and the subsequent generation of UML-based model visualisations.

Pradella et al. [37] describe a new temporal logic language that combines UML notation with a formal semantics and provides a formal notation, allowing developers to model non-critical system aspects in either UML or using the provided formal notation. Modelling system aspects is relevant to our work, as is the provision of complementary formal or semi-formal notation (since we also used our own existing DevOps metamodels for modelling systems in UML in this study and extended them with MLOps-specific metamodel types); however, the focus of their work was not within the context of understandability and MLOps.

Lavazza et al. [38] describe an approach whereby developers model systems in UML, and the models are then automatically translated into a formal notation which can subsequently be used to verify various properties of systems such as safety, utility, and liveness. They propose enhancements to UML and provide a formal semantics for some UML constructs. Again, the focus on system properties and formal or semi-formal modelling is relevant, but this study did not specifically focus on understandability or MLOps systems.

Rodano and Giammarco [39] use a general logical notation to formalise architectural models of systems and quantify system characteristics and quality attributes (such as performance) to ascertain the quality of architectural models (as opposed to quantifying user understanding). They suggest that their general notation can be combined with many system architecture tools and adapted to suit specific architectural frameworks or projects rather than MLOps systems directly.

Li and Horgan [40] discuss systematic and quantified architectures of software systems and present a method to construct formal models of software architectures and simulate them to predict behaviour, reliability, and performance. They then use the quantified simulation results to evaluate alternative software architectural designs. This is interesting since we are also concerned with system properties and behaviour. However, we would like to determine user understanding of systems with informal text and semi-formal architecture view descriptions rather than generate alternative architectures and predict system qualities.

III. Experiment Planning

A. Research Process

Figure 2 overviews our overall research process. In the pursuit of our research objectives, the research methodology unfolded through a series of carefully orchestrated steps. These steps encompassed diverse aspects, from initial data collection, described in Sections IV and V-A to the rigorous analysis and interpretation of the experiment's outcomes documented in Sections V and VI.

Our journey commenced with us iteratively and systematically scouring online resources for MLOps systems. Through each iteration, we applied meticulous inclusion-exclusion criteria (described in Section III-B until three suitable MLOps systems emerged, forming the foundational pillars of our study.

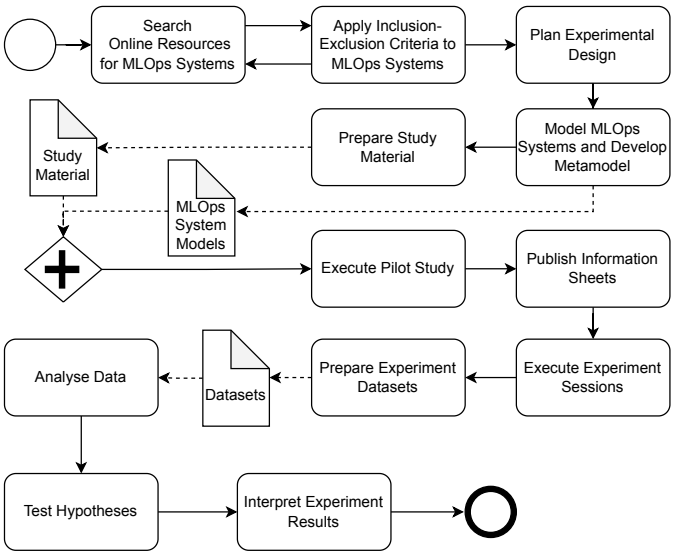

Fig. 2: Overview of the research process followed in this study.

Subsequently, we outlined the guidelines for our research and the planning of the experimental design, which we describe in Sections III-D and III-E, respectively. This crucial stage laid the groundwork for the controlled and structured study execution, ensuring the integrity of the results.

A pivotal phase of our research involved the intricate modelling of the selected MLOps systems, as described in Section III-C. This modelling endeavour was intertwined with developing a comprehensive metamodel for MLOps, creating a set of intricate MLOps system models comprising component and pipeline diagrams, and providing a detailed and holistic representation of MLOps systems.

To provide study participants with the necessary resources for a successful engagement with the experiment, we dedicated much effort to preparing and producing the study material. We meticulously crafted this material, including the information sheets and experiment task sheets, to facilitate a comprehensive understanding of MLOps systems and document it in Section III-G.

A critical milestone in our research journey was the execution of a pilot study, which we describe in Section IV-A. This pilot study was a vital validation step, employing the study material and MLOps system models to ensure their clarity, comprehensibility, and suitability for the main experiment.

To maximise accessibility and ensure participants were well-prepared, we published the experiment information sheets on our online learning platform. This platform served as a central hub for disseminating essential information to participants.

The core of our research was embodied in the execution of the experiment sessions. We carried out the experiment sessions meticulously with precision and adherence to predefined procedures and protocols, as described in Section IV.

After the experimental sessions, the research entered the data compilation and preparation phase. We carefully curated the result dataset in preparation for analysis. We document these activities in Section V-A.

With datasets in hand, we performed comprehensive data analysis, unearthing valuable insights, demographic information, quantitative findings, and meaningful statistical relationships. This analysis, which we detail in Section V, provided the groundwork for our hypothesis testing, enabling the rigorous assessment of observed differences and checking our research hypotheses.

In the final stretch of our research journey, we diligently interpreted the experiment results. This interpretive phase allowed us to synthesise the accumulated knowledge and insights, gaining a comprehensive understanding of the research's overarching outcomes and implications. These concluding insights provided the foundation for our contributions to the field of MLOps and our research's broader significance. The interpretation and implications are discussed in Sections VI and VII-A.

B. System Selection Method

We utilised popular search engines like Google and DuckDuckGo, along with topic portals such as InfoQ and DZone, to source pertinent MLOps systems. The selection of each system adhered to predefined inclusion-exclusion criteria.

Inclusion criteria encompassed:

- commercial products/platforms, open source systems, workflows featuring architectural descriptions, and comprehensive examples or tutorials that facilitated the derivation of system architectures;
- demonstrations or example projects by individuals or companies with high ratings on GitHub;
- the need to originate from reputable authors, companies, or publishers;
- the need for practitioners to be the intended audience.

Conversely, exclusion criteria encompassed:

- student projects, including those from universities, coursework, pet projects, or experimental examples;
- open-source projects categorised as tools or libraries, distinct from MLOps implementations or system descriptions;
- tutorial articles lacking examples or containing only code snippets;
- sources focusing solely on specific detailed aspects of the ML workflow, as opposed to the overall workflow or system;
- sources describing the ML workflow exclusively in a theoretical context (e.g., for informational or educational purposes)
- sources that served as advertisements or did not describe real architectural examples applicable in an industrial context;
- sources detailing the consulting approaches of companies without accompanying example projects.

The first selected MLOps system[4] is a guide based on Amazon AWS intended for data scientists and ML engineers looking to implement DevOps principles in the context of ML

[4]https://tinyurl.com/mlops-system-ap

systems, i.e. MLOps, endorsing the automation and continuous monitoring of every phase in the construction of ML systems, and covering areas such as integration, testing, release management, deployment, and infrastructure administration.

Our second selected MLOps system[5] by Google describes their level 1 architecture, the primary objective of which is to ensure uninterrupted model training through the automation of the ML pipeline, facilitating the seamless deployment of model prediction services. The pipeline includes automated data and model validation procedures to streamline the incorporation of new data for model retraining within a production environment. It encompasses triggers and metadata management to ensure efficient workflow automation.

Our final selected MLOps system[6] was a reference architecture outlining the procedures for establishing a continuous integration (CI), continuous delivery (CD), and retraining pipeline for an AI application through the utilisation of Microsoft Azure DevOps and Azure Machine Learning. A practical demonstration of this architectural framework was accessible via a reference implementation on GitHub.

The selected MLOps systems are representative of real-world MLOps systems. We achieved this through the meticulous application of our inclusion-exclusion criteria, which guided the selection of systems offered by notable vendors, including Amazon (AWS), Google and Microsoft.

C. Modelling Method

To create our models of the selected systems, we harnessed Codeable Models, an effective modelling tool available in Python, allowing us to specify metamodels and model instances. In this study, we initiated the development of an MLOps system architecture metamodel for the specification of components, connectors, and their relationships. Our metamodel serves as the cornerstone for the semi-formal representation of MLOps systems, utilising modelling primitives derived from our understanding of various aspects of the MLOps systems described in Section III-B.

The semi-formal, UML-based MLOps system representations in this study typically encompass component views and pipeline views, each consisting of nodes and connectors of various types. In component views, nodes represent different component types or pipeline types, with connectors delineating their relationships. These component nodes may signify various entities, including execution environments, cloud components, platforms, orchestrators, repositories, clusters, pipelines, etc. Connectors within a component view articulate relationships like artefact providers, data triggers, deployment targets, and pipeline triggers, among others.

Pipeline nodes describe the internal workings of the pipeline and consist of an initial node, a sequence of intermediate pipeline nodes, and a concluding node. The relations between these nodes establish the execution order. To address decisions or branching within pipelines, we utilise fork nodes followed by parallel pipeline nodes culminating in a joint node. This approach represents diverse pipeline aspects and steps, encompassing pipeline triggers, data processing, model training, container image creation, testing, model registration, and deployment. Pipeline nodes also house metadata detailing information like automatic execution invoked pipeline tasks or execution environments.

To visualise our models, we employed PlantUML[7] to generate the corresponding UML diagrams. Examples of a component and pipeline diagram for the Microsoft Azure DevOps and Azure Machine Learning system described in Section III-B may be found in Appendices A and B respectively.

D. Guidelines

When planning our study, we followed the template provided by Jedlitschka et al. [41], which details robust guidelines for empirical research in software engineering. Additionally, we adhered to guidelines according to Kitchenham, Pfleeger et al. [42], who offer overarching guidelines for conducting software engineering experiments and provide recommendations related to the design, implementation, analysis, and presentation of empirical research studies, Wohlin et al. [43], who go into more detail, and discuss statistical tests and their suitability for various study types, and Juristo and Moreno [44], who also offer valuable insights into conducting empirical research in software engineering. We also used the robust statistical method guidelines for empirical software engineering by Kitchenham, Madeyski et al. [45] as a basis for evaluating the data from the experiment sessions.

E. Study Design

Conducting a controlled experiment to evaluate the impact of providing semi-formal MLOps system diagrams has distinct advantages, ensuring causal relationships and facilitating future research replication. This method allows for the collection of quantitative data and the generalisation of findings to broader populations. Other methods, such as observational studies, eye tracking, surveys, comparative studies, qualitative interviews, and usability studies, were considered but ruled out due to limitations in establishing causality and controlling biases. Given the need for reliable, objective, and causal inferences, a controlled experiment was deemed the most appropriate for this study, providing the necessary control over variables and ensuring the validity of the results.

We deliberately chose a between-subject [46] experimental design, which involves distinct groups of participants exposed to different conditions, for several reasons. Firstly, within-subject designs, where the same participants experience all conditions in sequence, are susceptible to order effects like practice improvement and fatigue decline, which can confound results. Using a between-subject design, we ensured that each group encountered only one condition, effectively mitigating these order-related biases.

Secondly, within-subject designs can be problematic in studies where participants learn from the first condition and apply that learning to subsequent conditions. Opting for a between-subject design allowed us to start each group with

[5]https://tinyurl.com/mlops-system-gm

[6]https://tinyurl.com/mlops-system-ma

[7]https://plantuml.com

a clean slate, minimising the influence of habituation on our findings.

Thirdly, random group assignment enhanced internal validity, mitigating bias due to participant demographics, such as prior knowledge or exposure to different experiment material. Finally, equal incentives for all participants reduced the likelihood of participants biassing the survey and self assessment responses.

F. Participants

The study consisted of controlled experiment sessions with seventy-two participant submissions. After applying the objective inclusion-exclusion criteria described in Section V-A, we included sixty-three unique participant contributions for evaluation.

Notably, much widely-cited related work either used even smaller sample sizes (see Allodi, Biagioni et al. [32]: $n = 29$, Labunets et al. [33]: $n = 29$, Shafari et al. [34]: $n = 28$, Heijstek et al. [35]: $n = 47$) or similar sample sizes (see Allodi, Cremonini et al. [31]: $n = 73$).

We randomly assigned each participant to one of two groups: a control group ($n_{\text{CONTROL}} = 31$) and an experimental group ($n_{\text{EXPERIMENTAL}} = 32$), hereafter referred to as CONTROL and EXPERIMENTAL, respectively. Each group had a different experiment sheet, as detailed in Section III-G. All participants were either BSc or MSc students enrolled in at least one of three different courses spanning two semesters:

- In the summer and winter semesters of 2022, we invited students enrolled in Distributed Systems Engineering (DSE)[8][9] to participate. This is an optional bachelor and master-level course.
- In the summer semester of 2022, students enrolled in Advanced Software Engineering (ASE)[10], which is a mandatory master-level course, could also take part.
- In the winter semester of 2022, students enrolled in Software Engineering 2 (SE2)[11] were also able to participate. SE2 is a mandatory bachelor-level course.

We obtained written consent from all participants, participation was entirely voluntary, and we awarded students extra credit in the form of bonus points on top of the standard course points as an incentive to participate. The material covered in the study was related to the subjects of the various courses (software engineering and distributed systems). However, it was not integral to the course material, i.e. we didn't teach the students the material in classes or lectures, nor did we expect them to learn it for the courses, and it was not examinable. Thus, it made sense for us to offer students taking part in the study bonus points rather than standard course points, also meaning that we were not disadvantaging those students who opted not to participate at all since they could still earn the full standard course points. It was also possible for students to receive the bonus points by carrying out the experiment tasks but opt out of having their contribution included in the experiment.

We pseudonymised the participants' submissions after each experiment session so their identities could not influence the evaluation of their submissions. Similarly, their participation, non-participation or opting out of including their submissions in the experiment results did not affect their grading other than the awarding or non-awarding of bonus points at the end of the semester.

Please note that a vote by the University of Vienna's Ethics Committee[12] was according to their guidelines not necessary, as our study could not threaten the research subjects' physical and psychological integrity, the right to privacy was preserved through complete anonymisation through double-blinding (we used a procedure that was checked by University of Vienna's data protection officer), and the participants could participate in the bonus point activity of the class but opt out of the experiment without any possible negative consequences (due to the double-blinding neither the course instructors nor the authors of the study knew who has opted out).

Including participants from courses at different stages of the degree programmes allowed us to use the participants as proxies for novice (SE2), novice to moderately advanced (DSE), and moderately advanced (ASE) software architects, designers, or developers. While our study did not specifically involve software architects or ML engineers, the demographic data we collected supports the idea that the students who participated in our experiment can effectively represent professional software developers. These students possess a broad spectrum of knowledge and experience in software development. To illustrate, when we compare their demographics to those of a 2016 survey that encompassed fifty thousand developers on the online platform Stack Overflow[13], we observe similarities in the most pertinent demographics compared to our participants, particularly that all of our participants had some degree of programming experience (please refer to Section V-B).

Kitchenham et al. [42] provide further justification for using students in research experiments and state that using students *"is not a major issue as long as you are interested in evaluating the use of a technique by novice or nonexpert software engineers. Students are the next generation of software professionals and, so, are relatively close to the population of interest"*. Furthermore, as noted in Höst et al. [47], Runeson [48], Svahnberg et al. [49], Salman et al. [50], and Falessi et al. [51], students may also represent professionals in empirical software engineering studies. We further discuss the use of students in our study in Section VI-B.

G. Material and Tasks

Material

We based the experiment on a selection of publically available MLOps system descriptions, which we describe in Section III-B. The systems and subject matter were relevant to the topics covered in SE2, DSE, and ASE but not

[8]https://ufind.univie.ac.at/en/course.html?lv=052500&semester=2022S
[9]https://ufind.univie.ac.at/en/course.html?lv=052500&semester=2022W
[10]https://ufind.univie.ac.at/en/course.html?lv=053020&semester=2022S
[11]https://ufind.univie.ac.at/en/course.html?lv=051050&semester=2022W

[12]https://www.qs.univie.ac.at/services/ethikkommission
[13]https://insights.stackoverflow.com/survey/2016

formally part of the course content. The study consisted of the following material[14]:

- An *experiment information document*, providing background information on MLOps concepts and informal MLOps system descriptions and diagrams, and an example system description.
- An *experiment document* containing a survey requesting participant background information (such as age, gender, education level, programming and industry experience), three informal textual system descriptions and corresponding informal MLOps system diagrams with associated tasks (described below), task surveys (also described below), and a concluding survey (we did not use this in the final evaluation). We did not release this document to participants until the start of their experiment session. A brief excerpt from an informal textual system description follows[15]:

> *Release pipeline*
>
> This pipeline shows how to operationalize and promote the scoring image safely across different environments. This pipeline is subdivided into two environments, QA and production:
>
> *QA environment:*
> - **Model Artifact trigger.** Release pipelines get triggered every time a new artifact is available. A new model registered to Azure Machine Learning Model Management is treated as a release artifact. In this case, a pipeline is triggered for each new model is registered.
> - **Create a scoring image.** The registered model is packaged together with a scoring script and Python dependencies (**Conda YAML file**) into an operationalization Docker image. The image automatically gets versioned through Azure Container Registry.
> - **Deploy on Container Instances.** This service is used to create a non-production environment. The scoring image is also deployed here, and this is mostly used for testing. Container Instances provides an easy and quick way to test the Docker image.
> - **Test web service.** A simple API test makes sure the image is successfully deployed.
>
> *Production environment:*
> - **Deploy on Azure Kubernetes Service.** This service is used for deploying a scoring image as a web service at scale in a production environment.
> - **Test web service.** A simple API test makes sure the image is successfully deployed.
>
> The release pipeline publishes a real-time scoring web service. A release to the QA environment is done using Container Instances for convenience, but you can use another Kubernetes cluster running in the QA/staging environment.

- We provided EXPERIMENTAL with supplementary information, including a description of how to understand semi-formal MLOps system diagrams with reference to UML-based generic component and pipeline diagramsWe also provided participants in EXPERIMENTAL with UML-based semi-formal MLOps system component and pipeline diagrams for each of the three systems, which are described in Section III-B, and examples of a component and pipeline diagram for one of the systems may be found in Appendices A and B respectively.

We provided two types of semi-formal, UML-based MLOps architecture diagrams (component and pipeline), rather than just a single diagram, as they represent the high-level constructs resulting from our prior empirical studies of architectural design decisions (ADDs) for the ML workflow and deployment [3], [4]. We automatically annotated the UML diagrams with behaviour and properties via stereotypes and connectors and generated them based on the Python-based model, similar to how we generated the ADD diagrams in those studies.

As discussed in Section II-A, MLOps systems are inherently intricate, often characterised by a high degree of complexity due to the amalgamation of ML models, data pipelines, deployment processes, and other aspects. Semi-formal UML-based MLOps systems diagrams, such as those provided in the experiment document and depicted in Appendices A and B, are invaluable tools for breaking down this complexity into more manageable components and are pivotal in enhancing understanding of MLOps-specific challenges.

One notable benefit of UML-based MLOps system diagrams is that they can be designed to emphasise best practices and principles specific to MLOps. They are a visual guide for ensuring the system is implemented consistently with industry standards. This alignment with known practices, including those presented in our previous work [3], [4], contributes to the efficiency and reliability of the system.

These diagrams incorporate annotations or labels that bring known anti-patterns, such as those described by Sculley et al. [17], to the forefront. Practitioners can promptly recognise anti-patterns in the system's design, simplifying the process of identifying and addressing problematic structural elements. When visualised within the broader context of the system, anti-patterns become more apparent, aiding practitioners in understanding their implications and addressing them effectively.

As in other areas of software engineering, the use of semi-

[14]All relevant artefacts are available in our replication package, which we will publish as an open access dataset on the long term archive https://zenodo.org upon acceptance, but make available for review as an anonymously accessible link here: https://ucloud.univie.ac.at/index.php/s/lzxEa8smkmhL8Ug – password: gZCUDSczwaxf

[15]We took the excerpt from the following source, which provides the full text and a corresponding informal MLOps system diagram: https://tinyurl.com/mlops-system-ma

formal representations has been shown to improve understanding (see Stevanetic et al. [52] and Haitzer and Zdun [53]), and we speculate that ML practitioners may also benefit when utilising semi-formal representations, for instance, to identify areas where technical debt (see Sculley et al. [17]) may accumulate within the system. These diagrams offer a visual guide to software modules and their dependencies, making recognising segments that may require refactoring or enhancements easier. Annotations or metadata within the diagrams can highlight potential sources of technical debt, providing a visual cue for practitioners to focus on addressing these areas effectively.

Lastly, semi-formal MLOps system architecture view descriptions serve as instrumental tools in illustrating the use of ML models, including the strategies for model selection, storage, versioning and reuse. This visual representation gives practitioners insights into the criteria and processes guiding model choices and their integration into the system. Practitioners can discern how models interact with various components, facilitating their comprehension of the design and decision-making processes related to model selection and reuse.

Tasks

We associated each of the three systems with three tasks relating to aspects common to MLOps systems, to be answered after reading and understanding each system description. An example of each task type follows:

Pipeline behaviour: a "select the true statements"-style question with four statements to choose from, for example:

- "Please keep time records and tick (✓) the true statements:

  ☐ The retraining pipeline always finishes by registering a model.
  ☐ The retraining pipeline can be triggered via a suitable API call to an appropriate component.
  ☐ The build pipeline can trigger the retraining pipeline.
  ☐ In the release pipeline, releases of images to production can be approved independently of whether the images were first deployed to staging and QA."

Components: a "list all components..."-style question with four different types of components to list, for example:

- "Please keep time records and list all components that...

  ...are cloud components: _____
  _____.
  ...are orchestration components: _____
  _____.
  ...provide an artefact to another component: _____
  _____.
  ...receive trigger data from another component: ____
  _____."

Pipeline properties: an "enter the correct number"-style question, for example:

- "Please keep time records and enter the correct number for each of the following statements:

  The various pipelines can be triggered by _____ type(s) of trigger in total.
  The pipelines contain _____ node(s) responsible for testing.
  The pipelines contain _____ step(s) that do not run automatically.
  The total number of pipelines triggered via a commit to the code base is _____."

The task design in the controlled experiment, encompassing questions related to pipeline behaviour, components, and pipeline properties, is based on findings from our previous grey literature studies [3], [4] on architectural design decisions, patterns and practices for MLOps. Technical components and their related pipelines are important in various MLOps practitioner roles for task execution and the successful implementation of MLOps principles such as automation, workflow orchestration, and reproducibility, as discussed by Kreuzberger et al. [2]. Thus, we consider our study tasks to be relevant to factors important for the decision-making processes regularly undertaken by practitioners, particularly those engaged in MLOps, as part of their routine work. These tasks serve as a bridge between the higher-level architectural understanding and the low-level implementation details, rendering them highly pertinent to real-world engineering practice.

Practitioner sources from our prior studies [3], [4] and others, such as Lakshmanan et al. [54] and Treveil et al. [55], highlight the importance and relevance of pipelines to MLOps. Unlike traditional CI/CD setups, such as those described by Humble and Farley [25] and Shahin et al. [27], MLOps architectures may consist of a wider variety of interconnected, fundamental ML-specific pipeline types and steps, such as data preprocessing pipelines, model training and evaluation pipelines, data ingestion steps, feature engineering steps and hyperparameter tuning steps, as well as pipeline trigger types for new training data availability, model performance degradation and changes in the data distribution. We put forth that our tasks concerning pipeline behaviour facilitate the assessment of the extent to which participants comprehend the functioning of MLOps systems for their pipelines. They shed light on, for instance, how pipelines register models, trigger other pipelines, or release images to different environments, thereby ensuring the correct and reliable operation of the system. This knowledge is highly representative of the insights required by practitioners, particularly those specialising in MLOps, within real-world contexts.

Counting, for instance, different trigger types, testing nodes, non-automatic steps, or pipelines initiated through code commits mirrors the evaluation of specific properties within the system. We regard such evaluations as a common practice among developers and conjecture that they are integral to comprehending system behaviour when undertaking activities such as identifying bottlenecks and ensuring the accurate configuration of pipelines. The tasks directly correlate with the decisions developers make when configuring and automating

various facets of MLOps systems. For instance, the process of determining how pipelines are triggered and specifying the actions they should take is a practical necessity when constructing dependable CI/CD pipelines.

The task of quantifying the number of specific properties within the pipelines aligns with the practical requirement to evaluate these properties for quality assurance and system performance. Developers routinely perform such assessments to unearth areas for improvement or potential issues.

According to Humble and Farley [25], almost all modern software systems consist of a collection of components, and the relations between them in build and deployment processes can influence the complexity of a system. They state that considering the interactions between them when implementing a deployment pipeline is challenging. The relations they describe between components and pipelines are also relevant for MLOps, for instance, how a deployment pipeline interacts with an artefact repository component or the level of interaction between a version control system and build infrastructure components. On this basis, we regard understanding MLOps components and their relations as important to MLOps practitioners, and our study task involving identifying components and their interactions should help assess how well participants understand relevant aspects of the system architectures described in the experiment.

Identifying distinct component types and articulating their roles mirror the real-world decisions developers frequently make when selecting and integrating components within a system. This process of identification is a pivotal aspect of ensuring that the selected components work harmoniously together.

The assigned tasks exhibit parallels with high-level architectural analysis, a sphere in which developers strive to grasp the general structure and behaviour of MLOps systems. Questions pertaining to the understanding and identification of pipeline behaviours and components are expected, in our considered opinion, to be suited to ascertaining the comprehension of a system's overarching architecture. Concurrently, these tasks delve into the minutiae by considering properties like the number of trigger types or steps that do not run automatically. In our view, analysing such low-level details is pivotal in system implementation and fine-tuning.

Tasks for both groups were identical. Each task concluded with a SELF-ASSESSMENT survey on a Likert scale, representing a subjective evaluation of each participant's performance. The survey questions, to be answered on a scale from "strongly agree" to "strongly disagree" were as follows:

- "I am confident that my provided answers and solutions for this task are correct."
- "It was easy for me to understand the textual system description."
- "It was easy for me to understand the system diagram(s)."
- "It was easy for me to identify pipelines."
- "It was easy for me to identify components."
- "It was easy for me to understand pipeline behaviour."
- "It was easy for me to understand components and their relations."
- "It was easy for me to understand properties of pipelines."

H. Variables, Hypotheses and Research Questions

The independent variable is the provision or non-provision of the supplementary material (formal MLOps system diagrams). Two dependent variables are defined: the CORRECTNESS of the answers provided to the tasks and the DURATION, i.e. the time taken to complete the tasks.

The two dependent variables, particularly the CORRECTNESS, may be used to gain insight into the overall understandability of system descriptions [13]–[15], [56], [57]. In the context of our experiment, CORRECTNESS measures whether the experiment participants could correctly identify and understand the different components, pipelines, architectural roles, behaviour, properties and relations in the described software architecture. A high level of CORRECTNESS indicates that the architecture is easy to understand and navigate. By contrast, a low level of CORRECTNESS suggests that the architecture is more complex and challenging to comprehend. DURATION measures the time it takes participants to complete the tasks assigned to them in the experiment. A shorter DURATION indicates that the participants could quickly and efficiently understand the software architecture, while a longer DURATION suggests that the architecture is more challenging to comprehend.

We hypothesised that MLOps system descriptions are easier to understand when semi-formal MLOps system diagrams are provided in addition to informal textual system descriptions and informal MLOps system diagrams. Thus, for *understanding*, we defined the following null and corresponding alternative hypotheses:

- $\mathbf{H_0 1}$: There is no significant difference in task CORRECTNESS when semi-formal MLOps system diagrams, in addition to informal textual descriptions and informal graphical system diagrams, are provided.
- $\mathbf{H_a 1}$: Task CORRECTNESS increases significantly when semi-formal MLOps system diagrams, in addition to informal textual descriptions and informal graphical system diagrams, are provided.

Furthermore, we wished to investigate the effects of providing supplementary material in the form of semi-formal MLOps system diagrams to EXPERIMENTAL on task DURATION. We formulated the following null and alternative hypotheses accordingly:

- $\mathbf{H_0 2}$: There is no significant difference in task DURATION when semi-formal MLOps system diagrams, in addition to informal textual descriptions and informal graphical system diagrams, are provided.
- $\mathbf{H_a 2}$: Task DURATION increases significantly when semi-formal MLOps system diagrams, in addition to informal textual descriptions and informal graphical system diagrams, are provided.

Finally, we wished to investigate any relationship between task DURATION and task CORRECTNESS, and thus formulated the following null and alternative hypotheses:

- $\mathbf{H_0 3}$: There is no significant increase in task CORRECTNESS as task DURATION increases when only informal

textual descriptions and informal graphical system diagrams are provided.
- **H_a3**: Task CORRECTNESS increases significantly as task DURATION increases when only informal textual descriptions and informal graphical system diagrams are provided.
- **H_04**: There is no significant increase in task CORRECTNESS as task DURATION increases when semi-formal MLOps system diagrams, in addition to informal textual descriptions and informal graphical system diagrams, are provided.
- **H_a4**: Task CORRECTNESS increases significantly as task DURATION increases when semi-formal MLOps system diagrams, in addition to informal textual descriptions and informal graphical system diagrams, are provided.

We were also interested in how confident participants of each group were in their performance, and defined a suitable research question:

- **RQ** How does the exclusion or inclusion of semi-formal MLOps system diagrams affect the accuracy with which participants assess their performance in the context of their perceived task correctness?

IV. Experiment Execution

A. Pilot Test

After preparing our experiment materials, we conducted preliminary tests involving student tutors from our research group. Much like the participants in the subsequent stages, we gave the tutors the information sheet two weeks in advance, allowing them to acquaint themselves with the necessary knowledge to address the tasks. We randomly allocated one tutor to each of the CONTROL and EXPERIMENTAL groups. They engaged in the experiment under the same conditions we had planned for the official experiment sessions. These conditions included the option to seek clarifications regarding the experimental procedure, a seventy-five-minute timeframe for responding to the experiment questions, and no supplementary support beyond the provided materials.

After the pilot tests, we sought feedback from the tutors regarding their experiences. Their responses indicated that we had crafted the information sheet exceptionally well and had offered ample guidance for addressing the experiment questions. They found the experiment to be straightforward to navigate, and they noted that even participants who were new to the subject matter and concepts would likely find it accessible, primarily due to the comprehensive materials and examples provided. Given this positive feedback, we made no alterations to our experiment design.

B. Preparation

To help participants prepare for their experiment sessions, we provided registered participants with the information document described in Section III-G via our e-learning platform[16] two weeks before their session. The purpose of providing the information sheet was to ensure that the participants had the same minimum prerequisite knowledge and understanding of the subject matter to reduce the in-session learning curve and reduce potential bias among the participants due to factors such as education level, prior knowledge, and programming or industry experience. All participants worked through three tasks per the described MLOps system. Once we had concluded all experiment sessions, we consolidated the collected data and derived and analysed descriptive statistics. The analysis aimed to determine whether the statistics supported or refuted the hypotheses defined in Section III-H. At various stages in the experiment sheets, we also asked participants to complete short surveys to describe how well they believed they had performed on a Likert scale. We used these survey responses to gain insight into how well participants thought they had performed compared to how well they actually performed (in terms of CORRECTNESS) and answer our research question.

[16]https://moodle.univie.ac.at

C. Procedure

We conducted the experiment sessions as a pen-and-paper exercise under controlled conditions akin to a traditional closed-book written examination, with no supplementary tools or resources permitted except for the information sheets distributed in advance. To ensure equal access to the supplementary information, we provided copies of these sheets in case participants had not brought their own. We set an upper limit on the DURATION, which was the same for all participants; we did not allow collaboration between participants; we alternately assigned participants to either CONTROL or EXPERIMENTAL, and we ensured that participants sat at a sufficient distance from one another to prevent collaboration or copying. After being provided with a fifteen-minute overview of the experimental procedure and the structure of the experiment sheets at the start of the session, we issued each participant with an experiment sheet associated with either EXPERIMENTAL or CONTROL. We instructed participants to enter their background information and complete the tasks and embedded surveys in consecutive order. We allowed the participants a maximum of seventy-five minutes to do so and required them to record time intervals for reading the system descriptions and completing tasks. We projected a clock with seconds granularity onto a screen to facilitate the participants' recording of timestamps.

V. Analysis

A. Dataset Preparation

We could not include all participants' contributions in the study and defined objective exclusion criteria as follows:

- Participants recorded timestamps but made systematic errors throughout the experiment, meaning that we could not infer the timestamps' correct values ($n = 1$). This single participant excluded for this reason did not record timestamps correctly, recording them to whole minutes rather than whole seconds.
- Participants systematically recorded timestamps throughout the experiment that led to obviously implausible DURATION values ($n = 1$). The single participant excluded

for this reason recorded taking four seconds to read the supplementary material, seven to thirteen seconds to read the system descriptions, and times ranging from zero seconds to five seconds to complete the tasks. In contrast, most other participants would typically take several minutes for these activities.
- Participants did not confirm reading and understanding the information sheet before the experiment session ($n = 3$). This involved ticking the relevant checkbox on the initial survey page of the experiment document.
- Participants did not consent to their contribution being used in the study ($n = 1$). This likewise involved ticking the relevant checkbox on the initial survey page of the experiment document. The single participant who opted out of having their contribution included didn't confirm reading and understanding the information sheet before the experiment session either. We would have excluded them anyway for that reason and also included them in the three students counted above.
- Students could participate in both semesters to gain extra credit in multiple courses. However, in such cases of repeated participation, we only included the contribution from the first experiment session the student participated in ($n = 4$).

Thus, we excluded nine out of seventy-two contributions, leaving contributions from sixty-three unique participants for inclusion. Of the included contributions, we conducted some necessary further manual processing:

- If an individual timestamp for a task was missing but could be inferred, then we used the inferred value.
- If at least one timestamp for a task was missing and could not be inferred, then we excluded the time for this task.
- If a task had not been attempted, we excluded it.
- If a survey response wasn't completed, then we excluded this individual survey response from the overall results. However, we still included the remainder of the participant's survey responses (barring any other incomplete survey responses).
- One participant misunderstood a task and communicated this in writing on the experiment sheet. We excluded their contribution for this individual task from the results but still included the remainder of their contribution (barring any other excluded tasks).

The implications of excluding individual tasks and survey responses are as follows: if we excluded a task for any reason, then we recorded no score for the task and did not use it when calculating the average CORRECTNESS, nor did we use it when calculating the total DURATION; if a survey response was missing, then we did not use it to gain insight into how well participants thought they had performed compared to how well they actually performed. We entered the data from the contributions to be included in the study, including the participants' background information, survey responses, time intervals, and task answers, into a LibreOffice [58] OpenDocument Spreadsheet (ODS)[17] file. We converted the participants' start and stop timestamps into a DURATION in seconds.

We exported the ODS file to a comma-separated value (CSV) [59] file and, having already installed the necessary R[18] [60] dependencies, we then input the CSV file into an R script, which performed some initial processing and generated an RDS [61] file. We then input the RDS file into a different R script, which analysed the data and generated plots and tables of statistics.

B. Participant Demographics

Participants' background information that we collected included age (see Figure 3[19]), gender, course, education level, programming experience (see Figure 4), modelling experience, software industry experience (see Figure 5), hardware industry experience, and programming and modelling language experience. We also collected data on whether participants had prior knowledge of ML, CI/CD, and CI/CD for ML. Neither group appeared to have an unfair advantage over the other, and any imbalances for specific demographics were compensated elsewhere. Of the sixty-three participants, thirty-four had no higher education qualifications ($n_{\text{CONTROL}} = 15$, $n_{\text{EXPERIMENTAL}} = 19$)[20], twenty-six had a bachelor's degree ($n_{\text{CONTROL}} = 14$, $n_{\text{EXPERIMENTAL}} = 12$), and three had a master's degree ($n_{\text{CONTROL}} = 2$, $n_{\text{EXPERIMENTAL}} = 1$).

Across both groups, forty-two ($n_{\text{CONTROL}} = 19$, $n_{\text{EXPERIMENTAL}} = 23$) participants had prior knowledge of ML, thirty-four ($n_{\text{CONTROL}} = 13$, $n_{\text{EXPERIMENTAL}} = 21$) had prior knowledge of CI/CD, and two ($n_{\text{CONTROL}} = 0$, $n_{\text{EXPERIMENTAL}} = 2$) had prior knowledge of CI/CD for ML. Thirty-five ($n_{\text{CONTROL}} = 19$, $n_{\text{EXPERIMENTAL}} = 16$) had prior knowledge of modelling. Seven ($n_{\text{CONTROL}} = 5$, $n_{\text{EXPERIMENTAL}} = 2$) participants had no prior knowledge of any of the above.

Regarding education level, CONTROL was slightly better qualified, having four fewer participants without a university education, two more with a bachelor's degree and one more with a master's degree. Regarding prior knowledge of ML, CI/CD, EXPERIMENTAL appeared to be more experienced since that group had eight more participants with prior knowledge of CI/CD and two more with prior knowledge of CI/CD for ML. On the other hand, CONTROL had three more participants with prior modelling knowledge. The average age in CONTROL was 24.43. In *Experimemtal* 25.16, i.e. roughly the same, the mean number of years of experience in the software industry was very similar, with 1.34 for CONTROL and 1.25 for EXPERIMENTAL. The number of years of programming experience was also similar between groups, with CONTROL having an average of 3.85 and EXPERIMENTAL having an average of 3.62. Thus, neither of the groups had an overall

[17]https://www.oasis-open.org/committees/tc_home.php?wg_abbrev=office

[18]We used the programming language R (https://www.r-project.org) for the statistical analysis. We will publish the source code in our replication package on https://zenodo.org upon acceptance, but for review we have made it available as an anonymously accessible link here: https://ucloud.univie.ac.at/index.php/s/lzxEa8smkmhL8Ug – password: gZCUDSczwaxf

[19]Two participants did not provide their ages.

[20]n_{CONTROL} is the number of observations in CONTROL and $n_{\text{EXPERIMENTAL}}$ is the number of observations in EXPERIMENTAL.

This article has been accepted for publication in IEEE Transactions on Software Engineering. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TSE.2024.3367488

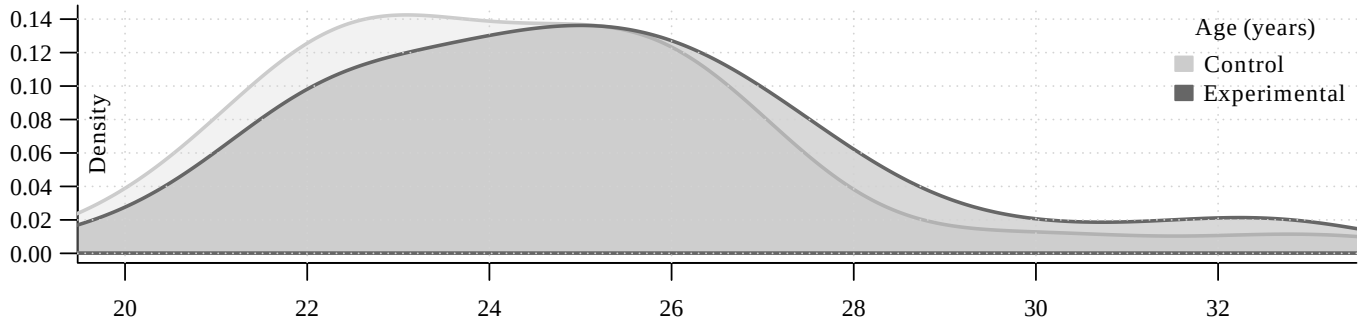

Fig. 3: Kernel Density Plot of Participants' Ages.

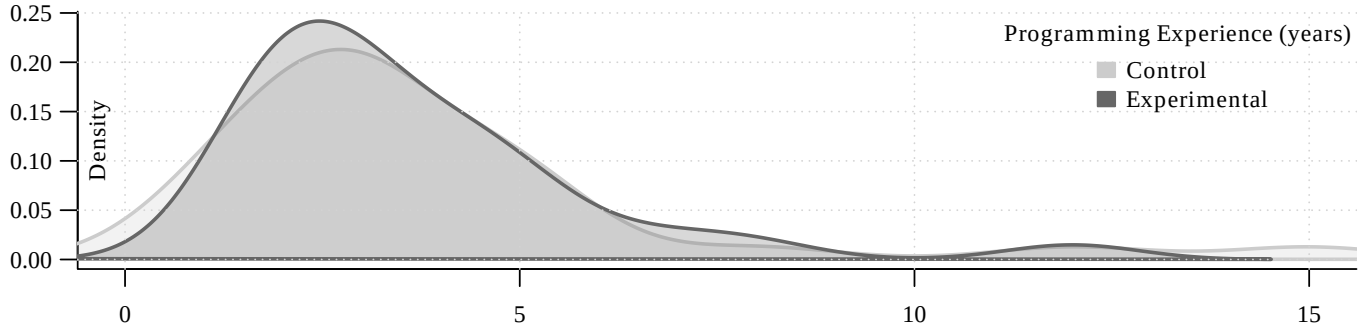

Fig. 4: Kernel Density Plot of Participants' Programming Experience.

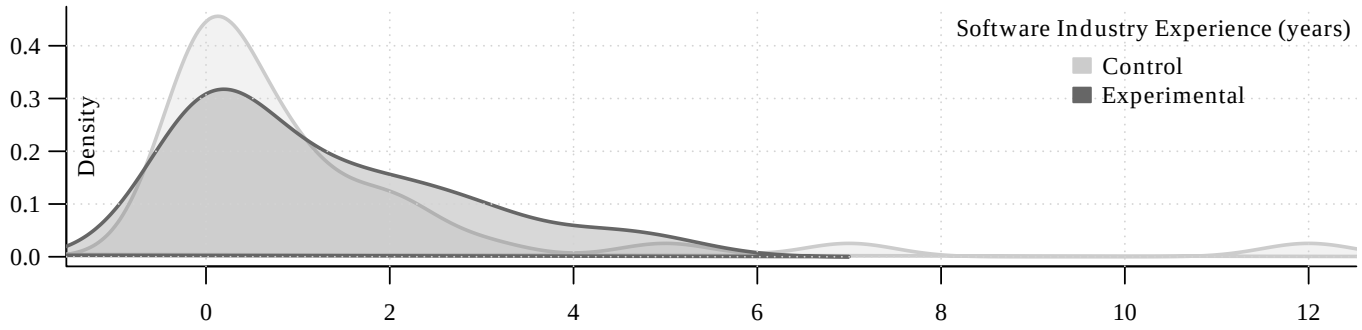

Fig. 5: Kernel Density Plot of Participants' Software Industry Experience.

advantage over the other in multiple demographic aspects, and any advantage that a group may have had in one area was balanced in a different area for the other group.

C. Descriptive Statistics

CORRECTNESS

In Table I, We note the group-specific statistics for the dependent variable CORRECTNESS, which we define within the range $[0, 1] \cap \mathbb{R}$. This table presents the number of observations along with measures of central tendency and dispersion for each group. In this paper, we use various representations of the data, each offering important information for understanding the data and guiding our selection of the appropriate statistical tests. The kernel density plots reveal insights into the distribution, the Q-Q plots are used to assess normality, the box plots are used for summary statistics, the scatter plots reveal any potential correlation between two dependent variables, and the tables display a variety of descriptive statistics. These statistics can be viewed as a kernel density plot presented in Figure 6, as well as a box plot provided in Figure 7. In the box plot displayed in Figure 7, it's clearly noticeable that the median and the entire interquartile range of EXPERIMENTAL exceed those of CONTROL and even surpass the maximum value of CONTROL. The boxes, which do not overlap, emphasise the difference between the two groups in terms of CORRECTNESS. Furthermore, EXPERIMENTAL features an extended lower whisker due to some participants' underperformance compared to other group members. Additionally, EXPERIMENTAL exhibits two outliers with notably low CORRECTNESS values[21]. Upon visually examining Figure 6 and considering the skew values provided in Table I, it becomes evident that the distribution of CONTROL closely approximates a nearly symmetrical or lightly negatively skewed form. In contrast, EXPERIMENTAL's distribution displays a highly negative skew. The kurtosis values, both < 3, suggest

[21]It's worth noting that we identified these outliers as natural variations within the population rather than arising from measurement, data entry, or data processing errors, thus representing true outliers.

This article has been accepted for publication in IEEE Transactions on Software Engineering. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TSE.2024.3367488

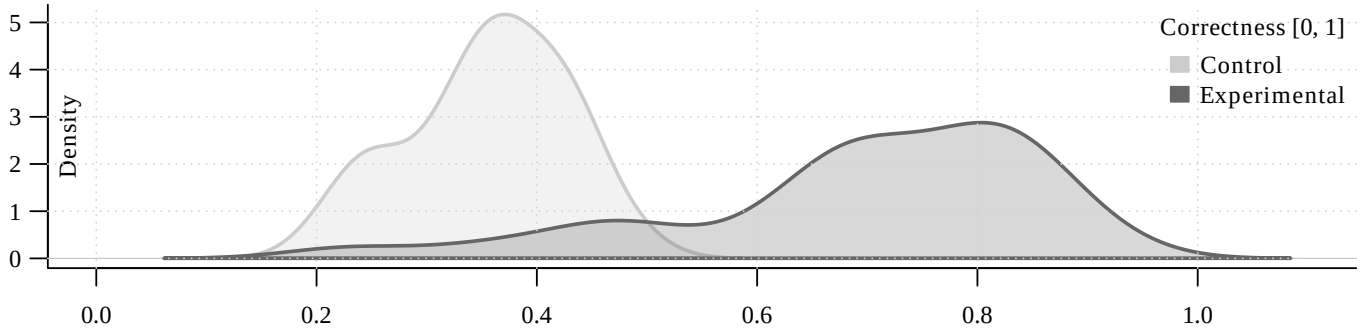

Fig. 6: Kernel Density Plot of CORRECTNESS.

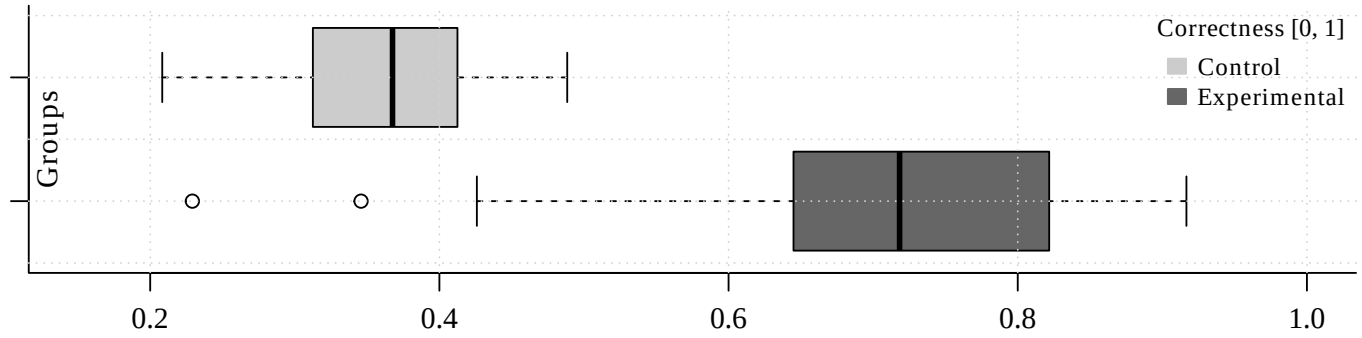

Fig. 7: Box Plot of CORRECTNESS.

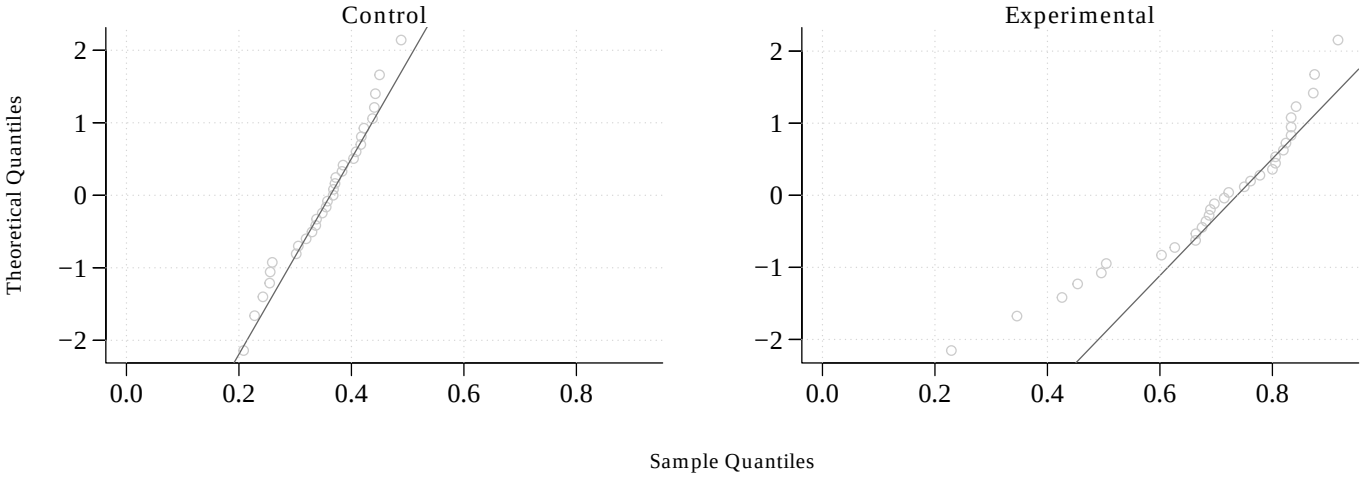

Fig. 8: Normal Q-Q Plot of CORRECTNESS per Group.

that the distributions in both groups are platykurtic.

Examining the normal Q-Q plots in Figure 8 for both groups and CORRECTNESS, it appears that the data from CONTROL exhibits a visual resemblance to a normally distributed pattern. However, the normality of EXPERIMENTAL's data remains inconclusive based on these visual cues. To test for normality, we chose the Shapiro-Wilk [62] normality test since, according to Razali and Yap [63], it is more potent than alternatives (such as Anderson-Darling [64], Lilliefors [65], and Kolmogorov-Smirnov [66]). Assuming $\alpha = 0.05$, this test indicated that the CONTROL's distribution is not significantly different from the normal distribution, with a value $p > \alpha$, whereas EXPERIMENTAL's distribution significantly deviates from the normal distribution, with a value $p \leq \alpha$.

DURATION

Table I shows the number of observations, central tendency, and dispersion measures per group for the dependent variable DURATION[22] These statistics are visualised as a kernel density plot in Figure 9 and as a box plot in Figure 10. The skews shown in Table I indicate that CONTROL's distribution is nearly symmetrical to lightly negatively skewed, whereas EXPERIMENTAL's distribution is moderately negatively skewed. The kurtosis values < 3 for both groups indicate that the groups' distributions are platykurtic.

In the box plot of Figure 10, the groups' interquartile ranges overlap, with EXPERIMENTAL's interquartile range almost being contained entirely within CONTROL's, with a

[22]We denote DURATION in seconds.

This article has been accepted for publication in IEEE Transactions on Software Engineering. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TSE.2024.3367488

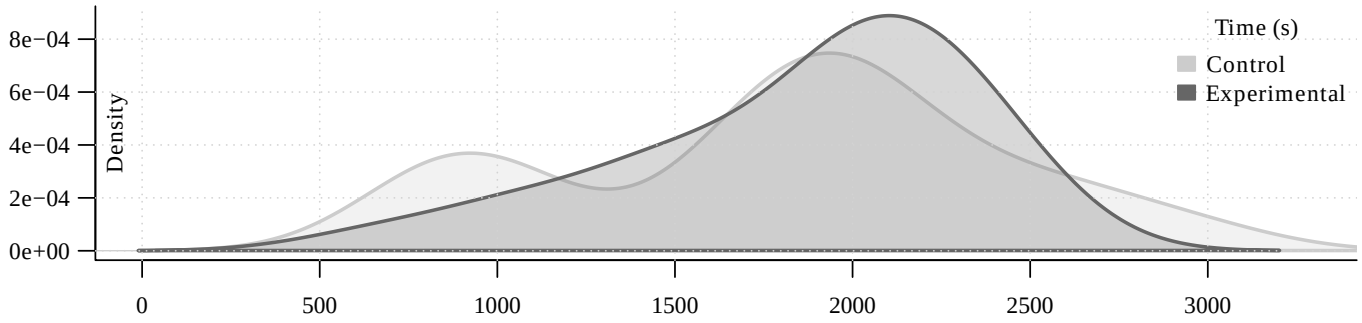

Fig. 9: Kernel Density Plot of DURATION.

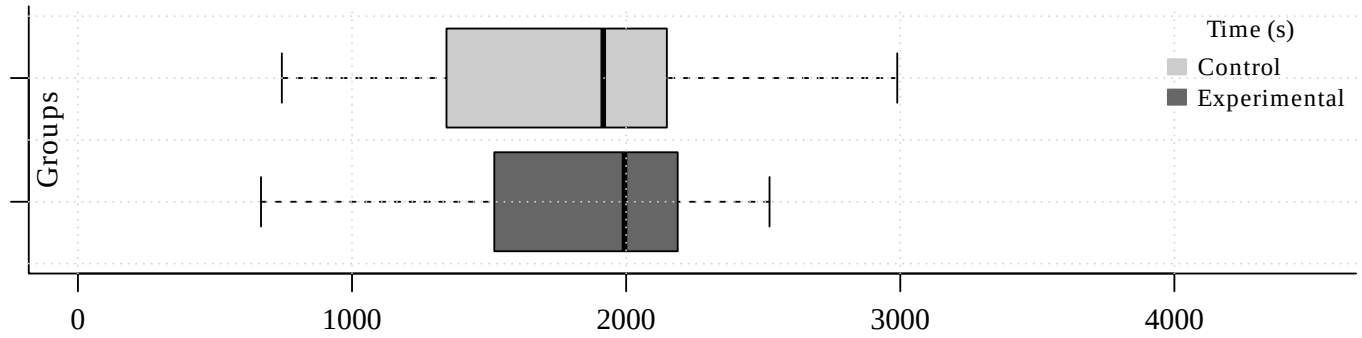

Fig. 10: Box Plot of DURATION.

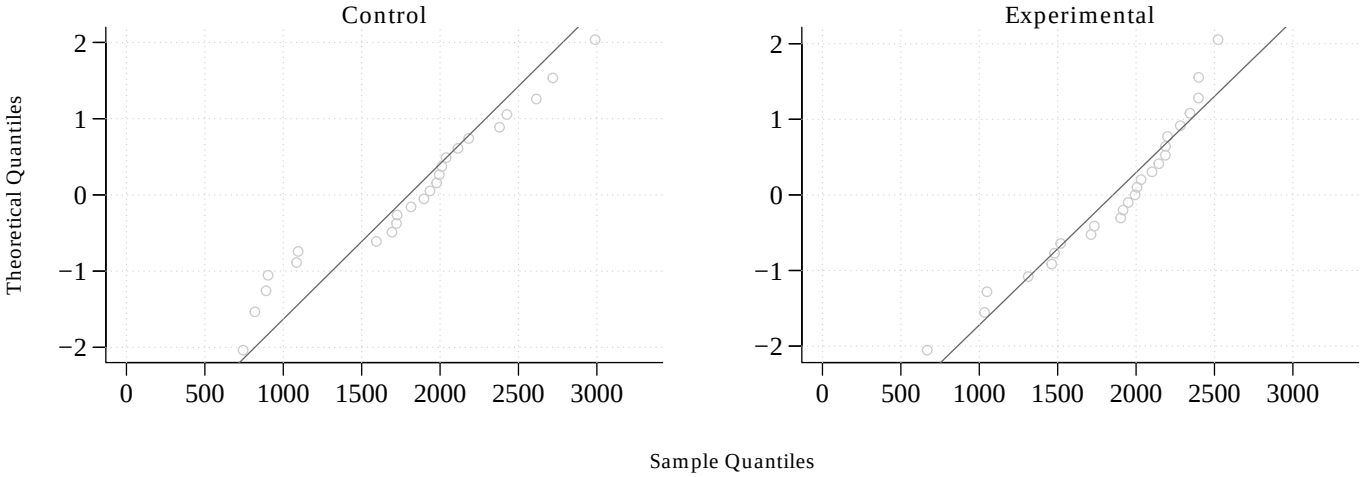

Fig. 11: Normal Q-Q Plot of DURATION per Group.

slightly higher median. EXPERIMENTAL's box plot's whiskers indicate a smaller spread of values than CONTROL's, which is also confirmed by the lower standard deviation (478.73 for EXPERIMENTAL versus 621.19 for CONTROL). While both groups' minimum values were similar, CONTROL's maximum value was relatively high, but we did not deem it an outlier for that group.

Visual inspection of the normal Q-Q plots for both groups for DURATION, visible in Figure 11, was insufficient for us to determine whether each group's data were normally distributed. The Shapiro-Wilk normality test indicated that, for DURATION, neither group's distribution is significantly different to normal, with values $p > \alpha$ for both groups.

D. Hypothesis Testing

CORRECTNESS and DURATION

When comparing two or more groups on two or more metric dependent variables, and provided certain other assumptions are met, it is usually possible to apply the *Multivariate Analysis of Variance* (MANOVA) [67] statistical test to determine whether independent variables on their own affect dependent variables. As noted in Section V-C, CONTROL's distribution is not significantly different from the normal distribution for CORRECTNESS and DURATION, whereas EXPERIMENTAL's distribution is significantly different from the normal distribution for CORRECTNESS but not for DURATION. Still, MANOVA requires all distributions to be normally distributed. It was thus necessary for us to compare the variances of both

TABLE I: Descriptive Statistics per Group of Dependent Variables CORRECTNESS and DURATION.

CORRECTNESS		
	CONTROL	EXPERIMENTAL
Observations	31	32
Mean	0.3553	0.6947
Standard deviation	0.0721	0.1643
Median	0.3676	0.7183
Median abs. deviation	0.0728	0.1431
Minimum	0.2083	0.2292
Maximum	0.4884	0.9167
Skew	-0.3261	-1.0567
Kurtosis	-0.8388	0.4515
Shapiro-Wilk Test p	0.4359	0.0056

DURATION		
	CONTROL	EXPERIMENTAL
Observations	24	25
Mean	1806.88	1861.56
Standard deviation	621.19	478.73
Median	1916.50	1994
Median abs. deviation	436.63	416.61
Minimum	744	668
Maximum	2989	2523
Skew	-0.1642	-0.7997
Kurtosis	-0.9032	-0.2763
Shapiro-Wilk Test p	0.2826	0.0711

groups for both dependent variables to determine their equality since this would further inform our choice of statistical tests. We selected Bonett's [68] method as a two-sided test to compare variances. This was a suitable choice since even though EXPERIMENTAL's CORRECTNESS data was heavily skewed, the sample sizes for all four combinations of group and variable was $n \geq 20$. Had the latter condition not been the case, then another test, such as Levene's [69] or Brown-Forsythe's [70] method, may have been more appropriate to reduce the risk of a high type I error rate when data is highly skewed and $n < 20$. Bonett's test is also accurate for any continuous distribution of quantitative values and does not require that the data be normally distributed. Given the above conditions, it is usually more powerful and reliable than Levene's or Brown-Forsythe's tests.

For CORRECTNESS, the estimated ratio is 2.143862, with a 95% confidence interval of $[1.401176, 3.280205]$ and $p = 0.0004406616$ with $\alpha = 0.05$. Since $p \leq \alpha$, the estimated ratio of the groups' standard deviations is statistically significant, i.e. the variances for CORRECTNESS differ between the groups. For DURATION, the estimated ratio is 0.76211, with a 95% confidence interval of $[0.4590661, 1.265203]$ and $p = 0.2935268$ with $\alpha = 0.05$. Since $p > \alpha$, the estimated ratio of the groups' standard deviations for DURATION is not statistically significant, i.e. the variances for DURATION do not differ.

Given that the dependant variable CORRECTNESS had differing variances between groups, we could not use the Kruskal-Wallis [71] test or Wilcoxon's [72] rank sum test [45], [73]. Welch's [74] t-test is suitable for unequal population variances but assumes normally distributed data, so this was not an option either.

Cliff's δ [75] is a robust, nonparametric test that makes no assumptions about data distribution, differing distributions between populations, or unequal variances and is recommended as a suitable method in this scenario by Kitchenham [45]. Even though Cliff's δ was originally intended to measure ordinal data, it is equally applicable [76], [77] to the quantitative, continuous data in this study. Cliff's δ estimates the probability that a randomly selected observation from one group is larger than a randomly selected observation from a second group, subtracting the reverse probability [78].

When testing multiple hypotheses with a single method (we applied Cliff's δ twice), it is necessary to lower the level of α to avoid type I errors[23]. Various methods to adjust α can be chosen from, such as the false discovery rate [79], or the Bonferroni-Dunn [80], [81] correction. The latter is the strictest form of correction and is given by Equation 1:

$$\alpha' = \frac{\alpha}{n} \tag{1}$$

where n is the number of times a test was applied. In our study, this results in $\alpha' = \frac{0.05}{2} = 0.025$ where $\alpha = 0.05$ and $n = 2$. The results of the one-tailed Cliff's δ test are shown in Table II for CORRECTNESS and Table II for DURATION. For CORRECTNESS, Cliff's δ indicates by $p \leq \alpha'$ that EXPERIMENTAL scored significantly higher than CONTROL. For DURATION, Cliff's δ yielded $p > \alpha'$, so we cannot conclude that EXPERIMENTAL took significantly longer than CONTROL to complete the experiment. To verify these results, we ran a one-tailed Brunner-Munzel [82] test for each dependent variable. This test is suitable for differing distributions and can also be applied to arbitrary data distributions. Adjusting α to derive α' for repeated tests via Bonferroni-Dunn correction as previously described, the results for the Brunner-Munzel test as indicated in Table II align with those of Cliff's δ, with $p \leq \alpha'$ for CORRECTNESS and $p > \alpha'$ for DURATION.

Based on both tests, concerning CORRECTNESS, we can reject the null hypothesis $\mathbf{H_0 1}$ and thus accept the alternative hypothesis $\mathbf{H_a 1}$. Thus, we found a significant difference in task CORRECTNESS in favour of providing semi-formal MLOps system diagrams in addition to informal textual descriptions and informal graphical system diagrams.

Conversely, concerning DURATION, we must accept the null hypothesis $\mathbf{H_0 2}$ and reject the alternative hypothesis $\mathbf{H_a 2}$, since we found no significant difference in task DURATION between CONTROL and EXPERIMENTAL.

Correlation Between CORRECTNESS and DURATION

Visual inspection of the scatter plot depicting any potential correlation between the two dependent variables CORRECTNESS and DURATION in Figure 12 per group didn't reveal an apparent linear correlation. For CONTROL, the CORRECTNESS barely increased with time. In the case of EXPERIMENTAL,

[23]Note that the α adjustment is not necessary when testing for normality or when comparing variances.

TABLE II: Hypothesis Tests per Group Combination of the Dependent Variables CORRECTNESS and DURATION.

CORRECTNESS ($H_0$1)	
	EXPERIMENTAL vs CONTROL
Cliff's δ Test	
Cliff's δ	-0.8633
s_δ	0.0886
v_δ	0.0079
z_δ	-9.7401
CI_{low}	-0.9528
CI_{high}	-0.6359
$P(X > Y)$	0.9317
$P(X = Y)$	0.0000
$P(X < Y)$	0.0683
p	3.719e−13
Brunner-Munzel Test p	3.063e−10

DURATION ($H_0$2)	
	EXPERIMENTAL vs CONTROL
Cliff's δ Test	
Cliff's δ	-0.0767
s_δ	0.1688
v_δ	0.0285
z_δ	-0.4542
CI_{low}	-0.3434
CI_{high}	0.2015
$P(X > Y)$	0.5383
$P(X = Y)$	0.0000
$P(X < Y)$	0.4617
p	0.3259
Brunner-Munzel Test p	0.3275

despite an apparent increase in CORRECTNESS with DURATION, the data points lie so far from the indicated reference line that we cannot assume a linear correlation. Following this visual inspection, we deemed it prudent to test for correlation. Various tests were available, with three well-known ones being Pearson's correlation coefficient, Kendall's τ, and Spearman's ρ.

Pearson's [83] correlation coefficient is suitable for continuous variables that are both normally distributed. We established earlier in this section that this is not the case for EXPERIMENTAL's CORRECTNESS data; thus, we disregarded Pearson's correlation coefficient.

The Kendall [84] rank correlation coefficient (Kendall's τ) measures the relationship between two variables. It is suitable for our purposes since it may be used with continuous data and is non-parametric. We calculated the one-tailed Kendall rank correlation coefficient. For confirmation, we also calculated the one-tailed Spearman [85] rank correlation coefficient (Spearman's ρ), which is similar to Kendall's τ and makes similar assumptions about the provided data.

For CONTROL, Kendall's τ and Spearman's ρ coefficients indicated a very weak positive association between CORRECTNESS and DURATION. However, as signified by $p > \alpha'$ (where α' is derived from the adjustment of α as described earlier in this section) for both tests, we must accept the null hypothesis **$H_0$3** and thus reject the alternative hypothesis **H_a3**, since we found that there was no significant positive correlation between CORRECTNESS and DURATION.

For EXPERIMENTAL, both Kendall's τ and Spearman's ρ coefficients indicated a moderate positive association between CORRECTNESS and DURATION. The p-value $p > \alpha'$ for both tests indicates that the observed association is statistically significant, i.e. that there is a significant positive correlation between CORRECTNESS and DURATION, so we must reject the null hypothesis **$H_0$4** and thus accept the alternative hypothesis **H_a4**. Furthermore, the large z-value for Kendall's τ indicates that the observed value for this measure is quite far from the expected value under the null hypothesis, which supports our conclusion that there is a true association between CORRECTNESS and DURATION for this group. Additionally, the value of S yielded by Spearman's ρ test indicates that the observed ranks of the two variables under test are not identical, which provides further evidence still that a true association between the variables exists.

TABLE III: Correlation per Group of the Dependant Variables CORRECTNESS with DURATION per Group.

	CONTROL ($H_0$3)	EXPERIMENTAL ($H_0$4)
Kendall's τ	0.0580	0.3105
p	0.3457	0.0149
z	0.3969	2.1726
Spearman's ρ	0.0504	0.4339
p	0.4075	0.0151
S	2184	1471.7830

E. Research Question

As described in Sections III-H, we defined a research question where we expressed our interest in how accurately participants predicted their correctness.

As described in Section III-G, we required participants to complete a survey after each task, assessing their CONFIDENCE that their answers were correct so that we could determine a SELF-ASSESSMENT score. We derived a formula that considers both the CORRECTNESS of each participant's answers and the CONFIDENCE they had in the CORRECTNESS of their answers to arrive at an overall SELF-ASSESSMENT score. We give the formula we derived in Equation 2:

$$self_{\text{ASSESSMENT}} = self_{\text{CORRECTNESS}} - \frac{5 - self_{\text{CONFIDENCE}}}{5} \quad (2)$$

where $self_{\text{CORRECTNESS}}$ represents the participants' average CORRECTNESS as defined in V-C, with 0 indicating entirely incorrect answers and 1 indicating completely correct answers.

$self_{\text{CONFIDENCE}} \in [0, 5] \cap \mathbb{R}$ represents the participant's average CONFIDENCE according to a survey on a five-point Likert scale. We assigned each scale point an equidistant value within this range. A value of 0 indicates that a participant had high CONFIDENCE in the CORRECTNESS of their answers, whereas a value of 5 indicates low CONFIDENCE.

$self_{\text{ASSESSMENT}} \in [-1, 1] \cap \mathbb{R}$ represents the participant's average SELF-ASSESSMENT score. A value $self_{\text{ASSESSMENT}} < 0$

This article has been accepted for publication in IEEE Transactions on Software Engineering. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TSE.2024.3367488

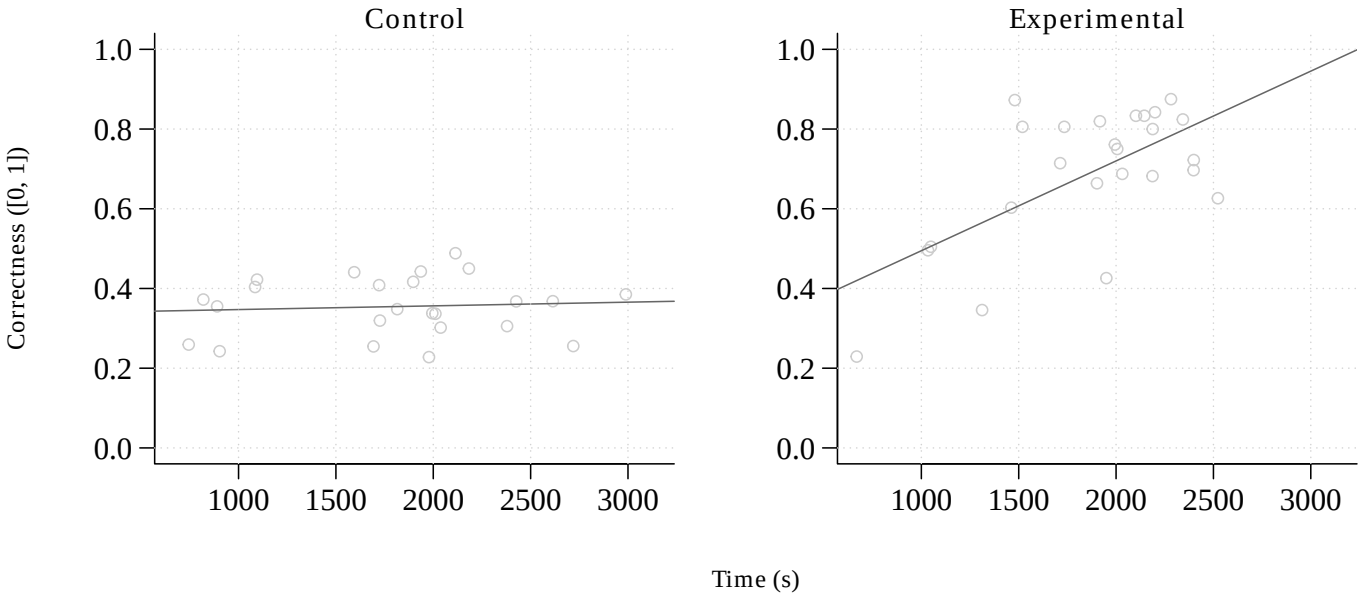

Fig. 12: Scatter Plot of the Dependent Variables CORRECTNESS to DURATION per Group.

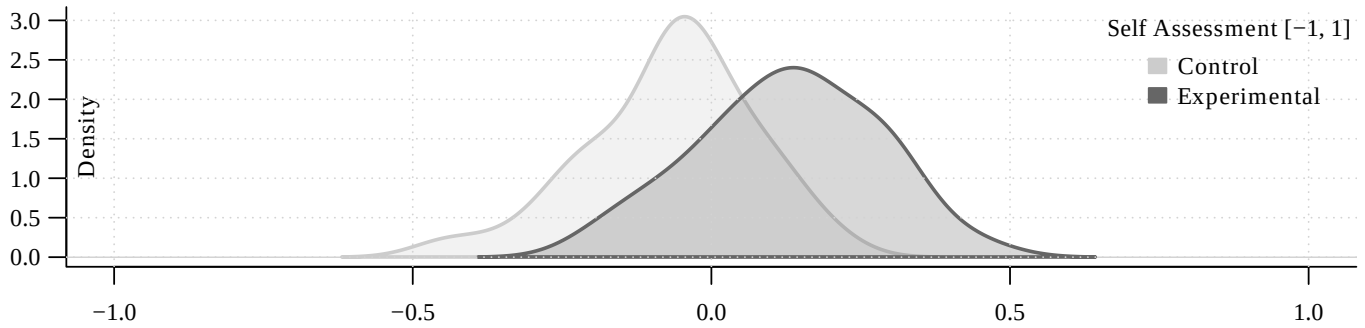

Fig. 13: Kernel Density Plot per Group of Participants' Self Assessment.

indicates that a participant overestimated the CORRECTNESS of their answers, $self_{\text{ASSESSMENT}} = 0$ indicates that a participant correctly estimated the CORRECTNESS of their answers and $self_{\text{ASSESSMENT}} > 0$ indicates that a participant underestimated the CORRECTNESS of their answers.

Figure 13 depicts the participants' overall SELF-ASSESSMENT score per group as a kernel density plot and indicates that CONTROL slightly overestimated their CORRECTNESS on average across all tasks, whereas EXPERIMENTAL slightly underestimated theirs.

VI. DISCUSSION

A. Discussion of the Results

CORRECTNESS and DURATION

Overall, our results strongly support the assertion that the provision of semi-formal MLOps system diagrams significantly aids the understanding of MLOps system descriptions when understanding is quantified by CORRECTNESS. Providing practitioners with semi-formal MLOps system diagrams is advantageous when asking them to understand MLOps system descriptions and complete tasks based on them. This result aligns with our expectations.

Participants in EXPERIMENTAL did not take significantly longer to complete the tasks compared with CONTROL once they had initially read and understood the material, despite having more reference material during task completion. A possible explanation for no significant difference in DURATION is that, when completing the tasks, participants in EXPERIMENTAL might have mainly consulted the UML-based MLOps system diagrams to locate the relevant information and were satisfied that they had found it relatively quickly. This contrasts with participants in CONTROL, who could only search for details in informal textual descriptions and informal graphical system diagrams. Assuming that DURATION is inversely correlated with understanding, this result shows that providing semi-formal MLOps system diagrams as supplementary material does not hinder understanding. Had it taken participants in EXPERIMENTAL significantly longer to complete the tasks (independent of their CORRECTNESS), then this would potentially have indicated that semi-formal MLOps system diagrams can represent a barrier to, or steepen the learning curve of, understanding MLOps system descriptions, at least to some extent. Consequently, when first attempting to understand a system and complete such tasks, practitioners

should not expect to take significantly longer when provided with semi-formal MLOps system diagrams (and, in combination with the findings for CORRECTNESS, they can expect better results than if they had not been provided with the semi-formal MLOps system diagrams). This result did not align with our expectations – we expected to see participants taking significantly longer to complete the tasks with semi-formal MLOps system diagrams due to the greater amount of reference material to look through when completing the tasks – but it is, in fact, a positive result since it is a strong argument for providing practitioners with semi-formal MLOps system diagrams.

Correlation Between CORRECTNESS and DURATION

Intuitively, one would expect CORRECTNESS and DURATION to be strongly correlated, and this was our assumption. In contrast to our expectations, there was no significant positive correlation between CORRECTNESS and DURATION for CONTROL. This suggests that participants in CONTROL believed that they had found the correct answers within a reasonable DURATION[24] and moved on to the next task. Thus, practitioners completing tasks based on informal system descriptions are advised that consuming excessive time to complete such tasks does not necessarily yield better performance in terms of CORRECTNESS.

Conversely, and more closely aligned with our expectations, we note a significant positive correlation between CORRECTNESS and DURATION for EXPERIMENTAL. There are multiple possible explanations for this correlation. It could indicate that members of EXPERIMENTAL applied a different methodology when completing the tasks. Given that the only difference between CONTROL and EXPERIMENTAL was the presence of semi-formal MLOps system diagrams, we might assume that members of EXPERIMENTAL mainly relied on the semi-formal MLOps system diagrams as a resource. Alternatively, the positive correlation between CORRECTNESS and DURATION might be explained by the participants first checking the informal textual descriptions and informal graphical system diagrams for answers and then subsequently checking and correcting their answers using the semi-formal MLOps system diagrams (hence leading to an improved CORRECTNESS on average with increased DURATION). This result greatly strengthens the case for providing practitioners with semi-formal MLOps system diagrams - the additional time involved in studying semi-formal MLOps system diagrams has a direct positive correlation with the correct understanding of the modelled systems. So practitioners are advised to make use of such model-based architectural documentation where at all possible. In our study, providing semi-formal MLOps system diagrams appears to be the decisive positive factor, given equal time constraints, with members of EXPERIMENTAL taking a similar DURATION on average to those in CONTROL before being satisfied with their answers, but with EXPERIMENTAL nevertheless performing significantly better in terms of CORRECTNESS.

[24] Any "reasonable DURATION" in this context has, broadly, two potential aspects: (a) the overall time allocated for the experiment session and (b) a possible self-imposed time limit for an individual task, given that participants were conscious that they did not have unlimited time to complete the tasks.

Self Assessment

In Section V-E, we observed after processing participants' survey responses that the participants whom we did not provide with semi-formal MLOps system diagrams slightly overestimated their performance, and conversely. We speculate that the additional semi-formal MLOps system diagrams reduced the CONFIDENCE of those participants whom we provided with them. Those participants possibly felt overwhelmed by the supplementary material or did not feel they had fully understood the semi-formal MLOps system diagrams. Again, this appears to be a counterintuitive result since we would expect these participants to be more confident than their counterparts, especially given that they had performed much better. We posit that this result is neutral in arguing for or against semi-formal MLOps system diagrams since the preference for practitioners to be overconfident, underconfident, or somewhere in between is more context-dependent, qualitative, and subjective than CORRECTNESS and DURATION. We consider this a neutral observation and result with no positive or negative impact or influence on the results of the hypothesis tests or our conclusions. Still, we nevertheless view it as a factor to bear in mind and deem our associated research question answered within the context of the study.

B. Threats to Validity

Various kinds of threats to validity must be considered when conducting controlled experiments. We follow the threat classification schemes for experiment validity described by Cook and Campbell [86] and Haynes et al. [87].

Threats to Internal Validity

Threats to internal validity encompass variables or circumstances that can potentially generate inaccurate conclusions regarding a cause-and-effect relationship. These elements serve as plausible sources of error that can compromise the capacity to deduce that alterations in the independent variable directly influenced changes in the dependent variable. Threats to internal validity encompass various factors, such as historical events (external influences), maturation (natural changes in subjects over time), or testing effects (how repeated testing impacts outcomes).

During the experiment sessions, no disturbances or interference occurred. We provided participants with an introduction and an opportunity to answer questions. No questions that affected the entire sessions arose, and we answered any of the participants' questions individually.

Due to the restricted time allocated to each session, the risk of maturation effects was limited, and indeed, we did not observe such effects. Owing to the experimental design, each participant contributed only once to the experiment results. Thus, learning between sessions could be ruled out. We consider any learning effect within a session across the three tasks to favour neither CONTROL nor EXPERIMENTAL. Each participant could score an equal number of bonus points for their course, regardless of group. This reward equality

precluded instrumental bias, and we prevented selection bias thanks to the random assignment of participants to groups.

There may have been potential for cross-contamination between experimental sessions and groups since preventing participants from discussing the experiment with future participants was impossible. Still, since we did not permit the participants to take a copy of the experiment sheet with them after their experiment sessions, we did not permit the use of electronic devices during the experiment; the systems and tasks were quite complex. We scheduled the sessions either consecutively or several weeks apart; we think it is unlikely that participants gained knowledge unfairly in advance of their sessions. Furthermore, any advantage would likely be roughly even across both groups due to the random group assignment, thus not favouring either group. As described in Section IV-C, the ban on electronic devices also ensured that participants could not consult external information sources. The only reference material allowed was the printed information document described in Section III-G so that potential effects of participants consulting other sources could be prevented. Lastly, based on the participant demographics described in Section III-F, we think it is unlikely that either group had an unfair advantage due to neither group dominating the most relevant demographics.

There is a potential risk to internal validity associated with the MLOps system architecture metamodel development and modelling process, and, as a consequence, the models we developed and the resulting diagrams we then generated from the models and provided to the participants. Although the authors are highly experienced in the applied modelling procedure, the potential for interpretive bias remains. Different researchers may have understood or modelled the MLOps systems differently, resulting in a dissimilar metamodel or system models. However, since the modelling involved developing metamodel constructs that accurately represent observed phenomena found within the informal system descriptions, and we successfully used them to model the selected systems, we do not think that this risk significantly impacts our study, even if the models created by other researchers may appear different.

The informal diagrams as source material may also threaten internal validity since we relied on their accuracy as source material for the study. Since textual descriptions accompanied the informal diagrams, we could disambiguate any inclarities in the original informal diagrams. The disambiguation process required a degree of subjectivity, and other authors may have had differing opinions on how to interpret some aspects of the original diagrams. However, since the UML-based diagrams generated from our models accurately reflected the informal descriptions, we anticipate that this phenomenon is insignificant regarding its impact on our methods and positive findings.

Threats to External Validity

Threats to external validity pertain to constraints on the extent to which study findings can be extended to contexts beyond the specific conditions in which the research was conducted. These threats to external validity raise questions about the broader applicability of the results to different populations, environments, or timeframes. Notable concerns regarding external validity include selection bias (where the sample may not accurately reflect the larger population) and interaction effects (indicating that the treatment's impact might differ across various groups).

An external validity concern regarding the applicability of our study's findings in practical settings arises from using students rather than non-student professionals. To address this concern, we took measures to educate the students about MLOps-related concepts in the experiment. Participants possessed varying levels of theoretical knowledge in software engineering, distributed systems, programming experience and industry experience, and we are not aware of factors that preclude a student population of this nature from being representative of a broader population of software developers. Considering the Stack Overflow industry survey mentioned in Section III-F in more detail, sixty-nine per cent of respondents to the Stack Overflow survey stated that they are at least partially self-taught, and only forty-five per cent of developers stated they have a bachelor's degree in computer science or a related field, with only thirteen per cent of respondents declaring that they have a master's degree and two per cent stating that they have a PhD. Thus, most developers participating in this industry survey of fifty thousand developers conducted by a well-respected software development portal were self-taught. Most did not even have a bachelor's degree, so these factors do not appear to be requisite for qualifying as a professional developer, further supporting our claim that students may serve as substitutes for developers in our study. Consequently, our findings may be extended to professional software developers, at least to some extent. However, to ascertain no significant differences in comparison with the population of professional developers, it would be necessary to replicate similar experiments with practitioners.

A further threat to external validity involves the risk that the tasks we designed may lack relevance to the complexities and nuances of industry practices. This risk introduces a potential limitation that could undermine the generalisability of the study's findings to real-world scenarios in the industry. We assert that this risk is possible but not likely since we are an experienced author team, and our perspectives in the study were based on findings in practitioner and scientific literature (see Section III-G).

Threats to Construct Validity

Threats to construct validity revolve around uncertainties regarding the degree to which the measurement and operationalisation of variables faithfully reflect the theoretical constructs under investigation. These threats to construct validity pertain to the suitability of the chosen measurement methods. They encompass issues such as insufficient operational definitions, instrumentation lacking the necessary sensitivity, or potential biases introduced by the experimenter.

As described in Section III-H, we considered CORRECTNESS and DURATION, which are two dependent variables customarily used when measuring understandability. Still, we cannot exclude the possibility that other metrics may be more appropriate for determining understandability. Similarly, more suitable methods may exist for assessing the participants'

confidence when addressing our research question.

When participants record times themselves, as in this study, it is vital to ensure that the measurements taken during the experiment accurately represent the time taken for them to complete the tasks. In this context, various threats to construct validity may arise.

One possibility is that "task completion" was ambiguous or imprecisely defined, leading to inconsistent timekeeping by individual participants across tasks and comparatively between participants. We mitigated this risk by instructing participants at the start of each experiment session on precisely when to start and stop recording task time.

Regarding the threat of self timing to reliability, we assessed the reliability of each recorded timestamp during the dataset preparation phase based on objective exclusion criteria as described in Section V-A. We determined, for each participant, whether their systematic recording of timestamps was consistent and realistic. If we determined that a participant's timekeeping was systematically inconsistent or unrealistic, then we excluded this participant's timestamps. For each timestamp, if its value was missing, incomplete or could not be inferred (e.g. due to illegible handwriting), then we excluded the time for the whole task. If an individual timestamp led to an implausible DURATION, we likewise excluded the time for the whole task. By following this methodology, we reduced the risk of unreliable timekeeping adversely affecting our conclusions.

Erroneous measurement accuracy may pose yet another threat to construct validity in the context of self timing. Due to participants being instructed to use a single, common clock under our control, with high readability due to being projected onto a large screen and an accuracy of seconds, we eliminated the risk of inaccurate timekeeping, e.g. due to participants having to start or stop a timer manually, misread the time due to poor readability, or enter in accurate times due to using different clocks.

A threat to construct validity arises through the potential for interpretation inconsistencies among participants when understanding UML diagrams, stemming from factors such as diverse interpretations of UML symbols, ambiguity in diagram elements, subjectivity in stereotype usage, varied understanding of relationships, and participants' prior experience with UML. To mitigate this threat, participants in EXPERIMENTAL received supplementary information, including a detailed description of how to interpret the UML-based MLOps system diagrams. Additionally, a pilot study was conducted, which did not highlight any ambiguities in the UML diagrams. The statistically significant results favouring the provision of UML diagrams suggest a consistent understanding among participants in EXPERIMENTAL, providing reassurance regarding the effectiveness of the mitigation strategies.

Threats to Content Validity

Content validity refers to the extent to which the experimental tasks, measurements, and materials used in the study accurately and adequately represent the research questions or objectives. In other words, it assesses whether the content of the experiment, including the variables, tasks, and instruments, is relevant and comprehensive enough to measure what the study aims to investigate.

We thoroughly considered the concept of content validity in our study design, evaluating the alignment between the experimental tasks, measurements, and materials with the research questions and objectives. In our meticulous planning, we scrutinised the relevance and comprehensiveness of the variables, tasks, and instruments employed in the experiment to ensure they effectively capture and measure the aspects under investigation. Consequently, after a comprehensive review, we couldn't identify any discernible threats to content validity, as our focus has consistently been on crafting an experimental framework that authentically addresses the problem statement and research objectives outlined in Section I.

Threats to Conclusion Validity

Threats to conclusion validity pertain to the extent to which the conclusions derived from a study align with the findings and can be considered well-founded. In essence, it scrutinises the accuracy of deducing a causal connection. Threats to conclusion validity may include insufficient statistical power, sampling discrepancies, or the excessive extrapolation of results.

Following the experiment sessions, a certain degree of inference and interpretation was necessary, mainly due to the pen-and-paper nature of the experiment, allowing handwritten, free-text responses to some survey and task questions rather than selection from pre-prepared responses. We deliberately made the pen-and-paper format a fundamental part of the experiment design: we did not want to provide pre-prepared answers to select from since we wanted participants to discover, e.g. component and pipeline names for themselves. Also, for some values, it simply made more sense to allow for free text. For instance, for the tasks involving identifying and counting element types, free text made more sense than providing a list of integers to choose from.
Any interpretations or inferences were as follows:

- We interpreted ranges for prior experience in the form of, e.g. "2-3", as "2.5".
- Some participants selected "none" for university education but wrote that they have an unspecified degree in "telecommunications engineering", "business analytics", "biomedical engineering" or "business administration". We recorded these students as having no prior university qualifications since we were only interested in computer science qualifications.
- If a participant left the university education field blank, we assumed they had no relevant university education.
- Some participants entered a non-zero value for the number of years of modelling experience but did not tick the box signifying prior knowledge of system modelling. These students may have understood these modelling activities as distinct.
- Since participants could enter free text, they often renamed components and pipelines ("elements") when completing the tasks. There were also slight name variations within the source material (both within and between

informal textual descriptions and informal graphical system diagrams). Indeed, this is one of the issues that arise with informal MLOps system descriptions and informal MLOps system diagrams. We used consistent, standardised naming in our UML-based MLOps system diagrams. Still, where names were vague or inconsistent in participants' answers, we inferred the referenced element when evident. We omitted the element from the answer for ambiguous, non-existent or unreadable element names. We also omitted catch-all answers (such as "all model-building components"). We corrected obvious spelling mistakes, applying this strategy consistently for both groups, thus eliminating the risk of bias.

An improvement to increase the construct validity of the study could be to survey the participants for their knowledge of MLOps-specific technologies. We used the same list of technologies as in our previous studies for consistency and potential comparison with our previous work.

VII. Conclusion

A. Impact and Relevance

We report on a controlled experiment with sixty-three participants on the understandability of MLOps system architecture descriptions. Our study focuses on whether providing semi-formal MLOps system diagrams supplementary to informal textual descriptions and informal graphical system diagrams affects the understanding of the described systems.

Our findings support our assumption that providing supplementary semi-formal MLOps system diagrams significantly aids understanding system descriptions. The respectable score for CORRECTNESS achieved by EXPERIMENTAL, which on average ($\overline{x}_{\text{EXPERIMENTAL}} = 0.6947$), was around double that of CONTROL ($\overline{x}_{\text{CONTROL}} = 0.3553$), whose members scored relatively poorly, provided supporting evidence for this assertion. In practical MLOps contexts, the implications are profound. The use of semi-formal MLOps system diagrams can significantly enhance task CORRECTNESS, which directly relates to operational efficiency and quality. Practitioners should consider integrating semi-formal diagrams into their documentation and communication practices. The significant difference in task CORRECTNESS in favour of providing semi-formal MLOps system diagrams suggests that researchers should consider the inclusion of semi-formal diagrams in their studies. This finding encourages further exploration into how visual aids, such as diagrams, can enhance performance in similar activities.

The outcome of our study indicates that, when provided supplementary semi-formal MLOps system diagrams, it does not take significantly longer to complete tasks of the nature of those in the study once the provided material has been read and understood. For practitioners, it is thus vital to remember that this additional information provided in the form of semi-formal MLOps system diagrams should not lead to the expectation of extra time being necessary when understanding a system.

We also found no correlation between task CORRECTNESS and task DURATION when no semi-formal MLOps system diagrams are provided. This signifies that the time invested in understanding MLOps system descriptions is not necessarily worthwhile if the representation of the system itself is not explicitly designed to enhance understanding. Practitioners should invest their time in developing and using semi-formal MLOps system diagrams of their MLOps systems since our results indicated a significant increase in CORRECTNESS and a measurable positive return in terms of CORRECTNESS for the time spent using our semi-formal MLOps system diagrams when understanding MLOps system descriptions. This insight is vital for project planning and resource allocation in real-world MLOps scenarios. The correlation between CORRECTNESS and DURATION in EXPERIMENTAL also highlights an exciting avenue for further research. Investigating the relationship between task CORRECTNESS and task DURATION with a focus on the impact of semi-formal diagrams can yield valuable insights into optimising MLOps system descriptions for better task outcomes.

We noted that members of CONTROL slightly overestimated their CORRECTNESS, whereas their counterparts slightly underestimated their CORRECTNESS. This serves to remind practitioners not to assume, given informal textual descriptions and informal graphical system diagrams, that they will sufficiently understand the described system. Conversely, practitioners faced with semi-formal MLOps system diagrams that are perhaps complex in appearance should not necessarily find them daunting. They should be confident that the information they seek lies within the semi-formal MLOps system diagrams. Observing CORRECTNESS estimation behaviour also highlights the importance of self assessment in practice. Practitioners should be aware of potential bias in their self assessments and adopt strategies for more accurate self evaluation.

The use of semi-formal diagrams can be considered a best practice in the description of MLOps systems in research studies. This approach aligns with the broader scientific principle of improving the replicability and reliability of experiments by providing a standardised visual framework.

We are unaware of previous empirical studies on how best to represent MLOps system architectures to foster understanding. The findings of this study are an indication that practitioners should feel confident in adopting a semi-formal MLOps graphical architecture view representation of their MLOps system architectures as a supplementary resource and that they should expect such diagrams to enhance understanding without the expectation of having to spend significantly more time gaining that understanding.

The results of our controlled experiment investigating the impact of using semi-formal MLOps system diagrams have several implications in education. The significant difference in task CORRECTNESS favouring the inclusion of semi-formal diagrams suggests that incorporating this visual aid in university courses can substantially enhance students' understanding of MLOps system architectures. Students exposed to semi-formal diagrams are more likely to grasp the concepts accurately and apply them effectively. The observation that participants not provided with semi-formal diagrams tended to overestimate their CORRECTNESS indicates that they may have misconceptions or gaps in their understanding. This underscores

the importance of semi-formal diagrams in preventing the overconfidence effect, which can hinder learning. By including these diagrams, instructors can ensure that students' self assessments are more closely with their actual knowledge. While there was no significant difference in task DURATION between CONTROL and EXPERIMENTAL, it is essential to recognise the potential trade-off between CORRECTNESS and task DURATION. Students who have access to semi-formal diagrams may invest more time in understanding the material and, in turn, achieve higher CORRECTNESS. Instructors should strike a balance in course design to optimise learning outcomes without overextending course duration or administrative overhead. Students have diverse learning styles and preferences. Since we observed a significant positive correlation between CORRECTNESS and DURATION for EXPERIMENTAL, educators need to emphasise practical, hands-on exercises where students can engage with semi-formal diagrams. This hands-on experience is crucial for reinforcing the understanding of MLOps system architectures. MLOps systems encompass both software engineering and machine learning, making them a suitable topic for interdisciplinary courses. Our findings highlight the value of incorporating visual aids to bridge the gap between these domains, enabling students from diverse backgrounds to collaborate effectively in real-world projects.

Our carefully designed and conducted empirical study builds on our previous work [3], [4] and contributes a solid foundation for further empirical work in the domain of architectural modelling of MLOps systems. The study also represents a positive initial step towards determining how helpful semi-formal MLOps system diagrams may be in understanding MLOps system architecture. Our results encourage ML practitioners to create and use semi-formal MLOps system diagrams of their MLOps system architectures.

This study helps advance the field of MLOps by promoting the use of semi-formal diagrams as a best practice for improving task CORRECTNESS. These findings stimulate further scientific investigation into the cognitive aspects of CORRECTNESS estimation. Additionally, our findings provide practical guidance for MLOps practitioners, emphasising the potential benefits of semi-formal diagrams in enhancing task CORRECTNESS and the importance of realistic self assessment.

Finally, our results have the potential to transform how software architecture is taught in university courses. By leveraging semi-formal MLOps system diagrams, educators can improve learning outcomes, address common misconceptions, cater to different learning styles, and foster a deeper understanding of complex architectural concepts. Our study also encourages a more holistic and interdisciplinary approach to teaching software architecture, aligning with the evolving demands of the industry.

B. Future Work

Our study examined whether providing semi-formal, UML-based MLOps system diagrams with informal textual descriptions and informal graphical system diagrams enhances the understandability of systems, and its results invite further investigation into the topic. There is much scope for further empirical evaluations within MLOps system architecture modelling, and this study serves as an appropriate starting point.

An area of potential interest, given that our UML-based MLOps system diagrams appear to facilitate understanding would be to compare different kinds of graphical architecture view representations, e.g. SysML[25], the C4 model[26] or a novel, custom diagram format to see if understanding in the context of MLOps can be improved further and discover what factors might help or hinder understanding. Alternatively, textual descriptions could be reconsidered, and a follow-up experiment could compare graphical architecture representations with varying textual architecture design descriptions as in [36].

Participants provided with UML-based MLOps system diagrams performed better without taking significantly longer. Still, the recorded times did not consider the effort and time involved in creating our MLOps system models and diagrams. A further study could investigate the trade-off between the time to develop semi-formal MLOps system models and diagrams and the improvement in understanding to determine whether the additional effort is worthwhile. We would assume that it is worth the effort since EXPERIMENTAL performed significantly better than CONTROL in terms of CORRECTNESS, and CORRECTNESS and DURATION were correlated in our study when we provided semi-formal, UML-based MLOps system diagrams.

Further to the previous idea concerning semi-formal MLOps system model and diagram creation, it would be interesting to compare how different subjects create MLOps system models based on various types of source media (such as informal text and diagrams) to measure consistency and identify features of informal representations that can lead to confusion and ambiguity, and further refine our MLOps system metamodel.

As discussed in III-F, students may represent professionals in empirical software engineering studies. Nevertheless, it would be interesting to conduct a study with software industry professionals of different backgrounds (e.g. software architecture, software engineering, and machine learning) and with varying levels of experience and perhaps compare them directly with students [31] to gain further knowledge of factors affecting understanding of MLOps system architecture descriptions.

A future study could focus on semi-formal MLOps system diagrams and utilise eye-tracking as in Shafari et al. [34], think-aloud protocols as in Heijstek et al. [35], and interviews or surveys to gain insight into which aspects of semi-formal MLOps system diagrams are most helpful, to refine further and improve them. A similar study could investigate how different levels of detail and abstraction influence the understandability of semi-formal MLOps system diagrams.

VIII. ACKNOWLEDGEMENTS

This work was supported by the FFG (Austrian Research Promotion Agency) projects AMMONIS (no. 879705) and MODIS (no. FO999895431).

[25]https://sysml.org
[26]https://c4model.com

REFERENCES

[1] 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.

[2] D. Kreuzberger, N. Kühl, and S. Hirschl, "Machine learning operations (MLOps): Overview, definition, and architecture," *ArXiv*, vol. abs/2205.02302, 2022.

[3] S. J. Warnett and U. Zdun, "Architectural design decisions for the machine learning workflow," *Computer*, vol. 55, no. 3, pp. 40–51, 2022.

[4] ——, "Architectural design decisions for machine learning deployment," in *2022 IEEE 19th International Conference on Software Architecture (ICSA)*. IEEE, 2022, pp. 90–100.

[5] D. Soni, R. L. Nord, and C. Hofmeister, "Software architecture in industrial applications," in *Proceedings of the 17th International Conference on Software Engineering*, ser. ICSE '95. New York, NY, USA: Association for Computing Machinery, 1995, p. 196–207. [Online]. Available: https://doi.org/10.1145/225014.225033

[6] F. Bachmann, J. Carriere, P. Clements, D. Garlan, J. Ivers, R. Nord, and R. Little, "Software architecture documentation in practice: Documenting architectural layers," 01 2000.

[7] A. Eden and R. Kazman, "Architecture, design, implementation." 01 2003, pp. 149–159.

[8] S. Cook, C. Bock, P. Rivett, T. Rutt, E. Seidewitz, B. Selic, and D. Tolbert, "Unified Modeling Language (UML) Version 2.5.1," Object Management Group (OMG), Standard, Dec. 2017. [Online]. Available: https://www.omg.org/spec/UML/2.5.1

[9] M. Völter, "Architecture as language," *Software, IEEE*, vol. 27, pp. 56–64, 03 2010.

[10] B. Rumpe and R. France, "Variability in UML language and semantics," pp. 439–440, 2011.

[11] J. Whittle, "Formal approaches to systems analysis using UML: an overview," *Advanced Topics in Database Research, Volume 1*, pp. 324–341, 2002.

[12] R. France, A. Evans, K. Lano, and B. Rumpe, "The UML as a formal modeling notation," *Computer Standards & Interfaces*, vol. 19, no. 7, pp. 325–334, 1998. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0920548998000208

[13] C. Czepa and U. Zdun, "How understandable are pattern-based behavioral constraints for novice software designers?" *ACM Transactions on Software Engineering and Methodology (TOSEM)*, vol. 28, pp. 1 – 38, 2019.

[14] ——, "On the understandability of temporal properties formalized in linear temporal logic, property specification patterns and event processing language," *IEEE Transactions on Software Engineering*, vol. 46, pp. 100–112, 2020.

[15] P. Paulweber, G. Simhandl, and U. Zdun, "On the understandability of language constructs to structure the state and behavior in abstract state machine specifications: A controlled experiment," *J. Syst. Softw.*, vol. 178, p. 110987, 2021.

[16] R. Solingen, V. Basili, G. Caldiera, and D. Rombach, *Goal Question Metric (GQM) Approach*, 01 2002.

[17] D. Sculley, G. Holt, D. Golovin, E. Davydov, T. Phillips, D. Ebner, V. Chaudhary, M. Young, J.-F. Crespo, and D. Dennison, "Hidden technical debt in machine learning systems," in *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 2*, ser. NIPS'15. Cambridge, MA, USA: MIT Press, 2015, p. 2503–2511.

[18] P. Kriens and T. Verbelen, "Software engineering practices for machine learning," *CoRR*, vol. abs/1906.10366, 2019. [Online]. Available: http://arxiv.org/abs/1906.10366

[19] S. Amershi, A. Begel, C. Bird, R. A. Deline, H. C. Gall, E. Kamar, N. Nagappan, B. Nushi, and T. Zimmermann, "Software engineering for machine learning: A case study," *2019 IEEE/ACM 41st International Conference on Software Engineering: Software Engineering in Practice (ICSE-SEIP)*, pp. 291–300, 2019. [Online]. Available: https://api.semanticscholar.org/CorpusID:88499204

[20] M. S. Rahman, E. Rivera, F. Khomh, Y. Guéhéneuc, and B. Lehnert, "Machine learning software engineering in practice: An industrial case study," *CoRR*, vol. abs/1906.07154, 2019. [Online]. Available: http://arxiv.org/abs/1906.07154

[21] Q. Yao, M. Wang, Y. Chen, W. Dai, Y.-F. Li, W.-W. Tu, Q. Yang, and Y. Yu, "Taking human out of learning applications: A survey on automated machine learning," 2019.

[22] M.-A. Zöller and M. F. Huber, "Benchmark and survey of automated machine learning frameworks," 2021.

[23] N. Hewage and D. Meedeniya, "Machine learning operations: A survey on MLOps tool support," 2022. [Online]. Available: https://arxiv.org/abs/2202.10169

[24] L. Bass, I. Weber, and L. Zhu, *DevOps: A Software Architect's Perspective*, 1st ed. Addison-Wesley Professional, 2015.

[25] J. Humble and D. Farley, *Continuous Delivery: Reliable Software Releases through Build, Test, and Deployment Automation*, 1st ed. Addison-Wesley Professional, 2010.

[26] U. Zdun, E. Ntentos, K. Plakidas, A. El Malki, D. Schall, and F. Li, "On the design and architecture of deployment pipelines in cloud- and service-based computing - a model-based qualitative study," 07 2019, pp. 141–145.

[27] M. Shahin, M. Zahedi, M. Ali Babar, and L. Zhu, "An empirical study of architecting for continuous delivery and deployment," *Empirical Software Engineering*, vol. 24, 08 2018.

[28] G. Schermann, J. Cito, P. Leitner, U. Zdun, and H. Gall, "An empirical study on principles and practices of continuous delivery and deployment," PeerJ Preprints, Tech. Rep., 2016.

[29] D. Tamburri, "Sustainable MLOps: Trends and challenges," 09 2020, pp. 17–23.

[30] A. Paleyes, R.-G. Urma, and N. D. Lawrence, "Challenges in deploying machine learning: A survey of case studies," *ACM Computing Surveys*, vol. 55, pp. 1 – 29, 2020. [Online]. Available: https://api.semanticscholar.org/CorpusID:227053929

[31] L. Allodi, M. Cremonini, F. Massacci, and W. Shim, "Measuring the accuracy of software vulnerability assessments: experiments with students and professionals," *Empirical Software Engineering*, vol. 25, 03 2020.

[32] L. Allodi, S. Biagioni, B. Crispo, K. Labunets, F. Massacci, and W. Santos, "Estimating the assessment difficulty of CVSS environmental metrics: An experiment," in *Future Data and Security Engineering*, T. K. Dang, R. Wagner, J. Küng, N. Thoai, M. Takizawa, and E. J. Neuhold, Eds. Cham: Springer International Publishing, 2017, pp. 23–39.

[33] K. Labunets, F. Paci, F. Massacci, and R. Ruprai, "An experiment on comparing textual vs. visual industrial methods for security risk assessment," 08 2014.

[34] Z. Sharafi, A. Marchetto, A. Susi, G. Antoniol, and Y.-G. Guéhéneuc, "An empirical study on the efficiency of graphical vs. textual representations in requirements comprehension," 05 2013, pp. 33–42.

[35] W. Heijstek, T. Kühne, and M. R. V. Chaudron, "Experimental analysis of textual and graphical representations for software architecture design," *2011 International Symposium on Empirical Software Engineering and Measurement*, pp. 167–176, 2011.

[36] R. Jolak, M. Savary-Leblanc, M. Dalibor, A. Wortmann, R. Hebig, J. Vincur, I. Polasek, X. Le Pallec, S. Gérard, and M. R. V. Chaudron, "Software engineering whispers: The effect of textual vs. graphical software design descriptions on software design communication," *Empirical Softw. Engg.*, vol. 25, no. 6, p. 4427–4471, nov 2020. [Online]. Available: https://doi.org/10.1007/s10664-020-09835-6

[37] M. Pradella, M. Rossi, and D. Mandrioli, "A UML-compatible formal language for system architecture description," vol. 3530, 06 2005, pp. 234–246.

[38] L. Lavazza, G. Quaroni, and M. Venturelli, "Combining UML and formal notations for modelling real-time systems," *ACM SIGSOFT Software Engineering Notes*, vol. 26, 12 2001.

[39] M. Rodano and K. Giammarco, "A formal method for evaluation of a modeled system architecture," *Procedia Computer Science*, vol. 20, 12 2013.

[40] J. li and J. Horgan, "Applying formal description techniques to software architectural design," *Computer Communications*, vol. 23, pp. 1169–1178, 07 2000.

[41] A. Jedlitschka, M. Ciolkowski, and D. Pfahl, *Reporting Experiments in Software Engineering*. London: Springer London, 2008, pp. 201–228. [Online]. Available: https://doi.org/10.1007/978-1-84800-044-5_8

[42] B. Kitchenham, S. Pfleeger, L. Pickard, P. Jones, D. Hoaglin, K. Emam, and J. Rosenberg, "Preliminary guidelines for empirical research in software engineering," *Software Engineering, IEEE Transactions on*, vol. 28, pp. 721– 734, 09 2002.

[43] C. Wohlin, P. Runeson, M. Höst, M. C. Ohlsson, B. Regnell, and A. Wesslén, *Experimentation in Software Engineering*. Springer Publishing Company, Incorporated, 2012.

[44] N. Juristo and A. Moreno, *Basics of Software Engineering Experimentation*, 01 2001.

[45] B. Kitchenham, L. Madeyski, D. Budgen, J. Keung, P. Brereton, S. Charters, S. Gibbs, and A. Pohthong, "Robust statistical methods for empirical software engineering," *Empirical Softw. Engg.*, vol. 22,

no. 2, p. 579–630, apr 2017. [Online]. Available: https://doi.org/10.1007/s10664-016-9437-5

[46] G. Charness, U. Gneezy, and M. A. Kuhn, "Experimental methods: Between-subject and within-subject design," *Journal of Economic Behavior & Organization*, vol. 81, no. 1, pp. 1–8, 2012. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167268111002289

[47] M. Höst, B. Regnell, and C. Wohlin, "Using students as subjects - a comparative study of students and professionals in lead-time impact assessment," *Empirical Software Engineering*, vol. 5, pp. 201–214, 11 2000.

[48] Runeson, Per, "Using Students as Experiment Subjects – An Analysis on Graduate and Freshmen Student Data," in *Proceedings 7th International Conference on Empirical Assessment & Evaluation in Software Engineering*, 2003, pp. 95–102.

[49] M. Svahnberg, A. Aurum, and C. Wohlin, "Using students as subjects - an empirical evaluation," in *Proceedings of the Second ACM-IEEE International Symposium on Empirical Software Engineering and Measurement*, ser. ESEM '08. New York, NY, USA: Association for Computing Machinery, 2008, p. 288–290. [Online]. Available: https://doi.org/10.1145/1414004.1414055

[50] I. Salman, A. T. Misirli, and N. J. Juzgado, "Are students representatives of professionals in software engineering experiments?" *2015 IEEE/ACM 37th IEEE International Conference on Software Engineering*, vol. 1, pp. 666–676, 2015.

[51] D. Falessi, N. Juristo, C. Wohlin, B. Turhan, J. Münch, A. Jedlitschka, and M. Oivo, "Empirical software engineering experts on the use of students and professionals in experiments," *Empirical Software Engineering*, vol. 23, 02 2018.

[52] S. Stevanetic, M. A. Javed, and U. Zdun, "Empirical evaluation of the understandability of architectural component diagrams," in *Proceedings of the WICSA 2014 Companion Volume*, ser. WICSA '14 Companion. New York, NY, USA: Association for Computing Machinery, 2014. [Online]. Available: https://doi.org/10.1145/2578128.2578230

[53] T. Haitzer and U. Zdun, "Controlled experiment on the supportive effect of architectural component diagrams for design understanding of novice architects," in *Proceedings of the 7th European Conference on Software Architecture*, ser. ECSA'13. Berlin, Heidelberg: Springer-Verlag, 2013, p. 54–71. [Online]. Available: https://doi.org/10.1007/978-3-642-39031-9_6

[54] L. V. Lakshmanan, M. Munn, and S. Robinson, *Machine Learning Design Patterns*. O'Reilly Media, Incorporated, 2020.

[55] M. Treveil, N. Omont, C. Stenac, K. Lefevre, D. Phan, J. Zentici, A. Lavoillotte, L. Heidmann, and M. Miyazaki, *Introducing MLOps: How to Scale Machine Learning in the Enterprise*. O'Reilly Media, Incorporated, 2020.

[56] J. Siegmund, C. Kästner, S. Apel, J. Liebig, M. Schulze, R. Dachselt, M. Papendieck, T. Leich, and G. Saake, "Do background colors improve program comprehension in the #ifdef hell?" *Empirical Software Engineering*, vol. 18, pp. 1–47, 08 2012.

[57] B. Hoisl, S. Sobernig, and M. Strembeck, "Comparing three notations for defining scenario-based model tests: A controlled experiment," *Proceedings - 2014 9th International Conference on the Quality of Information and Communications Technology, QUATIC 2014*, pp. 95–104, 12 2014.

[58] The Document Foundation, "Libreoffice." [Online]. Available: https://www.libreoffice.org

[59] Y. Shafranovich, "Common Format and MIME Type for Comma-Separated Values (CSV) Files," RFC 4180, Oct. 2005. [Online]. Available: https://www.rfc-editor.org/info/rfc4180

[60] R Core Team, *R: A Language and Environment for Statistical Computing*, R Foundation for Statistical Computing, Vienna, Austria, 2021. [Online]. Available: https://www.R-project.org/

[61] M. S. Handcock, I. E. Fellows, and K. J. Gile, *RDS: Respondent-Driven Sampling*, Los Angeles, CA, 2023, R package version 0.9-6. [Online]. Available: https://CRAN.R-project.org/package=RDS

[62] S. S. Shapiro and M. B. Wilk, "An analysis of variance test for normality (complete samples)," *Biometrika*, vol. 52, pp. 591–611, 1965.

[63] N. Mohd Razali and B. Yap, "Power comparisons of Shapiro-Wilk, Kolmogorov-Smirnov, Lilliefors and Anderson-Darling tests," *J. Stat. Model. Analytics*, vol. 2, 01 2011.

[64] T. W. Anderson and D. A. Darling, "A test of goodness of fit," *Journal of the American Statistical Association*, vol. 49, no. 268, pp. 765–769, 1954. [Online]. Available: https://www.tandfonline.com/doi/abs/10.1080/01621459.1954.10501232

[65] H. W. Lilliefors, "On the Kolmogorov-Smirnov test for normality with mean and variance unknown," *Journal of the American Statistical Association*, vol. 62, pp. 399–402, 1967.

[66] A. N. Kolmogorov, "Sulla determinazione empirica di una legge didistribuzione," *Giorn Dell'inst Ital Degli Att*, vol. 4, pp. 89–91, 1933.

[67] J. H. Bray and S. E. Maxwell, "Analyzing and interpreting significant MANOVAs," *Review of Educational Research*, vol. 52, no. 3, pp. 340–367, 1982. [Online]. Available: http://www.jstor.org/stable/1170422

[68] D. Bonett, "Robust confidence interval for a ratio of standard deviations," *Applied Psychological Measurement - APPL PSYCHOL MEAS*, vol. 30, pp. 432–439, 09 2006.

[69] H. Levene, "Robust tests for equality of variances," *Contributions to Probability and Statistics: Essays in Honor of Harold Hotelling*, p. 278–292, 1960.

[70] M. B. Brown and A. B. Forsythe, "Robust tests for the equality of variances," *Journal of the American Statistical Association*, vol. 69, no. 346, pp. 364–367, 1974. [Online]. Available: https://www.tandfonline.com/doi/abs/10.1080/01621459.1974.10482955

[71] W. H. Kruskal and W. A. Wallis, "Use of ranks in one-criterion variance analysis," *Journal of the American Statistical Association*, vol. 47, pp. 583–621, 1952.

[72] F. Wilcoxon, "Individual comparisons by ranking methods," *Biometrics Bulletin*, vol. 1, no. 6, pp. 80–83, 1945. [Online]. Available: http://www.jstor.org/stable/3001968

[73] D. W. Zimmerman, "Statistical significance levels of nonparametric tests biased by heterogeneous variances of treatment groups," *The Journal of General Psychology*, vol. 127, no. 4, pp. 354–364, 2000, pMID: 11109998. [Online]. Available: https://doi.org/10.1080/00221300009598589

[74] B. L. Welch, "The generalization of "Student's" problem when several different population variances are involved," *Biometrika*, vol. 34, no. 1-2, pp. 28–35, 01 1947. [Online]. Available: https://doi.org/10.1093/biomet/34.1-2.28

[75] N. Cliff, "Dominance statistics: Ordinal analyses to answer ordinal questions." *Psychological Bulletin*, vol. 114, pp. 494–509, 1993.

[76] H. D. Delaney and A. Vargha, "Comparing several robust tests of stochastic equality with ordinally scaled variables and small to moderate sized samples." *Psychological methods*, vol. 7 4, pp. 485–503, 2002.

[77] L. M. Hsu, "Biases of success rate differences shown in binomial effect size displays." *Psychological methods*, vol. 9 2, pp. 183–97, 2004.

[78] N. Cliff, "Answering ordinal questions with ordinal data using ordinal statistics," *Multivariate Behavioral Research*, vol. 31, pp. 331–350, 06 2010.

[79] Y. Benjamini and Y. Hochberg, "Controlling the false discovery rate - a practical and powerful approach to multiple testing," *J. Royal Statist. Soc., Series B*, vol. 57, pp. 289 – 300, 11 1995.

[80] C. Bonferroni, "Teoria statistica delle classi e calcolo delle probabilita," *Pubblicazioni del R Istituto Superiore di Scienze Economiche e Commerciali di Firenze*, vol. 8, pp. 3–62, 1936.

[81] O. J. Dunn, "Multiple comparisons among means," *Journal of the American Statistical Association*, vol. 56, pp. 52–64, 1961.

[82] E. Brunner and U. Munzel, "The nonparametric Behrens-Fisher problem: Asymptotic theory and a small-sample approximation," *Biometrical Journal*, vol. 42, pp. 17–25, 2000.

[83] K. Pearson, "Notes on regression and inheritance in the case of two parents," *Proceedings of the Royal Society of London*, vol. 58, pp. 240–242, 1895.

[84] M. G. Kendall, "A new measure of rank correlation," *Biometrika*, vol. 30, no. 1-2, pp. 81–93, 06 1938. [Online]. Available: https://doi.org/10.1093/biomet/30.1-2.81

[85] C. Spearman, "The proof and measurement of association between two things. By C. Spearman, 1904." *The American Journal of Psychology*, vol. 100 3-4, pp. 441–71, 1987.

[86] T. Cook and D. Campbell, *Quasi-experimentation: Design & Analysis Issues for Field Settings*. Houghton Mifflin, 1979. [Online]. Available: https://books.google.at/books?id=BFNqAAAAMAAJ

[87] S. N. Haynes, D. Richard, and E. S. Kubany, "Content validity in psychological assessment: A functional approach to concepts and methods." *Psychological assessment*, vol. 7, no. 3, p. 238, 1995.