Trust and Costs for Evolving Architectural Performance Models: A Survey

Technical Report

Martin Armbruster[1], Manar Mazkatli[1], and Anne Koziolek[1]

[1]KASTEL – Institute of Information Security and Dependability, Karlsruhe Institute of Technology, Germany

November 25, 2024

Contents

Abstract

Architectural performance models allow assessing a system's performance during different development phases by predicting performance characteristics. At the same time, by representing a software's architecture, they can support software developers in various scenarios. Due to time constraints and frequent changes, their application is hindered. Thus, researchers proposed approaches, which continuously update architectural performance models during the development. However, we do not know how software professionals perceive the trust and costs in such approaches and if they address current challenges in performance management. Therefore, we conducted a survey among software professionals to investigate these topics.

Based on 110 responses, our findings indicate that performance prediction approaches are less used and gain less trust than monitoring tools or performance tests. We also found out that certain properties can improve the trust. Additionally, the results suggest that software professionals consider different cost factors and are willing to spend some time when adopting a new tool.

Important note: This technical report reports on the current state of our work. Thus, it is not complete yet, and we mark gaps accordingly.

1. Introduction

Performance is an important quality attribute of software [13]. As a consequence, there are different techniques which enable the assessment of a system's performance. In particular, performance models, for example, simulate a system to predict performance characteristics [1]. Such simulations usually need no changes to the system, its configuration, or source code to explore design alternatives during software evolution. In addition, architectural performance models (PM) also represent a system's architecture, supporting various activities (e.g., evolution to prevent architectural decay [9]).

However, the required manual effort for co-evolving PMs along with the source code can hinder the adoption of PMs, as a result. Especially in agile software development processes with frequent changes of requirements and source code, this can further raise the barrier. Therefore, we proposed the Continuous Integration of architectural Performance Models (CIPM) approach [6]. It keeps an PM continuously consistent and evolving with the source code during the development. Based on monitoring data, the performance model is calibrated (i.e., its parameters are estimated). Afterward, a self-validation is performed to enable accurate performance predictions.

By providing these automated activities, including the calibration and self-validation, we want to increase the perceived trust in the evolving PMs and reduce the costs in manually updating them. Nevertheless, we do not know how software professionals perceive the trust and costs in the CIPM and similar approaches and if they address current challenges in performance management. As a consequence, we decided to conduct a survey among software professionals with the following objectives and research questions:

G1 *Understanding the Challenges in Performance Management:* First, we want to obtain insights into current practices and challenges in performance management. Based on these findings, we can get indications for future research directions and for aligning approaches with current challenges.

RQ1.1 How do software professionals currently manage performance?

RQ1.2 Which challenges do software professionals face in performance management?

G2 *Potential Trust in Approaches for Evolving PMs:* Next, we want to evaluate the potential trust in such approaches and compare it to the general perceived trust in performance management approaches as a baseline.

RQ2.1 How do software professionals perceive trust in the results of performance management approaches?

- RQ2.2 To what extent do software professionals perceive trust in the results of model-based performance predictions generated by an approach for evolving PMs?

G3 *Potential Costs for Adopting an Approach for Evolving PMs:* Regarding the adoption costs, we want to investigate influencing factors and get coarse-grained cost estimations by software professionals. Here, we express costs as the working time to adopt an approach for evolving PMs.

- RQ3.1 Which factors influence the decision process for adopting a new approach for evolving PMs?
- RQ3.2 How much working time are software professionals willing to spend for adopting a new approach for evolving PMs?

In the remainder of this report, we introduce the CIPM approach as an exemplar for approaches for evolving PMs in chapter 2. Then, we explain the survey design in chapter 3. Afterward, we present the results and their discussion in chapter 4 and chapter 5, respectively. The threats to validity follow in chapter 6 and related work in chapter 7. Finally, we summarize our findings and give an outlook to future work in chapter 8.

2. Foundation: The CIPM Approach

The CIPM approach by Mazkatli et al. [5] employs a Continuous Integration (CI) pipeline which extracts source code changes from version control systems. Based on these changes, it updates an intermediate model of the source code. Then, the execution of rules update an PM. These rules are derived from technologies applied in the source code and define how the PM should be kept consistent with the source code. As a result, the CIPM approach keeps the PM up-to-date and evolving with the source code during the development.

In a next step, the source code is monitored in a test or production environment to collect measurements. With a training set of these measurements, the PM is calibrated (i.e., its parameters are estimated). Afterward, a self-validation is performed by predicting the performance and comparing the results with a validation set from the measurements [7]. The comparison is based on statistical measures which express the accuracy of the performance prediction. If the prediction is accurate enough, the PM can be used to analyze the software system (e.g., explore design alternatives or investigate performance characteristics). Otherwise, the monitoring and calibration continues until the self-validation estimates the prediction as sufficiently accurate.

Following the CIPM approach [5], at least two conditions need to be fulfilled to adopt the CIPM approach to new applications. First, an intermediate model for every used programming language is required. Second, the rules for updating the PM are defined for the technologies applied in the new application's source code.

3. Survey Design

In this section, we present the survey. This includes its design process in section 3.1 and the actual content in section 3.2.

3.1. Design Process

For the design of the survey, we followed the guidelines by Linåker et al. [4] and Ralph et al. [8]. The latter one provides quality criteria and essential and desirable attributes in questionnaire surveys.[1] Linåker et al. describe a step-by-step process for designing and conducting surveys in software engineering [4].

Based on the objectives and research questions introduced in chapter 1, we defined the target population as all people from industry involved in software development. This covers different roles and positions in professional software development, which allows us to get insights from different perspectives. For example, project managers can consider different cost factors compared to software developers. Moreover, there are other roles than a software developer (e.g., scrum masters or software architects) involved in software development and performance management.

With this target population, we chose a non-probabilistic sampling method in the form of a mostly accidental sampling and small extension to snowballing sampling. Concretely, we performed two recruitment rounds.

For the first round, we invited personal contacts, people via email lists, and people via posts on a business social media platform.[2] Based on the email lists' sizes and unique post views, we determined that we approached approximately more than 2,500 people. In the invitation, we asked the recipients to distribute the survey if they knew suited potential participants. Since we cannot assess how often the survey was distributed, we cannot further estimate the number of approached people and the response rate. We run this first round from February 28th, 2024, until May 18th, 2024.

For the second round, we used a survey platform for recruitment.[3] There, people participated on a first come-first serve basis and were paid for their participation. Overall, 20 people participated on October 2nd, 2024, and 52 people on November 1st, 2024.

[1] https://www2.sigsoft.org/EmpiricalStandards/docs/standards?standard=QuestionnaireSurveys
[2] LinkedIn: https://linkedin.com
[3] Prolific: https://www.prolific.com/

Section / Questions	Research Question
D	-
C1 - C8	RQ1.1
C9	RQ2.1
Ch	RQ1.2
N	RQ2.2
Co1	RQ3.1
Co2 - Co3	RQ3.2

Table 3.1.: Mapping of sections or questions to related research questions.

For the questionnaire design process, two authors of this paper with a background in performance engineering and software development formed a team and designed a first version of a web-based questionnaire. Then, the initial version was reviewed by two researchers with expertise in survey design. After a refinement, we conducted a pretest with three researchers. Based on their feedback, we refined and finalized the design.

3.2. Questionnaire

The survey consists of mostly closed-ended or hybrid (i.e., closed-ended questions with an option to enter free text if neither of the predefined options apply) questions and only a few open-ended questions to increase the likeliness of participation. In total, there are 28 questions for the first round and 31 questions for the second round. Certain questions are only asked if a participant gives specific answers, serving as conditions. As a result, they get 20 to 23 questions, which take 10 to 15 minutes to answer. Before a person could start the survey, they were informed about the survey and data processing and requested to give their consent. Without a consent, they could not participate.

We divided the survey into five sections to cover our areas of interest: *demographic information* (D), *contextual information* (C), *challenges* (Ch), *new tool* (N), and *costs* (Co). The mapping between these sections and related research questions is listed in Table 3.1. In the following, we omit a few questions because of space limitations. The full questionnaire can be found online.[4] All questions are encoded in the format $X0$ where X is the abbreviation of a section and 0 denotes the question number within the section. Additionally, certain predefined answers are encoded in the format $X0z$ where XO equals the question code for which the answer is predefined and z is a lowercase letter identifying the answer within the question. These codes are aligned with our internal codes.

The *demographic information* targets specific attributes of the participants (e.g., their age, professional experience (D3), or roles in the company (D4)) to understand their composition.

Similar to the demographic information, the *contextual information* covers general questions regarding the current performance management practices (RQ1.1). Therefore, we ask what

[4] https://doi.org/10.5281/zenodo.12743765

techniques are employed to manage performance (C3). Here, we differentiate between monitoring tools (C3a), performance tests (C3b), and model-based performance predictions (C3c) providing the following definitions to the participants.

- *Monitoring tools* take measurements of a software during its runtime in the production environment and allow analyzing the software's performance based on these measurements [13].
- *Performance tests* execute a part or the complete software with different workloads to measure the software's performance [13].
- *Model-based performance predictions* employ a performance model (e.g., a Petri net or UML) which represents a software and describes how resources are used [1, 13]. Based on such a performance model, different techniques predict the performance characteristics of the software.

Besides, we are interested in why a certain technique is not used (C4/C5/C6). This question is only asked for a technique if the technique is not used. Additionally, there are questions for the purposes of the performance management (C7), and its relevance (C8). The contextual information also contains a question (C9) about the perceived general trust in the results of the different performance management techniques (RQ2.1) to avoid influences from the *new tool* section as outlined below.

Next, the *challenges* section includes two open-ended questions to gather challenges the participants face (Ch2) and missing features in performance management tools (Ch3) (RQ1.2). Moreover, we want to know how satisfied the participants are with certain quality attributes of performance tools (Ch1) to get an indication for potential issues. The last question (Ch4) deals with desired features in performance prediction tools to compare the results with approaches in evolving PMs.

With the *new tool* section, we begin a second part of the survey by introducing the concept of the CIPM approach as a potential new tool for model-based performance predictions. Thus, we added the question about the general trust in the contextual information to avoid influences by the introduction of the CIPM approach. In the new tool section, we extend upon the general trust question with an additional question (N2). This new question regards the perceived trust in performance predictions when they have the following properties of the CIPM approach (RQ2.2): metrics about the accuracy of the prediction results are available (N2a), the performance model is calibrated based on monitoring data (N2b), and the prediction results are validated with monitoring data (N2c). Based on the responses for the question, we want to analyze if the properties of the CIPM approach can improve the perception of trust in the results of performance predictions.

At last, the questions about the *costs* support us in identifying factors that are considered when introducing a new performance prediction tool (Co1) (RQ3.1). Moreover, further questions allow estimating the effort that the participants are willing to invest in the setup, learning, and adoption of a new tool (Co2/Co3) (RQ3.2). We express the effort as work time. In addition, the questions are split between project managers (question codes with a trailing *P*) and non-project managers (question codes with a trailing *D*) since these groups can have different perspectives on the costs. Therefore, both groups receive question texts formulated for their perspective. We differentiate the participants based on their roles: participants which included project manager as one role are considered as project managers, while the remaining participants are considered as non-project managers.

4. Results

In the following, we describe the calculated statistical measures shortly in section 4.1 and report the results of the survey in section 4.2.

4.1. Statistical Measures

Based on the objectives and research questions, we designed a descriptive and exploratory survey [4]. Therefore, we use descriptive statistics for the analysis. For all close-ended and hybrid questions, we calculate the frequency of responses for each predefined option. Additionally, we report the 0%, 25%, 50% (median), 75%, and 100% quantiles for every matrix question which we handle as ordinal data. For the open-ended questions, we follow a qualitative inductive approach [10] to analyze the answers. At last, we provide a replication package with all anonymized single responses and analysis scripts.[4]

4.2. Results

In the following, we report the results of the survey. Maximum values in certain tables are given in bold italic with an arrow pointing towards the value.

Overall, 121 participations were registered. During the first round, 16 participants completed all answers, and 22 answers were incomplete. In case of four responses, the survey was started, but no answer was given. In the second round, there were six dropouts and one duplicated participation, which we filtered out. The reported results refer to the remaining 110 participations.

4.2.1. Demography

The participants are between 18 and 65 years old (n=108) with the most ones (44.4 %) being between 26 and 35 followed by 27.8 % between 36 and 45. They have less than one year professional experience up to more than 10 years. In addition, the company and team sizes vary.

Regarding the roles displayed in Table 4.1, most people work as software developers (58.6 %) followed by software architects (15.0 %) and project managers (12.1 %). Beside the predefined options, 10 additional roles were entered as "Other".

Role	Frequency (n=140)
Software Developer	82 (58.6 %)
Software Architect	21 (15.0 %)
Project Manager	17 (12.1 %)
Product Owner	6 (4.3 %)
Scrum Master	3 (2.1 %)
QA/Test Engineer	11 (7.9 %)

Table 4.1.: Frequencies of responses for each role (D4). 10 additional roles are omitted.

Reason	Monitoring Tools (C4)	Performance Tests (C5)	Performance Predictions (C6)
Not required	***12 (25.5 %)*** ←	8 (12.0 %)	26 (16.6 %)
Time constraints	7 (14.9 %)	***16 (23.9 %)*** ←	22 (14.0 %)
No expertise in team	9 (19.2 %)	10 (14.93 %)	***34 (21.7 %)*** ←
Never used before	7 (14.9 %)	14 (20.9 %)	25 (15.9 %)
No tool support	1 (2.1 %)	2 (3.0 %)	1 (0.6 %)
Limited tool support	4 (8.5 %)	5 (7.5 %)	15 (9.6 %)
Inaccurate results	2 (4.3 %)	3 (4.5 %)	8 (5.1 %)
High costs	5 (10.6 %)	7 (10.5 %)	15 (9.6 %)
I do not know what monitoring tools / performance tests / performance predictions are	0 (0 %)	2 (3.0 %)	11 (7.0 %)

Table 4.2.: Frequencies of responses for the reasons why a specific performance management technique is not used. The arrow marks the maximum value in a column.

4.2.2. Context: Current Performance Management (RQ1.1)

Regarding the current performance management techniques, 72 participants (49.7 % of the respondents) answered that they employ monitoring tools, followed by 63 participants (43.5 %) with performance tests. Only 10 people (6.9 %) utilize model-based performance predictions. There are varying reasons why a technique is not used as shown in Table 4.2. For the monitoring tools, 25.5 % of the respondents report that they are not required. In comparison, 12.0 % stated this reason for performance tests. In addition, 23.9 % responded with time constraints. In the case of the performance predictions, all reasons except "No tool support" were selected at least eight times. Most people (21.7 %) have no expertise in the team.

In Table 4.3, the frequency of responses for each purpose is listed. Except the fulfillment of regulations with 11 responses (3.5 %), every purpose has at least 32 responses up to 66 responses (20.7 %) for "analyze performance behavior". The relevance (Figure 4.1) of performance management is perceived as important (the median) in both companies and teams.

Purpose	Frequency
Improve performance	63 (19.8 %)
Analyze performance behavior	***66 (20.7 %)*** ←
Observe performance over time	55 (17.2 %)
Evaluate performance for design alternatives	42 (13.2 %)
Find performance regressions	32 (10.0 %)
Find performance bottlenecks	50 (15.7 %)
Fulfill regulations	11 (3.5 %)

Table 4.3.: Frequency of responses for the purposes of the performance management (C7). The arrow marks the maximum.

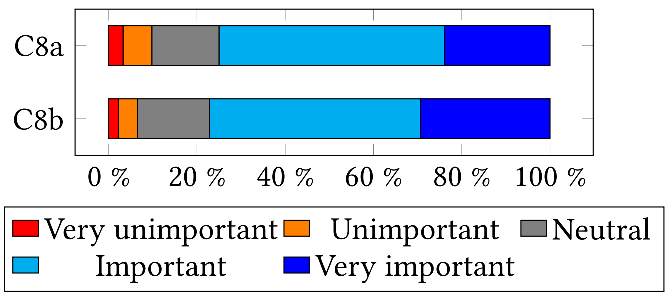

Figure 4.1.: Frequencies of responses (n = 92) for the relevance of the performance management in the company (C8a) or team (C8b).

4.2.3. Challenges (RQ1.2)

Figure 4.2 shows the satisfaction of the participants with specific quality attributes of performance management tools and tests. With accuracy and reliability, no participant is unsatisfied. Both options have a median of satisfied. User-friendliness, performance, and effort to apply received a few unsatisfied responses. The median for user-friendliness and effort to apply is neutral while the median for performance is between neutral and satisfied.

For the free text questions about challenges (Ch2) and missing features (Ch3), we received responses, which we have not analyzed yet.

In Figure 4.3, we depict the results for missing features in model-based performance prediction tools. All predefined responses obtained at least 60 % positive agreement. While there was a slight disagreement, only visualization received strong disagreement.

4.2.4. Trust (RQ2.1, RQ2.2)

Figure 4.4 shows the results for the trust-related questions. At first, the perceived general trust differs slightly between the performance management techniques. While monitoring tools and performance tests have mostly positive responses with a median of agree, performance predictions received more mixed responses with a median of slightly agree. In combination,

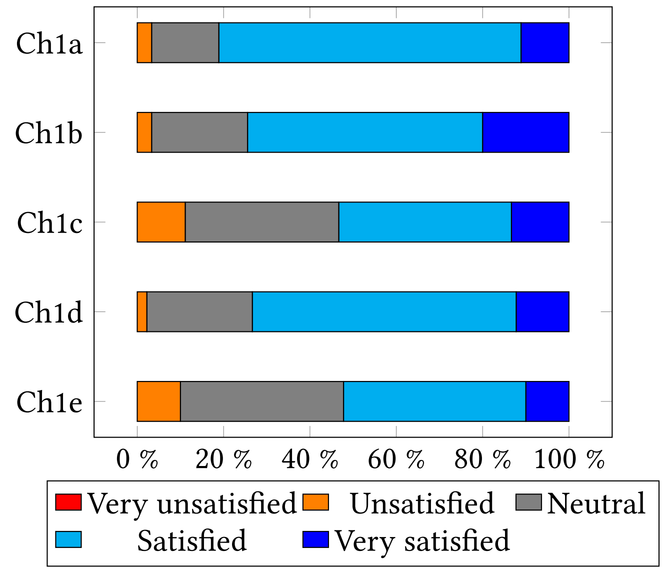

Figure 4.2.: Frequencies of responses (n = 90) for the satisfaction with specific quality attributes of performance tools and tests: accuracy (Ch1a), reliability (Ch1b), user-friendliness (Ch1c), performance (Ch1d), and effort to apply (Ch1e).

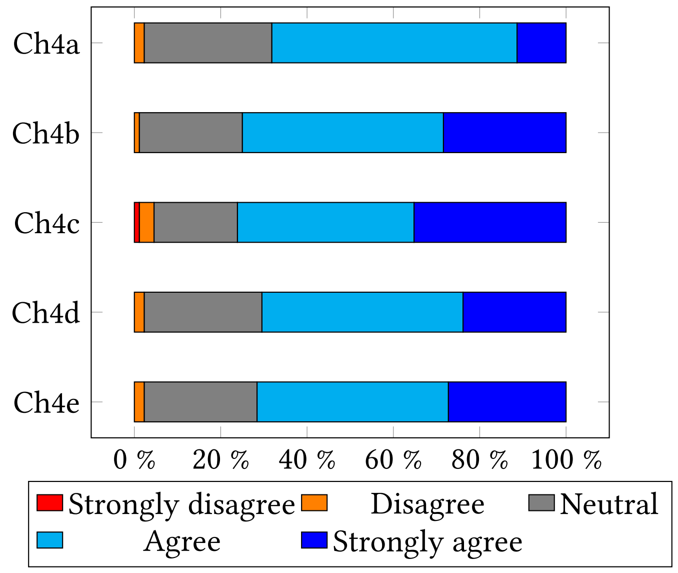

Figure 4.3.: Frequencies of responses (n = 88) for missing features in performance prediction tools: support for design decisions (Ch4a), statements about prediction accuracy (Ch4b), visualization (Ch4c), support for software architecture and architectural design (Ch4d), and fast feedback (Ch4e).

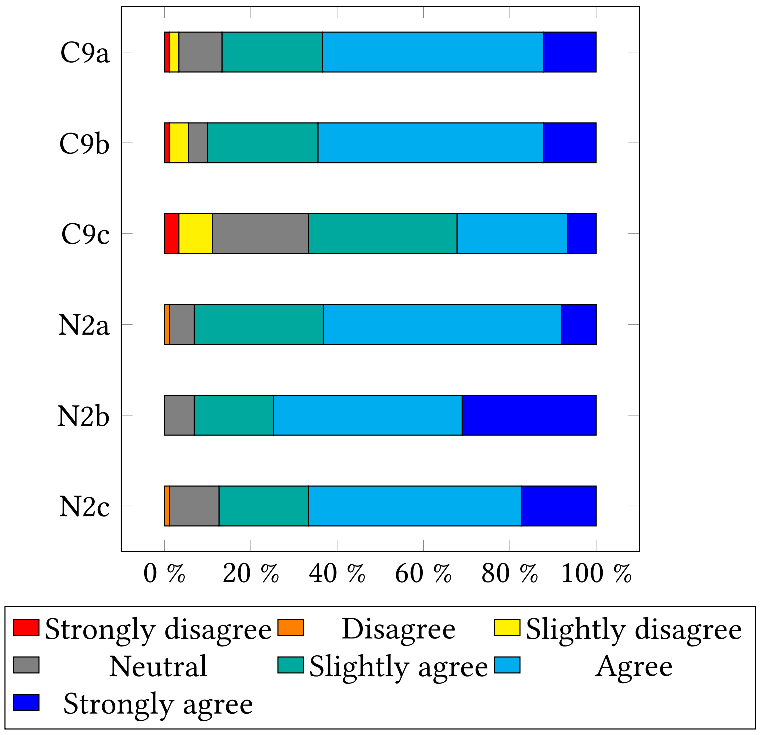

Figure 4.4.: Frequencies of the responses for the trust-related questions (C9/N2) based on 90 responses for C9 and 87 responses for N2.

11.1 % expressed their disagreement with performance predictions. For monitoring tools, it is 3.3 %, and for performance tests, it is 5.6 %.

Regarding the perceived trust in performance predictions with the presented properties, the responses are mostly positive. If metrics about the accuracy are available, 1.2 % disagree, and 93.1 % slightly agree, agree, or strongly agree. Similarly, if the performance model is calibrated with monitoring data, 1.2 % disagree, and 87.4 % slightly agree, agree, or strongly agree. At last, if the prediction results are validated with monitoring data, there is no disagreement, and 93.1 % slightly agree, agree, or strongly agree. The median in all three cases is at agree.

4.2.5. Costs (RQ3.1, RQ3.2)

In Table 4.4, the different cost factors and their frequency of responses are shown. For non-project managers, all factors were mentioned at least 20 times. The highest frequency of responses received benefits and quality (n=55) followed by easy setup (n=52). In contrast, project managers selected all factors except benefits and quality (n = 12), license costs (n = 8), and possibilities for extensions (n = 6) 11 times.

Figure 4.5 and Figure 4.6 depict the results regarding the time participants are willing to spend on learning, setting up, and adopting a new tool. The most non-project managers (n = 29) want to spend between two and three working days to learn and set up a new tool, while the second most selected response (n = 22) ranges between four and five working days. Four

Factor	**Non-Project Managers (Co1D)**	**Project Managers (Co1P)**
Benefits and Quality	***55 (16.2 %)*** ←	***12 (14.8 %)*** ←
Easy setup	52 (15.3 %)	11 (13.6 %)
Learning curve	46 (13.5 %)	11 (13.6 %)
Effort to adapt the tool for the team	44 (12.9 %)	11 (13.6 %)
License costs	46 (13.5 %)	8 (9.9 %)
Maintenance costs	40 (11.8 %)	11 (13.6 %)
Setup costs	37 (10.9 %)	11 (13.6 %)
Possibilities for extensions to the tool	20 (5.9 %)	6 (7.4 %)

Table 4.4.: Frequencies of responses for the cost factors (Co1) sorted by their total frequency (Co1D+Co1P). The arrow marks the maximum.

project managers are willing to spend between two and three working days, four and five working days, and six and ten working days each. Regarding the adoption of a new tool for the employed programming languages and technologies, five project managers are willing to spend between one and two weeks. For non-project managers, 27 participants stated one to two weeks, followed by 15 participants for less than one week.

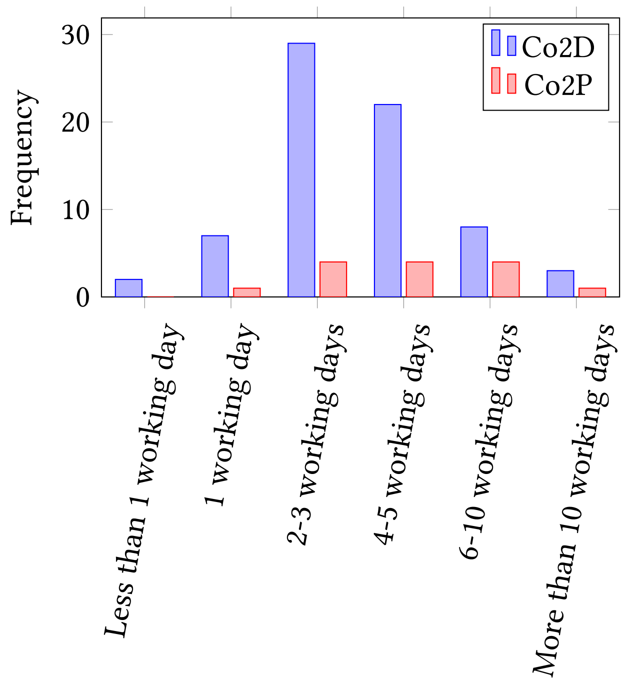

Figure 4.5.: Frequencies of responses for the willingness to spend time to set up and learn a new tool (Co2).

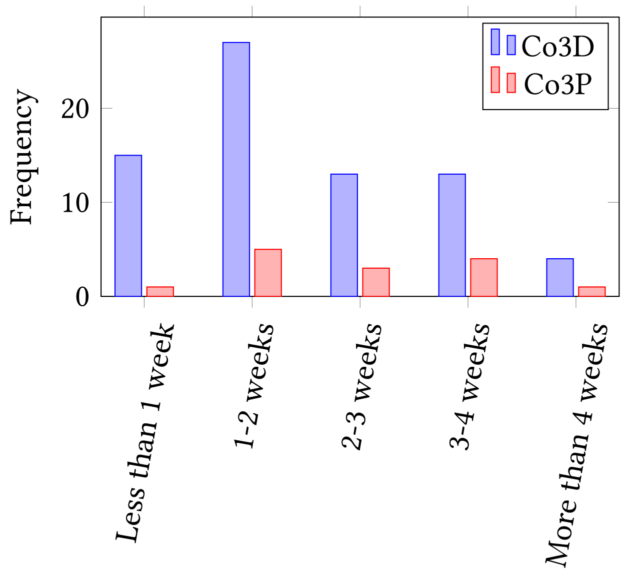

Figure 4.6.: Frequencies of responses for the willingness to spend time to adopt a new tool to employed programming languages and technologies (Co3).

5. Discussion

In the following, we discuss the results to answer the research questions.

5.1. Context: Current Performance Management (RQ1.1)

Based on the presented results for the current performance management, we can directly answer RQ1.1.

Answer for RQ1.1: The results indicate that software professionals mostly use monitoring tools and performance tests for various purposes regarding the important quality attribute performance.

5.2. Challenges (RQ1.2)

We omitted the discussion here since we did not analyze the responses yet.

5.3. Trust (RQ2.1, RQ2.2)

Considering the general perceived trust in the performance management techniques, we observe that the participated software professionals have less trust in model-based performance predictions compared to monitoring tools and performance tests which answers RQ2.1. Nevertheless, the responses for the performance predictions range from strongly disagree to strongly agree so that there are software professionals which have more or less trust.

By comparing the general perceived trust in performance predictions with the perceived trust in performance predictions with the explicit properties of the CIPM approach, we perceive an improvement in the trust. The calibration based on monitoring data has the most potential to increase the trust in performance predictions as no one disagrees and it has the most participants who strongly agree. Then, the validation with monitoring data and the available accuracy metrics follow. For both options, there is a slight improvement in the perceived trust since there are fewer disagreements. In summary, to answer RQ2.2, we conclude that all three properties of the CIPM approach can improve the trust in performance predictions and that their combination should be able to increase the trust, too.

Answers for RQ2.1 and RQ2.2: Software professionals have less trust in performance predictions compared to monitoring tools and performance tests. However, accuracy metrics, calibration with monitoring data, and validation with monitoring data can improve the trust in performance predictions.

Here, an in-depth discussion of the results will follow.

5.4. Costs (RQ3.1, RQ3.2)

Answer for RQ3.1: All factors we provided can influence the decision process for adopting a new approach for evolving PMs. The extent to which a factor contributes varies.

Answer for RQ3.2: In case of the time the participants are willing to spend on learning and setting up a new tool, there is a trend for two to five working days. For the adoption towards the employed programming languages and technologies, the time ranges between one and two weeks.

Here, an in-depth discussion of the results will follow.

6. Threats to Validity

In this section, we discuss the threats to the following validity dimensions as described by Wohlin et al. [12]: *construct validity*, *internal validity*, *conclusion validity*, and *external validity*.

Construct validity expresses how well suited and designed the questionnaire is to answer the research questions [12]. Different aspects of the questionnaire, for instance, the order of questions, their wording, or the wording of predefined responses, could guide the participants' answers towards our expectations or could not provide adequate measurements for answering the research questions. Thus, we followed the guidelines by Linåker et al. [4] and Ralph et al. [8] as described in chapter 3 to mitigate these threats. In particular, the guidelines include a design process with which the questionnaire was carefully and systematically created. Its structure consists of two parts: a general one about current practices and challenges in performance management, followed by a specific one about approaches for evolving PMs. This separation and order should avoid influences of the specific part on the general part. By applying established question types and scales, such as the Likert scale, we could obtain appropriate metrics. Additionally, we conducted expert reviews and a pretest to validate and refine the questionnaire. Moreover, we used a web-based tool for the questionnaire so that every participant had an equal presentation of the questionnaire. This allowed us to limit the direct contact to participants to invitation mails and posts.

Internal validity ensures that the survey results reflect the participants' general point of view which can be negatively or positively influenced by different factors [12]. Here, the survey design is also one factor, which we mitigated by the systematic design process as outlined in the threats for construct validity. Another factor is maturation (i.e., the software professionals' experience can impact their responses). As mentioned in section 4.2, there is a variation in the professional experience and demographics so that maturation effects should be mitigated. Certain factors potentially influencing the internal validity (e.g., repeated participations or disturbances during a participation) are partly out of our control. The payment of participants is another factor which we still need to discuss.

Conclusion validity refers to the ability to draw the correct conclusions from the survey results [12]. It relies on the execution of the survey, the measurements, and the usage of adequate statistics. As outlined in the threats for construct validity, we followed guidelines for the survey design and used a web-based questionnaire so that the execution should be the same for every participant. In addition, the measurements are based on established question types and scales. Therefore, we applied corresponding descriptive statistics such as quantiles in our analysis.

External validity deals with the generalization of the results [12]. As presented in section 4.2, the participants have diverse backgrounds with different ages, years of experience, company sizes, and team sizes. Due to the small sample size and included personal contacts, the sample is not representative limiting the generalizability of the results. While they serve as an indicator, further surveys need to be conducted to generalize the results.

7. Related Work

Bezemer et al. [2] conducted a survey among software professionals to investigate the performance management in DevOps. Based on 26 responses, they found out that 88 % of the participants do not use performance models, although there is an interest in applying them. As a consequence, they conclude that performance management techniques need to be lightweight and to smoothly integrate into DevOps in order to lower the barrier for adoption. These results are in line with our findings. In addition, we asked for the challenges and perceived trust in performance management techniques in general.

In a survey by Waseem et al. [11], they asked software professionals about the design, monitoring, and testing of Microservice-based applications. Based on 106 responses in a questionnaire and six interviewees, some of their findings indicate that software professionals mostly use informal or semiformal approaches to describe Microservice architectures. The use of formal approaches is hindered by several reasons (e.g., required learning effort). Moreover, the most important quality attributes include security, availability, performance, and scalability. Waseem et al. also asked about different aspects of monitoring Microservices. The responses imply a wide usage of monitoring tools. While this survey focuses on Microservice applications and our survey on performance management in general, certain results are similar. Our findings indicate that monitoring tools are widely employed and performance is an important quality attribute, too.

8. Conclusion

In this paper, we presented and discussed the results of a survey we conducted to understand the challenges in performance management and to observe the perceived trust and costs for approaches for evolving PMs. Based on 110 participations, our findings indicate that monitoring tools and performance tests are commonly used and gain more trust in general compared to model-based performance predictions. However, the properties of the example CIPM approach (accuracy metrics, calibration based on monitoring data, validation with monitoring data) can improve the perceived trust in performance predictions. Different cost factors are considered when adopting a new performance prediction tool. In addition, software professionals are willing to spend some time on adopting such a new tool.

Acknowledgements

This work was supported by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) - SFB 1608 - 501798263. This publication is also based on the project SofDCar (19S21002) that the German Federal Ministry for Economic Affairs and Climate Action funds. Additionally, this work was supported by the pilot program Core Informatics at KIT (KiKIT) of the Helmholtz Association (HGF).

Bibliography

[1] Simonetta Balsamo et al. “Model-Based Performance Prediction in Software Development: A Survey”. In: *IEEE Transactions on Software Engineering* 30.5 (May 2004), pp. 295–310.

[2] Cor-Paul Bezemer et al. “How is Performance Addressed in DevOps?” In: *Proceedings of the 2019 ACM/SPEC International Conference on Performance Engineering*. ICPE ’19. Mumbai, India: Association for Computing Machinery, 2019, pp. 45–50. ISBN: 9781450362399. DOI: 10.1145/3297663.3309672. URL: https://doi.org/10.1145/3297663.3309672.

[3] L. Giamattei et al. “Monitoring tools for DevOps and microservices: A systematic grey literature review”. In: *Journal of Systems and Software* 208 (2024), p. 111906. ISSN: 0164-1212. DOI: https://doi.org/10.1016/j.jss.2023.111906. URL: https://www.sciencedirect.com/science/article/pii/S0164121223003011.

[4] Johan Linåker et al. “Guidelines for Conducting Surveys in Software Engineering.” In: *ELLIIT: the Linköping-Lund initiative on IT and mobile communication* (2015). URL: https://lucris.lub.lu.se/ws/portalfiles/portal/6062997/5463412.pdf.

[5] Manar Mazkatli et al. *Continuous Integration of Architectural Performance Models with Parametric Dependencies – The CIPM Approach*. Tech. rep. 46.23.03; LK 01. 2022. 28 pp. DOI: 10.5445/IR/1000151086/v2.

[6] Manar Mazkatli et al. “Incremental Calibration of Architectural Performance Models with Parametric Dependencies”. In: *IEEE International Conference on Software Architecture (ICSA 2020)*. Salvador, Brazil, 2020, pp. 23–34. DOI: 10.1109/ICSA47634.2020.00011.

[7] David Monschein et al. “Enabling Consistency between Software Artefacts for Software Adaption and Evolution”. In: *2021 IEEE 18th International Conference on Software Architecture (ICSA)*. 2021, pp. 1–12. DOI: 10.1109/ICSA51549.2021.00009.

[8] Paul Ralph et al. *Empirical Standards for Software Engineering Research*. 2021. arXiv: 2010.03525 [cs.SE]. URL: https://arxiv.org/abs/2010.03525v2.

[9] Mehwish Riaz, Muhammad Sulayman, and Husnain Naqvi. “Architectural Decay during Continuous Software Evolution and Impact of ‘Design for Change’ on Software Architecture”. In: *Advances in Software Engineering*. Ed. by Dominik Ślęzak et al. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, pp. 119–126. ISBN: 978-3-642-10619-4.

[10] David R. Thomas. “A General Inductive Approach for Analyzing Qualitative Evaluation Data”. In: *American Journal of Evaluation* 27.2 (2006), pp. 237–246. DOI: 10.1177/1098214005283748. eprint: https://doi.org/10.1177/1098214005283748. URL: https://doi.org/10.1177/1098214005283748.

[11] Muhammad Waseem et al. “Design, monitoring, and testing of microservices systems: The practitioners’ perspective”. In: *Journal of Systems and Software* 182 (2021), p. 111061. ISSN: 0164-1212. DOI: https://doi.org/10.1016/j.jss.2021.111061. URL: https://www.sciencedirect.com/science/article/pii/S0164121221001588.

[12] Claes Wohlin et al. *Experimentation in Software Engineering*. Berlin Heidelberg, Germany: Springer, 2012. ISBN: 978-3-642-29043-5. DOI: 10.1007/978-3-642-29044-2.

[13] Murray Woodside, Greg Franks, and Dorina C. Petriu. “The Future of Software Performance Engineering”. In: *2007 Future of Software Engineering*. FOSE ’07. USA: IEEE Computer Society, 2007, pp. 171–187. ISBN: 0769528295. DOI: 10.1109/FOSE.2007.32. URL: https://doi.org/10.1109/FOSE.2007.32.

A. Survey

Welcome Page

Welcome to this survey about performance management. It aims to gather information about the current state of performance management for future research and research directions.

We invite you to participate if you are working in the industry, are involved in software development, and at least 18 years old. The participation is voluntarily, and we treat all data confidental.

The survey takes approximately 15-20 minutes to complete with approximately 20 questions. It consists of two parts of which the following first part is about the current state of performance management.

We appreciate your participation and thank you in advance.

For any matter, you can contact us: [contact details omitted].

Demographic Information

From which country do you come? (Single choice)

[Provided list of all countries to select from]

How old are you? (Single choice)

- 18-25 years old
- 26-35 years old
- 36-45 years old
- 46-55 years old
- 56-65 years old
- Over 65 years old

How much professional experience do you have? (Single choice)

Help text: Professional experience means all time spans in which you worked for a company or organization in the context of software development (i.e., also in different roles).

- Less than 1 year
- 1-5 years
- 6-10 years
- More than 10 years

What are your roles in the company? (Multiple choice, *Q-Role*)

- Software Developer
- Software Architect
- Project Manager
- Product Owner
- Scrum Master
- QA/Test Engineer
- Other (Free text)

How large is your company? (Single choice)

- Up to 9 employees
- 10-100 employees
- 101-250 employees
- 250-10,000 employees
- More than 10,000 employees

How large is your team? (Single choice)

- Up to 5 people
- 6-10 people
- More than 10 people

Contextual Information

Which of the following software development methods are commonly used in your team? (Multiple choice)

- Sequential development (e.g., Waterfall)
- Agile development (e.g., Scrum, XP)
- DevOps
- Other (Free text)

Which technologies and methods do you use for development? (Multiple choice)

- Cloud Computing (e.g., AWS, Azure)
- Microservices
- Container (e.g., Docker, Kubernetes)
- Big Data (e.g., Apache Hadoop)
- Technologies for CI/CD
- Databases
- Other (Free text)

How do you currently manage performance in your team? (Multiple choice, *Q-PManagement*)

Monitoring tools take measurements of a software during its runtime in the production environment and allow to analyze the software's performance based on these measurements.

Performance tests execute a part or the complete software with different workloads to measure the software's performance.

Model-based performance predictions employ a performance model (e.g., a Petri net or UML) which represents a software and describes how resources are used. Based on such a performance model, different techniques predict the performance characteristics of the software.

- By using monitoring tools
- By using performance tests
- By using model-based performance predictions
- Other (Free text)

[Only shown if monitoring tools are NOT selected in *Q-PManagement.*]

Which reasons hinder you in using monitoring tools? (Multiple choice)

- Not required
- Time constraints

- No expertise in team
- Never used before
- No tool support
- Limited tool support
- Inaccurate results
- High costs
- I don’t know what monitoring tools are
- Other (Free text)

[Only shown if monitoring tools are selected in *Q-PManagement*.]

Which monitoring tools do you currently use? (Multiple choice, [3])

- Dynatrace
- Prometheus
- Akamai mPulse
- DataDog
- GrayLog
- pyroscope
- Other (Free text)

[Only shown if performance tests are NOT selected in *Q-PManagement*.]

Which reasons hinder you in using performance tests? (Multiple choice)

- Not required
- Time constraints
- No expertise in team
- Never used before
- No tool support
- Limited tool support
- Inaccurate results
- High costs
- I don’t know what performance tests are
- Other (Free text)

[Only shown if performance predictions are NOT selected in *Q-PManagement.*]

Which reasons hinder you in using model-based performance predictions? (Multiple choice)

- Not required
- Time constraints
- No expertise in team
- Never used before
- No tool support
- Limited tool support
- Inaccurate results
- High costs
- I don't know what model-based performance predictions are
- Other (Free text)

[Only shown if performance predictions are selected in *Q-PManagement.*]

Which tools or techniques do you currently use for performance predictions? (Multiple choice)

- Analytical models (e.g., Petri nets)
- Architecture-based models (e.g., UML or UML Profiles)
- Machine Learning
- Spreadsheet
- Other (Free text)

For which purposes do you use the performance management tools or tests? (Multiple choice)

Help text: Performance management tools include monitoring tools and model-based performance prediction tools.

- Improve performance
- Analyze performance behavior
- Observe performance over time
- Evaluate performance for design alternatives
- Find performance regressions
- Find performance bottlenecks

- Fulfill regulations
- Other (Free text)

How important is the performance management? Please indicate on a scale from very unimportant to very important. (Matrix)

Scale: Very unimportant, Unimportant, Neutral, Important, Very important

- Performance management is important in our company.
- Performance management is important in our team.

Do you agree to the following statements? Please indicate on a scale from strongly disagree to strongly agree. (Matrix)

Scale: Strongly disagree, Disagree, Slightly disagree, Neutral, Slightly agree, Agree, Strongly agree

- I would trust monitoring data from monitoring tools.
- I would trust the results of performance tests.
- I would trust the results of model-based performance predictions.

Challenges

How satisfied are you with the following quality attributes of the performance management tools that you use? Please indicate on a scale from very unsatisfied to very satisfied. (Matrix)

Scale: Very unsatisfied, Unsatisfied, Neutral, Satisfied, Very satisfied

- Accuracy
- Reliability
- User-friendliness
- Performance
- Effort to apply

Are there any issues or challenges when you use the performance management tools and/or tests? If so, please name them. (Free text)

Do you miss features in performance management tools and/or tests? If so, please name them. (Free text)

Which of the following features do you want to have in a model-based performance prediction tool? Please indicate on a scale from strongly disagree to strongly agree. (Matrix)

Scale: Strongly disagree, Disagree, Neutral, Agree, Strongly agree

- Support for design decisions (e.g., deployment or system composition)
- Statements about the prediction accuracy
- Visualization
- Support for the software architecture and architectural design
- Fast feedback

New Tool

Introductory text: Welcome to the second part. It consists of four questions about a new model-based performance prediction tool. This tool extracts a performance model from code and monitors the code in a test or production environment. Based on a training set from these measurements, the performance model is calibrated (i.e., its parameters are estimated). Afterwards, the tool performs a self-validation by predicting the performance and comparing the results with a validation set from the measurements. The comparison is based on statistical measures which express the accuracy of the performance prediction. If the prediction is accurate enough, the model can be used to analyze the software system. Otherwise, the monitoring and calibration continues until the self-validation estimates the prediction as sufficiently accurate.

Do you agree to the following statements? Please indicate on a scale from strongly disagree to strongly agree. (Matrix)

Scale: Strongly disagree, Disagree, Slightly disagree, Neutral, Slightly agree, Agree, Strongly agree

- I would trust the results of model-based performance predictions when metrics about the accuracy of the performance prediction are available..
- I would trust the results of model-based performance predictions when the calibration of the performance model is based on monitoring data.
- I would trust the results of model-based performance predictions when the prediction results are validated with monitoring data.

Costs (Not project manager)

[Only shown if project manager is NOT selected as role in *Q-Role.*]

What factors would you consider when adopting a new performance prediction tool? (Multiple choice)

- Easy setup

- Learning curve
- Effort to adapt the tool for the team
- Possibilities for extensions to the tool
- License costs
- Setup costs
- Maintenance costs
- Benefits and Quality
- Other (Free text)

How much time would you be willing to invest in setting up and learning a new performance prediction tool? (Single choice)

- Less than 1 working day
- 1 working day
- 2-3 working days
- 4-5 working days
- 6-10 working days
- More than 10 working days

If a new performance prediction tool requires adoption to your specific technologies and programming languages, how much time would you be willing to invest in this adoption? (Single choice)

- Less than 1 week
- 1-2 weeks
- 2-3 weeks
- 3-4 weeks
- More than 4 weeks

Costs (Project manager)

[Only shown if role in company (*Q-Role*) includes project manager]

What factors would you consider when introducing a new performance prediction tool? (Multiple choice)

- Easy setup
- Learning curve
- Effort to adapt the tool for the team
- Possibilities for extensions to the tool
- License costs
- Setup costs
- Maintenance costs
- Benefits and Quality
- Other (Free text)

How much overall time would you be willing that people of your team invest in setting up and learning a new performance prediction tool? (Single choice)

- Less than 1 working day
- 1 working day
- 2-3 working days
- 4-5 working days
- 6-10 working days
- More than 10 working days

If a new performance prediction tool requires adoption to your team's specific technologies and programming languages, how much overall time would you be willing that people of your team invest in this adoption? (Single choice)

- Less than 1 week
- 1-2 weeks
- 2-3 weeks
- 3-4 weeks
- More than 4 weeks