# Advancing Software Monitoring: An Industry Survey on ML-Driven Alert Management Strategies

Hrusto, Adha; Runeson, Per; Engström, Emelie; Ohlsson, Magnus C

*Published in:*
50th Euromicro Conference Series on Software Engineering and Advanced Applications (SEAA) 2024

2024

*Document Version:*
Early version, also known as pre-print

# Advancing Software Monitoring: An Industry Survey on ML-Driven Alert Management Strategies

Adha Hrusto*
*Lund University, Department of Computer Science*
*System Verification Sweden AB*
*adha.hrusto@cs.lth.se*

Per Runeson
*Lund University, Department of Computer Science*
*per.runeson@cs.lth.se*

Emelie Engström
*Lund University, Department of Computer Science*
*emelie.engstrom@cs.lth.se*

Magnus C Ohlsson
*System Verification Sweden AB*
*magnus.c.ohlsson@systemverification.com*

***Abstract*—With the dynamic nature of modern software development and operations environments and the increasing complexity of cloud-based software systems, traditional monitoring practices are often insufficient to timely identify and handle unexpected operational failures. To address these challenges, this paper presents the findings from a quantitative industry survey focused on the application of Machine Learning (ML) to enhance software monitoring and alert management strategies. The survey targets industry professionals, aiming to understand the current challenges and future trends in ML-driven software monitoring. We analyze 25 responses from 11 different software companies to conclude if and how ML is being integrated into their monitoring systems. Key findings revealed a growing but still limited reliance on ML to intelligently filter raw monitoring data, prioritize issues, and respond to system alerts, thereby improving operational efficiency and system reliability. The paper also discusses the barriers to adopting ML-based solutions and provides insights into the future direction of software monitoring.**

***Index Terms*—monitoring, alert management, anomaly detection, machine learning**

## I. Introduction

Large-scale cloud computing systems can fail in various unpredictable ways, with faults cascading across components, leading to service outages or degraded performance. Understanding how these systems behave under failure conditions is crucial for planning effective failure management and prevention strategies [1]. Effective monitoring plays a vital role in detecting these operational failures. Software development companies rely heavily on monitoring tools [2] that collect and analyze vast amounts of operational data. However, the large volume and complexity of monitoring data pose challenges in identifying meaningful patterns and extracting actionable insights. Traditional monitoring systems often struggle to effectively manage data overload, leading to alert fatigue and the potential overlooking of critical alerts. Moreover, there is a lack of standardization and an overabundance of different monitoring tools, which leads to inconsistent and ineffective monitoring practices [3].

To address these challenges, Machine Learning (ML) techniques have emerged as a promising approach to enhance the analysis and management of monitoring data [4], [5]. ML can significantly improve system observability and enable extensive system performance analysis to identify anomalies that could indicate a potential failure. However, despite significant research and commercial solutions for advanced monitoring using ML, many organizations have yet to utilize these capabilities fully [6]. Current monitoring methods primarily depend on alert thresholding for key performance indicators (KPIs) or log querying, with limited use of ML-based solutions due to uncertainties about their usefulness, reliability, and cost-effectiveness. [6].

To advance understanding of the aforementioned challenges in monitoring cloud computing systems and adopting ML-based approaches, our industry-academia collaboration research team conducted a national quantitative survey among industry practitioners. We targeted companies responsible for developing, testing, and operating cloud-based software systems. By engaging with practitioners who are directly involved in the lifecycle of such systems, we obtained a comprehensive understanding of the current monitoring practices, their limitations, and the potential impact of incorporating advanced ML techniques for more efficient monitoring strategies. More details about the survey setup can be found in Section III.

Despite the growing interest in ML technologies and their potential applications [1], [7], there is a scarcity of comprehensive survey studies specifically focusing on how industries are implementing ML for monitoring purposes and detecting operational failures. Most recent and relevant surveys [3], [6] provide valuable insights, but each from unique and differing viewpoints regarding cloud monitoring practices and the adoption of machine learning in the industry, particularly for software failure prediction. Thus, there is a need for more focused research in this area that integrates both perspectives, to understand the benefits, practical challenges, and the extent of ML adoption in industrial monitoring and proactive alert management. Therefore, this study aims to contribute to this area by providing empirical insights into the current state of ML usage in industrial monitoring and exploring the factors influencing its adoption and effectiveness for early detection of operational failures.

## II. Background and Related Work

Recent studies have significantly advanced software monitoring, particularly through ML-driven approaches, highlighting the challenges and innovations in this area. The review by Giamattei et al. [2] explores a variety of monitoring tools for large-scale systems like microservices, noting challenges in their selection and usage. Research by Candido et al. [8] addresses complexities in log data and introduces AIOps for improving operational workflows. *The IntelligentMonitor* study [9] discusses an adaptive system that reduces data overload and alert fatigue through ML, enhancing monitoring efficiency. Additionally, Gill and Hevary [10] identify major challenges in cloud monitoring, including issues in technology, virtualization, and performance, emphasizing the need for innovative solutions.

According to the recent survey by Tamburri et al., [3], it is evident that monitoring practices are crucial for detecting operational failures. Additionally, monitoring should be recognized as a strategic asset for improving system observability [3]. There are examples of successful intentions to address the early detection of failures by leveraging data accessible through monitoring tools. Mariani et al. [5] introduce PreMiSE, a method that predicts failures in multi-tier distributed systems. This approach, tested on a telecommunication system prototype, showcases high precision in failure prediction with minimal false positives. Similarly, Cotroneo et al. [1] propose a method for analyzing failure data in cloud systems, leveraging Deep Embedded Clustering to classify failures efficiently without manual feature engineering.

However, despite the availability of numerous monitoring tools, Tamburri et al. [3] find that adopting advanced monitoring technology in the industry is still in its early stages due to required substantial investments and lack of industry standards. To investigate this further, our survey study aims to understand to what extent the companies leverage monitoring tools and data to detect operational failures and perform root-cause analysis.

Hrusto et al. have undertaken two case studies in collaboration with industrial partners [11]–[13]. They reveal current alert management practices and the limitations of existing monitoring solutions. Interviews and observations in the case studies highlight specific challenges, such as undetected operational failures, alert flooding, difficulty in interpreting alerts, and the need for more efficient alert mechanisms, such as autonomous monitors [13]. To address these, they developed and evaluated a cloud-based solution for monitoring, detecting anomalies, and reporting interpretable alerts.

Additionally, it is crucial to understand the integration of AI and ML in the industry, as these technologies play a significant role in enhancing productivity and decision-making. Surveys by Rana et al. [6] and Holmström [14] offer insights into the factors influencing AI/ML adoption. Our study adds a practical perspective by evaluating the real-world applicability of AI/ML in software development, helping to bridge the gap between theoretical frameworks and industry implementation.

## III. Research methodology

We conduct an industrial survey study following the recommendations and guidelines from three key publications on designing and conducting surveys in software engineering authored by Molléri et al. [15], Kasunic [16], and Linåker et al. [17]. The survey process involves several key steps [15], detailed in subsequent subsections, including reflections on threats to validity important for ensuring the reliability and credibility of research findings.

### A. Research objective

The main objective of this survey study is to describe the current challenges of handling operational failures and relevant monitoring data. By surveying a sample of the large population of software development companies in Sweden, we aim to report the frequency, detection mechanisms, and types of operational failures they encounter, while considering the monitoring data that could be used for in-depth understanding and designing prevention mechanisms. Collected survey data are used to reason about the benefits and limitations of a ML-based solution [13] for anomaly detection and reporting smart alerts. In this way, we may reason about its wider applicability within the software industry. Additionally, we describe the effect of AI/ML solutions on developing, testing, and operating large-scale software systems deployed in the cloud. This will include an in-depth analysis of the organizations' capabilities to adopt and rely on AI/ML-based tools.

### B. Research questions

We defined the research questions based on a synthesis of the authors' industry experiences and the latest contributions in the field, as outlined in Section II. This approach ensured that our questions were grounded in both practical insights and aligned with the latest research. We defined the following research questions:

- **RQ1:** How do different types of operational failures impact software development environments, considering their frequency and consequences?
- **RQ2:** To what extent are monitoring data and its specific types used for detecting and analyzing operational failures?
- **RQ3:** Are there recognized needs for more advanced detection (alert) mechanisms in operations based on state-of-the-art machine learning approaches?
- **RQ4:** What is the current level of readiness and attitude of software development, testing, and operations teams towards adopting and relying on ML-based solutions, given the latest advancements in AI?

### C. Defining and sampling the population

We target the population of Swedish software companies that develop and operate software systems deployed in the cloud. We refer to them as *software development companies*, and their names are not disclosed at their request to remain

anonymous. The target audience consists of intended respondents from these companies, to whom we refer as *software practitioners*.

The first and fourth authors are employed at a global software quality assurance (GSQA) company with headquarters in Sweden that offers a wide range of services within the Software Development Life Cycle (SDLC). The company has a significantly large network of loyal customers across Sweden who share a passion for high-quality software products and processes, which we aim to examine in our study. We used a non-probabilistic sample from this customer network, combining convenience and purposive sampling as discussed by Baltes and Ralph [18]. To minimize biases, we expand the sample with the authors' LinkedIn connections, specifically software practitioners working in companies with similar preferences towards software quality.

### D. Designing and validating the instrument

We used a structured questionnaire as a survey instrument for data collection. The questionnaire was carefully designed to include closed-ended questions, which offered respondents both single- and multiple-response options. This format was chosen to simplify the response process and to ensure consistency and comparability in the collected data. We utilized a specialized survey design tool provided by our university to develop the questionnaire, ensuring a rigorous and systematic approach to data collection. We formulated questions with predefined categorical responses specifically designed based on the authors' experiences and considering definitions of quality characteristics standardized by ISO[1].

The questionnaire form, now openly available, begins with an overview section that includes the project title, the research team involved, the purpose of the survey, and information on data privacy. This part sets the context for the respondents and assures them of the confidentiality of their responses. The section of the questionnaire is divided into distinct sections:

- Respondent Profile: This section comprises six questions designed to gather respondents' demographic and professional background information.
- Operational Failures and Monitoring Data Usage: It consists of nine questions, targeting RQ1 and RQ2 to collect respondents' experiences and perceptions regarding operational failures and monitoring practices in their respective environments.
- ML for Smart Monitoring & Alerts: This segment includes eight questions focused on using ML in smart monitoring and alert management (RQ3).
- Adoption of ML-based Solutions: The final section, with seven questions, explores the attitudes and readiness of respondents towards adopting ML-based solutions in their work processes (RQ4).

To ensure the construct validity of our survey, we leveraged the substantial expertise within our team of authors to validate the defined constructs and their alignment with the survey questions. The initial questionnaire design was undertaken by one author, who possesses in-depth knowledge in this domain. Subsequently, three other authors, each bringing their academic and industrial insights and perspectives, conducted a thorough review of the questionnaire. During this review process, several concerns were identified, leading to a collaborative discussion among the authors. These discussions guided the resolution of identified issues, thereby refining and validating the constructs and questions. In this way, driven by the collective expertise of our team, we ensured that the survey accurately reflects the constructs it intends to measure.

[1]https://iso25000.com/index.php/en/iso-25000-standards/iso-25010

### E. Managing participants and responses

In managing participant engagement and responses, our approach involved the distribution of the online questionnaire through three primary digital channels: email, Microsoft Teams, and Viva Engage. To effectively reach potential respondents, we shared the survey URL along with a brief context description, explaining the purpose of the study and its significance. We directly reached out to 30 potential respondents through our professional network, including consultants at GSQA company and contacts on LinkedIn, and relied on them to further distribute the questionnaire among their respective teams and companies. Throughout the survey period of four weeks, we actively monitored the response rate to gauge participant involvement. Based on these observations, we regularly reached out to all previously contacted individuals to check on the status of their participation, as well as to express our gratitude and encourage their ongoing involvement, acknowledging their valuable contributions. This strategy of regular communication and appreciation played a crucial role in maintaining a good response rate.

### F. Analyzing and reporting results

We initially exported the results into an Excel file to analyze and report our survey results. We manually inspected the file and determined the most efficient approach for detailed analysis. To facilitate this, we divided the results into five separate CSV files corresponding to the respondent profile and the four research questions. Each question and its corresponding answers were coded systematically (e.g., F1 for a question and F1A1 for its answer related to operational failures), streamlining the process of data handling. This structured coding system enabled us to efficiently write Python code snippets for loading, analyzing, and extracting relevant information that directly addressed our constructs and survey objectives. For comprehensive transparency and reference, the auto-generated report from the survey tool, encompassing all the detailed results, has been included in the appendix of our study (see Appendix A).

### G. Threats to validity

Analyzing potential threats to validity before conducting a survey is crucial for ensuring accuracy in measuring intended constructs, improving overall quality, saving time and resources by resolving issues early, and enhancing the credibility

of the survey's findings [15]. We addressed the following threats to validity:

- **Internal Validity** – We ensured through internal reviews that questions were designed to minimize misunderstanding or ambiguity and that the survey was distributed to a representative sample of the target population.
- **Construct Validity** – Leveraging our expertise, we precisely defined each construct while also considering the definitions of quality characteristics provided by the ISO standard, wherever applicable.
- **External Validity** – The selected sample was non-probabilistic. Therefore, we acknowledge that it may limit the generalizability of our findings to the broader population. However, we aimed to include a diverse range of participants from various software engineering subfields to capture the population's broader properties.
- **Conclusion Validity** – We ensured this by employing a systematic approach that combined manual data inspection with automated analysis using a Python script. This approach enabled us to thoroughly examine the data while efficiently extracting information relevant to our research questions.

## IV. Results

This section presents the outcomes of our quantitative survey, structured to address each of our four research questions in Section III-B. The results are analyzed based on 25 responses across 11 different software development companies, out of 30 invitations. We examined the data from both the company and software practitioner perspectives.

### A. Respondent profiles

The respondent profiles include a diverse spectrum of positions primarily in the software industry. Quality assurance is the most prevalent role within the target audience (10/25 from 8/11 companies), but DevOps engineers (6/25 from 6/11 companies) and software developers (5/25 from 4/11 companies) are also dominant positions. These practitioners are sourced from a diverse range of companies, where the number of companies corresponding to each position follows a similar distribution. In terms of professional experience, there's a wide range, from those with less than a year to those with over a decade in their field. The experience with AI tools varies, with a significant number of respondents having engaged with AI either a few times or moderately, indicating a growing interest in using AI to enhance their working processes.

### B. RQ1: Operational failures

The main objective of this research question was to investigate one of the critical aspects of software development related to the types, frequency, and consequences of operational failures. The results showed that the frequency of operational failures varies across companies, reflecting the differing resilience and vulnerability of systems in the industry. With seven companies experiencing weekly failures and three of them encountering them monthly, it's evident that operational failures are a common and recurrent challenge. Interestingly, respondents in different roles within the same company often reported different frequencies of operational failures, indicating role-specific challenges and perspectives on issues. To address this discrepancy, we considered the worst-case scenario as the reference. This means that when respondents from a single company reported different frequencies of failures, we considered the more frequent occurrence as the baseline. In this way, we focused on addressing the most significant and recurrent operational failures.

The reported methods for detecting operational failures show the industry's reliance on technology and human oversight. Companies are increasingly integrating technology-driven methods, such as automated alert systems, used by all eleven surveyed companies, with human-centric approaches like manual monitoring and user reports, employed by eight companies each. Interestingly, some companies employ a mix of these methods to create a more robust and comprehensive detection strategy. These blended strategies for detecting operational failures highlight the need for more comprehensive monitoring and detection mechanisms.

Continuing the analysis, the collected data from the surveyed companies revealed that the most frequent problems arise from the interaction between different software components. This was reported by nine companies, indicating the importance of the integration or compatibility issues. Following closely are performance issues, signalizing the difficulties the companies face in maintaining optimal system performance and response times. System outages or crashes also represent a major concern, as they were experienced by six companies. This demonstrated an urgent need for robust infrastructure and proactive maintenance to minimize downtime. Although data-related issues and security vulnerabilities are less frequent among operational failures, their relevance is still highly important. Data corruption or loss can have severe consequences for data integrity and reliability, while even a single security incident can cause data breaches, financial losses, and damage to a company's reputation. Considering the broad range of potential failures, their early identification is crucial for maintaining overall system health.

An additional overview of operational failures per company is given in Figure 1. It becomes evident that certain operational failures are more prevalent in specific companies, indicating potentially unique challenges or vulnerabilities within their operational environments. For instance, the results suggest that many companies should direct their improvement efforts to address performance issues.

Complementing these findings, the results also indicate consequences of operational failures, shown in Figure 2. It was evident that several companies experienced a range of failure consequences simultaneously. Among the closely distributed four responses (blue, green, purple, and yellow), the highest rate of companies, 9/11 (81.8%), reported increased development time and costs due to operational failures, highlighting a burden on resource allocation and project timelines.

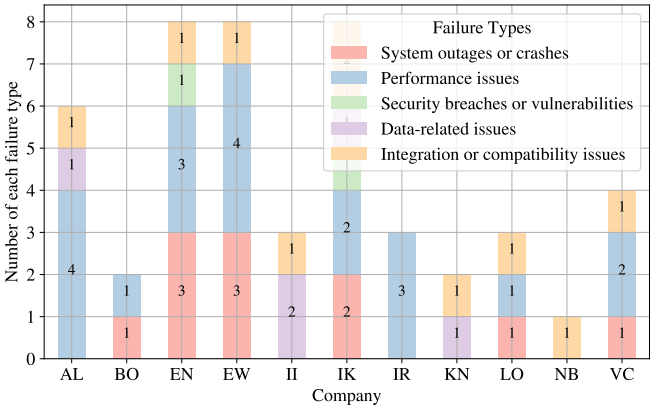


Fig. 1. Distribution of different failure types per company

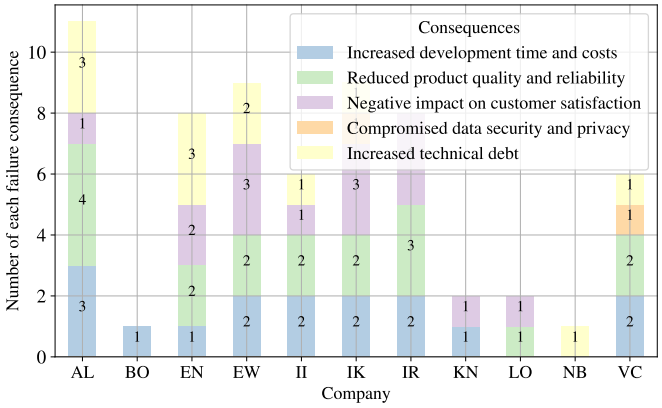


Fig. 2. Distribution of different types of failure consequences per company

A comparable rate of 8/11 (72.7%) experienced impacts on both product quality and customer satisfaction, indicating the extended effects on business reputation and profitability. An increase in technical debt, reported by 7/11 (63.3%), points to the accumulating challenges in software maintenance and development efficiency. These insights collectively revealed the complex nature of operational failures yet nearly even distribution of consequences affecting various aspects of business.

The findings of RQ1 highlight the extensive and persistent challenges regarding operational failures in the software industry. This indicates an urgent need for proactive and effective mitigation strategies. The diversity of these failures, as shown in Figure 1, emphasizes the necessity for robust monitoring solutions capable of addressing a wide range of issues.

### C. RQ2: Usage of monitoring data

With RQ2, we aimed to understand the evolving landscape of monitoring data management, which is crucial in detecting and analyzing operational failures. Furthermore, we examined the utilization of diverse monitoring tools and alert mechanisms and considered their impact in providing real-time insights and proactive responses to operational anomalies.

The data reveals how different organizational roles utilize monitoring data. DevOps engineers and quality assurance teams primarily engage with it for continuous integration and quality checks, focusing on system vulnerabilities and integration issues. Managers, though less represented, primarily rely on data for decision-making and oversight. In contrast, CloudOps engineers and software developers use data to track performance trends and identify vulnerabilities, aligning with their responsibilities in cloud infrastructure and software development. These patterns highlight the tailored applications of monitoring data to meet specific role-based demands.

Next, we analyze the results related to the usage of different monitoring tools and types of monitoring data, shown in Figure 3. Among the monitoring tools, Amazon CloudWatch, Azure Monitor, and Grafana emerge as the organizational leaders, signaling their widespread acceptance likely due to robust features and seamless integration capabilities with respective cloud environments. In contrast, tools like Prometheus and Kibana, while still utilized, show a relatively lesser degree of adoption, which may reflect preferences based on specific organizational needs or system compatibility.

When analyzing the types of monitoring data, performance metrics, particularly focusing on CPU and memory usage, stand out as the most monitored data type in Figure 3. This dominant usage highlights the critical importance of system performance and cloud infrastructure optimization in maintaining operational efficiency. Log files, comprising system, application, and error logs, demonstrate a slightly higher usage compared to performance metrics, highlighting their importance in diagnosing issues and gaining insights into operational failures. We have additionally identified that there is a consistent distribution of different types of monitoring data across a range of monitoring tools. This uniformity suggests that irrespective of the specific monitoring tool employed, the aforementioned data types are always prioritized.

In comparison, resource utilization, user activity, and network traffic, although integral to comprehensive system monitoring, appear to be less prioritized. This may suggest an area for increased focus, especially considering the insights they can provide into user behavior and network efficiency. Security alerts remain a significant concern even though they don't

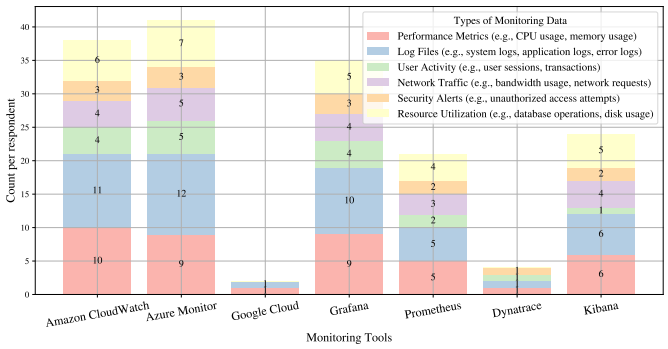


Fig. 3. Overview of monitoring tools and types of monitoring data per respondent out of 25

dominate in usage, indicating a relatively limited emphasis on identifying and mitigating potential security risks. The data collectively illustrates diverse monitoring approaches using various tools tailored to specific system health and security needs, emphasizing strategic organizational choices.

Many commercial monitoring tools offer the possibility to configure different alert mechanisms, which can be highly significant for timely identification of operations failures. The survey results reveal that a larger proportion of companies reported having basic or limited alert systems, compared to those with fully integrated systems, suggesting a trend towards incremental adoption of sophisticated monitoring tools and alert strategies. Interestingly, most of these companies consider their systems to be very effective, indicating a positive reception towards the existing alert mechanisms despite their varying levels of complexity. However, fewer companies are in the process of implementing or planning to implement alert systems. This shows a growing awareness and need for advanced monitoring solutions. This industry shift towards proactive, efficient monitoring strategies underscores the growing importance of effective alert systems for organizational resilience and efficiency.

### D. RQ3: ML for advanced monitoring and alerting

The survey findings revealed a diverse perspective among different organizational roles, as shown in Figure 4. Quality assurance practitioners mainly reported a moderate need, with a few noting a strong need, which may denote a balanced perspective on integrating advanced technologies and the possible benefits for quality assurance processes. DevOps and CloudOps engineers, with the majority perceiving little and moderate need, expressed satisfaction with current monitoring systems or possible concerns associated with implementing and maintaining such ML-driven solutions. On the contrary, developers recognized a strong need for advanced monitoring, most likely because they are usually impacted by operational failures and are mainly responsible for their resolution. Interestingly, manager roles pointed towards a strong and moderate need for ML-driven monitoring solutions, which may reflect their prioritization of efficiency and risk management in project delivery. These varied responses highlight the complexity of organizational readiness for advanced ML monitoring solutions, emphasizing the relevance of our study to understanding current and future trends.

As previously explained in Section II, one of the reasons for examining the applicability of ML-based monitoring solutions was inspired by the authors' recent work [13]. For this purpose, we specifically constructed questions to target the benefits and challenges of such solutions. Table I highlights the top three benefits and challenges identified in our analysis. Among the benefits, *the fastest detection of operational failures* stands out as the most acknowledged advantage, marked by 23 out of 25 respondents, highlighting its significance. This suggests a strong agreement on the importance of ML in quickly identifying operational issues, which is crucial for timely problem-solving and efficiency. The second most frequent

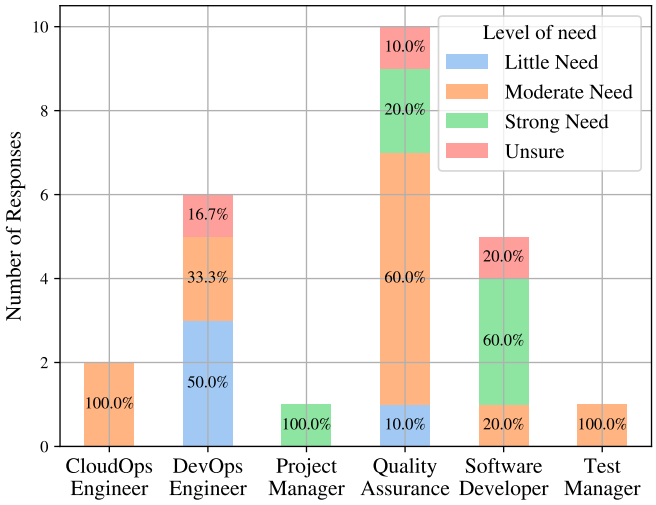


Fig. 4. Need for advanced analysis of monitoring data perceived by different organizational roles

answer, *the reduction in manual monitoring efforts*, indicates a significant awareness of ML capabilities, highlighting its role in easing the workload of human operators. *More accurate root cause analysis*, although rated lower, is still notably recognized as a benefit, pointing to the analytical strengths of ML in diagnosing issues.

The top three identified challenges, including *integration with existing systems*, *cost and resource allocation*, and *data privacy and security concerns*, shared the highest score, with each receiving 15 responses. This uniformity in responses suggests a balanced perception of these challenges, emphasizing that while ML implementation is promising, it comes with a set of equally important considerations. These include technical integration complexities, significant investment requirements, and the critical need to maintain data integrity and security. The balanced view on these challenges reflects a well-rounded understanding of what implementing such a solution entails, emphasizing the need for strategic planning and resource management.

The aforementioned ML-based solution for proactive monitoring [13] utilized the capabilities of the GPT3.5 Turbo model for getting interpretations of the log data. Therefore,

TABLE I
TOP THREE BENEFITS AND CHALLENGES OF ML SOLUTION FOR REPORTING INTERPRETABLE ALERTS

| **Benefits** | **# respondents** |
|---|---|
| Faster detection of operational failures | 23 |
| Reduction in manual monitoring efforts | 19 |
| More accurate root cause analysis | 15 |
| **Challenges** | **# respondents** |
| Integration with existing systems | 15 |
| Cost and resource allocation | 15 |
| Data privacy and security concerns | 15 |

TABLE II
READINESS TO ADOPT NEW AI/ML-BASED SOLUTIONS BY ORGANIZATIONAL ROLE

| Organizational role | Not ready or hesitant | Somewhat ready, facing constraints | From cautious to proactive |
|---|---|---|---|
| CloudOps Engineer | 0 | 2 | 0 |
| DevOps Engineer | 4 | 1 | 1 |
| Project Manager | 0 | 0 | 1 |
| Test Manager | 0 | 1 | 0 |
| Quality Assurance | 3 | 4 | 3 |
| Software Developer | 0 | 1 | 4 |

we aimed to understand how confident different organizational roles are in the predictions from such a large language model. A significant percentage of participants found *GPT suggestions somewhat effective* and primarily helpful for basic alerts and routine tasks. A relatively smaller group acknowledged the effectiveness of GPT in accurately identifying issues and suggesting solutions. However, only a few respondents rated GPT as highly effective, pointing to its potential for delivering in-depth analysis and actionable solutions, while a minimal number perceived it as not effective, highlighting limitations in certain contexts. In terms of confidence in GPT's predictions, a similar pattern emerges. Many participants trust GPT for routine tasks but remain cautious about its use in critical decisions. This disparity underscores the necessity of continually assessing GPT's reliability and effectiveness in diverse operational contexts.

### *E. RQ4: Overall adoption of ML-based solutions*

The current level of readiness towards adopting AI/ML-based solutions is a crucial indicator of how these cutting-edge technologies are currently being integrated into various software environments. The collected survey data, as shown in Table II, reflects varied levels of readiness among different organizational roles. For instance, a significant number of quality assurance practitioners and software developers indicated cautious readiness or somewhat readiness, facing constraints. This implies a recognition of the potential benefits of AI/ML but with a raised awareness of the challenges and constraints involved. The data also indicates a hesitancy among some roles, particularly DevOps engineers, who showed a mix of hesitation and cautious readiness. These results could be related to potential concerns regarding the technical complexities, integration challenges, or possible disruption to established workflows and processes. Overall, the distribution of answers highlights the need for tailored strategies in AI/ML integration, considering the unique needs and constraints of each role within the software development and operations.

Another significant observation from the survey results shown in Table III is the contrast between the current and forecasted impacts of AI/ML-based solutions. A major percentage (56%) of respondents reported no significant impact from recent AI advancements on their development and testing processes. However, there are positive expectations about the future, with an equal proportion of respondents (56%) anticipating a noticeable increase in development cycle speed due to AI/ML solutions. Therefore, while the immediate benefits of AI are not yet widely recognized across organizational units, there is strong optimism in its potential to enhance efficiency and delivery pace in the near future.

TABLE III
TOP THREE ANSWERS REGARDING THE CURRENT AND FORECASTED IMPACT OF AI/ML-BASED TOOLS ON DEVELOPMENT, TESTING, AND OPERATIONS

| **Current impact on DevOps** | **Answers (%)** |
|---|---|
| No significant impact | 56% |
| Slightly improved processes | 20% |
| Moderately improved processes | 20% |
| **Forecasted impact on delivery pace** | **Answers (%)** |
| Noticeable increase in speed | 56% |
| Minor increase in speed | 36% |
| No effect on pace | 4% |

Even though practitioners in the majority of organizational units do not perceive the immediate effects of AI/ML tools on the software development life-cycle, a significant percentage (60%) have already started using some of the basic tools, such as chatbots and simple analytics. This captures a typical transitional phase in the adoption of new technology. Industry professionals acknowledge the potential of AI in enhancing software development and operations, as they are keen to explore widely used tools on the market. However, their practical outcomes in software development environments are still not apparent but are enthusiastically awaited.

## V. DISCUSSION AND CONCLUSION

In this section, we discuss the main findings of this survey study, focusing on the implications and future directions of each research question.

The diversity of identified operational failures and their frequencies across different organizational units and roles **(RQ1)** highlights the dynamic and complex nature of software development environments. A need for more advanced monitoring solutions is evident to ensure operational resilience and efficiency in such environments. This may include interpretable monitoring strategies that detect issues and provide natural language suggestions for addressing them. This evolution towards systems that can interpret operational failures and propose solutions in a human-readable format represents a significant step forward. It merges the efficiency and precision of technology with the intuitive understanding of human experts, aiming to enhance decision-making and streamline the resolution process in software development environments.

Detailed insights into the use of monitoring data revealed its significance for investigating underlying issues and maintaining software quality **(RQ2)**. The diverse usage of tools like Amazon CloudWatch and Microsoft Azure, favored for their comprehensive features and ease of integration, highlights the need for tailored approaches to meet varied operational demands. Commonly analyzed data includes performance met-

rics and log files, with operational teams prioritizing system performance monitoring and issue diagnosis. Organizations adopt alert systems of varying sophistication to enhance effectiveness, and there is industry consensus on the value of alert strategies for the early detection of operational failures.

Even though some organizational roles are currently confident in existing monitoring solutions, there is still a growing interest in integrating advanced technologies to enhance alert detection **(RQ3)**. This is crucial for minimizing downtime and resolving issues proactively. Additionally, the potential reduction in manual monitoring efforts through ML suggests an important shift towards automation, freeing up human resources for more complex tasks. However, the possibilities of ML in monitoring and alerting are approached with caution. There are concerns about integration, costs, and data security, as well as the varied levels of confidence in ML predictions, particularly in critical operational tasks. This indicates a need for further validation and refinement of these ML-enabled systems. The common perspective leans towards a gradual, thoughtful integration of ML, with a focus on balancing innovation with practicality, efficiency with reliability, and automation with human oversight.

Regarding **RQ4**, the findings showed a cautious but growing readiness among different organizational roles towards adopting AI/ML solutions. The varied readiness levels indicate a recognition of the potential benefits and challenges associated with these technologies. The contrast between the current limited impact and the optimistic future outlook for AI/ML in software development suggests an ongoing transition phase. As practitioners start to engage with basic AI/ML tools, there is a growing expectation for these technologies to considerably enhance operational efficiency and development processes in the future. This points to a period of exploration and gradual integration, where the full potential of AI/ML in software environments is yet to be realized.

The collective findings from our survey study offer valuable insights into the evolving dynamics of software development and monitoring, particularly in the context of operational failures, monitoring practices, alert strategies, and the integration of AI/ML technologies. The results revealed a combination of complex challenges and emerging opportunities where organizations increasingly recognize the potential of advanced technologies to enhance efficiency and problem-solving capabilities. The anticipated future impact of AI/ML on software development and operations promises a new era of innovation and productivity, considering that the challenges of integration, cost, and data security will be effectively addressed in the near future.

## Acknowledgment

This work was partially supported by the Wallenberg AI, Autonomous Systems and Software Program (WASP) funded by the Knut and Alice Wallenberg Foundation. We thank all the industry practitioners who participated in this survey, whose invaluable insights and expertise have greatly contributed to the success and depth of this research.

## References

[1] D. Cotroneo, L. De Simone, P. Liguori, and R. Natella, "Enhancing the analysis of software failures in cloud computing systems with deep learning," *Journal of Systems and Software*, vol. 181, p. 111043, Nov. 2021.

[2] L. Giamattei, A. Guerriero, R. Pietrantuono, S. Russo, I. Malavolta, T. Islam, M. Dînga, A. Koziolek, S. Singh, M. Armbruster, J. Gutierrez-Martinez, S. Caro-Alvaro, D. Rodriguez, S. Weber, J. Henss, E. F. Vogelin, and F. S. Panojo, "Monitoring tools for DevOps and microservices: A systematic grey literature review," *Journal of Systems and Software*, vol. 208, p. 111906, Feb. 2024.

[3] D. A. Tamburri, M. Miglierina, and E. D. Nitto, "Cloud applications monitoring: An industrial study," *Information and Software Technology*, vol. 127, p. 106376, Nov. 2020.

[4] L. Toka, G. Dobreff, D. Haja, and M. Szalay, "Predicting cloud-native application failures based on monitoring data of cloud infrastructure," in *2021 IFIP/IEEE International Symposium on Integrated Network Management (IM)*, 2021, pp. 842–847.

[5] L. Mariani, M. Pezzè, O. Riganelli, and R. Xin, "Predicting failures in multi-tier distributed systems," *Journal of Systems and Software*, vol. 161, p. 110464, Mar. 2020.

[6] R. Rana, M. Staron, J. Hansson, M. Nilsson, and W. Meding, "A Framework for Adoption of Machine Learning in Industry for Software Defect Prediction:," in *Proceedings of the 9th International Conference on Software Engineering and Applications*. Vienna, Austria: SCITEPRESS - Science and and Technology Publications, 2014, pp. 383–392.

[7] J. Sillito and E. Kutomi, "Failures and Fixes: A Study of Software System Incident Response," in *2020 IEEE International Conference on Software Maintenance and Evolution (ICSME)*. Adelaide, SA, Australia: IEEE, Sep. 2020, pp. 185–195.

[8] J. Candido, M. Aniche, and A. van Deursen, "Contemporary Software Monitoring: A Systematic Literature Review," *arXiv:1912.05878 [cs]*, Dec. 2019.

[9] P. Thantharate, "IntelligentMonitor: Empowering DevOps Environments with Advanced Monitoring and Observability," in *2023 International Conference on Information Technology (ICIT)*. Amman, Jordan: IEEE, Aug. 2023, pp. 800–805.

[10] A. Q. Gill and S. Hevary, "Cloud Monitoring Data Challenges: A Systematic Review," in *Neural Information Processing*, A. Hirose, S. Ozawa, K. Doya, K. Ikeda, M. Lee, and D. Liu, Eds. Cham: Springer International Publishing, 2016, vol. 9947, pp. 72–79.

[11] A. Hrusto, P. Runeson, and E. Engström, "Closing the Feedback Loop in DevOps Through Autonomous Monitors in Operations," *SN Computer Science*, vol. 2, no. 6, p. 447, Aug. 2021.

[12] A. Hrusto, E. Engström, and P. Runeson, "Towards optimization of anomaly detection in devops," *Information and Software Technology*, vol. 160, p. 107241, 2023.

[13] A. Hrusto, P. Runeson, and M. C. Ohlsson, "Autonomous monitors for detecting failures early and reporting interpretable alerts in cloud operations," in *2024 IEEE/ACM 46th International Conference on Software Engineering: Software Engineering in Practice (ICSE-SEIP)*, 2024.

[14] J. Holmström, "From AI to digital transformation: The AI readiness framework," *Business Horizons*, vol. 65, no. 3, pp. 329–339, May 2022.

[15] J. S. Molléri, K. Petersen, and E. Mendes, "An empirically evaluated checklist for surveys in software engineering," *Information and Software Technology*, vol. 119, p. 106240, Mar. 2020.

[16] M. Kasunic, "Designing an effective survey," Carnegie Mellon Software Engineering Institute, Tech. Rep. CMU/SEI-2005-HB-004, 2005. [Online]. Available: https://apps.dtic.mil/sti/pdfs/ADA441817.pdf

[17] J. Linåker, S. Sulaman, M. Host, and R. de Mello, "Guidelines for conducting surveys in software engineering," Lund University, Tech. Rep., 05 2015. [Online]. Available: https://lup.lub.lu.se/search/files/6062997/5463412.pdf

[18] S. Baltes and P. Ralph, "Sampling in software engineering research: A critical review and guidelines," *Empirical Software Engineering*, vol. 27, no. 4, jul 2022.

## Appendix A
## Online questionnaire form

Online questionnaire form and results are available at https://doi.org/10.5281/zenodo.10986352.