Journal of Information Systems Engineering and Management
2026, 11(2s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

Decision Latency as a First-Class Performance Metric in AI-Native Engineering Organizations

Lakshmi Priya Gopalsamy

Independent Researcher, USA

ARTICLE INFO

Received: 03 Jan 2026

Revised: 08 Feb 2026

Accepted: 16 Feb 2026

ABSTRACT

The artificial intelligence features have essentially revolutionized the software development speed with technical implementations that took hours to minutes. State-of-the-art code generation systems can generate useful implementations at speeds never before seen, but the throughput of organizations is limited by decision-making architectures targeted at a lower-speed execution paradigm. Elite engineering organizations show frequencies of deployment and orders of magnitude better in lead times than low performers, which is largely due to the latency of decisions, compared to technical capability. The given framework defines decision latency as a performance metric of operations, which includes recognition latency, coordination overhead, approval queue, and implementation delays. Architectural designs are used to manage this limitation by distributing decision rights, automation using policy-as-code, synthesizing context, and progressive authority models, which balance governance demands with performance speed. Trackable proxies such as the approval queue time, number of review loops, frequency of escalation and rework churn allow systematic measurements to be made across organizational settings. There are different manifestations of decision architecture challenges in enterprise cases involving financial services, cloud infrastructure, and healthcare technology. To ensure the transformation of AI-enabled execution speed into a consistent organizational throughput, accountable authority structures, validation procedures, and coordination protocols need to be redesigned accordingly to align with the modern development capabilities without compromising accountability and risk management corresponding to the severity of consequences.

Keywords: Decision Latency, Organizational Architecture, Governance Automation, Policy-As-Code, AI-Augmented Engineering

Introduction

Advanced code generation systems have essentially transformed the speed of the technical execution of a software engineering organization. Studies of large language models that are trained on code show that solutions to programming problems can be developed at the rate that has been previously impossible to solve manually, with models being able to attain functional correctness on standardized coding tasks at the rate of approximately 28.8 on the first attempt [1]. This technology is a significant acceleration in the amount of time it takes to complete single technical tasks, including code synthesis to automated testing. Nevertheless, when the increased individual performance ability is translated into organizational throughput, there is more of a complicated reality. The software delivery performance analysis of thousands of engineering organizations shows that, despite the fact that the advancement of technological capabilities still goes on, the structure of organizational performance has not changed, that is, it is very stratified, and the elite performers constitute only 26% of surveyed teams [2].

The artificial intelligence enhancement of compression of technical execution time has revealed organizational decision-making as a vital limitation to end-to-end delivery velocity. When the work of providing basic code functionality can be done in minutes instead of hours by individual developers, the organizational workflows involved with such code such as approval processes, architectural review, security validation, and deployment authorization processes become prevalent points of bottleneck. Measuring the performance of software delivery offers some startling differences in the organizational ability to convert technical performance into finished products. The rate of execution is also 973 times greater in elite engineering organizations than in their worst performers, which proves that speed of execution is not sufficient to explain such a dramatic performance differences [2]. The gap between levels of organizational performance indicates that affecting the overall throughput is not controlled by technical capability.

Additional analysis of the delivery metrics supports this reading. Although the elite performers can obtain lead times to changes within one hour of code commit to the production deployment, the low performing organizations can take over six months to obtain the same [2]. This performance disparity, which is measured as the elite performers with 6,570 times faster lead times cannot be explained by both the technical tooling differences and individual developer capability [2]. The scale of such difference is an indication of the systematic organizational tension within organizational decision processes, the approval procedures, and systemic coordination. Technical execution that becomes order of minutes or hours leaves organizational processes that need days, weeks, or months to pass through approval lines as the bottlenecks to value delivery.

These observations define decision latency, the time taken to pass through the decision process between identifying the need to make a decision and the action taken, as a first-class performance measure that should be systematically measured and optimized. Conventional methods of enhancing software delivery tend to focus on technical practices and tooling investments and developer productivity. Although they are still relevant, the data indicates that the organizational decision architecture has become the main limitation to throughput in AI-enhanced engineering settings. Companies that want to leverage the rush in technical performance must restructure decision models, decentralize control and adopt systems fitted to the speed that modern development tools offer. The sections below give architectural models to measure decision latency, enumerated patterns of choice of architecture that add friction, and suggested principles of design used in designing low-latency decision architectures which can retain governance requirements but allow quick, responsible response.

Core Definitions and Conceptual Framework

The latency in decision making must be operationally accurate so as to facilitate systematic observation of various organizational situations. Decision latency is determined as a result of lapsed time between event recognition being triggered up to verified implementation as an outcome of the framework, not to be confused with decision throughput which is completed decisions per unit time. Studies on managing technical debt indicate similar problems in organizational decision-making where an enterprise is usually forced to make 8-12 handoffs before an implementation can occur, and every move adds queue time and information loss [3]. The multi-phase development of organizational approval pathways is a cause of compounding delays that can be seen especially where technologized execution is used to speed up.

This conceptual model splits decision latency into sub stages which reflect complexity found in a large-scale software development. Recognition lag is the time between the occurrence of the event and the initiation of a decision. Information synthesis and option evaluation take up analysis time, wherein teams tend to use large amounts of effort on what should otherwise be superfluous processes with more efficient mechanisms. Technical debt management studies indicate that developers spend about 23 percent of their working time dealing with technical debt that is related to activities that must be attended to because of the built-up organizational inefficiency [3]. This ratio indicates that there is significant room to improve by having lean decision architecture to decrease overheads in coordination.

Coordination overhead turns out to be the most fluctuated element in the organizational settings. Study of the patterns of automation artifact deployment shows that around 60 percent of the solutions identified are in the early evidence stages meaning that the organizations are not willing to make decisions on implementation [3]. This reluctance is indicative of the ambiguity of multi-stakeholder alignment within the enterprise setting. These coordination issues are exemplified by large-scale agile development projects, where implementation of 175 people, comprising 100 external consultants across 5 different vendor organizations took 800,000 person-hours to implement 2,500 user stories in 12 releases [4]. The project hit the highest level of complexity when it had 12 concurrent development teams that demanded the large scale of coordinating mechanisms that were not within the normal frameworks.

The framework includes the decision criticality categorization where irreversible high-consequence decisions are classified and reversible low-consequence decisions are classified. A systematic review of the literature on automation in technical debt management found 121 unique artifacts of automation in 178 original research articles and found that 47 percent are standalone tools, and 32 percent are automation plug-ins to be integrated with the current infrastructure [3]. The implication of this distribution is that organizational decisions about automation of tooling and processes are often complex integration issues, and not binary decisions. The technical debt study also shows that 82 out of 121 studies that have been conducted consider code-level technical debt, whereas only 58 out of the studies take into account design-level issues implying a different degree of organizational investment in different spheres of decision-making [3].

Implementation delay and verification periods show the organizational ability to deploy and verify within a short time. Megaprojects have significant resource demands, and major projects have budgets of up to 140 million EUR, and lifespans of several years [4]. Organizations that operate within such complexity are faced with the need of balancing in depth validation and velocity goals. The framework acknowledges that decision quality includes accuracy in getting desired results, sustainability in preventing duplication, and alignment with strategic goals of an organization. Technical debt management The interpretation of techniques used in detecting and managing technical debt has provided 45% of technical debt artifacts are based on the design-level of managing technical debt and 49 out of 58 are specifically based on identifying activities instead of remediation [3], indicating that decisions made within organizations often concentrate on identifying a problem rather than addressing it.

Journal of Information Systems Engineering and Management
2026, 11(2s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/ **Research Article**

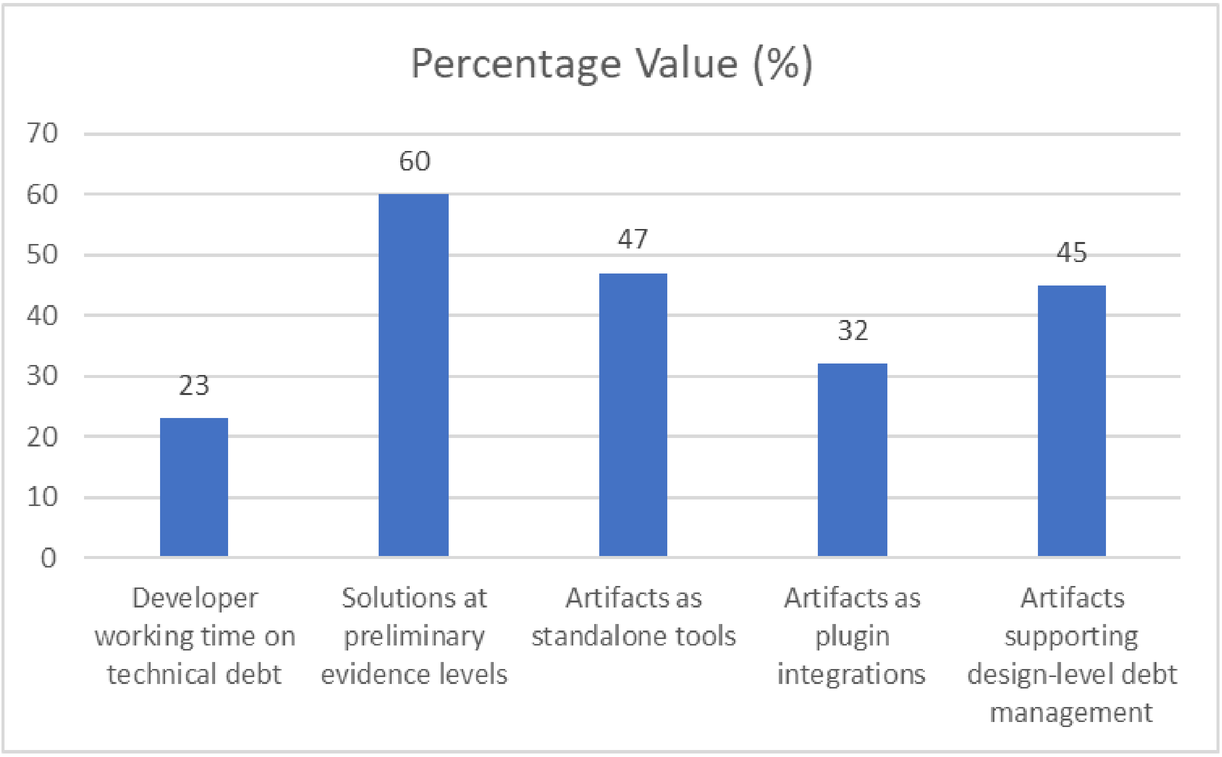

Fig 1: Decision Latency and Technical Debt Management - Percentage Distribution Metric [3, 4]

Architecture/Design Patterns and Principles

To minimize the time to make decisions and keep the governance, the patterns of architecture must allocate authority, validate it by automation, and limit the number of coordinated links. A study of the deployment of DevSecOps in 147 articles of the multivocal literature analysis study can find the structure of an organization that incorporates both speed and control and the study identifies 60 practices that are deployed at the organizational, process, technological, and business levels [5]. It is on these practices that empirical foundation can be placed when it comes to the way organizations design decision making to ensure that it is executed fast without compromising on accountability.

The Decision Rights Architecture pattern defines clear ownership boundaries that outline the decision scopes, any escalation and authority limits in hierarchical forms of the structure. DevSecOps adoption analysis shows that organizations with formalized governance structures in 178 primary studies indicate much higher efficiency of coordination [5]. The pattern defines default decision-makers of recurring scenarios, avoiding negotiating on a case-by-case basis overhead. Infrastructure decisions that affect more than one service are handled by the platform teams, service teams handle the details of implementation within a limited interface, and cross-cutting issues that lie outside the established guardrails are managed by the architecture review boards. The DevSecOps study revealed that there were 28 different issues encountered during implementation, and the organizational and cultural ones formed the biggest group, which also indicates that authority distribution is one of the main areas of tension in speeding up decisions [5].

The Policy-as-Code design eliminates the use of general human approvals with compliance verification against declarative policies. Infrastructure deployments test against security baselines, code commits test against quality gates and code changes are checked against operational standards without manual inspection of conforming changes. A study about the automation of technical debt management has determined 121 artifacts of automation in various organizational settings, 47 of which are used as standalone software, and 32 of which are used as a plug-in and need to be integrated [5]. This distribution implies that the automated policy enforcement should support different deployment models. The non-conforming changes initiate human examination including the provision of context as a matter of course, enabling the security and compliance teams to extend governance without a commensurate upsurge in headcount.

Context Synthesis pattern uses the power of computers to expedite the information collection and alignment of the stakeholders. Such massive software repositories prove the data complexity that organisations have to operate in and the platforms can handle 200,000 events per day with a load of 8,300 events per hour [6]. Companies that have a comprehensive record of development activities have large amounts of data, where repository mining programs have amassed more than 900GB of raw data and 10GB of queryable metadata of publicly available software projects [6]. At the levels of natural language processing, graph databases, and summarization, historical decisions and initiatives are identified as related and dependencies, technical specifications are derived into executive briefings.

The Progressive Authority pattern gives the teams widening the scope of decisions the growing reliability and judgment. Newer teams work under stricter approval standards and mature teams have greater autonomy and each team has an escalation requirement of decisions beyond a specified limit. DevSecOps research found that the organizations that were successful in applying security practices suggested 20 various metrics of measuring the effectiveness of the program on an organizational, process, and technology levels [5]. This trend strikes a balance between managing risks and managing latency, and the reason is to provide a level of oversight where the uncertainty is the greatest and provide the speed when the records of the tracks warrant belief.

Pattern / Principle	Purpose	Example / Metric
Decision Rights Architecture	Clear ownership + escalation rules	Improves coordination efficiency
Policy-as-Code	Automates compliance approvals	121 automation artifacts identified
Context Synthesis	Faster stakeholder alignment using data	Handles 200k events/day
Progressive Authority	More autonomy as teams mature	Uses 20 effectiveness metrics

Table 1: Architecture/Design Patterns to Reduce Decision Latency [5, 6]

Measurement Approach and Practical Metrics

To measure decision latency, the measurement has to be operationalized by defining observable proxies that represent friction in an organization without excessively instrumentating the proxies. Manufacturing operations whose pipelines of operation are complex offer educative comparisons to gauge the decision flow in organizations. Evidence of machine learning pipeline observability Studies into machine learning pipeline observability show that per component-level measurements in heterogeneous systems are difficult to measure, and practitioners need automated component collection mechanisms to integrate with current tooling instead of requiring an overhaul of their infrastructure [7]. Organizations with observability capabilities note that lighter weight instrumentation styles are adopted significantly more often, and open-source implementations receive over 400 GitHub stars in the first release cycles, meaning that minimally invasive measurement solutions are in high demand by practitioners [7].

Approval queue time is a metric that is introduced to measure the latency that is caused by sequential authorization requirements. This measure is used to describe the time elapsed since the time of submitting a decision to final acceptance that includes handoffs between organizations. The measurement infrastructure should differentiate the active reviewing periods and queue waiting time to separate the process inefficiencies with workload capacity constraints. Similar measurement issues can be seen in feature-level observability systems in production systems, where the practitioners monitor distributions over 256 factor buckets to detect changes in performance in high-dimensional operational spaces [7]. Other organizations using a decision tracking system can use such bucketing techniques to cluster decision processes by decision type, corporate department, or risk rating in order to optimize decision processes with high latency.

Review loop count gives a supplementary understanding of coordination costs by counting the loops of iteration needed to reach a stakeholder consensus. Every time the review is repeated, it is another organizational friction as the requests to clarify, solve the objection, or synthesize information extend decision time. Automated constraint validation systems provide informative measurement precedents and research has found 25 different single column validation constraints and more multi-column relationship verifications that companies should continuously tune to find the right balance between precision and recall [7]. The quality frameworks of decisions can also introduce graduated review, and monitor the outcome of decisions; either they achieve a consensus within reasonable limits of the iteration or a higher level of authority needs to be led into decision making.

Rework churn and escalation frequency measures reflect dimensions of decision quality which have an impact on sustainable organizational velocity. Escalation patterns demonstrate effectiveness of decision-rights architecture, which means the correspondence of authority distribution with organizational capability and risk tolerance. A study on the operational performance measurement performance in different organizational contexts was conducted on 355 organizations surveyed on the following: Performance measure selection, and patterns of deploying performance measures [8]. This level of empirical research indicated that the heterogeneity in metric selection practices was significant and organizations focused on various issues of operational effectiveness depending on the level of governance maturity and strategic goals. The frameworks of comprehensive performance revealed 28 different security-related measures across the operations, management and technical areas, which showed that multidimensional performance properties in organizations need various measurement portfolios [8].

Delay in incident decisions give real time indicators of the responsiveness of organizations during the time of pressure. Decision architecture weaknesses are revealed through production incidents since tight timelines do not allow the elaborate approval processes. Organizations with high incident decision latency exhibit structural disinformation within their structure between how authority is distributed and the requirements of the operations tempo. Measurement systems to keep track of these delays allow organizations to identify between the unavoidable technical diagnosis time and organizational coordination overhead and quantify the potential improvements of streamlined decision paths and increased availability of context to the decision-makers who have to operate under time limits.

Journal of Information Systems Engineering and Management
2026, 11(2s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/ **Research Article**

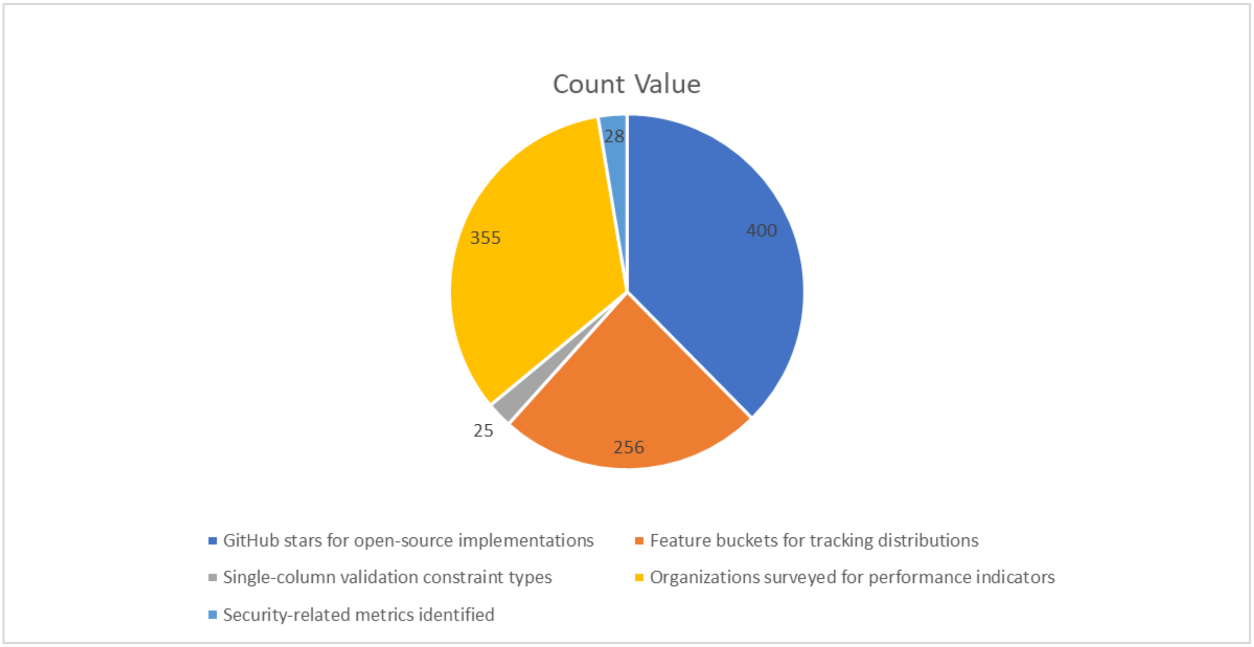

Fig 2: Measurement Framework - Discrete Count Metrics [7, 8]

Case Examples (Industry and Enterprise Scenarios)

Empirical research of organizational decision latency in various enterprise scenarios shows a wide range of difference in the speed of decision-making and the overhead of coordination. The study of experimentation in software-intensive companies can be instructively related to the study of decision architecture performance. A study of controlled experimentation processes based on 128 organizational case studies evidences that decision cycles that regulate feature deployment can predictable temporal behaviors, as the sequence of experiments take an average of 42 days with each experimental decision taking one week to be made [9]. These time limits set minimum standards of the latency of decisions in technology-based organizations that run on the data-driven governance frameworks. Importantly, it has been shown that out of these experimental decision sequences, it has proven that only 1/3rd of them eventually lead to production deployment, showing that enormous organizational resources are spent on decision processes, which do not lead to implementation results [9].

Organisations that deploy automated trading systems in the financial services sector are especially confronted with the decision latency issue whose regulatory approval concerns augment technical validation processes. Decisions related to portfolio rebalancing that must be verified by compliance, evaluate risk, and authorize by the executive often face latency due to handoffs over organization boundaries. When the coordination mechanisms do not support accelerated execution capabilities, which are brought about by algorithmic automation, teams working through these approval chains report decision timelines that extend beyond planning horizons. Organizations using agile practices show extensive awareness of the effect of coordination overhead, and 86% to 87% of agile practitioners are using daily synchronization meetings to minimize the delays in information propagation as reported by 86% to 87% [9]. This adoption pattern indicates that organizations that undergo narrow execution cycles naturally react by raising the frequency of their coordination but such reactions can create unwanted causes of latency by meeting overhead build-up.

The contexts of the cloud infrastructure management show the patterns of decision latency due to the distributed organizational structures. A survey study of coordination behavior in distributed software teams of different locations across the world yielded 160 responses in a study that offered empirical basis to learn more about cross-boundary decision dynamics [10]. The interaction of 152 subjects found that mean groupings had 7.19 team members and, therefore, defined the base formation units wherein the authority to make decisions function [10]. In configurations that exhibited a systemically larger organization compared to co-located teams, distributed teams suggested the complexities of coordination that is reflected in longer decision latencies when there are approval conditions across organizational boundaries. According to quantitative measures of the time spent on coordination the team members who were distributed across the world spent 7 hours and 45 minutes every week in planned coordination events, which were augmented by 8 hours and 54 minutes in unplanned coordination transactions [10]. These indicators further reveal how much of the organizational capacity is spent on coordination overhead as the unscheduled coordination time is much greater than the scheduled allocation in spite of clear attempts at process definition.

Healthcare technology implementations offer another dimension of decision latency in patient safety in which stringent validation requirements are required. The changes in the clinical decision support system mandate may involve institutional review board, clinical validation studies, and the stakeholder consensus processes which ultimately cause a cumulative delay. Managing these approval pathways, organizations note that the distributed stakeholder groups repeat coordination patterns that have been recorded in software development situations in which geographical distance and asynchronous communication channels can add to the duration of a decision cycle. The clash between the regulatory demands and the expedited technical skills generates tension within the organization between the governance demands and the rate of execution speed, showing that the optimization of decision latency in the zone should be balanced between the risk management demands and the throughput targets without neglecting accountability standards that are relevant to the severity of consequences.

Scenario	Main Issue	Key Metric
Software experimentation	Slow decisions + low deployment rate	42 days avg, only 1/3 deployed
Finance (trading systems)	Compliance approvals delay execution	Decisions extend beyond planning
Distributed cloud teams	Heavy coordination overhead	~16+ hrs/week coordination
Healthcare tech	Validation + stakeholder approvals slow change	Multi-step approvals cause delay

Table 2: Decision Latency Across Industry Scenarios [9, 10]

Conclusion

Compressed technical performance by artificial intelligence augmentation of organizations encounters decision-making structures that form major throughput bottlenecks. Conventional focus on technical practice and increases in productivity of developers is not enough when organizational approval processes take orders of magnitude longer than the code generation process. The decision latency model implements measurement based on measurable proxies that measure coordination friction without too much instrumentation overhead. Patterns in architecture such as decision rights allocation, policy validation by computer, computational context synthesis, and progressive authority models are blueprints that can be put into practice and achieve the reduction of latency without disrupting integrity in governance. Microsystems Enterprise situations in controlled industries and distributed organizations The applicability is universal, even in cases with domain manifestations. Effective optimization demands an orderly measurement of the approval queues, the number of review cycles, the escalation processes, and rework churn and planned redistribution of authority and automation of validation. Organizations that adopt these patterns are able to transform the individual execution velocity into sustainable organizational throughput, minimize thrash, enhance time-to-value and become more resilient to accelerated change. A shift in the system of manual approval to automated policy-based approach is an inherent organizational development that will be required to take advantage of the capabilities of artificial intelligence without compromising the principles of accountability that correspond to the severity of consequences of the decision and regulatory needs.

References

[1] Mark Chen et al., "Evaluating Large Language Models Trained on Code," arXiv:2107.03374v2, 2021. [Online]. Available: https://arxiv.org/pdf/2107.03374
[2] Dustin Smith et al., "State of DevOps Report," DORA Research, 2021. [Online]. Available: https://services.google.com/fh/files/misc/state-of-devops-2021.pdf
[3] João Paulo Biazotto et al., "Technical debt management automation: State of the art and future perspectives," ScienceDirect, 2024. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0950584923002306
[4] Torgeir Dingsøyr et al., "Agile Development at Scale: The Next Frontier," IEEE, 2019. [Online]. Available: https://arxiv.org/pdf/1901.00324
[5] Xiaofan Zhao et al., "Identifying the primary dimensions of DevSecOps: A multi-vocal literature review," ScienceDirect, 2024. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0164121224001080
[6] Georgios Gousios, "The GHTorent Dataset and Tool Suite". [Online]. Available: https://www.researchgate.net/profile/Georgios-Gousios/publication/261479894_The_GHTorent_dataset_and_tool_suite/links/5589137b08ae8c4f34067b51/The-GHTorent-dataset-and-tool-suite.pdf
[7] Shreya Shankar and Aditya G. Parameswaran, "Towards Observability for Production Machine Learning Pipelines," Proceedings of the VLDB Endowment. [Online]. Available: https://www.vldb.org/pvldb/vol15/p4015-shankar.pdf
[8] Joonas Forsberg and Tapio Frantti, "Technical performance metrics of a security operations center," ScienceDirect, 2023. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S016740482300439X
[9] Florian Auer et al., "Controlled experimentation in continuous experimentation: Knowledge and challenges," ScienceDirect, 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0950584921000367
[10]Viktoria Stray and Nils Brede Moe, "Understanding coordination in global software engineering: A mixed-methods study on the use of meetings and Slack," ScienceDirect, 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0164121220301564