International Research Journal on Advanced Engineering Hub (IRJAEH)
e ISSN: 2584-2137
Vol. 04 Issue: 10 October 2026
Page No: 5873 - 5882
https://irjaeh.com
https://doi.org/10.47392/IRJAEH.2026.0759

Scaling Enterprise Agility: Transforming Complex Digital Programs Through SAFe, Lean Governance, and AI-Ready Delivery Models

Aparna Gosala[1]
[1]*Sri Padmavati Mahila University, Tirupati, Andhra Pradesh, India*
Email id: *aparnagosala3@gmail.com*[1]

Abstract
Enterprise agility is becoming more and more a system level capability than the replication of a team level Scrum. This review summarizes 25 peer reviewed studies related to large-scale agile transformation, the Scaled Agile Framework (SAFe), Lean portfolio governance, DevOps, machine-learning operations (MLOps), and responsible artificial intelligence (AI) engineering. The evidence shows that scaling works when there are coordinated persistent value streams, local autonomy within clear decision boundaries, and continuous economic and risk feedback, instead of episodic project controls. While SAFe provides a familiar coordination architecture, Agile Release Trains, Program Increment planning, portfolio backlogs and architectural synchronization, empirical research cautions against ceremony, rigidity and too much hierarchy, which creates the delays that agility is supposed to eliminate. Lean governance is the balancing mechanism that provides funding to value streams, restricts work in process, pushes decision making for routine work to the lowest level, and escalates only material risk. AI-ready delivery is an extension of traditional DevOps as code, data, models, prompts, policies, and monitoring evidence all need to evolve in tandem. The review thus proposes an integrated operating model where strategy and policy trickle down through the governance guardrails and delivery, operational and model evidence trickle up to support investment and control decisions. Two analytical tables are provided which compare the scaling choices and map governance controls to delivery evidence. The synthesis is that enterprise agility is not measured by compliance to the frameworks or how frequently you deploy; it's the ability to shift investment, the ability to coordinate complex dependencies, the ability to safely deliver and the ability to learn from the evidence in production without losing sense of accountability.

Keywords: *enterprise agility; SAFe; Lean governance; digital programs; DevOps; MLOps; artificial intelligence; portfolio management.*

1. Introduction
Today, digital programs are composed of a mix of cloud platforms, legacy applications, data products, cybersecurity controls, external vendors and more and more AI driven services. Their performance is therefore more a function of how well the organization is able to coordinate decisions across portfolios, value streams, architecture, assurance and operations, rather than the speed of one development team. The most common challenges were identified as organizational culture, resistance to change, coordination and integrating with non-agile functions, whereas management support, model tailoring, coaching and shared alignment were identified as common enablers for large scale transformations [1]. Enterprise agility should therefore be thought of as a distributed organizational capability, and rapid learning and rapid redirection must go hand-in-hand with reliable integration and responsible control. This distinction is backed up by longitudinal evidence. In the context of a complex product environment, specialization was not something that could be eliminated as part of Ericsson's transition to continuous feature delivery, experimentation with

International Research Journal on Advanced Engineering Hub (IRJAEH)
e ISSN: 2584-2137
Vol. 04 Issue: 10 OCtober 2026
Page No: 5873 - 5882
https://irjaeh.com
https://doi.org/10.47392/IRJAEH.2026.0759

team structures, stepwise change and a shared direction across sites had to take place [2]. Another large organization found culture, previous Lean experience, management support and value unification to be key factors in adoption, while aggressive rollout schedules and lack of compatibility with the existing business processes hindered adoption [3]. The results contradict the notion that agility is a by-product of adding teams, ceremonies or an enterprise structure. According to comparative research, there are differences between the various models in terms of governance reach, coordination mechanisms, prescribed roles, and technical expectations between SAFe, Large-Scale Scrum (LeSS), Scrum@Scale, Disciplined Agile and the Spotify model [4]. More important, initial adoption isn't enough to sustain transformation. Normalisation research shows that new routines need to be coherent, collectively performed, monitored and constantly reworked in the daily routines [5]. This review will explore how SAFe, Lean governance, and AI-ready delivery can be integrated without losing agility to process compliance. Its contribution is a synthesis that is evidence based and links portfolio economics, program coordination, software delivery, AI lifecycle governance and operational feedback in one review framework.

2. Review Scope & Synthesis Approach

A structured narrative synthesis was used for the review due to the heterogeneity of the evidence base which comprised of systematic literature reviews, multivocal reviews, surveys, action research and longitudinal or retrospective case studies. Literature was categorized into four areas: large scale agile transformation, SAFe and portfolio coordination, Lean governance and continuous delivery, and MLOps or responsible-AI lifecycle management. Focus was placed on peer-reviewed research from 2016 onwards, with a focus on organizational scale, governance, delivery capability, or production AI. Twenty-five sources were kept, none of which were duplicated. Evidence was compared on a unit of analysis, claimed mechanism, reported benefit, contextual limitation and implication for complex digital programs basis. This is because the studies do not measure the same outcome or in the same setting and so do not estimate a pooled effect size. Rather, it aims to achieve a convergence of methods. Case studies on a very large scale reveal that the issues of customer involvement, architecture and inter-team coordination are constant challenges [6]. Additionally, a longitudinal study reveals that agile and plan-driven practices are not mutually exclusive and can be mutually enabling when there are legitimate coordination and assurance needs, due to scale. In 13 transformations, similar problems have been found in commercial and customized frameworks [8]. These converging results suggest that SAFe is one of the many possible configurations to coordinate a socio-technical system, rather than a context-free intervention.

3. From Team Agility to Enterprise Agility

Team agility is optimizing a bounded product backlog with the use of short feedback cycles. Enterprise agility also needs to apply to the limited investment, inter-team dynamics, shared platforms, regulatory requirements, and persistent operational risk. This is not linear replication, however, as the scaling problem. In the case of multiple teams using the same services, environments, data definitions and release constraints, local optimization can lead to higher system delay. Portfolio visibility, architectural interfaces, integration cadence and common evidence turn into coordination infrastructure. Meanwhile, queues are created in the centralization of all decisions and ownership is weakened. The design question is which decisions are to be synchronized, which decisions are to be kept local and which evidence is needed to safely decentralize authority [4, 6, 7]. The literature also distinguishes between the two concepts of transformation activity and transformed capability. Training, role changes, and launch events are inputs to adoption and don't necessarily mean that the enterprise can redirect resources or achieve outcomes more quickly. Normalisation involves making routines part of the budgeting, procurement, architecture, assurance and performance management [5]. A framework can thus look mature, and there is no change in the lead time, deployment risk or customer learning. An

International Research Journal on Advanced Engineering Hub (IRJAEH)
e ISSN: 2584-2137
Vol. 04 Issue: 10 OCtober 2026
Page No: 5873 - 5882
https://irjaeh.com
https://doi.org/10.47392/IRJAEH.2026.0759

evidence-based transformation evaluates the efficiency of flow, realization of outcomes, reliability of services, sustainability of employees, and time needed to change strategic direction. This is a change from teams 'following agile' to the operating system turning uncertainty into timely, governed learning [1] [3] [5].

4. SAFe as a Coordination Architecture

In complex programs, SAFe provides common terminology and concepts for managing the portfolio, development value streams, Agile Release Trains (ARTs), architecture, and delivery cadence. PI planning makes dependencies and capacity constraints visible and system demonstrations and inspect-and-adapt events provide cross-team feedback. Lean portfolio management links strategic themes to epics and value streams and architectural runway work acknowledges that autonomy of features in the near term requires the building of technical autonomy. They are designed to address issues that have been seen time and time again in large-scale programs, such as fragmented prioritisation, late integration, poor stakeholder alignment and dependency risk that is not transparent [4] [8]. The empirical evidence of SAFe is good, but limited. The research in 59 project portfolios across 22 large companies revealed mixed results on the hybrid model of SAFe, Scrum, and Spotify model, while communication, productivity and time-to-market were highlighted as benefits, organizational culture, resistance and strategic leadership were still significant [10]. This outcome is more aligned with the contextual configuration rather than standardized rollout. Comparative literature also cautions that commercial frameworks have tended to be more focused on than the principles, and evidence is not consistent across methods [4]. Therefore, a minimum coordination structure should be developed to manage the level of actual coupling and risk in an organization, and activities that do not contribute to the quality of decisions, integration or learning should be eliminated. As shown in Table 1 Comparative Characteristics of Enterprise Agile Scaling Approaches.

Table 1 Comparative Characteristics of Enterprise Agile Scaling Approaches

Approach	Primary coordination logic	Governance strength	Main advantage	Principal limitation
SAFe	ART cadence, PI planning, portfolio/value-stream alignment	High; explicit portfolio, architecture and roles	Strong visibility for regulated, multi-team programs	Can become role-heavy and ceremonial if adopted without tailoring
LeSS	One product backlog and feature teams with minimal added structure	Moderate; product-centered	Preserves simplicity and whole-product focus	Requires major structural change and mature product ownership
Scrum@Scale	Network of Scrum teams and coordinating cycles	Moderate; modular	Scales incrementally and supports decentralization	Cross-portfolio controls and architecture need local design
Spotify-inspired	Autonomous squads aligned through tribes, chapters and guilds	Low to moderate; culture-led	Encourages autonomy and communities of practice	Often copied without the enabling culture or operating context
Contextual hybrid	Selected practices combined with existing assurance and domain controls	Variable; intentionally designed	Fits legacy, vendor, regulatory and product constraints	Can become incoherent unless design principles and metrics are explicit

International Research Journal on Advanced Engineering Hub (IRJAEH)
e ISSN: 2584-2137
Vol. 04 Issue: 10 OCtober 2026
Page No: 5873 - 5882
https://irjaeh.com
https://doi.org/10.47392/IRJAEH.2026.0759

5. 5. Lean Governance: Control Through Flow and Evidence

Lean governance shifts the focus and when of control. Under traditional governance, there is typical approval of temporary projects with detailed forecasts and variance is reviewed once there is significant spend. For volatile digital work, an initial level of precision can be artificial, with requirements, technical feasibility and customer value being gradually uncovered. Lean governance, however, sets up investment horizons, capacity allocations, work-in-progress limits, thresholds for decision-making and measurable hypotheses. Portfolio Kanban helps to visualize demand and bottlenecks and frequent review helps to kill weak options before they suck up disproportionate capacity. A study of seven agile and Lean software companies revealed selective adaptation of traditional portfolio approaches and an increasing adoption of Kanban as a means to enable portfolio visibility and customer feedback [9]. This model is not an oversight elimination model. It distinguishes between routine decisions that are reversible and those that are consequential and/or regulated. Teams can deploy within guardrails, but when material changes happen, such as security, privacy, and model risk, financial exposure, or architecture boundaries, additional evidence and approval is required. Enterprise portfolio research suggests that scaling frameworks can help with alignment, if the leadership capability and organizational context are considered [10]. This also exemplifies how formal process expectations and agile portfolio practices are not necessarily at odds with each other, but can be aligned with CMMI V2.0 governance practices with SAFe 5.0 [11]. This results in a continuous, risk tiered and traceable control model. The choice of lean and agile practices should, however, be conscious. In a review of current agile/Lean practices, the potential dangers of implementing practices without considering the organizational context are pointed out [12]. There are 4 dimensions of useful governance metrics. Examples of flow measures are lead time, throughput, work-item age and dependency wait. Outcomes measures are related to adoption, customer benefit, revenue, mission impact or cost avoidance. Reliability measures are: change-failure rate, recovery time, escaped defects and security exposure. Experiment cycle time, hypothesis retirement and proportion of investment redirected using evidence are learning measures. A balanced set prevents teams from optimizing output at the expense of operational and/or governance costs [9] [11] [12].

6. The Execution Layer of Enterprise Agility: DevOps

If approved changes are stuck in integration, testing, release or infrastructure queues, then the agility of the portfolio is restricted. Continuous integration, delivery and deployment puts the strategic adaptability into executable capability. A systematic review of 69 studies revealed that automation, build and test acceleration, visibility, defect detection, deployment reliability, and secure scalable pipelines are central mechanisms, and the architecture, skills, infrastructure, and testing effort are contextual requirements [13]. So, DevOps is about organizational design and technical flow: ownership is extended from idea to production behavior and evidence is created continuously and not at release gates. Don't make DevOps about tooling. A critical review characterizes it by integrated development and operations responsibilities and points out that there is conceptual ambiguity itself as an obstacle to adoption [14]. Systematic mapping also demonstrates that DevOps is related to software quality, and that practices and results are dependent on the configuration of automation, collaboration and measurement [15]. According to a wide-ranging survey, DevOps can be defined as a multidisciplinary practice that is the combination of continuous delivery with correctness and reliability [16]. Platform teams can offer reusable pipelines, environments, observability, identity controls and policy-as-code, and stream-aligned teams can make use of these features without handing over product responsibility to a central release team. Lifecycle mapping has been conducted recently, which further underscores the importance of spreading DevOps capabilities throughout the planning, coding, integration, deployment, operations and

International Research Journal on Advanced Engineering Hub (IRJAEH)
e ISSN: 2584-2137
Vol. 04 Issue: 10 OCtober 2026
Page No: 5873 - 5882
https://irjaeh.com
https://doi.org/10.47392/IRJAEH.2026.0759

learning phases, instead of focusing them at the end of the development process [17]. This is a direct governance implication – evidence should be produced by the delivery system. An auditable chain can be built from automated tests, dependency scans, infrastructure validation, approval records, provenance attestations, deployment telemetry and rollback outcomes. Manual review is then kept for ambiguous, novel or high impact conditions. This way decreases the control latency and enhances the timeliness and repeatability of evidence of assurance [13], [16], [17].

7. AI-Ready Delivery Models

AI-ready delivery is longer software delivery as an AI service requires more than just executable software. Behavior might change due to training and evaluation data, features, model binaries, prompt, retrieval indexes, configuration, policies, and runtime context. A taxonomy of MLOps consists of a sequence from business understanding to data acquisition, model development and testing, continuous integration, continuous delivery, continuous training, continuous monitoring, explainability and sustainability [18]. Therefore, a release record needs to be able to track not just a source-code commit, but also the data and model lineage, evaluation protocol, intended use, approval basis and monitoring thresholds of the deployed system. The evidence also indicates that MLOps is still organisationally immature. A systematic survey of 93 studies revealed that there was a significant opportunity for the use of MLOps and AIOps, but the use was not consistent throughout the lifecycle stages [19]. A model that has been verified in three big software companies express maturity as an incremental integration of activities instead of the purchase of one platform [20]. In a subsequent multivocal review, practices and challenges from peer reviewed and practitioner literature are brought together, highlighting the socio-technical scope of production ML [21]. For enterprise programs, these findings suggest that the readiness of AI is dependent on the cross-functional product teams linking domain expertise, data engineering, ML engineering, software engineering, security, risk, and operations by using a shared backlog and evidence model. Trustworthiness too must be in operation, not as a post hoc ethics review. Monitoring is essential for data quality, data drift, performance degradation, failure modes, and recovery behavior in production in order to have robust MLOps [22]. A set of expert research mapping AI governance to design, development and operation revealed several concepts of governance across the various roles and stages of the lifecycle [23]. The responsible-AI patterns also categorize actionable practices according to multilevel governance, trustworthy processes and responsible-by-design product mechanisms [24]. A systematic review of AI governance structures poses the questions of who is responsible, what is being governed, when, and how [25]. These studies all combine to help support the use of risk-tiered controls that are integrated into delivery workflows and re-evaluated when the context changes.

8. Integrated Operating Model

The review is summarized in a 5-layer operating model in Figure 1. The foundation tests, codes and data, and models in CI/CD/continuous-training pipelines. Product and platform delivery provides ongoing ownership, re-use of engineering capabilities and operational telemetry. SAFe style program coordination only synchronizes the dependencies that need to be collectively planned. Lean governance establishes investment limits, risk levels and evidence requirements. Intelligence and feedback: Policy and investment revised with flow, outcome, reliability and model signals. So, policy cascades down as intent, constraint and threshold, and evidence cascades up as observed performance and risk. The loop is completed when governance decisions result in actual changes to funding, architecture priorities and/or controls [11, 17, 23-25]. As shown in Figure 1 Integrated operating model for SAFe coordination, Lean governance, and AI-ready delivery. As shown in Table 2. Governance Domains, Delivery Controls, and Evidence.

International Research Journal on Advanced Engineering Hub (IRJAEH)
e ISSN: 2584-2137
Vol. 04 Issue: 10 OCtober 2026
Page No: 5873 - 5882
https://irjaeh.com
https://doi.org/10.47392/IRJAEH.2026.0759

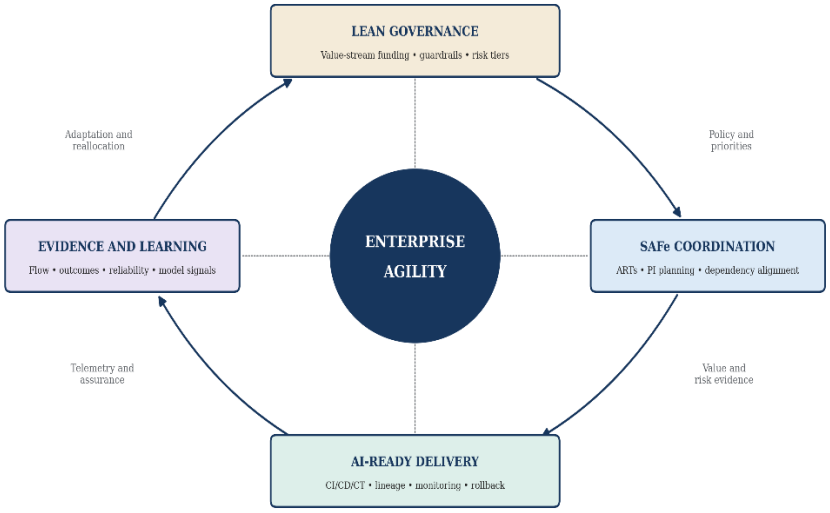


Figure 1 Integrated operating model for SAFe coordination, Lean governance, and AI-ready delivery

Governance domain	Embedded control	Primary evidence	Review trigger
Investment and value	Value-stream funding; hypotheses; WIP limits	Outcome trend, cost of delay, flow and benefit realization	Material outcome variance or persistent queue growth
Architecture and integration	Interfaces, architecture runway, automated compatibility tests	Build health, dependency age, integration frequency, system demo	Cross-stream dependency or architectural threshold breach
Software quality and security	CI/CD quality gates, policy-as-code, provenance and rollback	Test coverage, vulnerabilities, change-failure rate, recovery time	Failed gate, critical exposure or reliability degradation
Data and model integrity	Versioned data/model assets, evaluation suites and lineage	Data quality, reproducibility, model card, evaluation results	Material data change, retraining, or performance shift
Responsible AI	Risk classification, human oversight, explainability and use constraints	Bias/robustness tests, approvals, incident and monitoring records	New use case, affected population, drift or harm signal
Organizational learning	Inspect-and-adapt, retrospectives and portfolio review	Retired hypotheses, action closure, policy changes and learning time	Repeated exception, stalled improvement or changed strategy

International Research Journal on Advanced Engineering Hub (IRJAEH)
e ISSN: 2584-2137
Vol. 04 Issue: 10 OCtober 2026
Page No: 5873 - 5882
https://irjaeh.com
https://doi.org/10.47392/IRJAEH.2026.0759

9. Measurement and Evaluation Architecture

Evaluation should tie to strategic outcomes, delivery and operational signals, but not reduce to a maturity score. The main issues at portfolio level are: Is there any possibility to re-direct investment? Are there any realized benefits that justify continued funding? Is there capacity focused on the highest value constraints? Value-stream level – look at end-to-end lead time, work-item age, throughput stability, dependency delay, and the ratio of active to waiting time. Measures for delivery should involve deployment frequency, change-failure rate, recovery time, escaped defects, security remediation time and platform adoption. In addition, for AI services, data-quality trends, evaluation reproducibility, model or prompt version traceability, drift, subgroup performance, human override, incident recurrence, and readiness for rollback are all important considerations [13], [17]-[19], [22]. When indicators are turned into team targets, measurement is counterproductive. Velocity is a local measure and shouldn't be used to compare teams; frequent deployments are not useful if they create instability; if calibration, robustness, fairness, and/or operational utility are important, accuracy is not sufficient. Metrics should rather be considered as a set of evidence which is connected and be reviewed at a cadence suitable for the risk. Leading indicators (queue age, failed controls, drift etc.) are used for early intervention, while lagging indicators (realized value, incidents or customer retention etc.) are used to determine whether the intervention was effective. There should be an explicit decision to continue, adapt, stop, escalate or change a guardrail as a result of each review. This decision trace helps to complete the governance circle and enables organisations to differentiate between "learning" and "reporting" [9] [11] [16] [20] [23] – [25].

The implementation conditions and failure modes are listed below. Stable ownership around value is the first implementation condition. Temporary projects maximize delivery to a handoff date; persistent product or value-stream teams have the operating knowledge and can react to evidence after the project is handed off. The second one is selective synchronization. PI planning or similar is worth it if teams have interdependencies, but if everyone's work is scheduled to the same cadence, then batching can occur. The third one is the protected technical capacity. Funding of architecture, automation, security, data quality and observability should be considered delivery work instead of overhead. The fourth is governance participation: finance, risk, procurement, security, legal and operations have to change the way they interact with delivery teams from being outside an approval queue [1]-[4] and [10]. Failure modes tend to reinforce each other. When creating framework maximalism, you're adding more roles and meetings but not reducing the number of dependencies. Local velocity targets promote backlog production and hide system delay. Annual budgeting embeds low expectations, and makes it hard to stop work politically. When standards are not translated into reusable platforms and automated controls, queues are built up in the central architecture or on security boards. Without being coupled with software integration, data ownership, monitoring and accountability, AI pilots are particularly vulnerable. Lastly, transformations fade out when coaching is not continued beyond the end of the initial coaching. These problems provide an explanation for the apparent conformance and slow releases and limited strategic adaptability [5, 8, 14, 19, 21]. A safer transformation sequence starts with a set of diagnostic baselines and a few value streams. Organizations can capture the existing lead time, dependency wait, release frequency, failure recovery, outcome realization and control delay; visualize the decisions that introduce queues; and set guardrails to delegated authority. They can then add an ART or other coordination structure if there is a need for it because the dependency density is high, and develop platform capabilities that will lessen the need for coordination in the future. AI use cases should be categorized by impact and provided with impact, evaluation, monitoring, and fallback controls from the start. Reallocation of capacity should be based on evidence and be done on a quarterly basis (not just reapproving existing commitments) [9,13,18,23-25].

10. Discussion

International Research Journal on Advanced Engineering Hub (IRJAEH)
e ISSN: 2584-2137
Vol. 04 Issue: 10 OCtober 2026
Page No: 5873 - 5882
https://irjaeh.com
https://doi.org/10.47392/IRJAEH.2026.0759

The literature as a whole indicates that enterprise agility can be seen as a form of decentralization that is constrained. SAFe adds terms and synchronisation, Lean governance adds economic and risk boundaries, DevOps adds rapid and reproducible execution, and MLOps adds delivery evidence to data and model behaviour. No one is enough on its own. Bureaucracy can be digitized without Lean decision reform in SAFe. Lean portfolio practices and no engineering automation = no delivery shortening. The absence of strategic alignment can cause DevOps to drive the creation of low value output. If there is no lifecycle governance, MLOps can industrialize models that are not sufficiently controlled in terms of their use, accountability, or behavior [4, 11, 13, 18, 23].

The review also shows evidence gaps. Many of the large-scale agile studies are based on case studies and experience reports and it is challenging to make causal inferences and important to include contextual information [1, 4]. Adoption of commercial framework is frequently measured by perceptions, process measures or individual organizations, not by comparative results over time. Studies on MLOps and AI-governance are more recent and often taxonomic and conceptual or on a small number of cases [19] - [25]. Any statements that one framework always leads to increased productivity, quality or business value are thus beyond the current evidence. Research should be conducted to compare matched value streams over time, as well as include measures of implementation fidelity and adaptation, and include strategic response time, delivery, reliability, workforce, and risk outcomes. Yet, there are a number of mechanisms that have strong convergence: visible work, limited queues, cross functional ownership, rapid integration, automated evidence, explicit decision rights, architecture that reduces coupling, and feedback from actual use. These mechanisms are more long lasting than branded roles. Thus, the leader should think of frameworks as hypotheses about an organization that can be tested. Practices should be left in place if they enhance the performance of the system and changed or eliminated if they delay the system without any commensurate coordination or assurance. This position brings agility in line with scientific learning: intervention, observation and revision are in place of both the prescription of the curriculum and the local improvisation that is often unstructured [2,3,5,7].

Conclusion

Scaling enterprise agility is a re-engineering of the enterprise operating system for complex digital work. There is no evidence to support the unconditional adoption of a single scaling framework or replication of team level methods. For portfolios that have high dependency and assurance requirements, SAFe can be an important coordination facilitator, but it should be customized and assessed for its effectiveness, not for compliance. Lean governance provides timely investment and control via value-stream funding, work-in-process limits, delegated authority and risk-tiered evidence. DevOps and MLOps empower these principles to be actionable by turning code, data, model, security and operational changes into traceable, continuously evaluated delivery flows. A closed-loop system where policy is cascaded as guardrails and evidence is cascaded back as a means to shift strategy, funding, architecture and control is the most defensible target state. The key to enterprise agility is the ability to learn and pivot safely at the pace required by the environment.

References

[1]. K.-K. Dikert, M. Paasivaara, and C. Lassenius, “Challenges and success factors for large-scale agile transformations: A systematic literature review,” Journal of Systems and Software, vol. 119, pp. 87–108, 2016, doi: 10.1016/j.jss.2016.06.013.

[2]. M. Paasivaara, B. Behm, C. Lassenius, and M. Hallikainen, “Large-scale agile transformation at Ericsson: A case study,” Empirical Software Engineering, vol. 23, pp. 2550–2596, 2018, doi: 10.1007/s10664-017-9555-8.

[3]. M. Kalenda, P. Hyna, and B. Rossi, “Scaling agile in large organizations: Practices, challenges, and success factors,” Journal of Software: Evolution and Process, vol. 30, no. 10, Art. no. e1954, 2018, doi:

International Research Journal on Advanced Engineering Hub (IRJAEH)
e ISSN: 2584-2137
Vol. 04 Issue: 10 OCtober 2026
Page No: 5873 - 5882
https://irjaeh.com
https://doi.org/10.47392/IRJAEH.2026.0759

10.1002/smr.1954.

[4]. H. Edison, X. Wang, and K. Conboy, "Comparing methods for large-scale agile software development: A systematic literature review," IEEE Transactions on Software Engineering, vol. 48, no. 8, pp. 2709–2731, 2022, doi: 10.1109/TSE.2021.3069039.

[5]. N. Carroll, K. Conboy, and X. Wang, "From transformation to normalisation: An exploratory study of a large-scale agile transformation," Journal of Information Technology, vol. 38, no. 3, pp. 267–303, 2023, doi: 10.1177/02683962231164428.

[6]. T. Dingsøyr, N. B. Moe, T. E. Fægri, and E. A. Seim, "Exploring software development at the very large-scale: A revelatory case study and research agenda for agile method adaptation," Empirical Software Engineering, vol. 23, pp. 490–520, 2018, doi: 10.1007/s10664-017-9524-2.

[7]. K. H. Rolland, B. Fitzgerald, T. Dingsøyr, and K.-J. Stol, "Acrobats and safety nets: Problematizing large-scale agile software development," ACM Transactions on Software Engineering and Methodology, vol. 33, no. 1, pp. 1–45, 2024, doi: 10.1145/3617169.

[8]. K. Conboy and N. Carroll, "Implementing large-scale agile frameworks: Challenges and recommendations," IEEE Software, vol. 36, no. 2, pp. 44–50, 2019, doi: 10.1109/MS.2018.2884865.

[9]. M. O. Ahmad, L. E. Lwakatare, P. Kuvaja, M. Oivo, and J. Markkula, "An empirical study of portfolio management and Kanban in agile and lean software companies," Journal of Software: Evolution and Process, vol. 29, no. 6, Art. no. e1834, 2017, doi: 10.1002/smr.1834.

[10]. E. D. Suárez-Gómez and C. A. Hoyos-Vallejo, "Scalable agile frameworks in large enterprise project portfolio management," IEEE Access, vol. 11, pp. 98666–98684, 2023, doi: 10.1109/ACCESS.2023.3312728.

[11]. V. Henriquez, J. A. Calvo-Manzano, A. M. Moreno, and T. S. Feliu, "Agile governance practices by aligning CMMI V2.0 with portfolio SAFe 5.0," Computer Standards & Interfaces, vol. 91, Art. no. 103881, 2025, doi: 10.1016/j.csi.2024.103881.

[12]. S. Gutierrez-Rios, I. Morales, and M. Vargas-Lombardo, "A study of new decade of agile and lean methodologies for development software," in Proc. 4th International Conference on Information and Computer Technologies (ICICT), 2021, pp. 122–130, doi: 10.1109/ICICT52872.2021.00028.

[13]. M. Shahin, M. A. Babar, and L. Zhu, "Continuous integration, delivery and deployment: A systematic review on approaches, tools, challenges and practices," IEEE Access, vol. 5, pp. 3909–3943, 2017, doi: 10.1109/ACCESS.2017.2685629.

[14]. M. Gall and F. Pigni, "Taking DevOps mainstream: A critical review and conceptual framework," European Journal of Information Systems, vol. 31, no. 5, pp. 548–567, 2022, doi: 10.1080/0960085X.2021.1997100.

[15]. A. Mishra and Z. Otaiwi, "DevOps and software quality: A systematic mapping," Computer Science Review, vol. 38, Art. no. 100308, 2020, doi: 10.1016/j.cosrev.2020.100308.

[16]. L. Leite, C. Rocha, F. Kon, D. Milojicic, and P. Meirelles, "A survey of DevOps concepts and challenges," ACM Computing Surveys, vol. 52, no. 6, pp. 1–35, 2019, doi: 10.1145/3359981.

[17]. R. Amaro, R. Pereira, and M. M. da Silva, "Mapping DevOps capabilities to the software life cycle: A systematic literature review," Information and Software Technology, vol. 177, Art. no. 107583, 2025, doi: 10.1016/j.infsof.2024.107583.

[18]. M. Testi et al., "MLOps: A taxonomy and a methodology," IEEE Access, vol. 10, pp. 63606–63618, 2022, doi: 10.1109/ACCESS.2022.3181730.

[19]. J. Díaz-de-Arcaya, A. I. Torre-Bastida, G.

International Research Journal on Advanced Engineering Hub (IRJAEH)
e ISSN: 2584-2137
Vol. 04 Issue: 10 OCtober 2026
Page No: 5873 - 5882
https://irjaeh.com
https://doi.org/10.47392/IRJAEH.2026.0759

Zárate, R. Miñón, and A. Almeida, "A joint study of the challenges, opportunities, and roadmap of MLOps and AIOps: A systematic survey," ACM Computing Surveys, vol. 56, no. 4, pp. 1–30, 2024, doi: 10.1145/3625289.

[20]. M. M. John, H. H. Olsson, and J. Bosch, "Towards MLOps: A framework and maturity model," in Proc. 47th Euromicro Conference on Software Engineering and Advanced Applications (SEAA), 2021, pp. 1–8, doi: 10.1109/SEAA53835.2021.00050.

[21]. B. Eken, S. Pallewatta, N. Tran, A. Tosun, and M. A. Babar, "A multivocal review of MLOps practices, challenges and open issues," ACM Computing Surveys, vol. 58, no. 2, pp. 1–35, 2026, doi: 10.1145/3747346.

[22]. F. Bayram and B. S. Ahmed, "Towards trustworthy machine learning in production: An overview of the robustness in MLOps approach," ACM Computing Surveys, vol. 57, no. 5, pp. 1–35, 2025, doi: 10.1145/3708497.

[23]. S. Laato, T. Birkstedt, M. Mäntymäki, M. Minkkinen, and T. Mikkonen, "AI governance in the system development life cycle: Insights on responsible machine learning engineering," in Proc. 1st International Conference on AI Engineering: Software Engineering for AI (CAIN), 2022, pp. 113–123, doi: 10.1145/3522664.3528598.

[24]. Q. Lu, L. Zhu, X. Xu, J. Whittle, D. Zowghi, and A. Jacquet, "Responsible AI pattern catalogue: A collection of best practices for AI governance and engineering," ACM Computing Surveys, vol. 56, no. 7, pp. 1–35, 2024, doi: 10.1145/3626234.

[25]. [25] A. Batool, D. Zowghi, and M. Bano, "AI governance: A systematic literature review," AI and Ethics, vol. 5, pp. 3265–3279, 2025, doi: 10.1007/s43681-024-00653-w.