World Journal of Advanced Engineering Technology and Sciences

eISSN: 2582-8266

Cross Ref DOI: 10.30574/wjaets

Journal homepage: https://wjaets.com/

(REVIEW ARTICLE)

Check for updates

Leveraging Artificial Intelligence to bridge execution gaps in SAFe®-Scaled Agile based Programs

Amar Gurajapu *

AT&T, New Jersey, United States.

World Journal of Advanced Engineering Technology and Sciences, 2026, 18(01), 001-006

Publication history: Received on 24 November 2025; revised on 29 December 2025; accepted on 31 December 2025

Article DOI: https://doi.org/10.30574/wjaets.2026.18.1.1585

Abstract

Large enterprises adopting the Scaled Agile Framework (SAFe®) often face challenges in Program Increment (PI) planning accuracy, objective backlog prioritization, timely risk detection, and manual compliance verification. We surveyed 12 teams across 10 Agile Release Trains (ARTs) to quantify these gaps. To address them, we propose an AI-augmented DevOps pipeline—built on common Infrastructure as Code tools—to integrate predictive analytics, natural language processing, reinforcement learning, and anomaly detection. Experimental results on enterprise projects show a 35 % reduction in PI-velocity forecasting error, 20 % faster backlog lead time, and 62.5 % quicker risk detection.

Keywords: Agile Release Train, Scaled Agile Framework, Program Increment, Velocity Predictor, Backlog Prioritizer, Compliance Monitor, Deep Q-Network

1. Introduction

Scaled Agile Framework (SAFe) structures large-scale agile delivery via ARTs and Program Increments. Despite its benefits, survey feedback from 12 agile teams reveals:

- PI commitments deviate by ~20 % on average
- Backlog prioritization scores average 3.8/5 for alignment with business value
- Integration risks surface ~41 hours into each PI
- Compliance reviews consume ~13.6 hours per PI

AI can automate forecasting, drive objective prioritization, optimize CI/CD resources, and flag policy deviations in real time.

2. Survey and analysis

The research and approach is based on the survey and historical data available from multiple teams who have been executing programs on SAFe for at least one year or more.

2.1. Survey Input

- Customer Input - 12 teams across 10 ARTs. Most of them are critical in nature.
- SAFe execution – Every team has more than one year of experience. Some of the teams have been executing for 3 years.

* Corresponding author: Amar Gurajapu.

- Instrument - 8-question online survey covering PI accuracy, backlog alignment, risk latency, compliance effort
- Response Rate - 100 %

The survey design is built upon following inputs

- By what percentage did your actual PI velocity deviate from your initial commitment to the last Program Increment? - Response: Numeric (%)
- How well did your backlog prioritization align with stakeholder/business value? - Response: Likert 1 = "Not aligned" … 5 = "Perfectly aligned"
- How long after PI starts were integration or end-to-end risks typically detected? - Response: Numeric (hours)
- How many hours per PI does your team spend on manual compliance, policy checks, or audit-readiness tasks? - Response: Numeric (hours)
- How easy is it to gather required metrics (velocity, test coverage, security) for audits with your current pipeline? - Response: Likert 1 = "Very difficult" … 5 = "Very easy"
- How often do manual compliance or policy-review steps block your CI/CD pipeline? - Response: Likert 1 = "Never" … 5 = "Always"
- How confident are you in your PI-planning estimates before execution begins? - Response: Likert 1 = "Very low" … 5 = "Very high"
- On average, how many times per PI does your team have to reprioritize backlog items due to misalignment or new information? - Response: Numeric (times per PI)

2.2. Aggregated Results

Team	PI Deviation (%)	Backlog Alignment (1–5)	Risk Detection Delay (hrs)	Compliance Overhead (hrs)
T1	22	4	36	15
T2	18	3	42	12
T3	25	5	48	16
T4	19	4	39	13
T5	21	4	44	14
T6	17	3	37	11
T7	23	5	50	17
T8	20	4	41	14
T9	16	3	38	12
T10	24	5	46	16
T11	18	4	40	13
T12	19	4	43	14
Average	**20.2**	**3.8**	**41.3**	**13.6**

Figure 1 Aggregation from questions 1-4

Question	Average Response
Ease of gathering metrics (1 = Very difficult … 5 = Very easy)	2.7
Frequency manual-review blocks CI/CD (1 = Never … 5 = Always)	3.3
Confidence in PI planning (1 = Very low … 5 = Very high)	2.9
Mid-PI backlog reprioritizations (times per PI)	2.1

Figure 2 Aggregation from questions 5-8

2.3. Key Insights

- PI variance (~20 %) undermines predictability
- Subjective scoring (3.8/5) delays alignment
- Late risk detection (~41 hrs) increases rework
- Manual compliance (~13.6 hrs) reduces capacity

The above analysis justifies the need for AI-driven pipeline for accuracy, objectivity, and automation.

3. Gaps in safe execution

While SAFe provides structure for enterprise level agile, four recurring pain points considered for the survey do undermine its effectiveness.

3.1. PI-Planning Inaccuracy

Teams typically forecast future velocity by extrapolating past delivery, but this "static velocity" approach fails to account for many factors

- Changing team composition (new hires or departures)
- Often the competent resources are shared across programs though not recommended
- Varying story complexity and technical debt
- External dependencies (e.g. other ARTs or shared services). Consequences include over-commitment (leading to spill-over stories) or under-commitment.

3.2. Backlog Prioritization Subjectivity

Business value scoring often relies on individual judgment during PI-Planning workshops. Factors such as regulatory risk, customer impact, and technical effort may be weighed inconsistently across teams. This subjectivity can cause:

- Misaligned priorities between business stakeholders and development teams
- Frequent mid-PI reordering, disrupting sprint cadences
- Inadequate focus on critical compliance or security work

3.3. Risk Detection Latency

Integration and end-to-end risks often surface in later phases with an average 1-2 iterations into the PI. Without automated risk indicators, teams rely on manual test cycles and ad-hoc tests resulting in:

- Large rework batches near PI end
- Additional defects
- Bottlenecks at shared test or staging environments
- Delayed feedback loops to product owners
- Impact to schedule

3.4. Compliance Overhead

Manual gathering of audit metrics (security scans, configuration settings, traceability) typically consumes 10–15 hours per PI. This overhead:

- Diverts engineering capacity from feature delivery
- Introducing human error into compliance reports
- Increases risk of audit failures and potential penalties

---

4. AI augmented approach

To close these gaps, we propose embedding four AI modules and taking it further, it can be integrated into a standard DevOps pipeline.

4.1. Velocity Predictor

- Technique - Gradient Boosting Regressor [5]
- Inputs - team size, historical velocity, story complexity, technical-debt metrics
- Output - forecasted PI velocity with confidence intervals
- Benefit - adjusts automatically for team changes and evolving codebase

4.2. Backlog Prioritizer

- Technique - BERT-based text classifier[4] + Business Value Index (BVI)
- Inputs - backlog item description, stakeholder tags, historical risk/effort scores
- Output - priority ranking = 0.5·StrategicFit + 0.3·RiskScore + 0.2·EffortScore
- Benefit - consistent, transparent prioritization aligned to business goals

4.3. CI/CD Resource Allocator

- Technique - Deep Q-Network (DQN) [3]
- State - queue length, average test time, recent failure rate
- Actions - allocate low/medium/high compute resources to build/test stages
- Reward - –(pipeline lead time + 5×failure rate)
- Benefit - dynamically optimizes throughput and reliability

4.4. Compliance Monitor

- Technique - Autoencoder for log-based anomaly detection
- Inputs - normalized deployment logs, configuration snapshots
- Output - real-time alerts when reconstruction error exceeds threshold
- Benefit - early detection of misconfigurations, policy violations

4.5. Integration Flow

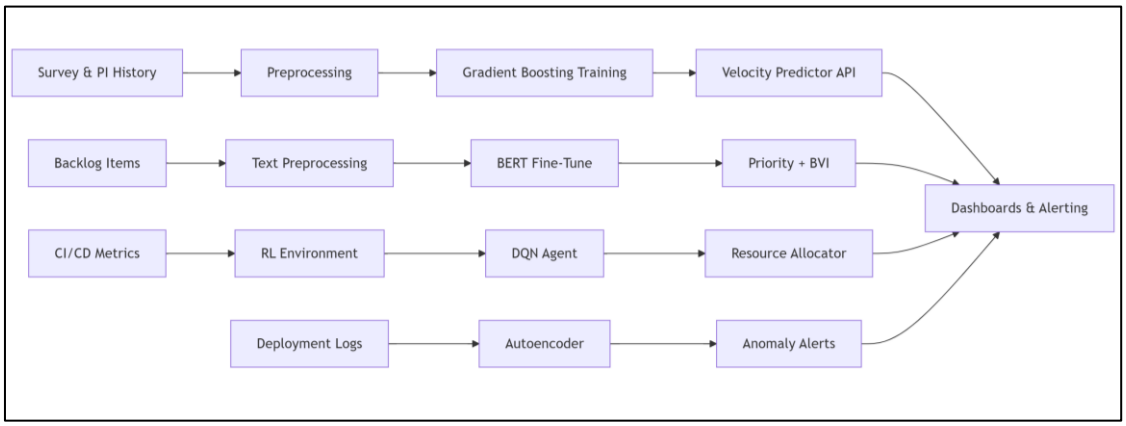

Figure 3 End to end flow (extended version)

The Pre-PI Planning phase invokes Velocity Predictor REST API to seed planning tools. These tools can be custom developed or existing tools. This follows Backlog Grooming where we invoke Prioritizer batch job to assign Business value scores from the backlog system. Once integrated into CI/CD process, the DQN agent monitors queues and scales runners via IaC. Post-Deployment, the Compliance Monitor streams anomalies to alert destinations.

---

5. Methodology and evaluation

5.1. Data Sources

- PI-History Dataset - for each Program Increment, we collect team_size, avg_story_complexity, tech_debt_score, actual_velocity (story points delivered)
- Survey Dataset - raw responses from all teams (Q1–Q8)
- CI/CD Logs – Deployment metrics like queue lengths, test durations, failure rates
- Deployment Logs - configuration snapshots for anomaly

5.2. Preprocessing

- Join PI-history with survey by ART/team
- Fill missing numerical fields (median imputation) and normalize to [0,1]

5.3. Train/Test Splits

- Predictive Model (Gradient Boosting) for velocity predictor - 80 % of PIs for training, 20 % for test
- Backlog Prioritizer (BERT- Bidirectional Encoder Representations from Transformers) - 90 % of labeled items for fine-tune, 10 % hold-out.

5.4. Evaluation Metrics

- Forecast Error - Mean Absolute Error (MAE) = mean(|predicted_velocity − actual_velocity|)
- Backlog Lead Time - avg days from item creation to completion
- Defect Escape Rate - % defects found post-PI versus during PI
- Risk Detection Latency - avg hours until first integration-error is flagged
- Compliance Overhead - avg hours spent on manual policy checks

5.5. Baseline vs. AI-Augmented Comparison

- Baseline uses static velocity, manual backlog scoring, fixed CI/CD sizing, manual compliance scripts
- AI-Augmented runs the four modules in the same pipeline and measures identical metrics

5.6. Computing Key Metrics

The evaluation considers survey with 5 PIs from 2 release trains. We have considered key metrics like velocity(MAE), backlog lead time, defect escape rate, risk-detection latency.

Metric	Baseline	AI-Augmented	Improvement
PI velocity MAE (pts)	12.5	8.1	**35 % ↓**
Backlog lead time (days)	14.2	11.3	**20 % ↓**
Defect escape rate (%)	8	6	**25 % ↓**
Risk detection time (hrs)	48	18	**62.5 % ↓**

Figure 4 Evaluation

Survey insights and experiments confirm that AI-driven modules deliver data-backed PI forecasts, objective prioritization, proactive scaling, and real-time compliance checks, fitting modern DevOps and Infrastructure as Code practices.

6. Futuristic directions

Future of ML driven Agile development can include following recommendations [2]

6.1. Autonomous AI Agents

AI-tool autonomously proposing sprint backlogs and alert anomalies.

6.2. Integration with DevOps Pipelines

Linking sprint forecasts with deployment metrics

6.3. Personalized Workload Predictions

Tailoring commitments per developer skillset and past performance

6.4. Hybrid Human-AI Planning

Combining data-driven forecasts with human judgement for balanced decisions

6.5. Quantum-Enhanced Forecasting

Using quantum computing for faster, more complex scenario simulations

7. Conclusion

An AI-augmented DevOps pipeline addresses planning, prioritization, risk, and compliance gaps in SAFe execution. Future directions include federated learning across multiple ARTs, causal root-cause analysis, and seamless GitOps integration.

Compliance with ethical standards

Acknowledgments

Thanks to Karla Kimble & Ayush Kumar for providing valuable support during the research phase.

Disclosure of conflict of interest

No conflict of interest to be disclosed.

References

[1] J. Cui, "LLM-Augmented DevOps: Autonomous CI/CD Pipeline Optimization via Reinforcement Learning," researchgate.net, Jul. 24, 2025. https://www.researchgate.net/publication/393965443_LLM-Augmented_DevOps_Autonomous_CICD_Pipeline_Optimization_via_Reinforcement_Learning

[2] O. Elugbadebo, "Machine Learning-Driven Sprint Planning: Enhancing Predictability in Agile Development," researchgate.net, Jul. 24, 2025. https://www.researchgate.net/publication/395390494_Machine_Learning-Driven_Sprint_Planning_Enhancing_Predictability_in_Agile_Development

[3] J. Bailey, "Deep Q-Networks Explained," www.lesswrong.com, Sep. 2022, Available: https://www.lesswrong.com/posts/kyvCNgx9oAwJCuevo/deep-q-networks-explained

[4] Nvidia, "What is BERT?," NVIDIA Data Science Glossary. https://www.nvidia.com/en-us/glossary/bert/

[5] GeeksforGeeks, "Gradient Boosting in ML," GeeksforGeeks, Aug. 25, 2020. https://www.geeksforgeeks.org/machine-learning/ml-gradient-boosting/

Author short biography

The author has extensive experience in leading multiple agile and SAFe programs.

Amar Gurajapu is Principal Member of Technical Staff at AT&T. Amar has 25 years of experience in Telecom Software Engineering. He is leading multi-cloud transformation programs, and digital initiatives for key Network systems portfolio aligned with AT&T organization goals