Post-Velocity Software Engineering: Assurance Saturation in AI-Enabled Socio-Technical Systems

Alessio Bucaioni
Mälardalen University
Sweden
alessio.bucaioni@mdu.se

Abstract

Automation and large language models are increasing the rate at which software changes can be produced in AI-enabled socio-technical systems. Yet responsible digital transformation is not only about generating code, models, or system modifications; it also requires understanding, validating, contesting, and approving changes before they can be trusted in operational, organizational, or public-interest settings. These assurance and governance activities depend on human attention, organizational processes, accountable oversight, audit evidence, and the participation of institutions and affected stakeholders, and they do not scale at the same pace as automated generation. We argue that software engineering is approaching a regime in which the rate of change can exceed the capacity of teams, institutions, and affected communities to review, contest, and govern it responsibly. We call this regime *post-velocity software engineering*. Beyond this threshold, increasing development speed may no longer improve outcomes and may instead weaken justified confidence, reduce oversight quality, and shift risk downstream into operations, users, communities, and public institutions. This article introduces post-velocity software engineering as a governance saturation problem in AI-enabled socio-technical systems. It describes signals that organizations may be approaching this threshold and outlines a research agenda for understanding how AI-accelerated development reshapes the balance between software production, assurance, accountability, contestability, and community-in-the-loop governance.

CCS Concepts

• **Software and its engineering** → **Software organization and properties**; • **Human-centered computing** → **Collaborative and social computing**; • **Social and professional topics** → *Computing / technology policy*.

Keywords

AI-enabled socio-technical systems, community-in-the-loop governance, software assurance, accountable oversight, contestability, safe velocity

The ACM 6th International Conference on Information Technology for Social Good (GoodIT 2026), Pisa, Italy

ACM ISBN 978-1-4503-XXXX-X/2026/06
https://doi.org/XXXXXXX.XXXXXXX

ACM Reference Format:
Alessio Bucaioni. 2026. Post-Velocity Software Engineering: Assurance Saturation in AI-Enabled Socio-Technical Systems. In *Proceedings of (The ACM 6th International Conference on Information Technology for Social Good (GoodIT 2026)).* ACM, New York, NY, USA, 4 pages. https://doi.org/XXXXXXX.XXXXXXX

1 Introduction

In discussions on transport optimization, Rory Sutherland recounts a lesson from high-speed rail: once trains became fast enough, further reductions in travel time delivered little additional value for passengers despite disproportionate investment.[1] Operators therefore shifted attention from speed to journey quality, prioritizing comfort, reliability, and overall experience. The broader lesson is that complex systems eventually need to optimize for quality, reliability, and service rather than speed alone.

Software engineering may be approaching a similar threshold, particularly in AI-enabled socio-technical systems that affect organizations, public institutions, communities, and citizens, including AI-supported public services, energy systems, transport infrastructures, health and care platforms, and community-facing digital twins. Continuous integration and deployment (CI/CD), large-scale automation, and large language models (LLMs) have reduced the effort required to produce software changes [3]. In many organizations, development velocity has become a central indicator of engineering performance, even though the governance capacity needed to validate, contest, and approve these changes may not grow at the same pace. Before software can be trusted in operational, organizational, or public-interest settings, each change must be reviewed, validated, understood, contested, and approved. These assurance and governance activities depend on human attention, expertise, shared understanding, institutional responsibility, audit evidence, and accountable oversight, and they do not scale as easily as automated generation.

We call the resulting situation *post-velocity software engineering*. In this regime, the primary constraint on safe software delivery is no longer producing changes but understanding, validating, contesting, and responsibly approving them. As change volume approaches the limits of socio-technical governance capacity, oversight weakens: reviews become more procedural, evidence thins, and issues that might have been caught during development surface later in operations, users, communities, public institutions, or incident response [7, 9].

LLMs intensify this imbalance. A single prompt can generate multiple implementations, tests, refactorings, and related modifications within seconds. Although these artifacts may appear locally correct,

[1] https://www.youtube.com/shorts/7-4C1AwkIXg

assessing their architectural implications, operational safety, governance consequences, stakeholder exposure, and long-term maintainability still requires deeper reasoning. The bottleneck therefore shifts from producing changes to evaluating and governing them. Once this threshold is crossed, further acceleration may produce diminishing, or even negative, returns because the mechanisms that establish trust, auditability, contestability, accountable approval, and community-in-the-loop governance cannot keep pace.

As a conceptual and agenda-setting contribution, this article introduces post-velocity software engineering as a hypothesis about governance saturation in AI-enabled socio-technical systems. It motivates the limits of velocity (Section 2), defines the post-velocity regime and its observable signals (Section 3), and outlines a research agenda on AI-accelerated development, assurance, accountability, and governance (Section 4).

2 Background and related work

Research on software engineering productivity has long examined the relationship between development speed and software quality. Studies show that when acceleration is not matched by stronger validation and governance, quality degrades, defects increase, and maintainability work is deferred [7, 9]. These findings matter for AI-enabled socio-technical systems because technical degradation can also weaken the evidence, traceability, and accountability required for responsible governance.

This literature highlights tensions between speed and quality, but often assumes that assurance can scale through automation and improved processes. In practice, many assurance and governance activities rely on scarce human and institutional capabilities, including expert review, architectural reasoning, shared understanding, contestability, and accountable decision-making. AI-assisted development may intensify this imbalance: LLMs can rapidly generate code, tests, and documentation, increasing the supply of changes while shifting human effort from authoring to evaluating outputs under uncertainty [5, 6, 8]. This raises cognitive workload and risks such as automation bias, especially when oversight must remain meaningful rather than procedural.

Similar concerns arise in research on governance and accountability in complex AI-enabled socio-technical systems. In safety- and mission-critical contexts, frameworks emphasize traceable reasoning, evidence-based validation, and human approval before deployment [1, 2, 4]. Studies of autonomous and self-adaptive systems also highlight risks when systems operate faster than humans can meaningfully supervise them. Together, these strands reveal a growing tension: automation and AI make software changes easier to produce, while justified confidence depends on human judgment, organizational processes, independent assurance, and governance mechanisms that scale more slowly. What remains underexplored is how these dynamics interact under sustained acceleration, and whether the capacity to review, validate, contest, and responsibly approve changes becomes the limiting factor for accountable AI-enabled socio-technical systems.

3 Post-velocity software engineering

What should organizations optimize once faster delivery stops improving outcomes? Post-velocity software engineering holds that, when change outpaces assurance and governance capacity, the bottleneck shifts from producing changes to validating, contesting, and responsibly approving them. The hypothesis is that, beyond this point, further acceleration may yield diminishing, or even negative, returns: more changes are shipped, but justified confidence, oversight quality, accountability, and stakeholder contestability weaken.

Post-velocity software engineering describes a regime in which the rate of change approaches or exceeds the socio-technical capacity available to validate and govern those changes. Beyond this threshold, validation, shared understanding, contestability, and accountable approval become the primary constraints on safe delivery in AI-enabled socio-technical systems. Increasing throughput may therefore reduce justified confidence in individual changes and shift operational, institutional, or societal risk downstream.

3.1 Conceptual model

The post-velocity model centers on four elements: the *rate of change* (R), *assurance and governance capacity* (A), *justified confidence per change* (J), and *safe velocity* (V_s), the delivery rate an organization can sustain without eroding confidence, accountability, contestability, or safety. When $R < A$, teams, institutions, independent reviewers, and affected stakeholders can maintain justified confidence through substantive review, validation, evidence collection, and governance. As automation and AI increase the supply of candidate changes, R may approach or exceed A. As Figure 1 illustrates, review effort and supporting evidence then thin out, reducing J. In this regime, sustainable delivery is bounded by V_s: the maximum rate that preserves acceptable confidence, oversight, stakeholder exposure, and risk.

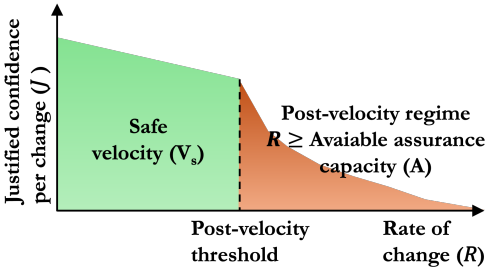

Figure 1: Conceptual view of post-velocity software engineering.

3.2 Observable signals

If post-velocity is real, it should leave traces in engineering and governance work. We expect three signals to be especially visible in development, operational, and governance data:

- *Validation lag (D1).* As change volume grows, substantive review and testing take longer, producing longer pull request queues, CI delays, review backlogs, or delayed assurance checks.
- *Procedural oversight (D2).* Approval shifts from evidence-based reasoning to symbolic approval, reflected in shorter review discussions, superficial approvals such as "LGTM", missing validation links, or incomplete audit trails.

Table 1: Illustrative indicators of post-velocity dynamics.

Element / signal	Operational proxies	Data sources	Expected signal when $R \gtrsim A$
R Rate of change	PRs/week; churn; diff size; deploy frequency; lead time	VCS; PR platforms; CI/CD logs; release logs	Change volume grows faster than validation and governance
A Assurance and governance capacity	Review minutes/change; reviewer load; CI minutes/PR; test evidence/change; triage, independent review, and governance capacity	PR traces; CI logs; incident systems; governance records	Review, testing, assurance, and governance queues saturate
J Justified confidence	Review depth; evidence completeness; post-change stability; auditability	PR text; test artefacts; static analysis; incident links; audit trails	Evidence thins; approvals become procedural
V_s Safe velocity	Risk-stratified max R under acceptable risk bounds	Governance policy; risk labels; component criticality; stakeholder exposure	Throughput exceeds accountable approval bandwidth
D1 Validation lag	Time-to-first-review; pending PRs; CI backlog; postmortem backlog	PR platforms; CI systems; incident tools	Validation backlog grows
D2 Procedural oversight	Rationale density vs. diff size; "LGTM" rate; missing evidence links	PR reviews; templates; policy checks; audit records	Symbolic approval replaces evidentiary review
D3 Downstream defect shifting	Rollbacks; change failure rate; incidents; production hotfixes; user- or community-reported issues	Deployment logs; incident reports; postmortems; support and feedback channels	Pre-release issues surface in operations, users, communities, or public institutions

- *Downstream defect shifting (D3).* Issues increasingly surface in operations or affected communities, indicated by rollbacks, change failures, incidents, production hotfixes, or user- and stakeholder-reported problems linked to recent deployments.

Individually, these signals may have multiple causes. Together, they suggest that change production is outpacing the organization's capacity to validate, understand, contest, and govern it. Table 1 links the model elements (R, A, J, V_s) and signals (D1–D3) to indicators in modern development and governance environments. In community-facing AI-STS, these indicators should also be read alongside governance traces such as stakeholder feedback, appeal records, audit records, and evidence used in institutional decision-making.

3.3 Boundary conditions

Post-velocity dynamics are not expected to arise uniformly across all projects. Threshold effects may be weak when changes are small and localized, modular architectures isolate components effectively, automated verification scales assurance evidence, or release cadences remain intentionally slow. Conversely, post-velocity conditions become more likely when changes are frequent and cross-cutting, systems are tightly coupled, and development occurs under strong reliability, safety, regulatory, public-interest, or governance constraints. In such contexts, maintaining justified confidence becomes a central challenge for accountable AI-enabled socio-technical systems, especially when operational decisions affect public institutions, communities, or citizens.

4 Research agenda

Our goal is to turn post-velocity software engineering from a conceptual framing into a research program for AI-enabled socio-technical systems. Building on the constructs and diagnostics in Section 3, we outline an empirical agenda for studying how change throughput, assurance capacity, accountability, contestability, and governance interact under sustained AI-driven acceleration across projects, organizations, public-interest settings, and domains.

4.1 Research questions

We highlight four research questions (RQs) that follow from the post-velocity perspective.

- *RQ1: Detecting post-velocity regimes.* Under what conditions does an organization enter post-velocity operation ($R \gtrsim A$), and which signals, such as validation lag (D1), procedural oversight (D2), or downstream defect shifting (D3), indicate assurance and governance saturation?
- *RQ2: Threshold effects on outcomes.* Does the relationship between development velocity and system outcomes show threshold behavior, with reliability, quality, governance, or accountability remaining stable up to a point and deteriorating once throughput exceeds assurance capacity?
- *RQ3: AI-driven generation–validation mismatch.* How do LLM-based tools increase the volume, size, or novelty of changes, and how do review depth, validation effort, evidence completeness, auditability, and justified confidence evolve as production accelerates?
- *RQ4: Accountability constraints and safe velocity.* In domains requiring human, institutional, or community-facing accountability, what defines a sustainable or *safe* velocity (V_s), and which governance mechanisms preserve meaningful oversight, contestability, and accountable approval as throughput increases?

4.2 Study designs

Addressing these questions requires complementary empirical approaches. The following study designs can contribute to a shared evidence base on assurance saturation, governance capacity, contestability, and safe velocity.

- *S1: Longitudinal field studies.* Analyze development traces before and after AI-assisted tooling adoption, or across teams with different levels of AI support, to examine how change throughput, validation practices, governance signals, audit evidence, and operational outcomes evolve.
- *S2: Controlled studies of validation under acceleration.* Examine how developers review and approve changes under

different levels of throughput pressure and AI assistance, measuring defect detection, review depth, confidence calibration, evidence linkage, auditability, and approval quality.
- *S3: High-assurance and public-interest case studies.* Study safety-critical, regulated, public-sector, or community-facing AI-enabled systems to understand how organizations define safe velocity, how governance mechanisms respond to increasing throughput, and how engineering and community evidence informs institutional decisions.
- *S4: Shared measurement infrastructure.* Develop community tools and replication packages that extract D1–D3 diagnostics and indicators of justified confidence from development and governance toolchains, enabling comparable studies across projects, organizations, public-interest settings, and domains.

4.3 Intervention space

Beyond measurement, the post-velocity perspective motivates design strategies for AI-accelerated workflows that treat acceleration as desirable only when assurance, accountability, contestability, and governance are preserved. Representative directions include:

- *(I1) Risk-stratified gating.* Require stronger evidence and deeper review for high-criticality or high-exposure changes, while allowing faster workflows for lower-risk modifications.
- *(I2) Differential assurance budgeting.* Allocate review effort, testing resources, CI capacity, and governance attention according to change volume, novelty, component criticality, and stakeholder exposure.
- *(I3) Evidence-carrying changes.* Require changes to include explicit validation evidence, such as linked tests, risk assessments, review rationales, audit trails, or assurance-case fragments.
- *(I4) Independent assurance and contestability mechanisms.* Support field-ready assurance cases, independent review, audit trails, and appeal or dispute-resolution pathways for changes affecting users, communities, or public institutions.

Evaluating these interventions requires measuring both outcomes and mechanisms, including whether stronger validation slows delivery in the short term but prevents downstream failures or governance breakdowns. More broadly, governance, accountability, contestability, independent assurance, and appeal mechanisms should be designed into AI-accelerated workflows rather than added as after-the-fact checks.

5 Implications and outlook

The post-velocity perspective suggests that development acceleration should not be treated as an unconditional goal. When the rate of change approaches the limits of validation, assurance, and governance capacity, additional throughput may increase operational, institutional, and societal risk rather than reduce it. Organizations therefore need to consider not only how quickly software can be produced, but also how confidently it can be validated, understood, contested, and responsibly approved in settings that affect users, communities, and public institutions. This shift has implications for both tooling and engineering practice. Development environments may need to prioritize mechanisms that support evaluation, oversight, and accountability, such as stronger links between changes and validation evidence, richer review context, audit trails, and tools that highlight architectural, operational, or governance risks introduced by proposed modifications. Engineering leadership may also need to reconsider performance metrics that reward throughput without accounting for assurance capacity, stakeholder exposure, contestability, or accountable approval.

Viewed through this lens, the goal of modern software engineering is not simply to maximize velocity but to sustain *safe velocity*: a rate of change that preserves justified confidence in deployed systems while allowing organizations to benefit from the productivity gains enabled by automation and AI. In AI-enabled socio-technical systems, safe velocity is also a governance concern, because trustworthy digital transformation depends on maintaining meaningful oversight, contestability, auditability, and public confidence as systems evolve. The post-velocity hypothesis introduced in this article provides a conceptual foundation for studying these dynamics. By connecting development throughput, assurance capacity, accountability, and system outcomes, it opens a research agenda for understanding the limits of AI-accelerated development and designing practices that maintain trust, public confidence, and accountable digital transformation in increasingly automated systems that affect users, communities, and public institutions.

Acknowledgments

This work is supported by: (a) the Swedish Agency for Innovation Systems through the project “Secure: Developing Predictable and Secure IoT for Autonomous Systems" (2023-01899), (b) the Key Digital Technologies Joint Undertaking through the project “MATISSE: Model-based engineering of digital twins for early verification and validation of industrial systems" (101140216), and (c) the Clean Energy Transition Partnership through the project “FLEXI: Human-centered AI and digital twin powered energy system integration for flexibility markets" (101069750).

References

[1] Alessio Bucaioni, Antonio Cicchetti, Gordana Dodig-Crnkovic, Romina Spalazzese, Emma Söderberg, and Dániel Varró. 2026. Engineering Future Critical CPSs with Trustworthy GenAI Across the Lifecycle. In *48th IEEE/ACM International Conference on Software Engineering*. http://www.es.mdu.se/publications/7308-

[2] Rogério De Lemos. 2020. Human in the loop: What is the point of no return?. In *Proceedings of the IEEE/ACM 15th International Symposium on Software Engineering for Adaptive and Self-Managing Systems*. 165–166.

[3] Angela Fan, Beliz Gokkaya, Mark Harman, Mitya Lyubarskiy, Shubho Sengupta, Shin Yoo, and Jie M. Zhang. 2023. Large Language Models for Software Engineering: Survey and Open Problems. In *2023 IEEE/ACM International Conference on Software Engineering: Future of Software Engineering (ICSE-FoSE)*. 31–53. doi:10.1109/ICSE-FoSE59343.2023.00008

[4] Odd Ivar Haugen, Dag McGeorge, Tore Myhrvold, Christian Agrell, and Andreas Hafver. 2024. Assurance of AI-enabled systems. In *Proceedings of the 2024 8th International Conference on Advances in Artificial Intelligence*. 100–106.

[5] Andreas Holzinger, Kurt Zatloukal, and Heimo Müller. 2025. Is human oversight to AI systems still possible? 59–62 pages.

[6] Syed Md Faisal Ali Khan and Salem Suhluli. 2025. Generative AI and Cognitive Challenges in Research: Balancing Cognitive Load, Fatigue, and Human Resilience. *Technologies* 13, 11 (2025), 486.

[7] Miikka Kuutila, Mika Mäntylä, Umar Farooq, and Maelick Claes. 2020. Time pressure in software engineering: A systematic review. *Information and Software Technology* 121 (2020), 106257.

[8] Joshua W Ohde, Lauren M Rost, and Joshua D Overgaard. 2025. The burden of reviewing LLM-generated content. AIp2400979 pages.

[9] Maluane Rubert and Kleinner Farias. 2022. On the effects of continuous delivery on code quality: A case study in industry. *Computer Standards & Interfaces* 81 (2022), 103588.