The Software Engineering Research Paradox: Evidence of Decay, Pathways to Relevance, and Visions for the Future

Audris Mockus
University of Tennessee
USA

Peter C. Rigby
Concordia University
Canada

Rui Abreu
University of Porto
Portugal

Abstract

Software engineering research aspires to improve real-world software practice, yet many of the most consequential advances in modern development and operations are created and disseminated outside mainstream SE venues. We argue that this disconnect is driven less by individual choices than by structural incentives: "puzzle-solving" on convenient artifacts, reliance on small or unrepresentative projects, disclosure norms that unintentionally exclude industry-scale studies, and slow, overloaded, high-variance peer review. Left unchanged, these forces can produce an "irrelevance cascade" in which SE venues remain academically active but increasingly disconnected from practice.

We propose near-term reforms that raise the weight of demonstrated practical impact, encourage industry-scale evaluation (including open-source "twins"), and use AI to reduce the mechanical burden and variance of review while improving reproducibility checks. We also advocate repositioning major conferences such as ICSE and FSE as impact-curated selection venues: dissemination is continuous through preprints and tool releases, authors provide impact statements, community members publicly champion or critique work, and program committees curate the most consequential results for presentation and synthesis. Finally, we sketch a longer-term vision of an AI-augmented, agentic research ecosystem that lowers the transaction costs of experimentation, replication packaging, and evaluation, enabling the community to return to harder, higher-leverage problems.

1 Introduction

Software engineering is a field whose success should ultimately be measured by improvements to real software practice. Yet a paradox is becoming increasingly visible: many of the most consequential advances in how software is built and operated at scale are created and disseminated outside of mainstream SE venues. New tools and practices often emerge first in deployed systems, open-source releases, practitioner communities, and increasingly AI/ML venues, while SE conferences and journals struggle to remain the primary forum where these ideas are first debated and adopted.

We argue that this gap is not mainly the result of individual choices, but of structural incentives that favor reviewer-checkable contributions over deployable value. Symptoms include "puzzle-solving" on convenient artifacts, reliance on small or unrepresentative projects, disclosure expectations that unintentionally exclude industry-scale work, and review processes that are slow, overloaded, and high-variance. If left unchanged, these forces can produce an "irrelevance cascade" in which SE venues remain academically active but have diminishing influence on practice.

This paper offers two paths forward. In the near term, we propose reforms that raise the weight of demonstrated practical impact, encourage industry-scale evaluation (including open-source "twins"), and use AI to reduce the mechanical burden and variance of peer review. We also propose repositioning major conferences such as ICSE and FSE as *impact-curated selection venues*: dissemination is continuous through preprints and tool releases, authors provide impact statements, community championing and critique become visible artifacts, and program committees curate the most consequential work for presentation and synthesis. In the longer term, we sketch an AI-augmented, agentic research ecosystem that lowers the transaction costs of experimentation, replication packaging, and evaluation, enabling the community to return to harder, higher-leverage problems.

2 Evidence of Decay (Current Problems)

The software engineering (SE) research community, much like other scientific disciplines analyzed by Thomas Kuhn [1], appears to be suffering from a systemic decay, moving from addressing real-world, high-impact problems to engaging in "puzzle-solving" of increasingly superficial and academically convenient issues.

The "Puzzle-Solving" Trap. Rooted in real practical challenges like software quality, defect prediction, and testing efficiency, the academic analogs of these problems have become detached from practical relevance. Research cycles now often produce minor, incremental, and approximate solutions that, while yielding numerous papers and PhDs, fail to offer substantial, deployable benefits to industry. This proliferation of unconstrained, low-impact work overwhelms the few endeavors aimed at the original, difficult problems.

The Academia-Industry Divide. The clearest evidence of this decay is the widening chasm between academia and industry. Truly impactful software engineering innovations—from large-scale version control systems to modern continuous integration/delivery practices—almost exclusively originate in industry. The current academic review system, which often demands infeasible disclosures of proprietary data and context, acts as a barrier, effectively excluding research derived from real-world, large-scale innovation from mainstream academic forums. This structural mechanism may inadvertently be delaying the necessary reckoning by prioritizing disclosure over genuine impact.

Furthermore, compounding this issue is the slow turnaround cycle of academic conferences and journals. Multiple rounds of review and revision, while intended to ensure rigor, are ill-suited to the fast-paced nature of the industry. By the time research is published, the state of practice may have already moved on, further diminishing the relevance and timeliness of academic contributions.

Table 1: Symptoms of misalignment between SE research incentives and real-world practice: recurring patterns (puzzle-solving, incremental approximations, exclusionary disclosure norms, and toy projects) that reduce practical relevance and limit industry-scale contributions.

Problem Area	Description	Impact on Research
Puzzle-Solving	Focus on minor, solvable variations of a large problem.	Research lacks practical relevance and scale.
Infinite Approximations	The search for slightly better approximate solutions.	Overwhelming paper volume masks true innovation.
Exclusionary Review	Requirements for data disclosure (e.g., NDA-protected data).	Real-world industry innovations are rarely published in main SE venues.
Toy Projects	Reliance on small, non-representative open-source projects.	Assumptions made often fail when applied to real, resource-constrained industry environments.

This lag undermines the potential for meaningful collaboration and knowledge transfer between academia and industry, and risks leaving academic research perpetually a step behind the needs of practitioners. A summary of symptoms is provided in Table 1

2.1 Vision A, the Status Quo: The Irrelevance Cascade

This future represents a continuation of today's incentive structure rather than an explicit failure of the community. No single decision causes collapse. Instead, small, individually rational choices by authors, reviewers, and institutions compound over time.

In this scenario, software engineering venues continue to reward novelty claims that are easy to evaluate on public artifacts, even when the artifacts are unrepresentative of modern industrial practice. As a result, many papers optimize for methodological tidiness and reviewer-checkable contributions rather than for deployable value. The center of gravity of the field shifts toward problems that are convenient to publish, not necessarily problems that matter most to practitioners.

At the same time, the highest-leverage innovations in developer productivity and software quality increasingly arise inside large organizations or at the intersection of SE and AI. Much of this work is disseminated through alternative channels: industrial blogs and engineering reports, open-source releases, patents, practitioner conferences, and AI/ML venues that already have the visibility, pace, and incentives aligned with rapid tool evolution. Traditional SE venues remain active and competitive, but their influence on real-world practice weakens because they are no longer where the most consequential ideas are first introduced, debated, and adopted.

Over a 10-year horizon, the outcome is a gradual decoupling. Academia continues to produce rigorous work, but it is less frequently consulted by teams building the next generation of software tooling. Conversely, practitioners see the academic literature as increasingly costly to translate into production impact. Conferences and journals then drift toward serving primarily as training and credentialing systems for academic careers, while practical standards and dominant tool paradigms are set elsewhere.

This is not meant as a critique of individual researchers. It is a description of what happens when evaluation criteria, data access constraints, and publication timelines remain misaligned with the realities of modern software development.

3 Avoiding Vision A: Proposed Short-term Solutions

While Kuhn's perspective suggests that meaningful change only occurs through a disruptive "scientific revolution," proactive strategies must be pursued to increase the practical relevance and impact of SE research before total obsolescence.

Reforming Research Requirements and Scope. The core objective must shift from the volume of academic output to demonstrable, real-world utility.

Practical Impact Criteria. Require academic papers to clearly articulate and, ideally, demonstrate practical impact using measurable, industry-relevant criteria. Create special tracks in academic venues specifically for research that is co-developed with or validated by industry partners.

Industry-Scale Focus. Mandate the use of real, industry-scale open-source projects (e.g., Chromium, large-scale cloud systems) in research, moving away from "toy" or small-scale OSS projects where assumptions often simplify or misrepresent real-world constraints.

Open-Source Twins. Promote the creation and use of "open-source twins"—projects designed to mimic the architectural or development constraints of internal company systems—as a viable bridge between private industry data and academic reproducibility.

Leveraging AI for a More Effective Research Cycle. AI offers tools not just for research subjects but for reforming the research process itself.

AI-Assisted Review. Deploy AI agents for initial technical reviews, focusing on novelty, methodological soundness, and basic reproducibility checks. Human reviewers would then assume a higher-level role, acting as checks on the practical appropriateness and significance of the work.

AI-Generated Meta-Review. Replace the current human meta-review role with an AI-generated summary and synthesis of all reviewer comments, freeing the human facilitator to focus purely on managing discussion and ensuring fairness.

Fostering Cross-Disciplinary Engagement. If current SE venues become resistant to change, high-impact research will naturally migrate. Researchers should actively seek relevance in related, growing fields.

Integration with AI/ML Venues. Actively pursue presenting SE research focused on tooling, MLOps, or large-scale systems at premier AI/ML conferences, acknowledging where SE innovation has the most traction.

3.1 Impact-Curated Major Conferences with Community Championing

A practical near-term step toward improving relevance and reducing friction is to change the function of major SE conferences such as ICSE and FSE from being primarily *submission-driven publication venues* to being *impact-curated selection venues*. The central shift is to treat dissemination as continuous (via preprints, technical reports, and open-source releases) while having ICSE/FSE curate and present work that is both current and already showing evidence of influence. In this model, the conference is not the primary distribution channel for PDFs; it is where the community gathers to debate, validate, and synthesize the most consequential results of the year.

One motivation is reviewer fatigue. The current system routinely produces short, low-signal reviews—the familiar "paragraph review" written by an overloaded reviewer under time pressure. This outcome is costly and unreliable: authors receive limited actionable feedback, reviewers feel their time is wasted, and acceptance decisions are made on thin evidence. An impact-curated model reduces dependence on single-shot, deadline-driven reviewing by allowing conference selection to incorporate signals that accumulate over time, including early adoption and external scrutiny.

A second motivation is the mismatch between assigned reviewing and genuine engagement. Reviewers are often asked to evaluate papers they are only partially interested in or that sit at the margins of their expertise. Even with good intentions, that tends to produce mechanical feedback rather than deep critique. In contrast, the highest-value evaluation typically comes from self-selected readers who care enough to replicate, adopt, extend, or rebut a result. An impact-curated ICSE/FSE would explicitly value these signals and make them part of the selection process, rather than relying almost exclusively on private, time-boxed reviews.

A third motivation is that the community already consumes research through fast, open channels. Many researchers learn about new results from preprint servers, repositories, tool releases, and informal dissemination. When a contribution is genuinely useful, it attracts citations, adopters, forks, issue discussions, reproduction attempts, blog posts, and practitioner commentary. The current conference model largely ignores these signals at decision time, because it must decide before impact is observable. Impact-curated conferences instead embrace the reality that influence is often visible early and can be evaluated directly.

Critically, this model changes what authors provide for consideration. Instead of submitting a paper that primarily claims novelty and promises future relevance, authors provide a **paper (or stable preprint) accompanied by an impact statement** describing observed influence to date. Depending on the maturity of the work, this can include citations (including preprint citations), documented adoption in open-source or commercial settings, downstream artifacts (forks, packages, plugins), evidence of integration into toolchains, replication attempts (successful or not), usage metrics where appropriate, and practitioner feedback. The objective is not to reward hype, but to make *evidence of use and scrutiny* an explicit input to selection.

Alongside impact statements, the model encourages **community championing**. Rather than concentrating most evaluation power in anonymous reviews of assigned submissions, researchers publicly champion work they believe matters—through signed commentaries, replication notes, adoption reports, critical reviews, or rebuttals. These artifacts create durable, citable evidence of engagement. They also provide a richer signal than traditional reviews: championing is costly (it takes time to adopt or replicate), and thus is harder to fake than superficial novelty claims. Importantly, championing can be positive or adversarial; both indicate that the work matters and is being seriously tested.

In this framing, the role of the ICSE/FSE program committee shifts from predicting future value to **curating demonstrated impact**. The PC evaluates a portfolio of candidate papers (with their impact statements and public championing artifacts) and selects those that are most consequential for the community to discuss in person. Conferences then become the venue where high-impact results are presented, challenged, and contextualized via deeper Q&A, structured rebuttals, replication panels, and synthesis talks. This preserves the prestige and coordination function of top venues while aligning incentives toward deployable value.

This approach has real risks. Visible impact signals can correlate with seniority, institutional resources, and social reach; without safeguards, selection could drift toward popularity contests. A viable impact-curation model therefore needs explicit countermeasures: normalization windows by paper age, mechanisms to elevate high-quality replications and critical reviews, structured templates for impact statements (to reduce marketing), and active processes to surface strong work from less-visible authors. Even with these caveats, shifting ICSE/FSE toward impact-curated selection offers a concrete, immediately deployable path toward aligning incentives with real-world influence while reducing reliance on exhausted, low-signal reviewing.

4 Vision B: The AI and Agentic Research Ecosystem Revolution

In this future, summarized in Table 2, the software engineering research life cycle is reorganized around specialized AI agents that support, and in some cases carry out, substantial portions of the work. The goal is not simply to increase the number of papers. The goal is to lower the transaction costs of doing serious, large-scale, and reproducible research so that the community can return to harder problems that are currently avoided because they are too expensive, too slow, or too difficult to validate.

A key feature of this vision is a separation of concerns: different agents are responsible for experimentation, writing, reproducibility, and evaluation. Human researchers remain accountable for problem selection, ethical and legal constraints, and the interpretation of results, but delegate much of the routine execution and verification.

Table 2: Vision B (10–20 years): an AI-augmented, agentic SE research lifecycle in which specialized agents help run experiments, generate replication packages (including controlled validation), and reduce review variance through automated checks and richer post-publication discourse.

Phase	Current State	Future State (10–20 Years)
Authoring	Manual design, execution, writing.	**Agent-Driven Contribution:** Agents design and run experiments; separate text-generation agents draft papers, tables, and figures. Human authors provide initial prompts and light editing.
Replication Packages	Time-consuming manual preparation with partial coverage — or completely impossible with research involving proprietary data.	**Agent generated replication packages** from public digital twins, potentially NDA supported reviewer Agent access to environments capable of running a limited set of analysis operations without direct access to proprietary data.
Peer Review	Time-consuming manual review, high variance.	**Agent-Validated Review:** Reviewers fine-tune personal AI agents on their past reviews. These agents apply comprehensive checks and search for missing literature. Human reviewers validate agent output.
Scientific Meetings	Author presentation, brief Q&A.	**Agent-to-Agent Discourse:** Author agents discuss and explain points to participant agents. Detailed transcripts and summaries of this discussion become mandatory addendums (or retractions) to the published work.

Authoring becomes agent-driven experimentation and drafting. In the authoring phase, agents assist with framing research questions, mapping them to testable hypotheses, and designing the empirical protocol. For example, an experiment-design agent can propose datasets, baselines, ablations, and statistical tests based on norms in the literature and known threats to validity. Another agent can implement the analysis pipeline, run experiments, and track provenance of intermediate artifacts. A separate writing agent drafts the paper from structured experiment logs, producing tables, figures, and method descriptions that are consistent with the executed pipeline. Humans remain responsible for choosing the research direction, specifying constraints, validating that the experimental design is sensible, and ensuring that the conclusions are warranted. In other words, humans set the intent and take responsibility; agents execute and document.

Replication packages become largely automatic, and can be supported by controlled access. A persistent weakness of the current ecosystem is that replication packages are either incomplete (due to time cost) or impossible (due to proprietary data). In this vision, replication is treated as a first-class product of the workflow. Agents generate replication packages directly from the experiment specification and execution traces, including pinned dependencies, containerized environments, and scripts that regenerate every figure and table in the paper. When the underlying data or systems are proprietary, the vision shifts from full disclosure to verifiable execution. One approach, suggested in Table 2, is an NDA-supported reviewer agent that can run a restricted set of analyses within a protected environment. Another approach is the use of public "digital twins" that approximate key properties of internal systems. The central idea is to make replication a standard, low-friction byproduct of doing the work, rather than an expensive afterthought.

Peer review becomes agent-validated, reducing variance and increasing coverage. Peer review is currently expensive and inconsistent, with high variance across reviewers and limited capacity to check details. In the agent-driven ecosystem, each reviewer uses a personal review agent, fine-tuned on that reviewer's prior reviews and evaluation preferences. The agent performs comprehensive mechanical checks: it validates claims against presented evidence, checks for missing baselines and ablations, audits statistical reporting, searches for closely related work, and attempts to execute the replication package. Human reviewers then focus on higher-level judgments that remain difficult to mechanize: significance, novelty, clarity, fit to the venue, and whether the work advances understanding rather than optimizing benchmarks. This division is intended to reduce reviewer fatigue while increasing the thoroughness and consistency of evaluation.

Scientific meetings shift toward structured agent-to-agent discourse. In the final phase, scientific meetings evolve from short presentations and limited Q&A into deeper, more structured interaction. Participants bring agents that can ask detailed questions, request clarifications, and probe edge cases. Author agents can respond with references to the exact experimental logs, code paths, and supporting analyses. The output is not only a talk recording, but also a structured transcript and summary of the technical discussion. These artifacts can be attached as addenda to the paper, documenting clarifications, limitations, and any discovered issues. Over time, this mechanism can reduce repeated confusion, make limitations more explicit, and create a durable record of post-publication scrutiny.

Expected outcomes and risks. If successful, this ecosystem increases throughput in a way that favors rigor rather than volume: more comprehensive experimentation, more reproducible artifacts, and faster iteration on research directions. It also makes it easier to engage with industry-scale questions by reducing the cost of

validation and by enabling controlled evaluation in settings where full disclosure is impossible.

At the same time, this vision introduces new risks that must be managed: automation bias (humans over-trusting agent output), homogenization (agents pushing toward standard but stale methodologies), privacy and security concerns (especially for controlled-access replication), and strategic behavior (optimizing for what review agents check rather than what matters). A viable "revolution" therefore requires not only better agents, but also new norms, audit mechanisms, and governance structures that preserve human accountability.

5 Conclusion

When Meta (Facebook at the time) launched React in 2013, revolutioninzing frontend development for millions of developers worldwide, it was not published at any major software engineering venue. When Google introduced site reliability engineering practices that redefined how the industry thinks about operations, idem. When GitHub fundamentally changed collaborative software development through pull requests, the innovation did not appear in a software engineering journal. This pattern is not coincidental — it reveals a troubling reality: the most transformative software engineering innovations of the past two decade have bypassed academic software engineering venues entirely.

Software engineering research faces a paradoxical trajectory: a growing volume of technically competent work that increasingly misses the data, constraints, and timelines that define modern software practice. We argued that this drift is reinforced by structural incentives—puzzle-solving on convenient artifacts, over-reliance on toy projects, review norms that unintentionally exclude industry-scale work, and a publication process whose latency and reviewer load reward checkability over deployability. If left unchanged, SE venues may remain academically healthy while steadily losing their role as the primary forum where consequential ideas are first introduced, validated, and adopted.

To avoid this outcome, we outlined pragmatic interventions that can be pursued immediately. These include elevating demonstrated practical impact as a first-class evaluation criterion, prioritizing industry-scale evaluations (including open-source "twins"), and leveraging AI to reduce the mechanical burden and variance of peer review. We also proposed a more fundamental but still near-term change: reposition major conferences such as ICSE and FSE as *impact-curated selection venues*. In this model, dissemination is continuous through preprints and tool releases, authors provide impact statements, community championing and critique become visible scholarly artifacts, and program committees curate the most consequential work for presentation and synthesis. The goal is to shift the conference function from predicting impact under severe uncertainty to selecting work that is already demonstrating influence and scrutiny.

Finally, we described a longer-term "agentic" research ecosystem in which specialized AI agents reduce the transaction costs of experimentation, replication packaging, and evaluation. If governed carefully, this ecosystem could increase rigor at scale, make replication the default, and enable controlled validation even when full disclosure is impossible. The central challenge is not whether SE can produce more papers, but whether it can rebuild evaluation and dissemination mechanisms that reliably reward work that matters. The opportunity is to use the same AI and automation forces reshaping software development to also reshape how software engineering knowledge is produced, checked, and transferred into practice.

Acknowledgments

AI was used to help with editorial changes and manuscript checking.

References

[1] Thomas S Kuhn. 1997. *The structure of scientific revolutions*. Vol. 962. University of Chicago press Chicago.