From Papers to Progress: Rethinking Knowledge Accumulation in Software Engineering

Anonymous Author(s)

Abstract

Software engineering research has experienced sustained growth in both output and participation over the past decades. Yet concerns persist about the field's ability to accumulate, integrate, and reuse knowledge in ways that support long-term progress. To better understand how the community itself perceives these challenges, we analyze responses from the ICSE 2026 Future of Software Engineering pre-survey, which captures perspectives from a globally distributed and highly experienced set of researchers. Our analysis reveals a tension between increasing research productivity and the limited mechanisms available for synthesizing results, tracking evolving claims, and supporting cumulative understanding over time.

Building on these observations, we diagnose four interrelated structural breakdowns: papers function as isolated knowledge units with claims embedded in prose; context and provenance are lost as knowledge moves through the publication pipeline; claims evolve without systematic tracking; and incentive structures favor novelty over consolidation. We argue that addressing these barriers requires rethinking the fundamental properties of research artifacts.

We articulate four technology-agnostic principles for future research artifacts: structured and interpretable representations of claims and evidence; inspectable and provenance-aware documentation of methodological decisions; long-lived and reusable substrates that evolve beyond publication; and governance mechanisms that align individual incentives with collective knowledge-building goals. We discuss implications for research practice, publication norms, and community infrastructure, positioning FOSE as a venue for experimenting with alternative artifact designs that support cumulative scientific progress.

CCS Concepts

• **Software and its engineering** → **Software creation and management**; **Empirical software validation**; • **General and reference** → *Empirical studies.*

Keywords

knowledge accumulation, research artifacts, software engineering, cumulative progress, research infrastructure

ACM Reference Format:

Anonymous Author(s). 2026. From Papers to Progress: Rethinking Knowledge Accumulation in Software Engineering. In . ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

ICSE'26, Rio De Janeiro, Brazil
2026. ACM ISBN 978-x-xxxx-xxxx-x/YYYY/MM
https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 Introduction

Software engineering research has experienced remarkable growth over the past decades. Conference submissions have increased, publication venues have proliferated, and the community has become truly global. By many measures, the field is thriving: researchers are productive, techniques are advancing, and new areas of inquiry continue to emerge. Yet beneath this surface of activity lies a persistent challenge: the difficulty of building cumulative knowledge that connects, consolidates, and extends prior work in ways that support long-term scientific progress.

This challenge is not new, nor is it unique to software engineering. Across many scientific disciplines, researchers have observed that increasing publication rates do not automatically translate into deeper understanding or more integrated knowledge [5, 8]. In software engineering specifically, concerns about fragmentation, replication, and the ability to synthesize results across studies have been raised repeatedly [11, 12]. Despite these concerns, the structural factors that limit cumulative progress—and the properties that future research artifacts might need to address them—remain underexplored.

This paper offers a community-informed perspective on knowledge accumulation challenges in software engineering. Grounded in responses from 280 ICSE 2026 FOSE pre-survey participants—an experienced (57% with 10+ years), productive (44% with 11+ papers in 3 years), globally distributed community—we diagnose four interrelated structural breakdowns: papers as isolated units, lost context and provenance, untracked claim evolution, and misaligned incentives favoring novelty over consolidation. In response, we propose four technology-agnostic design principles for research artifacts: structured and interpretable representations, inspectable and provenance-aware documentation, long-lived and reusable substrates, and community-governed infrastructures. This FOSE contribution synthesizes community observations into a diagnosis of structural limitations and principles to guide future work, without prescribing specific tools or claiming to solve the challenges we identify.

2 Community Signals from the ICSE 2026 FOSE Pre-Survey

To ground our diagnosis in community perspectives, we draw on the ICSE 2026 Future of Software Engineering pre-survey (n=280). The respondents represent a mature, productive, and globally distributed research community: 57% have 10+ years of experience, 44% authored 11+ papers in the past three years, and participation spans six geographic regions (Europe 50%, North America 25%, Asia 15%, others 10%).

This profile is critical to our argument. If knowledge accumulation remains limited despite significant expertise, productivity, and global participation, then the barriers must be *structural* rather than individual. The community is not lacking effort or output capacity—respondents themselves recognize systemic challenges in knowledge synthesis, with concerns about fragmentation across resubmissions, difficulties building on prior work, and systems that reward novelty over consolidation. These observations motivate the structural diagnosis that follows.

3 Where Knowledge Accumulation Breaks Down

If the software engineering community is experienced, productive, and globally distributed, yet faces persistent challenges synthesizing results and building on prior work, then the barriers must be structural. We diagnose four interrelated breakdowns arising from how research artifacts are currently produced, represented, and connected.

3.1 Papers as Isolated Knowledge Units

Research papers package claims, evidence, and context into narrative prose optimized for dissemination. While effective for presenting new ideas, this format creates barriers to cumulative building. Claims are embedded in paragraphs, interleaved with motivation and methods, making extraction and comparison across papers require reading entire documents and reconciling terminological differences. Evidence linkages remain implicit—readers must infer which results support which claims under what assumptions. Even when artifacts are shared, results are static at publication time; updates to datasets or baselines require new papers citing the old, while original claims remain unchanged. The consequence is fragmentation: each paper is an island connected only through citations and prose, requiring repeated manual synthesis that does not scale as literature grows.

3.2 Loss of Context and Provenance

Research involves countless contextual decisions: which dataset to use, how to split data, which baselines to compare, how to handle edge cases. Papers describe these choices but often omit the rationale due to space constraints. As work is cited and summarized, motivation and assumptions fade—carefully qualified findings become unqualified facts in subsequent literature. Methodological decisions must be reverse-engineered, sometimes revealing that subtle choices significantly impacted results. Papers present polished final versions, hiding the evolutionary path from hypothesis to result. When context and provenance are lost, later researchers must either accept prior work at face value or invest substantial effort reconstructing reasoning, slowing progress and increasing misapplication risk.

3.3 Claims Evolve Without Tracking

Scientific claims are refined, qualified, contradicted, and superseded as evidence accumulates, yet the publication system provides limited tracking mechanisms. The literature may contain conflicting claims—technique A outperforms B in one paper, the reverse in another—but contradictions are not systematically flagged. Researchers discovering conflicts must investigate causes themselves, often finding subtle methodological differences account for divergence. Refinements are implicit: a follow-on paper may qualify a prior claim, but the original remains unchanged. Claim relationships—does one extend, contradict, or depend on another?—are expressed only in natural language like "building on [12]" or "in contrast to [34]." Without structured representations, knowledge remains fragmented and ambiguous, requiring extensive manual synthesis to determine what is currently believed, under what conditions, and with what confidence.

3.4 Incentive Structures Favor Novelty Over Accumulation

The final breakdown is embedded in incentive structures. Publication venues, hiring criteria, and funding mechanisms reward novelty and originality [10, 13]. Conference and journal reviews prioritize new techniques and findings; replication studies, negative results, and syntheses face higher acceptance bars even when they would advance collective understanding [3]. Researchers investing in replication, dataset curation, or shared infrastructure face opportunity costs—these efforts may not yield top-venue publications or count heavily in promotion reviews [4]. Building knowledge repositories, ontologies, and interoperability layers requires sustained effort often led by small groups without commensurate recognition. The result is a collective action problem: everyone benefits from better infrastructure, but individuals are disincentivized from contributing. As long as novelty is privileged over accumulation, knowledge remains fragmented and progress constrained.

These four breakdowns reinforce one another, creating a system where knowledge accumulates slowly despite community expertise and productivity. Incremental fixes—better citation practices, more replications, improved repositories—are necessary but insufficient. Addressing these barriers requires rethinking the fundamental properties of research artifacts themselves.

4 Rethinking Research Artifacts for Cumulative Progress

The structural barriers arise from design choices in how artifacts are produced and shared. Addressing them requires reconsidering fundamental artifact properties. We articulate four technology-agnostic principles describing properties that systems, tools, or practices aiming to support knowledge accumulation should strive for. Implementations will vary across domains, but these principles provide a shared foundation for evaluating research infrastructure.

4.1 Principle 1: Structured and Interpretable

Principle: Research artifacts should make claims, evidence, and context explicit and directly accessible, not only embedded in prose.

The first breakdown—papers as isolated units—stems from the fact that claims and evidence are woven into narrative text. While prose is valuable for explaining motivation and argumentation, it is not optimal for supporting cumulative synthesis. A reader wishing to compare claims across papers must extract and interpret statements from natural language, a process that does not scale as the literature grows.

Artifacts should represent claims, evidence, and context as structured, first-class entities. A claim should be identifiable: "Technique X improves metric Y on dataset Z by δ under conditions C." Evidence and context should be explicitly linked, not buried in prose.

Structured representations enable direct comparison and reasoning: conflicting claims become visible and resolvable by examining structured evidence, and claims referencing updated datasets can be systematically reevaluated. Instantiations could include semantic annotations (machine-readable metadata describing claims and results) or knowledge graphs [9, 14] (entities with typed relationships enabling queries like "Which papers claim improvements on dataset D?"). Structure should complement prose, making knowledge directly accessible while preserving narrative's explanatory power.

4.2 Principle 2: Inspectable and Provenance-Aware

Principle: Research artifacts should preserve the full provenance of claims—from raw data through methodological decisions to final results—and make this provenance inspectable.

The second breakdown—loss of context and provenance—occurs because reasoning behind decisions fades as knowledge moves through publication. Papers report final results but often omit the path taken: why this dataset, baseline, or protocol. Later researchers must reconstruct this reasoning, often discovering subtle choices had significant consequences.

Artifacts should document not only what was found but how and why. Every claim should trace to its sources: data, code, configuration, assumptions. Methodological decisions should be explicitly justified, and when results are updated, the provenance chain should track changes [6]. Provenance enables trust: researchers can inspect a claim's lineage, verify evidence, and understand conditions. When claims conflict, provenance diagnoses divergence sources. Instantiations could include versioned computational artifacts (code and data in version control with clear lineage) or provenance graphs (explicit representations linking claims to experiments to data). Artifacts should be transparent about origins and decisions, enabling future work to build on solid foundations.

4.3 Principle 3: Long-Lived and Reusable

Principle: Research artifacts should support evolution and reuse, not remain static at the moment of publication.

Papers are snapshots frozen in time. Once published, they do not update when evidence emerges, datasets are revised, or methods improve. Follow-on work cites and describes differences in prose, but original claims remain unchanged.

Artifacts should be living substrates that can be updated, extended, and reused. When datasets are corrected, dependent claims should be re-evaluable. When techniques are refined, prior results should be comparable to new results. When claims are qualified, these relationships should be reflected in the artifact. This does not mean rewriting papers—rather, underlying structured representations should decouple from narrative documents. Narratives remain stable as historical records while structured substrates evolve. Instantiations could include living knowledge bases (repositories where claims and datasets are versioned and updated) or executable benchmarks (evaluation frameworks rerun with updated data). Knowledge should outlive individual papers, accumulating in shared substrates that reduce redundancy and enable direct building.

4.4 Principle 4: Governed with Human Oversight

Principle: Research artifacts and infrastructures should be governed by community processes that ensure quality, integrity, and ethical responsibility.

Even with perfect technical infrastructure, cumulative progress requires coordination: quality standards, conflict resolution mechanisms, and recognition for consolidation work. Artifacts cannot govern themselves—structured representations and provenance tracking are valuable only if the community trusts, maintains, and uses them responsibly. This requires human oversight: peer review for knowledge contributions, curation, dispute resolution processes, and credit systems valuing infrastructure work alongside novel research.

Governance also addresses ethical concerns about ownership, attribution, bias, and access. Who controls shared knowledge bases? How is credit assigned for incremental contributions? How do we prevent infrastructures from perpetuating biases? These social and ethical questions require community deliberation and ongoing stewardship. Instantiations could include community curation processes (peer-reviewed contributions with clear criteria) or credit systems recognizing infrastructure, replication, and consolidation work. Cumulative progress is a collective endeavor—technical solutions must be embedded in social practices aligning individual incentives with collective goals.

4.5 From Principles to Practice

These principles are aspirational, requiring significant changes in how research is conducted, reviewed, and rewarded. They provide a framework for evaluating incremental steps: Does a proposed tool or practice make artifacts more structured, inspectable, reusable, or better governed? The principles are technology-agnostic—knowledge graphs, computational notebooks, and repositories are potential instantiations, but the principles describe *properties* that artifacts should have, leaving room for diverse implementations. The challenge is designing systems and reforming incentives to make cumulative knowledge building not only possible but rewarded.

5 Implications for the Future of Software Engineering

Realizing these principles would require changes in how research is conducted, reviewed, published, and rewarded.

5.1 Research Practice and Publication

Research practice would shift toward documenting process alongside outcomes: recording methodological decisions, their rationales, and impacts using computational notebooks, version control, and workflow platforms. Researchers would supplement narrative papers with structured representations—semantic annotations or knowledge graph entries—making contributions directly accessible for synthesis. Designing for reuse would become standard: datasets documented with schemas, code modular and documented, protocols reproducible. Structured artifacts would enable new collaboration forms, with researchers contributing incrementally to shared knowledge bases rather than isolated papers.

Publication and review would need to value consolidation alongside novelty. Papers that resolve contradictions, curate benchmarks, or provide infrastructure should be recognized. Conferences could establish expectations for structured artifacts as first-class contributions, not optional supplements. If artifacts evolve post-publication, papers become snapshots while living artifacts accumulate evidence and track refinements, requiring new norms for citing evolving work.

5.2 Community Infrastructure

The community would need platforms for storing, querying, and updating structured artifacts—knowledge graphs, benchmark repositories, collaborative platforms—maintained through community governance [2]. Shared standards (ontologies, schemas, reporting protocols) would enable reuse across studies [15]. Infrastructure contributions, replications, and refinements would require recognition through alternative metrics, tenure criteria changes, or contribution-tracking platforms [1]. Governance structures would manage stewardship: maintaining repositories, resolving disputes, updating standards, addressing ethical concerns. Building such infrastructure requires sustained investment and coordination, but without it, the principles remain aspirational.

5.3 FOSE as a Venue for Experimentation

The Future of Software Engineering track is positioned to support experimentation with alternative artifact types. FOSE could encourage submissions experimenting with structured artifacts or living documents, judged on potential to demonstrate new knowledge-organizing approaches rather than solely novelty. FOSE sessions could facilitate dialogue about infrastructure needs and governance models, and track experiments over time to inform broader community decisions about adopting new norms.

6 Limitations

This paper diagnoses structural barriers and proposes principles for future research artifacts, but does not provide complete solutions. We acknowledge several limitations.

Survey limitations. Our analysis draws on 280 FOSE pre-survey responses, representing a small, self-selected fraction of the global community. The data are descriptive, not causal—they establish that the community is experienced and productive, supporting our argument that barriers are structural, but do not prove specific breakdowns. The survey reflects perspectives at one moment; community concerns evolve over time.

Feasibility challenges. Realizing these principles faces practical barriers. Producing structured, provenance-aware artifacts requires effort that researchers may not have without reduced demands or increased support. The needed infrastructure—shared repositories, standards, governance—does not yet exist in many areas and requires sustained investment and coordination [7]. Adoption depends on changing incentives at multiple levels (funding agencies, universities, conferences), which is difficult and slow. Cultural norms resist change, especially from those who succeed under current systems.

Ethical concerns. Structured knowledge infrastructures raise questions about ownership (who controls shared knowledge bases?), bias (artifacts may perpetuate creators' biases if diversity is lacking), access (resources may concentrate in well-funded institutions), and privacy (structured data about research processes requires ethical use and consent). These are not purely technical issues—they require community deliberation, ethical oversight, and ongoing stewardship. Any future infrastructure must address these concerns.

7 Conclusion

Software engineering research is productive and globally distributed, yet knowledge accumulation lags behind knowledge production. This paper has argued that the barriers are structural: papers function as isolated documents, provenance is lost, claims evolve without tracking, and incentives favor novelty over consolidation. Addressing these barriers requires rethinking research artifacts themselves. We have proposed four principles—structured and interpretable, inspectable and provenance-aware, long-lived and reusable, and governed with human oversight—to guide future work.

These principles are aspirational and technology-agnostic. Realizing them requires changes in practice, publication norms, infrastructure, and incentives. The challenges are significant, but the alternative—continuing with practices optimized for dissemination rather than accumulation—will perpetuate fragmentation as the field grows.

The Future of Software Engineering track provides space for this conversation. Next steps include experimenting with alternative artifact designs, developing infrastructure aligned with these principles, and revising recognition systems to value consolidation alongside novelty. Most importantly, the community must engage in dialogue about what cumulative progress requires and what trade-offs are acceptable.

The future of software engineering depends not only on what we discover but on how we organize and preserve what we know. Building infrastructures that support cumulative knowledge building—making it rewarded, not just possible—is essential to ensuring the next generation inherits not just a literature to read but a knowledge base to build upon.

References

[1] Liz Allen, Jo Scott, Amy Brand, Marjorie Hlava, and Micah Altman. 2014. Publishing: Credit where credit is due. *Nature* 508, 7496 (2014), 312–313. doi:10.1038/508312a

[2] Daniel E. Atkins, Kelvin K. Droegemeier, Stuart I. Feldman, Hector Garcia-Molina, Michael L. Klein, David G. Messerschmitt, Paul Messina, Jeremiah P. Ostriker, and Margaret H. Wright. 2003. Revolutionizing Science and Engineering Through Cyberinfrastructure: Report of the National Science Foundation Blue Ribbon Advisory Panel on Cyberinfrastructure. *National Science Foundation* (2003).

[3] Victor R. Basili, Forrest Shull, and Filippo Lanubile. 1999. Building knowledge through families of experiments. *IEEE Transactions on Software Engineering* 25, 4, 456–473. doi:10.1109/32.799955

[4] Christian Collberg and Todd A. Proebsting. 2016. Repeatability in Computer Systems Research. *Commun. ACM* 59, 3, 62–69. doi:10.1145/2812803

[5] Santo Fortunato, Carl T. Bergstrom, Katy Börner, James A. Evans, Dirk Helbing, Staša Milojević, Alexander M. Petersen, Filippo Radicchi, Roberta Sinatra, Brian Uzzi, Alessandro Vespignani, Ludo Waltman, Dashun Wang, and Albert-László Barabási. 2018. Science of science. *Science* 359, 6379 (2018), eaao0185. doi:10.1126/science.aao0185

[6] Jesús M. González-Barahona and Gregorio Robles. 2012. On the reproducibility of empirical software engineering studies based on data retrieved from development repositories. *Empirical Software Engineering* 17, 1-2 (2012), 75–89. doi:10.1007/s10664-011-9181-9

[7] Ben Herman, Emily Winter, Michail D. Papamichail, Georgios Gousios, and Andy Zaidman. 2020. A 2020 perspective on the replicability of software engineering research. In *Proceedings of the 14th ACM/IEEE International Symposium on Empirical Software Engineering and Measurement (ESEM)*. 1–12. doi:10.1145/3382494.3410652

[8] John P. A. Ioannidis. 2005. Why Most Published Research Findings Are False. *PLOS Medicine* 2, 8 (2005), e124. doi:10.1371/journal.pmed.0020124

[9] Mohamad Yaser Jaradeh, Allard Oelen, Kheir Eddine Farfar, Manuel Prinz, Jennifer D'Souza, Gábor Kismihók, Markus Stocker, and Sören Auer. 2019. Open Research Knowledge Graph: Next Generation Infrastructure for Semantic Scholarly Knowledge. In *Proceedings of the 10th International Conference on Knowledge Capture*. 243–246. doi:10.1145/3360901.3364435

[10] Brian A. Nosek, Jeffrey R. Spies, and Matt Motyl. 2012. Scientific Utopia: II. Restructuring Incentives and Practices to Promote Truth Over Publishability. *Perspectives on Psychological Science* 7, 6 (2012), 615–631. doi:10.1177/1745691612459058

[11] Forrest Shull, Jeffrey C. Carver, Sira Vegas, and Natalia Juristo. 2008. The role of replications in Empirical Software Engineering. In *Empirical Software Engineering and Verification*. Springer, 211–244. doi:10.1007/978-3-540-71301-2_8

[12] Dag I. K. Sjøberg, Tore Dybå, Bente C. D. Anda, and Jo E. Hannay. 2008. Building Theories in Software Engineering. *Guide to Advanced Empirical Software Engineering* (2008), 312–336. doi:10.1007/978-1-84800-044-5_12

[13] Paul E. Smaldino and Richard McElreath. 2016. The natural selection of bad science. *Royal Society Open Science* 3, 9 (2016), 160384. doi:10.1098/rsos.160384

[14] Kuansan Wang, Zhihong Shen, Chiyuan Huang, Chieh-Han Wu, Yuxiao Dong, and Anshul Kanakia. 2020. Microsoft Academic Graph: When experts are not enough. *Quantitative Science Studies* 1, 1, 396–413. doi:10.1162/qss_a_00021

[15] Claes Wohlin, Per Runeson, Martin Höst, Magnus C. Ohlsson, Björn Regnell, and Anders Wesslén. 2012. Experimentation in Software Engineering. (2012). doi:10.1007/978-3-642-29044-2