Research Statement

Courtney Miller

I am fundamentally drawn to studying how developers navigate technological disruptions in software engineering. From digital infrastructure abandonment in open source software supply chains to generative AI (GenAI) tool adoption on industrial development teams, ***I study the intersection of technical capabilities and human practices that determine successful transitions during periods of technological disruption***. Such disruptions threaten the stability of critical infrastructure that form the foundation of modern technology including healthcare systems, financial services, and transportation networks. Addressing them is essential for ensuring software reliability, security, and the resilience of our broader digital economy. My research is grounded in the real needs of practitioners and guided by the challenges they face. To address these challenges, ***my distinctive approach combines rigorous empirical methods with interdisciplinary theoretical grounding***: I leverage my degree in Statistics and eight years of empirical research experience to develop multi-dimensional mixed-methods approaches, integrating human-centered qualitative techniques with large-scale data-driven analysis, modeling, and visualization. This allows me to investigate timely disruptions while systematically grounding my work in social science theories and established frameworks e.g., signaling theory, game theory, and organizational behavior. The result is impactful insights and tooling solutions that address sociotechnical challenges in software development and maintenance processes.

My research has been recognized by both the software engineering and open source communities. My first author conference publications have received three ACM Distinguished Paper Awards at the International Conference on Software Engineering (ICSE), and I am a recipient of the NSF GRFP. I have delivered invited talks at industry research venues (e.g., Linux Open Source Summit North America) and guest appearances on open source and academic podcasts (e.g., Sustain). I have contributed to 15 peer-reviewed publications at venues including ICSE, ESE/FSE, and MSR, conducted large-scale empirical studies analyzing millions of package dependency interactions [OSS'19], [ICSE'25], developed LLM-based classifiers achieving high performance in independent practitioner evaluations [ICSE'26], and lead mixed-methods studies with a total of 3,259 developer participants (3,148 survey participants and 111 interviewees) [ICSE'21], [ESEC/FSE'23].

My dissertation focuses on the disruption of dependency abandonment in software supply chains, while recent work investigates how Generative AI (GenAI) tooling disrupts established development workflows– which I argue is one of today's most urgent challenges. ***As technology evolves, I will continue applying my empirical expertise to study and support developers through the disruptions of tomorrow.***

Understanding Abandonment Disruptions in Software Supply Chains

Modern software relies on open source as digital infrastructure, creating the software supply chains that our daily lives rely on, yet many widely used open source packages are abandoned by maintainers. ***When a dependency is abandoned, it shifts from a free resource to a liability***, exposing downstream projects to the risk of unpatched vulnerabilities, supply chain attacks, and costly unplanned migrations. Because modern applications often rely on hundreds of dependencies, even a single abandonment can trigger cascading service disruptions, affecting critical systems across healthcare, finance, and government infrastructure.

My research is among the first to study dependency abandonment from the user perspective, with the goal of enabling the sustainable use of open source by supporting users when they face abandonment, combining (1) semi-structured interviews on the experiences and practices of developers who have faced abandonment; with (2) large-scale quantitative analysis of abandonment prevalence as well as downstream impact and response, informed by signaling theory. By systematically studying abandonment as a disruption, we can better understand the challenges developers face when confronted with its risks and realities and design solutions that support them in navigating abandonment.

To understand developers' experiences with abandonment, we conducted exploratory semi-structured interviews with 33 developers with abandonment experience [ESEC/FSE'23]. ***Developers reported struggling to respond due to limited resources and guidance, often relying on labor-intensive manual investigations to identify abandonment that delay action.*** Many wanted to detect abandonment earlier, before it triggered urgent problems. Guided by signaling theory, we therefore hypothesize that increasing transparency around abandonment can make it more visible and, in turn, support downstream responses.

Building on these insights, we conducted a large-scale quantitative study of abandonment prevalence and downstream responses across the JavaScript npm ecosystem [ICSE'25]. We found that 15% of widely used packages were abandoned between 2015–2020, yet only 18% of exposed projects removed them during that period. To test whether transparency affects remediation, we compared responses to explicit-notice abandonment (public end-of-support signals) with activity-based abandonment (sustained inactivity) using survival analysis. Projects facing explicit-notice abandonment were 58% more likely to remove abandoned dependencies, ***demonstrating not only that abandonment is both widespread and under-addressed, but that improved information transparency can significantly accelerate downstream response.*** These findings motivate the design of interventions to improve automated identification of abandonment.

Designing Interventions for Abandonment Disruptions in Software Supply Chains

Since not all abandonment matters equally to most developers [ESEC/FSE'23], indiscriminate alerts could overwhelm developers and cause notification fatigue– a major usability issue limiting the effectiveness of existing tools for other dependency management practices e.g., dependency updates. While manual user customization is a potential solution, it is often not practically viable given the scale of modern dependency trees and the number of urgent tasks competing for developer's time. Thus, the key questions become (1) what abandonment will be impactful to a particular project given the context of their dependency usage; and (2) how to make such predictions at scale. We hypothesize and later demonstrate that our method using ***large language models (LLMs), equipped with theory-driven reasoning and contextual information, can accurately predict the impact of abandonment better than LLMs alone to support developer decision-making*** [ICSE'26]. This method is also promising for other disruptions where traditional tooling approaches have failed and theory or design work is still essential, lowering the barrier to creating sophisticated tools, as we will discuss further in the Future Work section.

We begin by conducting formative need-finding interviews with 22 developers to develop a theoretical understanding of ***what factors influence the impactfulness of a dependency's abandonment on a project given the context of their dependency usage***, identifying four key factors: *depth of integration*, *availability of alternatives*, *importance of the functionality provided*, and *external environmental pressures*.

We then leverage this theoretical understanding to develop Abandabot-Predict, a theory-driven LLM-based classifier to predict the impact of abandonment using context-specific reasoning and information. Abandabot-Predict combines repository mining and static analysis to extract context-specific usage information sufficient for expert judgment using CodeQL enabling production-grade global data-flow analysis. Using the Retrieval-Augmented Generation pattern, we embed this information in a theory-based reasoning prompt fed to the LLM, which executes sequential reasoning steps to produce a binary prediction of *impactful/not impactful*.

Finally, through an independent evaluation study with 124 developers, we demonstrate that our classifier is effective at predicting project-specific impactfulness, achieving a Macro-F1 score of 0.840.

Beyond abandonment, I have investigated other critical disruptions in software supply chains. My work on maintainer disengagement revealed how burnout and life changes drive voluntary exits [OSS'19], informing retention strategies. Studies on toxicity [ICSE'22] and trust decomposition [ICSE-NIER'24] examined how social and security disruptions propagate through development communities. This broader perspective on disruptions enables more comprehensive and well-informed solutions.

Exploring Tomorrow's Disruptions: GenAI Tools in Development Workflows

The rapid deployment of GenAI tools has disrupted software development on a seismic scale, promising a paradigm shift in how software is built and maintained. Yet despite widespread availability, adoption remains uneven even within teams- undercutting expected productivity gains, frusturating management expectations, and casting uncertainty on the future roles of developers. ***To investigate why adoption varies among developers in seemingly similar contexts, we utilize a paired interview design with 54 developers representing 27 pairs from the same team matched on key factors—primary programming language, role, and seniority- but who exhibit contrasting usage patterns (one frequent and one infrequent user)***, identified using telemetry data from a large multinational software company [UNDER-REVIEW'26]. This unique design ensures both developers share the same team-level

context (e.g., codebase, manager, policies), ***enabling direct comparison of context-specific challenges and workarounds to yield an empirically grounded sociotechnical account of GenAI tool usage***.

Our analysis reveals three core insights:

1. **Usage patterns diverge based on mindset and approach**: Frequent users often perceive GenAI as a collaborative partner, integrate it continuously and experimentally, and persist through challenges. Infrequent users more often perceive it as a utility feature, apply it conservatively for narrow well-vetted tasks, and more often abandon it quickly when challenges arise.

2. **Organizational factors amplify adoption differences**: Organizational factors can actively shape individual factors through an amplification effect, e.g., team-specific demonstrations of applying tooling on common development tasks can transform developers' perception of tool usefulness.

3. **The Productivity Pressure Paradox**: A self-perpetuating organizational dynamic we coined where increased productivity expectations from management without corresponding support often creates a paradoxical effect, where developers lack the time necessary to develop the skills that would save time.

These findings challenge the prevailing GenAI deployment strategy across the software industry, which frames the challenge of determining how to effectively use these tools in order to yield the expected systematic productivity gains as the responsibility of individual developers.

Future Work

My long-term research vision centers on helping developers and organizations effectively navigate technological disruptions in software engineering. The software industry faces two major disruptions demanding immediate attention. First, the adoption of GenAI is fundamentally altering and challenging established development processes. This needs to be urgently understood and supported, particularly the long-term impacts on software quality, security, and maintainability. Second, software supply chains face escalating threats from dependency abandonment and sophisticated targeted attacks. ***My future work addresses these disruptions through three complementary research thrusts.*** First, I'll develop AI-powered approaches that transform how we respond to supply chain disruptions, shifting from reactive crisis management to proactive resilience. Second, I'll establish frameworks and infrastructure that enable effective organizational GenAI integration, ensuring usage enhances rather than erodes developer capabilities. Third, I'll investigate how practices like code review must evolve when GenAI fundamentally changes what it means to write, review, and understand code as well as best practices for safe, effective, and genuinely assistive use. ***My research recognizes that disruptions are not aberrations to be avoided but inevitable evolutionary forces that require systematic study and strategic response.***

AI-Powered Supply Chain Sustainability and Security

The exponential growth in supply chain attacks in recent years, including those leveraging abandoned dependencies as an attack vector, are disruptive to software security and highlight the urgent need for more research supporting the management of software supply chains and the associated risks they present. My future work will develop and leverage AI-powered tools and strategies to enable proactive resilience rather than reactive crisis management through three interconnected approaches. The unifying question driving this work: ***how can we leverage AI to transform supply chains from fragile dependency networks into resilient ecosystems that anticipate and adapt to disruptions?***

Approach #1: Intelligent Blueprints for Proactive Abandonment Response. Developers emphasize that just identifying abandonment is not enough– many require guidance on *how to respond* [ICSE'26]. As such, I will extend my dissertation and explore: ***How can we leverage community insights and context-specific analysis to create project specific response blueprints?*** I will conduct controlled experiments evaluating the effectiveness of various approaches for synthesizing wisdom-of-the crowd migration patterns, cross-platform community discussions, and project-specific architectural constraints extracted via static analysis to generate customized intelligent LLM-based response and migration strategies.

While supporting individual responses to abandonment represents progress, there have been growing calls for"*community-oriented solutions*", which show potential for significantly reducing the collective cost of responding to and recovering from digital infrastructure abandonment by reducing redundant effort and increasing coordination [ESEC/FSE'23]. ***How do open source communities successfully recover from dependency abandonment, and what patterns distinguish successful from failed recovery attempts?*** I will use a mixed methods approach combining large-scale software repository data mining, LLMs-as-classifiers, statistical analysis, and targeted qualitative investigation to identify effective coordination strategies, quantify recovery success metrics, and develop evidence-based actionable guidance for navigating critical transitions. This research advances resilient and trustworthy supply chain security by showing how communities can collectively address abandonment—an essential capability for sustaining critical infrastructure amid complex dependency graphs and regulatory requirements.

Approach #2: Enabling Responsible Sunsetting. Although providing explicit notice of package abandonment can help support downstream responses, many maintainers do not announce abandonment due to limited time and resources, uncertainty about how to do so, and a lack of awareness of their downstream users' need. This underscores the need to lower the barriers to responsible project sunsetting. ***Can we enable increased responsible sunsetting by understanding and designing support mechanisms to meet the needs of maintainers winding down projects?*** I will leverage participatory design, controlled A/B experiments, LLMs for artifact generation, and user studies to design and evaluate the effectiveness of tooling and interventions for navigating the process of responsibly sunsetting digital infrastructure projects, as well as best practices for graceful transitions.

Approach #3: Theory-Based Context Engineering Beyond Abandonment. Finally, the approach demonstrated by Abandabot-Predict, of replacing traditional ML pipelines with theory-based context engineering, enables rapid tool develop while maintaining interpretability. By systematically combining qualitative research to identify theoretical constructs with LLMs as scalable reasoning engines, this approach can address diverse software engineering supply chain disruptions that require context-aware decision support. ***I will adapt this approach to address other disruptions like vulnerability prioritization, demonstrating how theoretical frameworks combined with LLM reasoning can rapidly address diverse supply chain threats.*** For example, applying this approach to vulnerability prioritization would involve developing empirical models of exploitation likelihood, extracting application-specific attack surface metrics, and synthesizing risk assessments calibrated to deployment contexts.

GenAI Integration Infrastructure

Many organizations adopt GenAI tools expecting systematic productivity gains while simultaneously delegating the responsibility of determining how to effectively integrate these disruptive tools into existing workflows in order to yield the expected productivity gains to individual developers. However, my recent work demonstrates how increased productivity expectations from management without corresponding support can lead to pressure-induced reversion to familiar methods, hindering the investments into skill development necessary to achieve the expected productivity gains and creating a self-perpetuating organizational dynamic which we refer to as the Productivity Pressure Paradox. Highlighting the urgent need for frameworks outlining how organizations can actively support and shape GenAI tool usage through both support mechanisms and assistive infrastructure. ***What organizational support mechanisms are effective at supporting developer's GenAI workflow integration and skill development in various contexts?*** I will conduct randomized controlled trials with industry collaborators to quantify the effectiveness of different support mechanisms (e.g., context-specific resources, knowledge sharing structures, and AI champions) across diverse team contexts. ***What tooling and support infrastructure is needed to enable effective integration of GenAI tools into existing workflows?*** I will design, build, and evaluate novel integration support infrastructure using participatory design, contextual inquiry, and usability studies to inform prototype development. Initial prototypes include: lightweight templates for team task-specific prompt catalogs, version-controlled prompt evolution tracking systems, and workflow analyzers that identify automation opportunities through task decomposition analysis.

Finally, I will explore how we can move past shallow vanity adoption metrics (e.g., usage frequency) to better measure not just use but *effective use* by asking: ***What value-based metrics measure effective GenAI tool use and skill development, and how can we operationalize them at scale?*** I will

employ a mixed-methods empirical approach to develop metrics for AI skill progression: tool exploration time, effective use case discovery, and debugging efficiency for AI-generated artifacts. These metrics, validated through longitudinal studies tracking developer usage trajectories, will enable organizations to distinguish between superficial usage and meaningful skill development.

Software Process Evolution for the GenAI Era

AI-generated code fundamentally disrupts established software supply chain security practices [S3C2'25], compromising existing review protocols, security assessments, and trust frameworks that are essential for ensuring supply chain integrity. Traditional code review assumes human-scale changes and comprehensible implementation logic. However, as the size of pull requests increases from hundreds to thousands of lines, developers are unable to perform the same comprehensive review process. Code review becomes "*it seems to work*" rather than "*I understand this.*" My work will ***help shape the evolution of software development and maintenance processes in the new paradigm of GenAI-enabled software engineering***.

How do code review practices evolve when reviews can't understand implementations? I will empirically analyze how code review practices change when reviewing AI-generated modifications through comparative studies of thousands of pull requests, measuring review effectiveness (defect detection rates, security issue identification, review duration) across human-authored versus AI-generated changes. I will also examine the security implications of evolving review practices through longitudinal analysis of vulnerability introduction rates in packages pre/post AI adoption across npm and PyPI ecosystems. This will include quantifying the security impact of reduced human code comprehension on downstream supply chain integrity.

One emerging strategy for addressing the challenges of reviewing AI-generated code is to use GenAI tools for automated code review. This shift toward closed-loop systems where AI both produces and evaluates code raises urgent questions about oversight: despite vendor warnings, best practices for integrating human judgment remain unclear. ***How can GenAI automated code review be incorporated into review processes while ensuring strategic human oversight to preserve quality guarantees?*** Through large-scale repository mining, natural experiments, and strategic qualitative analysis, I will triangulate evidence to reveal interaction patterns that balance automated review with effective human oversight.

Vibe coding is the next big GenAI disruption to software development processes. Vibe coding is an emergent software development paradigm in which developers specify software functionality through natural-language interaction with large language models rather than direct code authoring. Tools for vibe coding including autonomous coding agents like *Claude Code*, which relies on natural language prompts in a terminal window, and *Github Copilot coding agent*, which can be assigned an issue on GitHub and then independency generate code and submit a pull request for review, ***represent a proposed shift in the human role in software development from implementation to iterative intent-driven guidance and validation.***

This shift raises fundamental questions about software quality, developer expertise, and the nature of programming itself while also providing opportunities for developing augmented maintenance support tooling. ***How can vibe coding and autonomous coding agents be leveraged to reduce maintenance toil while preserving software security, compliance, and maintainability?*** My research will analyze real-world developer interaction patterns with coding agents, identifying friction points where current approaches fail and developing theory to explain when coding agents might erode code quality and when good practices can mitigate it. My work will help answer the critical question of: ***what is the difference between what LLMs could be used for and what they should should be used for***, in order to sustain software quality, meet compliance standards, and ensure that tools provide genuinely assistive support.

References

[(OSS'19)] Courtney Miller, David Gray Widder, Christian Kästner, and Bogdan Vasilescu. "Why Do People Give Up FLOSSing? A Study of Contributor Disengagement in Open Source". In: *IFIP International Conference on Open Source Systems*. 2019, pp. 116–129.

[(ICSE'25)] Courtney Miller, Mahmoud Jahanshahi, Audris Mockus, Bogdan Vasilescu, and Christian Kästner. "Understanding the Response to Open-Source Dependency Abandonment in the npm Ecosystem". In: *Proc. Int'l Conf. Software Engineering (ICSE)*. 2025. ACM Distinguished Paper Award.

[(ICSE'26)] Courtney Miller, Hao He, Weigen Chen, Elizabeth Lin, Chenyang Yang, Bogdan Vasilescu, and Christian Kästner. "Designing Abandabot: When Does Open Source Dependency Abandonment Matter?" In: *Proc. Int'l Conf. Software Engineering (ICSE)*. 2026.

[(ICSE'21)] Courtney Miller, Paige Rodeghero, Margaret-Anne Storey, Denae Ford, and Thomas Zimmermann. ""How Was Your Weekend?" Software Development Teams Working From Home During COVID-19". In: *Proc. Int'l Conf. Software Engineering (ICSE)*. 2021. ACM Distinguished Paper Award.

[(ESEC/FSE'23)] Courtney Miller, Bogdan Vasilescu, and Christian Kästner. ""We Feel Like We're Winging It:" A Study on Navigating Open-Source Dependency Abandonment". In: *Proc. Europ. Software Engineering Conf./Foundations of Software Engineering (ESEC/FSE)*. 2023.

[(ICSE'22)] Courtney Miller, Sophie Cohen, Daniel Klug, Bogdan Vasilescu, and Christian Kästner. ""Did You Miss My Comment or What?" Understanding Toxicity in Open Source Discussions". In: *Proc. Int'l Conf. Software Engineering (ICSE)*. 2022. ACM Distinguished Paper Award.

[(ICSE-NIER'24)] Lina Boughton, Courtney Miller, Yasemin Acar, Dominik Wermke, and Christian Kästner. "Decomposing and Measuring Trust in Open-Source Software Supply Chains". In: *Proc. Int'l Conf. Software Engineering: New Ideas and Emerging Results (ICSE-NIER)*. 2024.

[(UNDER-REVIEW'26)] Courtney Miller, Rudrajit Choudhuri, Mara Ulloa, Sankeerti Haniyur, Robert DeLine, Margaret-Anne Storey, Emerson Murphy-Hill, Christian Bird, and Jenna L Butler. ""Maybe We Need Some More Examples:" Individual and Team Drivers of Developer GenAI Tool Use". In: *Under Review* (2026).

[(S3C2'25)] Courtney Miller, William Enck, Yasemin Acar, Michel Cukier, Alexandros Kapravelos, Christian Kästner, Dominik Wermke, and Laurie Williams. "S3C2 Summit 2024-08: Government Secure Supply Chain Summit". In: *arXiv preprint arXiv:2504.00924* (2025).