A euphoric engineer sprints a glowing track toward a banner marked PLANNED while, unseen behind, the track buckles under a freight car labeled REWORK

The AI Productivity Tax Your Schedule Is Not Pricing

Syed Umair Shoaiby
Technical Program Manager at Cirrus Logic | Leading Semiconductor Silicon NPI from Tape-out to Production | AI...

June 12, 2026

Sixteen seasoned developers. Their own repositories, code they had lived in for years. Half were given AI tools. Half were not.

The ones using AI finished their tasks nineteen percent slower.

Here is the part that should stop every program manager cold. Before they started, those same developers predicted AI would make them twenty-four percent faster. After they finished, slower, they still believed it had sped them up by about twenty percent.

That is not a story about bad tools. It is a story about a broken instrument. The people doing the work could not feel the difference between faster and slower. And the schedule you build sits directly on top of that feeling.

We plan programs on velocity. How fast does the team think it is moving. With AI in the loop, that signal is now unreliable in a specific and dangerous direction. The speed is real at the keystroke. The code appears faster. What does not show up in that moment is everything that happens after the merge.

And after the merge is where it lands. A review of 470 pull requests found AI-co-authored code carried about 1.7 times more issues per pull request than human-written code, 10.83 versus 6.45. Veracode's 2025 analysis across more than a hundred models found AI-generated code introduced

a security flaw in roughly 45 percent of samples, at close to three times the rate of human-written code. Google's DORA team put the system view most plainly. Adoption is near universal, four out of five developers feel more productive, and yet delivery throughput stalls near a ten percent ceiling because AI creates change faster than the stabilization systems downstream of the keystroke can absorb it.

I have watched this exact shape in semiconductor programs, just with different words on it. A step looks done. The engineer is confident, and the confidence is genuine. Then it surfaces three gates later, at the worst possible time, on the critical path where there is no slack left to absorb it. We have a name for the gap between what looks finished and what is actually finished. We call it yield. And we never plan a program assuming yield is one hundred percent.

That is the trap with AI velocity. The speedup is felt early, at the individual, in the moment. The cost is paid late, at the system, after the merge. A schedule priced on the felt speed books the gain now and quietly assumes the rework will be zero. It never is. So the plan is not slightly optimistic. It is structurally optimistic, and the error surfaces at exactly the point where you have no buffer.

To be fair to the data, this is not a fixed law. When METR re-ran a version of the study in early 2026, the measured slowdown shrank to about four percent, and a third to half of the developers refused to work without AI at all. The tools are improving and so are the people using them. The durable finding is not the exact number. It is the gap itself. Felt velocity and measured throughput are two different quantities, and AI widens the distance between them.

So the fix is not to ban the tools. It is to stop trusting the wrong instrument. Two changes earn their keep.

Estimate from measured throughput, not felt velocity. Instrument the things that actually move the finish line. Cycle time from commit to merged-and-stable. Defect-escape rate with AI in the loop. How much review churn each AI-assisted change generates. Felt faster is a vibe. Shipped and stable is a measurement.

And budget the rework tax up front. The way an NPI plan never assumes perfect yield, an AI-accelerated plan should never assume zero rework. Treat AI-generated code as a higher-defect-density input that needs a review and test buffer, not as free finished work. Put the line item in before the work starts, because the gate where it surfaces is the gate where the slip is already locked in.

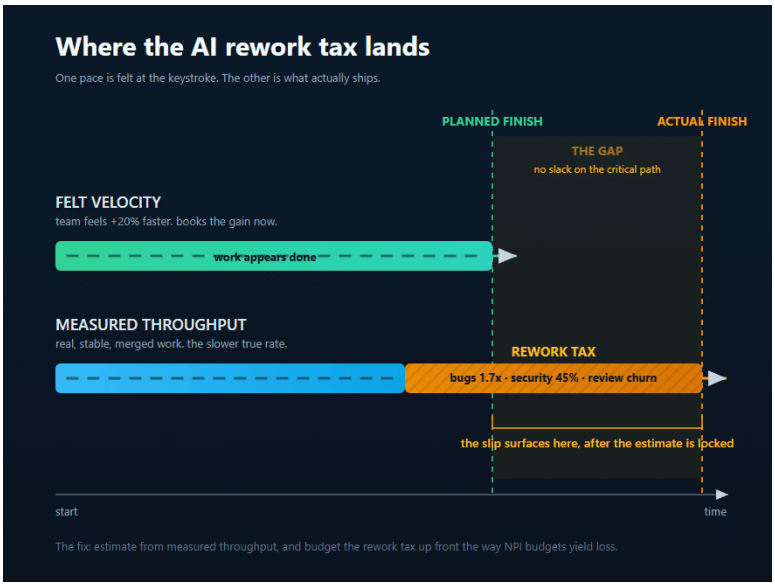
The teams that win with AI will not be the ones who move fastest. They will be the ones who can still tell the difference between moving fast and feeling fast.

Where have you seen AI make the work feel faster while the real finish line quietly moved out? And when it surfaced, where in the schedule did you find the slack?

Disclaimer: The views shared here are my own and do not represent the positions of my employer or any organization I am affiliated with.

Company examples are drawn from public reporting and are used for illustration, not as commentary on any specific business. All figures come from the sources linked with this article and reflect the data available at the time of writing.

#AI #ProgramManagement #AgenticAI #SoftwareEngineering



A program timeline where a fast felt-velocity bar reaches the planned finish early while a slower measured-throughput bar plus a stacked rework tax of bugs, security findings, and review churn pushes the real finish out past the plan.

REFERENCE GUIDE.

- "Experienced developers were 19 percent slower with AI, predicted 24 percent faster, still felt 20 percent faster afterward" METR randomized controlled trial, 16 experienced open-source developers on 246 real issues in mature repositories, July 10 2025.

https://metr.org/blog/2025-07-10-early-2025-ai-experienced-os-dev-study/

- "AI-co-authored code carried about 1.7 times more issues per pull request, 10.83 versus 6.45" CodeRabbit, State of AI vs Human Code Generation Report, 470 open-source GitHub pull requests, Dec 17 2025.

https://www.coderabbit.ai/blog/state-of-ai-vs-human-code-generation-report

- "AI-generated code introduced a security flaw in roughly 45 percent of samples, at close to three times (about 2.74x) the rate of human-written code" Veracode 2025 GenAI Code Security Report, 100-plus models across four languages.

https://www.veracode.com/resources/analyst-reports/2025-genai-code-security-report/

- "Near-universal adoption, 80-plus percent feel more productive, delivery throughput stalls near a 10 percent ceiling, AI overwhelms the stabilization

systems" Google DORA 2025 report, plus Faros AI summary on the system-level ceiling.

https://dora.dev/insights/balancing-ai-tensions/

https://www.faros.ai/blog/key-takeaways-from-the-dora-report-2025

- "METR early-2026 re-run measured about a 4 percent slowdown, and a third to half of developers refused to work without AI" METR, We Are Changing Our Developer Productivity Experiment Design, Feb 24 2026 (the durable finding is the felt-vs-measured gap, not the exact number).

https://metr.org/blog/2026-02-24-uplift-update/

- The semiconductor yield framing and the wafer-map / tapeout-gate analogies are the author's own lived practitioner experience, illustrative and not externally cited.

Syed Umair Shoaiby

Technical Program Manager at Cirrus Logic | Leading Semiconductor Silicon NPI from Tape-out to Production | AI Enabled Program Execution, Workflow Automation & Operational Strategy | Scaling Cross Functional Programs