The Copilot That Cannot Fly Straight: What the 150-Developer Study Really Measures

Weekly Analysis — https://ainews.social

The word *copilot* is doing a great deal of quiet work. A copilot sits in the right-hand seat, qualified to fly the aircraft, sharing the controls with someone equally qualified, ready to take over if the captain has a heart attack over the Atlantic. That is the image Microsoft wants in your head when it brands its assistant a Copilot: a second competent pilot, not a passenger, not a salesman, not a meter running on your data. The trouble is that the evidence gathered this week — the self-commissioned adoption surveys, the developer-productivity studies, the security disclosures, the pricing pages that quietly restructure who owns your workflow — describes something that cannot actually fly the plane. It can hold the yoke steady on a clear day. It also, on occasion, banks hard toward terrain while the instruments insist everything is nominal. The question worth asking is not whether these tools help. Sometimes they plainly do. The question is what the flood of confident measurement is measuring, and what it has arranged to leave uncounted.

Start with the most-cited number of the season. Microsoft's own [1] frames the coming year around agents and productivity, and it arrives, as these reports always do, with adoption figures that trend reassuringly upward. This is a document produced by the company that sells the product it is surveying interest in — a fact that should not be buried but placed at the front of the analysis, because it structures everything downstream. When Stanford's [20] tallied how professional developers described AI tools, the sentiments clustered around increased productivity, accelerated learning, and enhanced efficiency — the vocabulary of the vendor, reflected back through the user. The measurement instruments and the marketing instruments have converged. That convergence is the story.

[1] 2026 Work Trend Index report: Agents, human agency, and ...

[20] AI Index Report 2024

The 92% That Sells Itself

Consider how the adoption number is manufactured, and then consider what it is for. Microsoft's enterprise documentation is unusually

candid once you read it as a business plan rather than a manual. The [15] and the [18] describe an assistant that reaches across your mail, your calendar, your files, your meetings, your organizational graph — the whole record of who you talk to and what you produce. The pitch for this reach is "no additional cost," or something adjacent: it is *already there*, folded into the subscription you already pay, activated with a toggle. That phrasing is the trick. What is presented as generosity — the best of our AI, now simply included — is the mechanics of default. Google runs the identical play, announcing that [11], which is to say the AI now arrives whether or not anyone chose it, priced into the seat, waiting.

[15] Preguntas más frecuentes sobre Microsoft 365 Copilot Empresa

[18] Microsoft 365 Copilot - Service Descriptions | Microsoft Learn

[11] Le meilleur de l'IA de Google est désormais inclus dans les abonnements ...

The vendors have thought carefully about how to convert that default into a habit. Microsoft publishes a whole genre of enablement literature — the [12] and the operational [16] — instructing administrators not merely to install the tool but to drive usage: to identify champions, seed habits, measure engagement, and report adoption upward. Read those documents beside the survey and the circularity becomes visible. The company builds the funnel, meters the funnel, and then commissions research reporting that the funnel is full. A 92% adoption figure produced by this apparatus does not measure whether the tool is good. It measures whether the tool is unavoidable. Those are different findings, and only one of them is being advertised.

[12] Microsoft 365 Copilot adoption guide and overview for IT admins

[16] Rollout Microsoft 365 Copilot to your organization

This is what Kate Crawford, in [20], calls enchanted determinism: the presentation of a technology's spread as a natural fact, an inevitability to be adapted to rather than a corporate strategy to be evaluated. The enchantment obscures power, she argues, and closes off the possibility of informed public scrutiny or outright rejection. The "no additional cost" framing is enchantment in its purest retail form. Nothing is free. The price of the assistant that reads your entire working life is that your entire working life now trains, depends on, and is legible to a single vendor's stack — and the switching cost of that arrangement compounds every quarter you remain inside it. The genius of the framing is that the invoice never itemizes the thing you are actually paying.

[20] The Atlas of AI

When Task Completion Is the Only Variable

Which brings us to the developers, and to the study that keeps getting invoked as proof. The premise of the 150-developer comparison — quality against slop, measured across a population large enough to sound authoritative — sounds like exactly the rigor a skeptic should welcome. And in a narrow sense it is. But look at what the design

permits the study to see and, more importantly, what it structurally cannot. When the dependent variable is task completion time, or ticket throughput, or the ratio of accepted to rejected suggestions, the instrument can register speed with great precision. It cannot register the formation of competence, because competence formation is a longitudinal variable and completion time is a stopwatch. When the only thing you can measure is how fast the task closed, the slow accumulation of understanding — the thing that used to be the *point* of doing the task — becomes definitionally invisible. It is not that the study found competence unaffected. It is that the study was built in a way that could never have detected the difference.

The one piece of research this week that actually pointed its instrument at the missing variable found something the productivity numbers cannot. Anthropic's own investigation into [9] treats skill acquisition, not throughput, as the object of measurement — and once you ask that question the picture stops being flattering. The developer who accepts a correct completion has closed the ticket and learned nothing about why it was correct; the developer who fought the same problem for twenty minutes has closed it slower and knows the terrain. The productivity survey scores the first developer higher. It is optimizing for the metric that erodes the capability. This is the same asymmetry the [20] captures without quite naming: developers report "accelerated learning" as a sentiment, a feeling, a self-report collected in the moment of use — not a demonstrated retention measured six months later, when the scaffolding is gone and the underlying skill either exists or does not.

[9] How AI assistance impacts the formation of coding skills

[20] AI Index Report 2024

There is a deeper problem with taking completion as the proxy for value, and Janelle Shane names it precisely in [20]: the recurring failure of machine systems is that they solve a different, easier problem than the one you thought you posed — sorting medical case files by length rather than content, and passing the benchmark anyway. A coding assistant that produces code which compiles, passes the visible tests, and closes the ticket has satisfied every variable the productivity study tracks. Whether it solved *your* problem, or an adjacent easier one that merely resembles it, is a question the stopwatch was never designed to ask. The slop-versus-quality binary flatters us into thinking the hard distinction is between good output and bad output. The harder distinction, the one the measurement regime is engineered not to surface, is between output that closed a ticket and output that built a person who can close the next ticket without help.

[20] You Look Like a Thing and I Love You

I want to be fair to the tools here, because fairness is the whole point of skepticism worth the name. A great deal of coding is genuinely rote — boilerplate, glue, the fortieth CRUD endpoint — and on

that terrain an assistant that autocompletes the obvious is a real gift, returning hours to work that actually requires a mind. The vendor documentation is not lying when it promises that. The [8] and the [6] describe capabilities that are, on their own terms, real. The problem is not that the tools do nothing. The problem is that the discourse around them has collapsed into a single register — the tool-as-utility framing that dominates this week's coverage to the near-exclusion of everything else — and a register that can only speak the language of utility cannot articulate the cost. When roughly a quarter of all tools coverage frames these systems purely as neutral instruments of efficiency, the framing itself is doing ideological work: it pre-answers the question of what they are for.

[8] GitHub Copilot documentation - GitHub Docs
[6] Gemini Code Assist overview | Google for Developers

The Debugging Hours Nobody Counts

Here is where the productivity ledger gets genuinely dishonest, because the time the assistant saves at the keyboard reappears — often larger — somewhere the survey does not look. Generated code is not free code. It must be read, understood, integrated, and secured by someone who did not write it and therefore does not carry it in their head. The industry has begun, quietly, to document what this costs. The enterprise-facing [2] reads less like a celebration than a warning: it catalogues the ways in which assistant-generated code introduces vulnerabilities, leaks secrets, and expands the surface an organization must now defend. None of that debugging and hardening labor appears as a debit against the productivity gain. It is recorded, if at all, under a different budget line, on a different team's ticket, in a different quarter — which is precisely how a cost gets laundered into invisibility.

[2] AI Coding Assistant Security: Enterprise Guide 2026

The most instructive disclosure of the week is the one that shows the copilot banking toward terrain while the instruments read level. Security researchers at Wiz documented what they named [7] — a class of vulnerability in which the assistant, trusted to sit inside the developer's workflow, could be induced to cross a boundary it was never supposed to cross, taking actions the human believed it had approved when the human had approved nothing of the sort. This is the copilot metaphor failing in the most literal way available. A real copilot does not silently reroute the aircraft while you assume you have the controls. The assistant, embedded deep enough in the toolchain to be useful, is embedded deep enough to be dangerous, and the same reach that Microsoft's service descriptions advertise as convenience is the reach a security researcher describes as attack surface. Convenience and exposure are not opposing properties of

[7] GhostApproval: AI Coding Assistant Trust Boundary Flaw | Wiz Blog

these tools. They are the same property, described by the marketing department and the red team respectively.

Even the vendors concede the point in the fine print. Amazon ships a code assistant and, in the same breath, ships the [17] documentation explaining how to scan the assistant's output for the vulnerabilities the assistant might introduce — a tool to catch the mistakes of the tool, sold alongside it. The full [3] treats security review of generated code as a standing requirement, not an edge case. Read charitably, this is responsible engineering. Read accurately, it is an admission that the productivity figure is gross, not net: the assistant produces suggestions fast, and then a second apparatus — human attention plus another scanner — must be run over those suggestions before they can be trusted. The velocity everyone celebrates is measured at the moment of emission, before the bill for verification comes due. A careful adopter learns to ask a single deflating question of every productivity claim: velocity measured *where*, and paid for by *whom*?

[17] Security scans - CodeWhisperer - docs.aws.amazon.com

[3] Amazon CodeWhisperer Documentation

The evidence from outside the developer world points the same direction. A [19] found the familiar pattern of enthusiastic uptake running well ahead of any settled understanding of effects, with the tools adopted for their immediate convenience while their downstream consequences for the user's own capability remained unexamined by the users themselves. The dynamic is not confined to any one sector; it is a property of how these tools enter a workflow. They arrive promising to remove friction, and friction — the resistance of a hard problem against an unprepared mind — turns out to be where a good deal of learning was happening. Remove it everywhere and you have a faster process staffed by people who understand it less. That trade may be worth making in some places. It is never worth making *unknowingly*, and unknowingly is exactly how the productivity framing arranges for it to be made.

[19] Survey on Undergraduate Student Use of Generative AI

Choice as a Menu of Dependencies

Zoom out from the individual keyboard to the industrial frontier, because the tools do not exist in a market of equals. They exist downstream of a capital expenditure so large it reorganizes what "choice" can even mean. The infrastructure wave now underway — hundreds of billions of dollars in compute, committed by a handful of firms — is the tools story of the decade, and it is the one the productivity coverage most reliably ignores, because it cannot be rendered as a feature. When four companies control the substrate on which every assistant runs, the alternatives are not defeated in the market so much

as starved of the inputs required to exist. Independent options do not lose a fair fight; they never get to have one. This is the material condition beneath every "which tool should we adopt" conversation, and it converts that conversation into something narrower than it appears.

Look at the branded surface and you will see genuine variety: [13] offers tiers and model choices, the [5] listing presents an alternative, Amazon's stack presents a third. Look beneath it and the variety thins to a question of vendor allegiance layered over a compute oligopoly. Choosing Copilot over Gemini Code Assist over CodeWhisperer is choosing which hyperscaler's data center your keystrokes will train, which billing relationship will deepen, which proprietary integration will make the next migration more expensive than the last. This is what it means to say that choice now denotes a flavor of dependency rather than an exit from it. The menu is real. The kitchen is owned by four families.

[13] Modelos y precios para GitHub Copilot

[5] Gemini Code Assist - Visual Studio Marketplace

Shoshana Zuboff's account in [20] supplies the term for what is being optimized here, and it is not the user's capability. The behavioral exhaust of a working population — every prompt, every accepted and rejected suggestion, every correction, every hesitation — is itself the raw material, extracted at a scale no independent competitor can match precisely because no independent competitor commands the installed base. The reach that Microsoft's adoption guides work so hard to maximize is not incidental to the business model; it *is* the business model. Crawford's [20] traces how the same platforms extract cheap, "frictionless" labor from crowdworkers to train the systems while presenting the results as autonomous intelligence — and the developer accepting completions all day is, from a certain angle, an unpaid annotator refining the model that will be sold back to the next developer, and eventually, perhaps, sold *instead of* them. The productivity narrative describes this as empowerment. The capital structure describes it as consolidation. Both descriptions are of the same event.

[20] The Age of Surveillance Capitalism

[20] The Atlas of AI

Noam Chomsky and Edward Herman's [20] supplies the older template for how a self-interested message comes to feel like ambient common sense: not through censorship but through the structure of who produces the information and who profits from its reception. The AI-tools discourse this week is a small, clean instance. The dominant sources of "evidence" about whether the tools help are the companies selling the tools; the dominant framing is utility; the skeptical stance, where it appears at all, is granted the status of a caveat rather than a finding. When the party with the most to gain also owns the survey, the benchmark, the documentation, and the distribution channel, the resulting consensus is not a discovery about the world. It is a reflection of who was permitted to speak.

[20] Manufacturing Consent

The Data Is the Actual Product

It is worth stating plainly what the tool extracts, because the vendors state it plainly too, if you read the right documents. OpenAI's own explanation of [4] describes a development process fed by vast quantities of data, and Google's [14] narrates a similar dependence on ingested material. These are not confessions; they are product descriptions. But read against the "no additional cost" framing they clarify the transaction. You are not the customer being served at no charge. You are the input being processed. The assistant is free the way the sample counter at a store is free — the point is not the sample, the point is that you are now inside the store, and the store is measuring everything you touch.

[4] Cómo se desarrollan ChatGPT y nuestros modelos fundamentales
[14] Más información sobre la IA generativa - Ayuda de Aplicaciones con Gemini

That extraction has a legal and ethical shadow the productivity coverage rarely enters. The reporting collected under [10] documents generative systems reproducing recognizable, copyrighted material — the model returning, under the banner of creation, something close to a copy of what it consumed. The developer-tools equivalent is not hard to imagine and not hard to find: an assistant trained on public and private codebases emitting suggestions that carry the fingerprints of their sources, licensing questions and all. The vendor documentation on models and pricing does not dwell on this. It is a cost that has been externalized onto the people whose work became training data without their meaningful consent, and onto the adopter who will be liable if the borrowed fingerprint turns out to matter. The productivity ledger, once again, does not carry the line.

[10] La IA generativa tiene un problema de plagio visual

Put the pieces together and the shape of the thing resolves. The tool is optimized for a set of metrics — keystrokes saved, tickets closed, tokens emitted, adoption rates climbing — every one of which is legible, quarterly, and flattering. The effects that would complicate the picture — the erosion of the user's own capability, the debugging and security labor displaced onto other budgets, the dependency that raises switching costs toward infinity, the data extraction that is the real product — are none of them measured, because none of them serve the party doing the measuring. This is not a conspiracy. It is an incentive structure operating exactly as designed. The instruments point where the money is, and the money is in adoption and lock-in, not in whether the developer six months from now can still read a stack trace unaided.

What a Careful Adopter Actually Knows

None of this argues for refusal, which would be its own kind of unearned certainty. The tools do real work; the boilerplate really does evaporate; some hours really are returned. A skeptic who pretends otherwise forfeits the argument. What the evidence argues for is a different relationship to the claims — a habit of asking, of every confident number, the questions the number was built to prevent you from asking. When a vendor reports adoption, ask whether adoption was measured or manufactured, and whether the survey's author sells the product. Microsoft's [1] is not therefore worthless; it is evidence that must be read as an artifact of the interest that produced it, the way you would read a tobacco company's research on smoking — not dismissed, but discounted, and cross-examined against sources with no stake in the answer.

When a study reports a productivity gain, ask what it measured and, harder, what its design made invisible. The 150-developer binary can tell you how fast a task closed; it cannot tell you what happened to the developer, and only research that actually points its instrument at skill formation — as Anthropic's work on [9] does — can begin to. When a tool promises to remove friction, ask where the friction was doing quiet work, because the [19] and the coding-skills research point at the same uncomfortable finding: the resistance you are paying to eliminate was sometimes the place the learning lived. And when a tool is offered at "no additional cost," ask what is being harvested in lieu of a fee — the answer, per the vendors' own [18] and OpenAI's account of [4], being the entire recorded texture of your working life.

The red flags, once you know to look, are not subtle. Be wary when the party measuring the benefit is the party selling the product. Be wary when velocity is the only variable and verification is off-screen. Be wary when "included" replaces "chosen," when a security scanner is sold to catch the errors of the tool it accompanies, when the disclosure of a trust-boundary flaw like [7] is treated as a footnote to a story of empowerment. Be wary, above all, of the word *copilot* itself — of any framing that promises a second competent hand on the controls while the industrial and financial structure beneath it, the one Zuboff's [20] and Crawford's [20] describe, arranges for you to fly a plane you increasingly cannot land without permission.

The tools are getting faster. That is measured, real, and repeated in every report. They are also, at the same time and by the same design, getting harder to leave — and that, the thing that actually determines your position ten years out, is the one variable nobody selling you the copilot has any incentive to put on the instrument

[1] 2026 Work Trend Index report: Agents, human agency, and ...

[9] How AI assistance impacts the formation of coding skills

[19] Survey on Undergraduate Student Use of Generative AI

[18] Service Descriptions | Microsoft Learn

[4] Cómo se desarrollan ChatGPT y nuestros modelos fundamentales

[7] GhostApproval: AI Coding Assistant Trust Boundary Flaw | Wiz Blog

[20] The Age of Surveillance Capitalism

[20] The Atlas of AI

panel. A careful adopter does not refuse the tool. A careful adopter refuses the framing, keeps their own hand on the yoke, and insists on counting the costs the vendor has so carefully arranged not to count. The copilot cannot fly straight. It was never built to. It was built to keep you in the seat.

References

1. 2026 Work Trend Index report: Agents, human agency, and ...
2. AI Coding Assistant Security: Enterprise Guide 2026
3. Amazon CodeWhisperer Documentation
4. Cómo se desarrollan ChatGPT y nuestros modelos fundamentales
5. Gemini Code Assist - Visual Studio Marketplace
6. Gemini Code Assist overview | Google for Developers
7. GhostApproval: AI Coding Assistant Trust Boundary Flaw | Wiz Blog
8. GitHub Copilot documentation - GitHub Docs
9. How AI assistance impacts the formation of coding skills
10. La IA generativa tiene un problema de plagio visual
11. Le meilleur de l'IA de Google est désormais inclus dans les abonnements ...
12. Microsoft 365 Copilot adoption guide and overview for IT admins
13. Modelos y precios para GitHub Copilot
14. Más información sobre la IA generativa - Ayuda de Aplicaciones con Gemini
15. Preguntas más frecuentes sobre Microsoft 365 Copilot Empresa
16. Rollout Microsoft 365 Copilot to your organization
17. Security scans - CodeWhisperer - docs.aws.amazon.com
18. Service Descriptions | Microsoft Learn
19. Survey on Undergraduate Student Use of Generative AI
20. The Atlas of AI