Research paper · November 2026

Activity is not output.

What AI really returns in software engineering, and how to measure it from the code.



One 355-engineer organisation, before and after AI coding tools. Stanford research, 2025. Shapes show the sequence, not the scale.

Summary

Count what AI produces, not what it consumes.

AI coding tools are now part of almost every software team, and so is the question that follows them: what did they return? Most organisations answer with activity counts, which rise with AI whether or not more gets delivered. This paper sets out what the research shows, why the usual measures miss it, and how output can be measured from the code itself.

+14% / +1%	In one 355-engineer organisation, pull requests rose 14% after AI coding tools arrived. Output, scored from the code, rose 1%. Rework rose 2.6 times.
+20% / −19%	Experienced developers using AI felt 20% faster and were measured 19% slower in a randomised trial.
15–20%	The typical net gain from AI, once the extra rework is subtracted.
About 4×	How much the gap between leading and lagging AI teams widened between April 2023 and July 2025.
$R^2 \approx 0.2$	How little token spend explains of the gain. Codebase hygiene explains more (R^2 0.3 to 0.4).

Bottom line

Output can be read from the Git history you already have, scored commit by commit and benchmarked against 120,000+ engineers, without surveys or timesheets. Measured that way, AI's return becomes a number you can price licences, suppliers and headcount on.

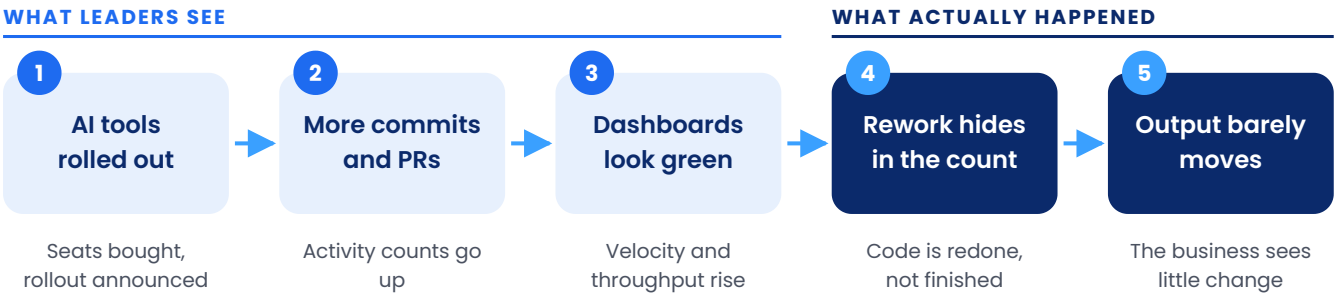
Contents

1	The question on every engineering leader's desk	3
2	What happened in one organisation	4
3	Why the usual numbers cannot see it	6
4	How output is measured from the code	7
5	What the research says about AI	9
6	Reading AI in the code	10
7	The read-out: seven views, seven decisions	12
8	How each reader uses it	14
9	Your code and data stay yours	16
10	From interest to a decision	17
	Sources and notes	18

1 The question on every engineering leader's desk

Everyone uses AI. Few can show what it returned.

Ninety per cent of technology professionals now use AI at work. Every supplier says it delivers with AI. Rate cards are under pressure and AI budgets are up for review. Engineering leaders are asked to prove the return, and the numbers they have measure activity.



Pull requests, commits and AI adoption all go up. Net output, scored from the code, moved by about 1%.

Every box on the left looks like progress on a dashboard. The two on the right are what the code shows.

Three decisions priced on activity

When activity is the only number available, three decisions that shape next year's engineering budget get priced against it. Each is a bet without a measure of output.

Licences

AI licences renewed on adoption, not on output per dollar.

Rate cards

Supplier rate cards negotiated on claims, not on measured output.

Headcount

Headcount plans sized on activity that includes rework.

90%

of technology professionals use AI at work (Google DORA, 2025)

3

budget decisions usually priced on activity: licences, rate cards and headcount

In short

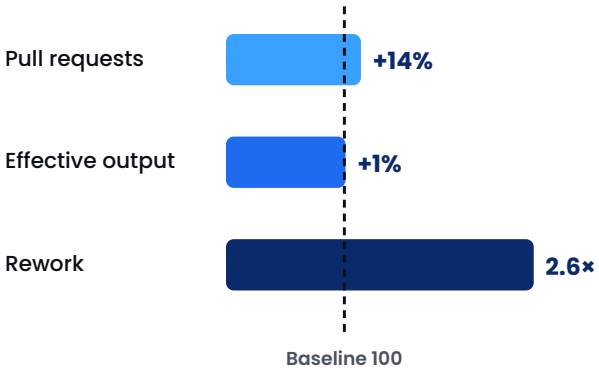
Pull requests, commits and adoption rates rise with AI whether or not more gets delivered. A measure of output is what turns these decisions from bets into evidence.

2 What happened in one organisation

Activity went up. Output barely moved.

One 355-engineer organisation rolled out AI coding tools in May 2025. Between January and August, pull requests rose 14%. Output, scored from the code, rose 1%. Most of the extra activity was people redoing recent work.

INDEX, PRE-AI BASELINE = 100



Indexed so that the period before AI is 100. Pull requests rounded from 13.6%.

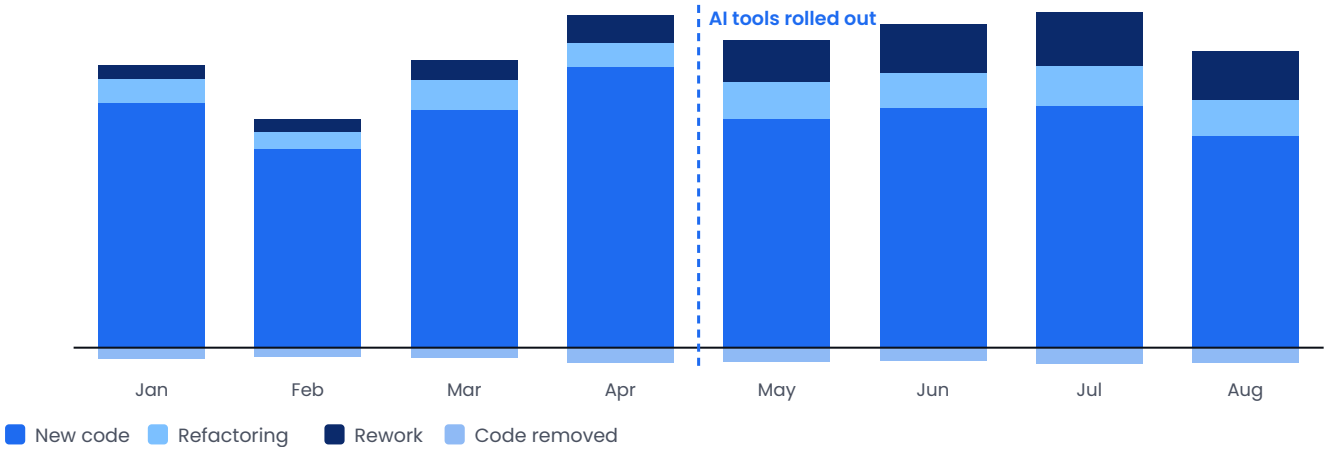
+14%
pull requests: more activity, not more output

+1%
effective output: no meaningful change

2.6×
rework: most of the extra code

−9%
code quality, with its variance up 3.6 times

WHERE EFFORT WENT, MONTH BY MONTH, JANUARY TO AUGUST 2025



Redrawn from the published research; values approximate. Code removed is shown below the line.

In short

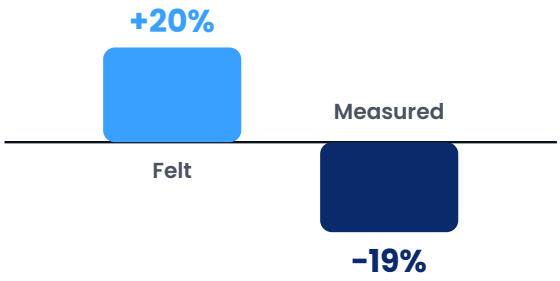
Pull-request counts said AI was working. The code said most of the extra activity was rework. Without measuring output, nobody would have known the return.

2 What happened in one organisation, continued

Asking people does not fix it.

If activity counts mislead, the instinct is to ask engineers how much faster they are. That does not work either: people are poor judges of their own productivity, with or without AI.

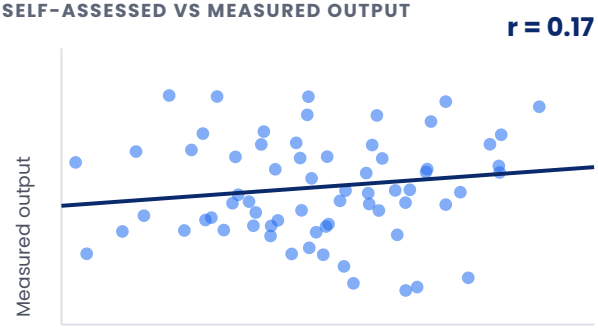
EXPERIENCED DEVELOPERS USING AI



METR randomised trial, 2025

In a randomised trial, experienced developers using AI tools felt about 20% faster. Measured, they were 19% slower.

SELF-ASSESSED VS MEASURED OUTPUT



Illustrative: what a correlation of 0.17 looks like

Across the research, self-assessment surveys are a weak predictor of real output (r = 0.17). A survey tells you how people feel about AI, which matters, but not what it returned.

What the two failures have in common

Dashboards count motion.
Commits, pull requests and story points rise with AI because AI makes it cheap to produce changes, including changes that are redone a week later.

People report feelings.
Feeling faster and being faster are different things, and surveys cannot separate them.

Neither sees the code.
Only the code shows what was delivered, what lasted and what was redone. That is where output has to be read.

More activity is not more output. Without measuring, nobody knows the return.

In short
Neither the dashboards nor the people can see the effect of AI. The code can.

3 Why the usual numbers cannot see it

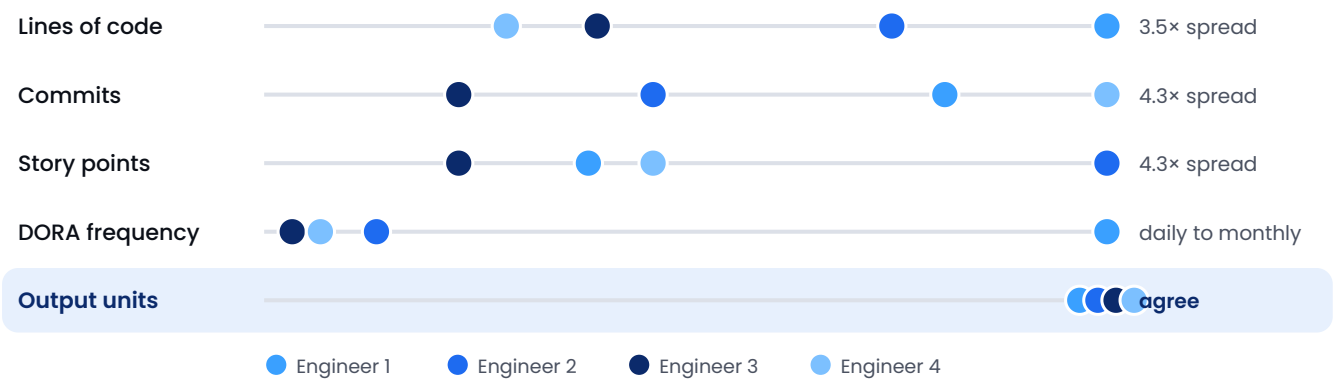
Activity measures disagree about the same work.

Take four engineers and four equivalent pieces of work. Lines of code, commits, story points and deployment frequency rate them differently by more than three times. Scored from the code, the four are equal.

Same work	Eng 1	Eng 2	Eng 3	Eng 4	What it misses
Lines of code	2,029	1,511	802	583	Verbose code is rewarded
Commits	21	12	6	26	Rewards splitting work up
Story points	5	13	3	6	Subjective, easy to inflate
DORA frequency	Daily	Weekly	Monthly	Bi-weekly	Speed, not what was delivered
Output units	**10**	**10**	**10**	**10**	**The same work scores the same**

Illustrative example of four equivalent pieces of work.

EACH MEASURE SCALED TO ITS OWN MAXIMUM · FOUR EQUIVALENT PIECES OF WORK



Lines of code
More lines are not more output, and counting them rewards verbose code.

Commits and pull requests
They reward splitting work into small pieces, not delivering more.

Story points
Subjective, easy to inflate and not comparable across teams.

Self-assessment surveys
A weak predictor of real output (r = 0.17).

DORA metrics
Useful for delivery speed and DevOps maturity, but they do not measure what was delivered.

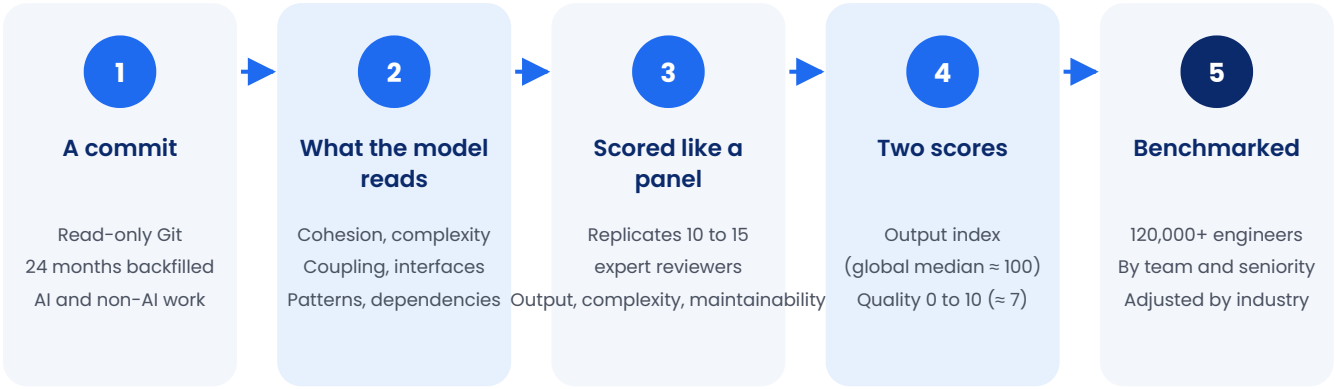
In short

Only a measure of what was delivered agrees with itself across teams, suppliers and time. That is the measure every AI, supplier and headcount decision needs.

4 How output is measured from the code

Every commit, read and scored like an expert panel.

The benchmark connects read-only to your Git repositories and scores up to 24 months of history, commit by commit, so you see before and after AI without waiting. Nothing is written back, and nobody fills anything in.



From a commit to a benchmarked score. History is scored on connection.

What the model reads in every commit

Cohesion	Complexity	Coupling	Data structures	Interfaces	Methods
Persistence layers	APIs consumed	Architectural patterns	Dependencies	Dependency injection	And more

Twelve attributes of the change, across 44+ languages and frameworks.

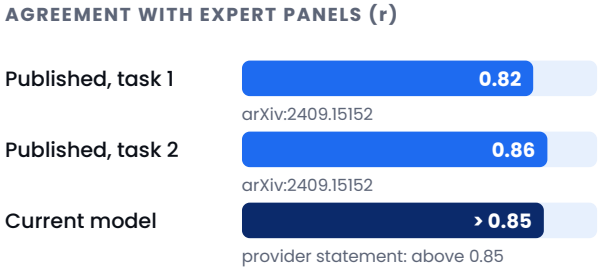
Two scores, one benchmark

Output index. The value delivered per engineer, team or partner, as an index against the dataset's global median of about 100. Output is averaged per contributor per month, so teams of different sizes compare fairly, and the benchmark is adjusted by industry.

Code quality, 0 to 10. The maintainability and complexity of what was written, against a global median of about 7.

Both are set against the research dataset of 600+ companies and 120,000+ engineers, quartile by quartile, by team and seniority, with teams and delivery partners side by side.

Why it holds



The model is trained to replicate what a panel of 10 to 15 independent expert reviewers would say about each commit. The published paper reported correlations of 0.82 and 0.86 on two evaluation tasks; the platform provider states that the current model exceeds 0.85.

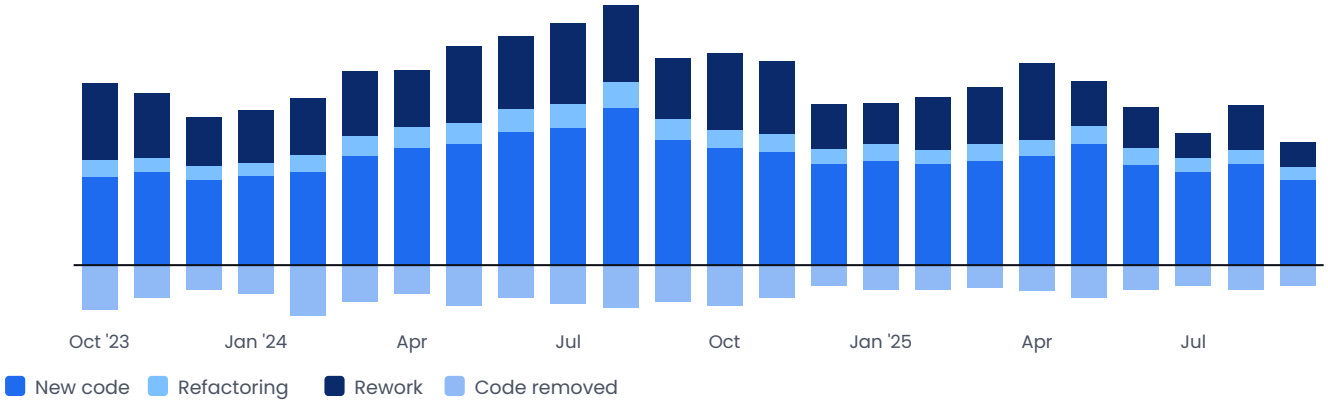
What counts as output, and what is waste.

Not every line written survives. The read-out splits each month's effort into new code, refactoring, rework and code removed. The first two count as output. The last two are waste: effort that does not last.

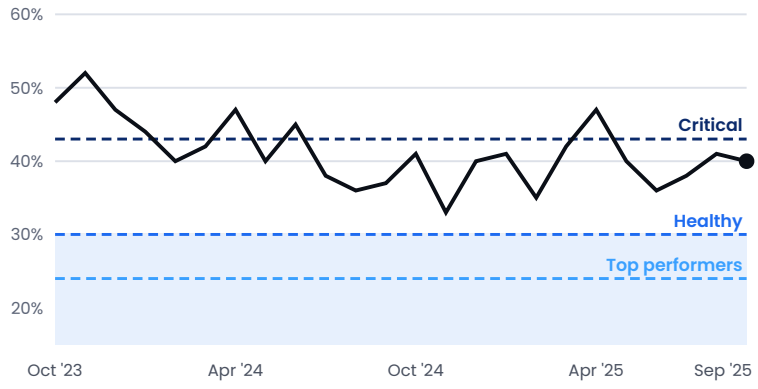
COUNTS AS OUTPUT		WASTE: EFFORT THAT DOES NOT LAST	
New code	Refactoring	Rework	Removed
Net new capability	Improving existing code	Rewritten soon after writing	Deleted without being used

Proportions illustrative. Waste rate = rework plus code removed, as a share of effort.

WHERE EFFORT GOES, EACH MONTH, OVER TWO YEARS (SAMPLE READ-OUT)

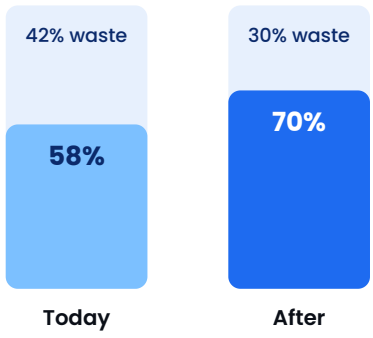


Illustrative, from an anonymised composite read-out. In this sample, rework runs at 1.5 to 2 times the industry average.



Waste rate each month. Healthy is below 30%; top performers are below 24%. Sample, illustrative.

IF WASTE FELL FROM 42% TO 30%



Effort that lasts, same headcount

(1 - 30%) ÷ (1 - 42%) - 1 ≈ +21% effective capacity with the same headcount.

In short

Rework is code rewritten soon after it was written, usually because specifications changed or tickets were unclear. It is a symptom upstream of engineering: fix ticket quality first. For a delivery partner, lower waste is margin.

5 What the research says about AI

AI raises activity almost everywhere. Output depends on the team and the codebase.

The Stanford Software Engineering Productivity Research Group has studied AI's effect on engineering output with the same method, across a dataset of 600+ companies and 120,000+ engineers. Five findings matter for anyone deciding where to spend on AI.

15–20%

The typical net gain from AI, once the extra rework is subtracted (n = 2,562). The net figure, not the gross rise in activity.



Stanford research, 2025

About 4×

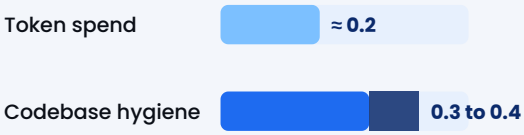
The gap between leading and lagging AI teams widened about four times: an interquartile difference in net gain of 4.8% in April 2023 and 19% in July 2025, 46 AI teams matched with 46 non-AI teams.



Stanford research, 2025 (A)

$R^2 \approx 0.2$

Spending more on AI tokens explained little of the difference between 27 teams. Cleaner engineering environments went with bigger gains (R^2 0.3 to 0.4).



Share of the difference in gain explained (R^2), 27 teams

Stanford research, 2025 (A)

28–38%

Coding agents brought 28 to 38% more commits at every maturity level, across 441 repositories. More commits, which is why commits alone cannot be the measure.

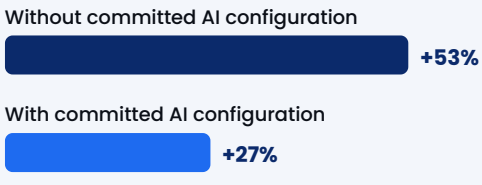


arXiv, 2026 (B)

53% vs 27%

Without committed AI configuration, cognitive complexity rose 53% after coding-agent adoption. With it, 27%. A few pages of committed rules and context for the agents went with a lower quality cost.

arXiv, 2026 (B)



Rise in cognitive complexity after coding-agent adoption

In short

Clean codebases and review guardrails get more out of AI than bigger token budgets do, and the leaders are pulling away. Measure the net gain, team by team, before deciding where the next AI pound or dollar goes.

6 Reading AI in the code

No surveys. AI use is read from the Git history itself.

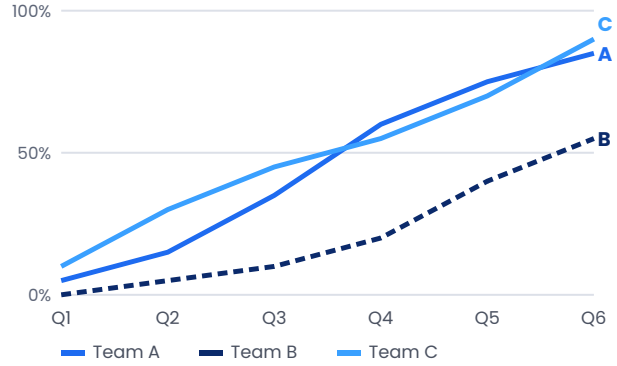
Every commit is checked for signs that AI was involved: an agent's branch, an agent as the author, a message pattern such as a co-author line, or a connected tool that reports the commits made with it. Output, rework and quality are then split between AI and non-AI work.

EVIDENCE OF AI, CHECKED ON EVERY COMMIT · 6 OF 8 SAMPLE COMMITS

a3f9c21	claude/add-retry-tests	Add retry tests for payment webhooks	Agent branch
7be0d44	copilot-swe-agent[bot]	Bump dependency and fix the breaking import	Agent as author
19c2e8a	feature/invoice-export	Export invoices to CSV (Co-authored-by: Cursor Agent)	Message pattern
e41f07b	feature/search-filters	Faceted search on the product list	Connected tool
5d8a913	fix/login-timeout	Increase the session timeout for SSO users	No evidence
c07b6e2	codex/refactor-pricing	Split the pricing service into two modules	Agent branch
8f3d2a0	hotfix/null-currency	Guard against a missing currency code	No evidence
2ae6c95	feature/audit-log	Audit log for admin changes (Generated with Claude Code)	Message pattern

"AI-assisted" means evidence that AI was involved, not a claim that AI wrote the code. Commits illustrative.

SHARE OF ENGINEERS WITH AI EVIDENCE



The rollout rarely spreads evenly. Illustrative.

Four signals, one rule

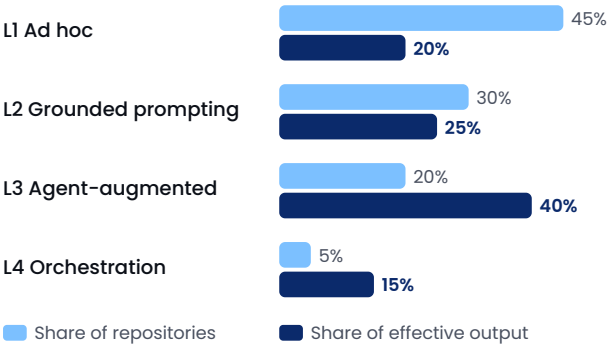
- **Agent branch**, for example a branch created by Claude Code or Codex.
- **Agent as author**, for example a Copilot agent bot.
- **Message pattern**, for example a co-author or "generated with" line.
- **Connected tool**, where the tool's integration confirms use.

Where there is no evidence, the work is treated as non-AI. "AI-assisted" means evidence that AI was involved, not a claim that AI wrote the code. It works from Git alone; connecting the AI tools adds usage and spend from Copilot, Cursor, Windsurf, Claude, OpenAI, Amazon Bedrock and others.

From adoption to payoff: which teams, which licences, at what cost.

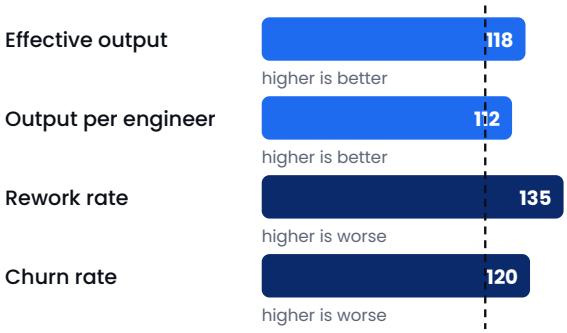
Adoption says how much AI is used. Three further views say how well, what it changed, and what it cost per unit of output.

AI MATURITY BY REPOSITORY



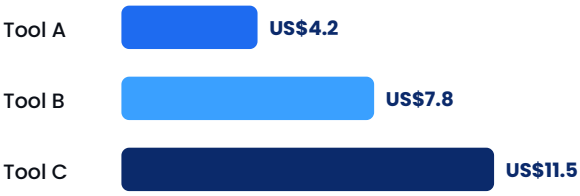
How well, not how much. Each repository is placed on a four-level ladder, from ad hoc use to orchestrated agents. Weighting by output shows whether the most mature practice sits on the work that matters.

BEFORE AND AFTER AI (PRE-AI = 100)



Before and after. History is scored on connection, so the twelve months before the AI adoption date become the baseline at once. A before-and-after view of one organisation, not proof that AI caused the change.

AI SPEND PER OUTPUT UNIT (US$, LOWER IS BETTER)



From each tool's own billing

Cost per output unit. Tokens are counted differently by every tool and are not comparable. Spend per unit of output is, so licences can be kept, grown or cut on what they produce per dollar.

What it tells you

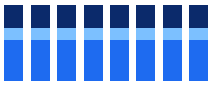
- Whether AI raised output on every team and supplier, not only on new builds
- Which practices produce the gain, and which add rework
- Where AI cost per unit of output is rising
- What to coach: replicate the strongest teams, support the rest

All figures on this page are illustrative, from an anonymised composite read-out.

7 The read-out

Seven views, each tied to a decision.

The read-out turns scored history into seven views. Each answers one question and supports one decision. The figures shown in this chapter come from an anonymised composite read-out and are illustrative.

View	What you see	The question it answers	The decision
1 **Output vs industry**		Where are we against the industry, and which way is it trending?	**Check team changes, onboarding load or a shift in scope**
2 **Where effort goes**		How much is new code, refactoring, rework and removal?	**Fix specifications and ticket quality**
3 **Waste**		How close are we to the critical level?	**Size the capacity gain from cutting waste**
4 **Teams**		Which team carries the waste, and is it quality or process?	**Fix process or requirements, not people**
5 **AI impact**		Did AI raise output, net of rework?	**Keep, grow or cut each licence**
6 **Output per dollar**		What does each delivery model return per US$?	**Choose or rebalance delivery partners**
7 **Staffing scenario**		What if the team mix changes?	**Model it, then add the manager's view**

Top 35%

Sample: output in the top 35% of the industry, but trending down over six months

42%

Sample: waste close to the critical level; at 30% the same team gains about a fifth more capacity

About 3×

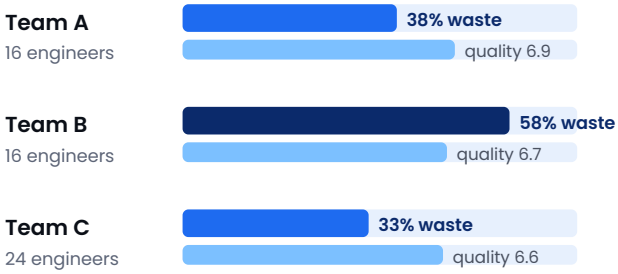
Sample: similar output at a quarter of the cost means about three times the output per dollar

In short

Every view ends in a decision you can take: on licences, suppliers, process or staffing. Views at engineer level are guidance only and can be switched off.

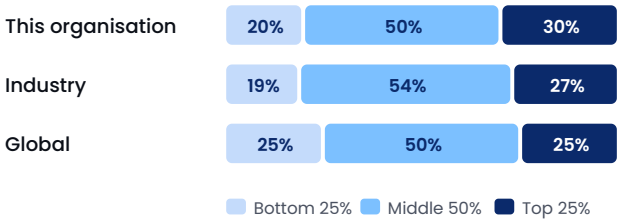
Teams, cost and staffing, on one measure.

WASTE RATE AND CODE QUALITY BY TEAM



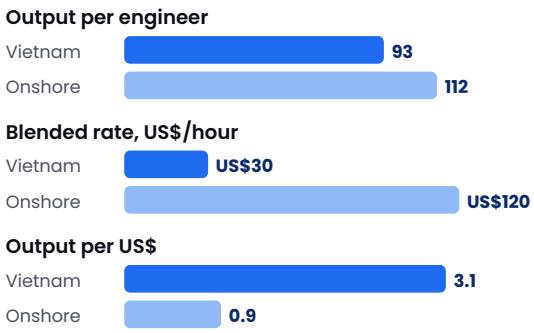
Look upstream before judging people. Team B carries far more waste than Teams A and C with comparable code quality. That points to a process or requirements problem, not to the engineers.

TALENT MIX BY PERFORMANCE QUARTILE



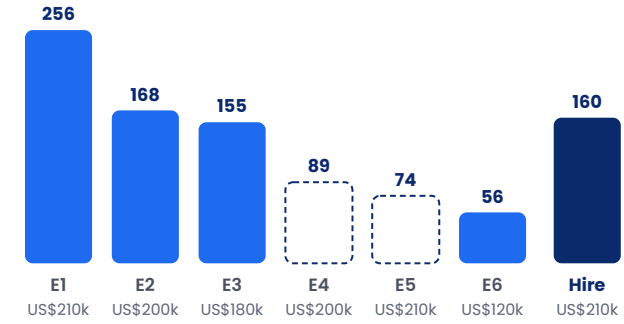
Talent mix. The organisation's spread across the bottom 25%, middle 50% and top 25% of performers, against the industry and the global mix.

VIETNAM AGAINST ONSHORE (US), ILLUSTRATIVE



Output per dollar. Output per engineer is an index against a global median of about 100, and rates are blended hourly rates. In a real read-out, your own cost bands per seniority replace these. It turns "we are cheaper" into evidence you can price on.

YEARLY OUTPUT UNITS, SIX ENGINEERS



Output 798 → 795 · Cost US$1,120k → US$920k

Staffing scenario. Replacing engineers 4 and 5 with one hire from the top 40% keeps output flat (798 to 795 units) and saves about US$200k a year. A scenario model only: people decisions always need the manager's view.

All figures on this page are illustrative, from an anonymised composite built from real engagements and published research. No client is identifiable.

8 How each reader uses it

One measure, used four ways.

The same measure answers different questions depending on who is asking. Engineering leaders in companies, consultancies, system integrators and software vendors each use it for their own decisions.

	Enterprise teams	Consultancies	System integrators	Software vendors
Show what AI returned, net of rework	✓	✓	✓	✓
Cost per unit of output, by tool or supplier	✓	✓	✓	✓
Compare suppliers on one measure	✓	✓	✓	·
Size teams and headcount on output	✓	✓	·	·
Due diligence, deals and takeovers	·	✓	✓	·
Evidence in bids and renewals	·	·	✓	✓
Price on output instead of hours	·	✓	✓	✓
Programme reviews and disputes	·	✓	·	·
Offer it under your own name	·	✓	✓	·

Engineering leaders in companies

Four answers within a month of connecting Git: effective output net of rework, AI work against non-AI work, cost per unit of output by tool and supplier, and your gain against others.

- AI coding tools rolled out, and no way to show what they returned
- AI licences up for renewal: keep, grow or cut on cost per unit of output
- Several suppliers, compared side by side on output, rework and quality
- Next year's headcount, sized on what teams deliver
- A board asking whether the AI spend pays off compared with others

Software vendors and delivery firms

Measure yourself first: one team of about 10 to 20 engineers for about a month, then roll it out, or install it in a client's environment if the client agrees.

- Prove your AI delivery claims from your own code
- Defend your rates with output per dollar, next to onshore alternatives
- Win bids with independent evidence rather than a claim every vendor makes
- Run a tighter delivery engine: rework hotspots fixed upstream
- Move to pricing on output instead of hours

Inside your engagements, or under your name.

Consultancies

Measure the AI transformation you are selling, and put evidence under your next recommendation.

- **AI engineering programmes:** a baseline before the rollout, then uplift by team and by tool, cost per unit for each licence and each repository's AI maturity, with rework as the guardrail
- **Deals:** buy-side and vendor due diligence, carve-outs (who owns which code, and who holds the knowledge) and post-merger integration, with a code assessment when the code itself needs auditing
- **Value creation across a portfolio:** baseline at acquisition, re-measure every quarter
- **Outcome-based contracts and sourcing:** a neutral unit of output both sides can sign
- **Programme reviews:** a fact base read from the history the programme already has
- **Engineering cost reviews:** at team level, not by ranking individuals
- **Disputes over delivered software:** evidence for your experts, not a legal opinion
- **Under your name:** an engineering effectiveness assessment as your own service

A global advisory firm has used it twice: an engineering benchmark of an organisation of several hundred engineers, then a technical due diligence of a software platform.

System integrators

Know your real AI uplift before you price it.

- **Price AI savings on evidence:** know your real uplift, net of rework, before you commit a number in a bid
- **Defend renewals:** output per dollar against the alternatives, before the client runs a tender
- **Due diligence before a takeover:** technical debt, key people and open-source risk in an estate, with a code assessment
- **Your subcontractor panel, on one measure:** we set it up and train your team, and you run the scoring yourselves; we are a subcontractor to integrators too, so we never score your panel
- **Win bids with independent evidence** instead of claims in a proposal
- **Measure the AI programmes you deliver,** team by team and tool by tool
- **Move to output-based pricing** on a shared, neutral unit of output
- **Under your name:** we run the platform and the analysis behind you

In short

Whoever asks, the measure is the same: output scored from the code, set against the same dataset. That is what makes it usable between a buyer and a supplier, and inside a client deliverable.

9 Your code and data stay yours

You decide what is connected, who sees which view, and what leaves the room.

Access is read-only and nothing is written back, in the cloud or on your own infrastructure. No result goes to anyone, a client included, without your written approval.



* Stated by the platform provider, as are its ISO 27001 and SOC 2 Type 2 certification and GDPR compliance. Confirm in your security review.

Three ways to connect

01 Cloud

GitHub, GitLab, Bitbucket, Azure DevOps and others, read-only. Fastest to start.

02 Cloud to your own GitLab

Self-hosted GitLab connects through your VPN.

03 On-premise

For fully air-gapped networks: run it on your own infrastructure.

What it works with

Git platforms: GitHub, GitLab, Bitbucket, Azure DevOps, AWS CodeCommit and Gitea, with self-hosted GitLab through your VPN.

Languages: 44+ languages and frameworks, from .NET, Java and Python to mobile, web front ends, infrastructure and data.

AI tools: usage and spend from Copilot, Cursor, Windsurf, Claude, OpenAI, Amazon Bedrock and others. We confirm each connection for your stack before anything is connected.

What the provider states

No source code is stored and data is deleted after processing. Commercial agreements allow a complete opt-out from model training. ISO 27001 and SOC 2 Type 2 certification and GDPR compliance, with documentation available for your security review.

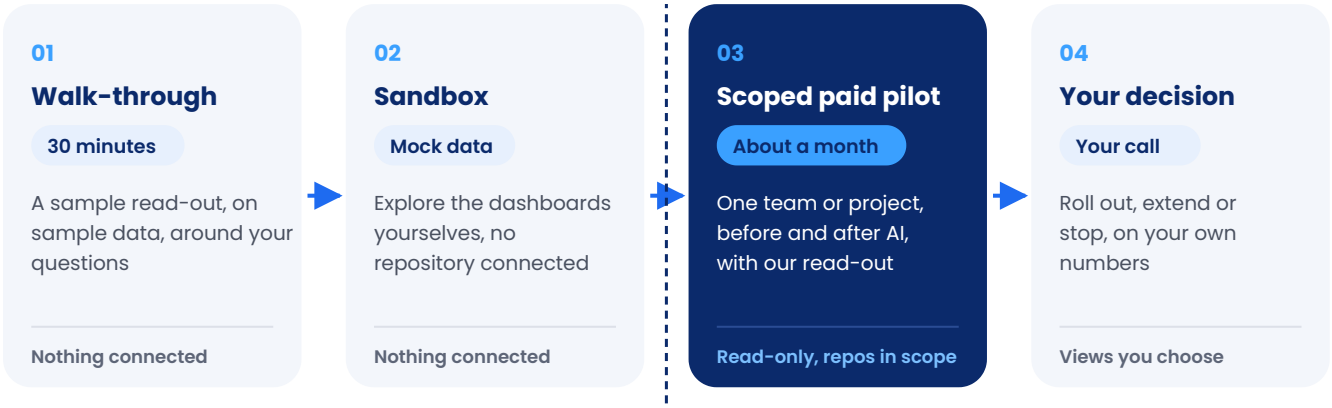
How big you need to be

The benchmark works for organisations with roughly 20 to 30 or more engineers.

10 From interest to a decision

Four steps, each one a chance to stop.

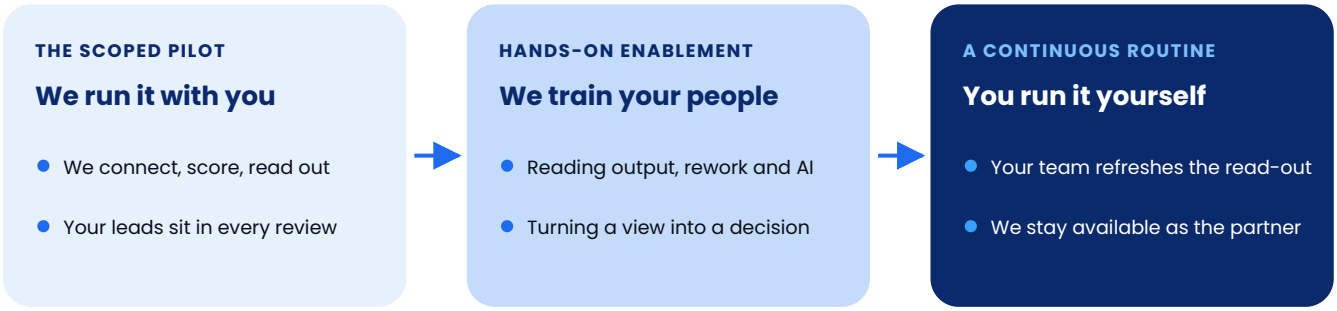
Nothing is connected to your code until the pilot, and the pilot is scoped with you first: the cohort, repositories, history, read-out, timing and terms are agreed before it starts.



Your gate: nothing is connected to your code before the pilot, and the pilot is scoped with you first

Then we train your people, so you can run it yourselves

We do not want you dependent on us. After the pilot, we train the people who will own the read-out, so your team refreshes it on its own cycle and ties each view to a decision.



No lock-in: the capability stays with you after the engagement. Scope and format of the training are agreed per client.

Book a 30-minute walk-through.
On sample data. No repository connected until you decide.

nessoris.com/en/benchmark
jakub@nessoris.com

Sources and notes

Where the figures come from.

A. Denisov-Blanch, "How to Quantify AI ROI in Software Engineering", Stanford Software Engineering Productivity Research Group, November 2025. The 355-engineer organisation (January to August 2025, AI coding tools rolled out in May): pull requests +13.6% (shown as 14%), effective output +1%, rework 2.6 times, code quality -9% with variance up 3.6 times. Also the widening gap between AI teams, token spend and codebase hygiene.

B. Denisov-Blanch et al., "A Few Pages of Markdown: Committed AI Configuration and Lower Quality Cost after Coding-Agent Adoption", 2026, arXiv:2608.25241. Commits after coding agents and cognitive complexity with and without committed AI configuration.

C. Denisov-Blanch et al., arXiv:2409.15152. The scoring model's agreement with expert panels: correlations of 0.82 and 0.86 on two evaluation tasks. That the current model exceeds 0.85 is a statement by the platform provider.

D. Stanford Software Engineering Productivity Research Group, 2025: the typical net gain from AI of 15 to 20% (n = 2,562); self-assessment surveys as a predictor of real output (r = 0.17); the research dataset of 600+ companies and 120,000+ engineers (November 2025).

E. METR, randomised controlled trial of experienced open-source developers using AI tools, 2025: felt about 20% faster, measured 19% slower.

F. Google DORA, 2025: 90% of technology professionals use AI at work.

G. Sample read-out: an anonymised composite built from real engagements and published research. Values are redrawn from report charts and rounded, and are illustrative only. No client is identifiable.

Notes

- The measurement platform is developed by an independent software company. Its methodology has been studied in academic research at Stanford; references to Stanford describe that research, not ownership or endorsement of the platform.
- Certifications, data deletion and the opt-out from model training are stated by the platform provider. Confirm them in your security review.
- The "four equivalent pieces of work" table is an illustrative example. Before-and-after views describe one organisation; they are not proof that AI caused the change.
- Nothing in this paper is a promise of a result for your organisation. The read-out shows what your own code says.



About Nessoris. An accountable delivery partner for software and AI delivery, measured on the value it realises, not the hours it bills. We work behind consulting firms and integrators, under their name, and alongside enterprise teams. The engineering benchmark is one of our assessments: it measures what engineering delivers, from the code itself.

Did AI raise your engineering output?

We answer it from your code, benchmarked against 120,000+ engineers.

Book a 30-minute walk-through at **nessoris.com/en/benchmark**

On sample data. No repository connected until you decide. Or write to jakub@nessoris.com.

Delivery that compounds.