Leadership in AI Assisted Engineering

LeadDev

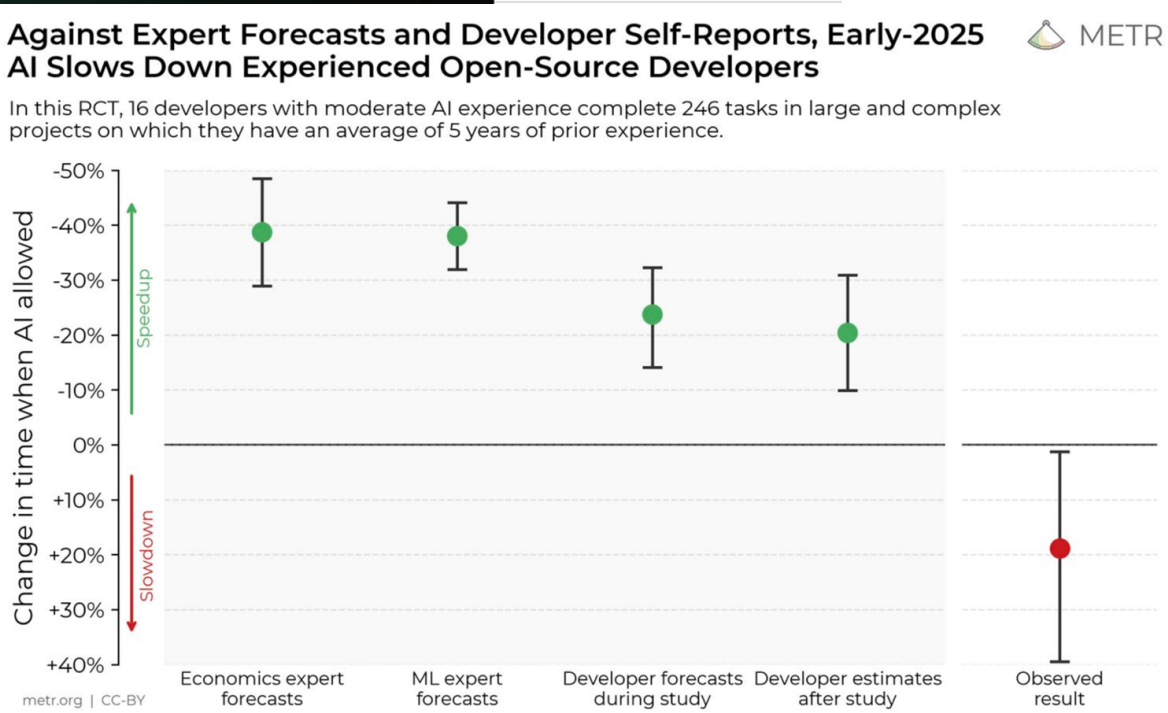
Agenda

- The impact of GenAI
- Challenges with AI Adoption
- Reducing fear of AI
- Measurement
- Employee Success
- Compliance and Trust
- Unblocking Usage
- SDLC Agent Use Cases

TECH

Sundar Pichai says AI is making Google engineers 10% more productive. Here's how it measures that.

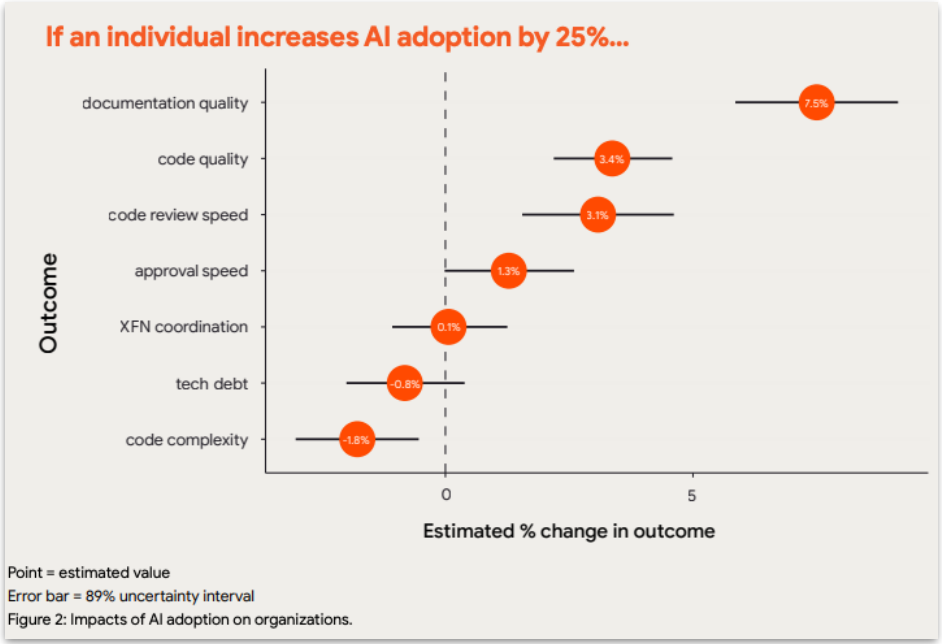
By **Hugh Langley**



GenAI is significantly impacting development

A 25% increase in AI adoption is associated with a...

- 7.5% increase in documentation quality
- 3.4% increase in code quality
- 3.1% increase in code review speed
- 1.3% increase in approval speed
- 1.8% decrease in code complexity



Point = estimated value
Error bar = 89% uncertainty interval
Figure 2: Impacts of AI adoption on organizations.

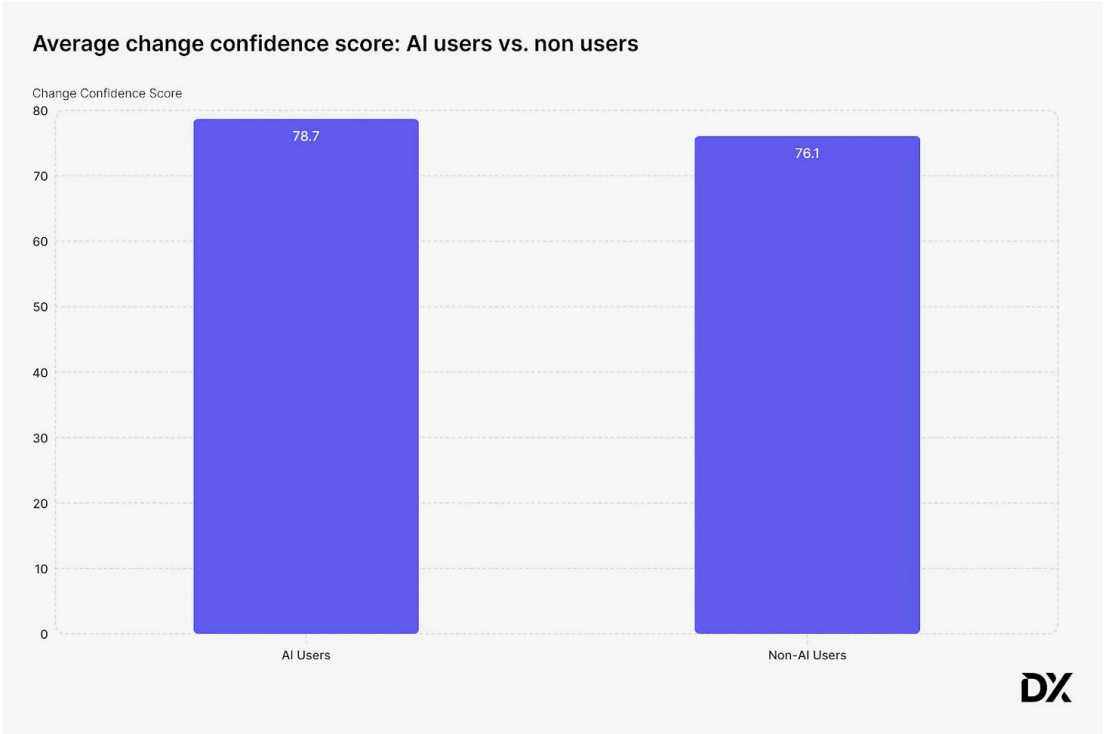
DORA Impact of GenAI in Software Development: https://dora.dev/research/ai/gen-ai-report/

AI seems to deliver modest gains in quality metrics

Average Change Confidence DXI driver for AI Users vs. Non-AI Users

Data from 19,251 developers

DXI Score



18 pt max gain

2.6 Point Average Gain from using AI

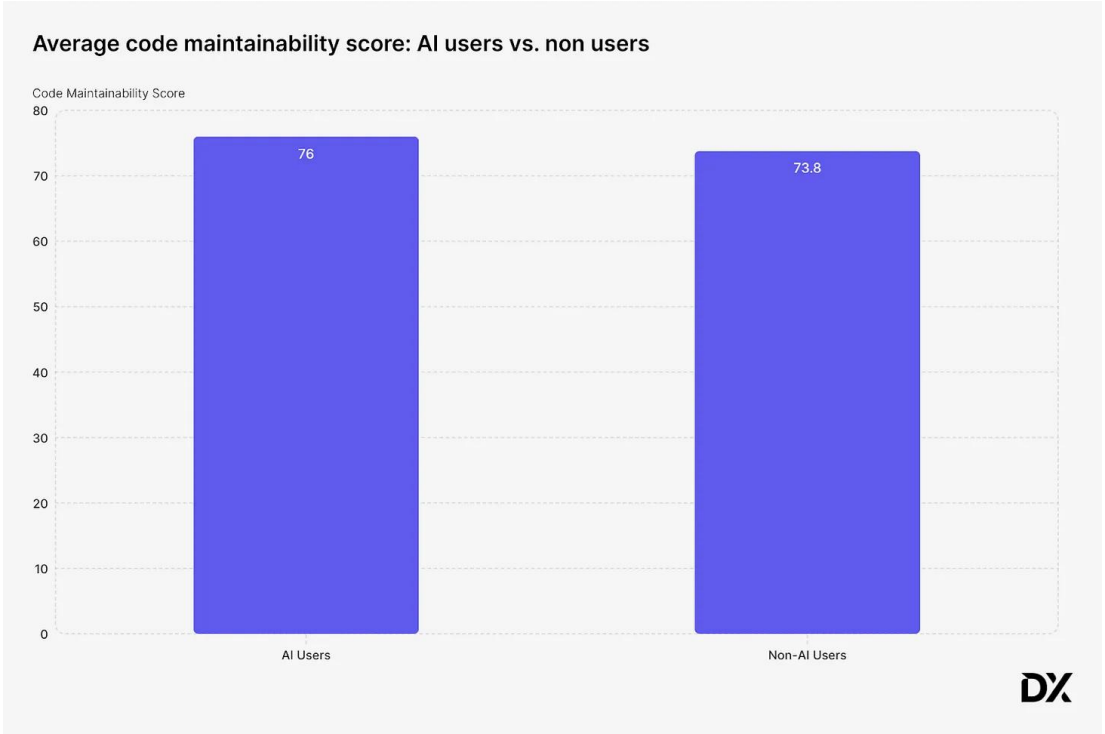
15 pt max loss

AI seems to deliver modest gains in quality metrics

Average Code Maintainability DXI driver for AI Users vs. Non-AI Users

Data from 20,012 developers

DXI Score



15 pt max gain

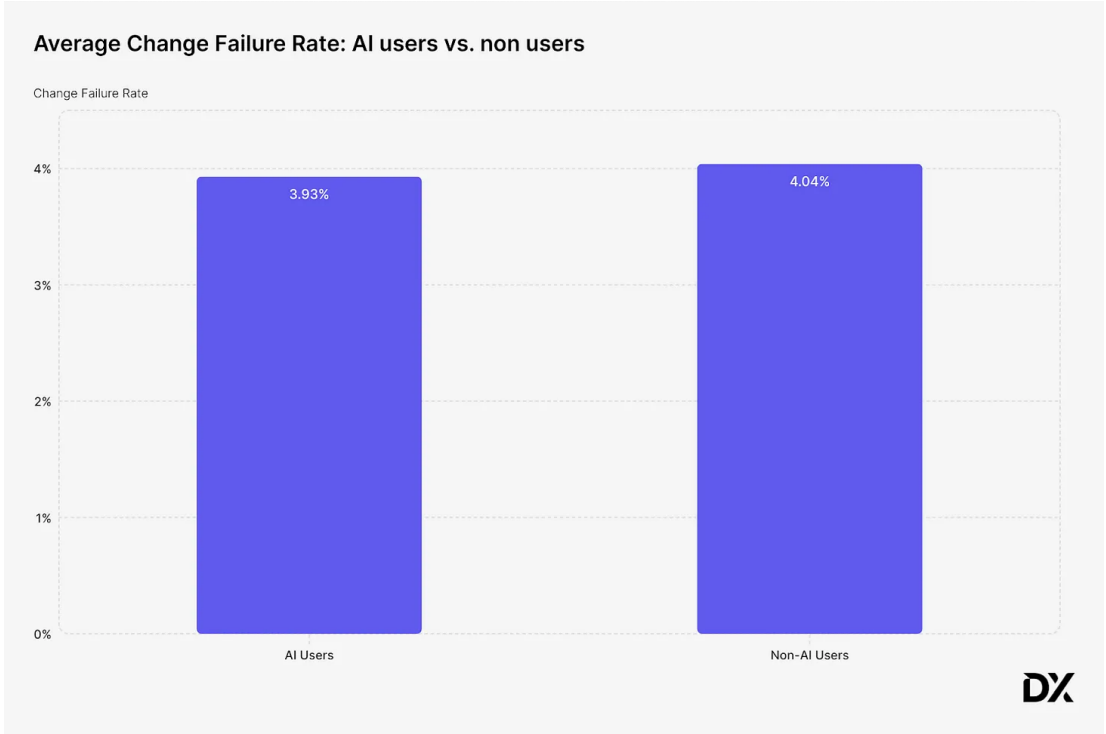
2.2 Point Average Gain from using AI

13 pt max loss

AI seems to deliver modest gains in quality metrics

Average Change Failure Rate% for AI Users vs. Non-AI Users

Data from 61 Companies



1.77% max gain

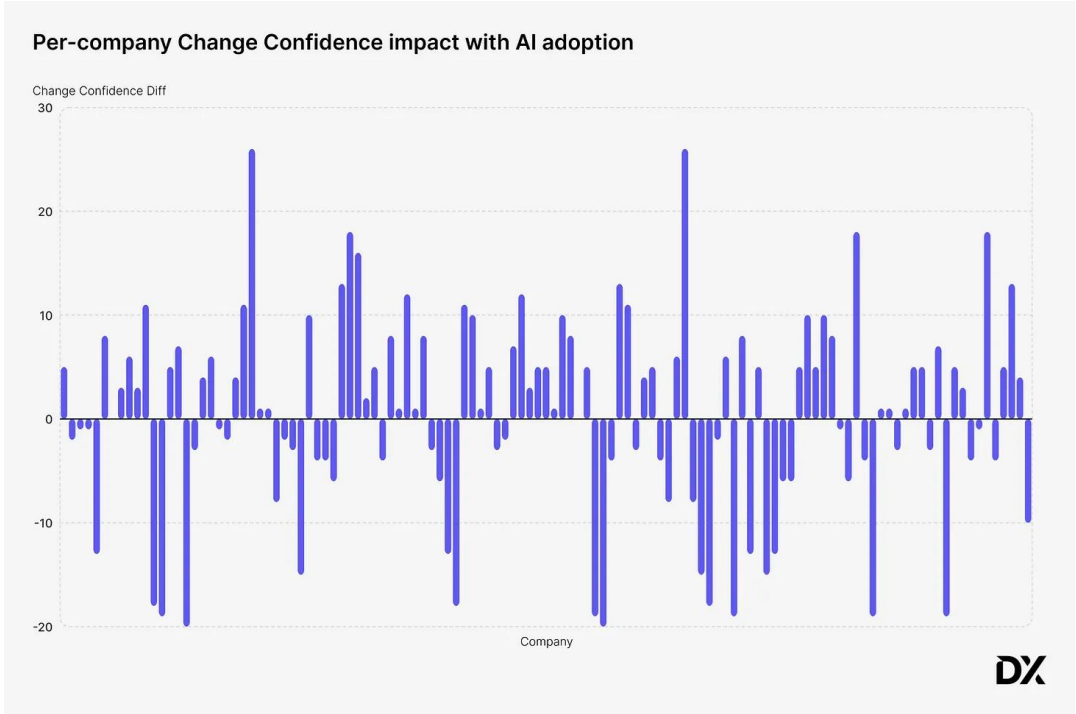
0.11% Reduction from using AI

1.07% max loss

Industry averages hide the big picture of quality impact

Per-company Change Confidence DXI impact for AI Users vs. Non-AI Users

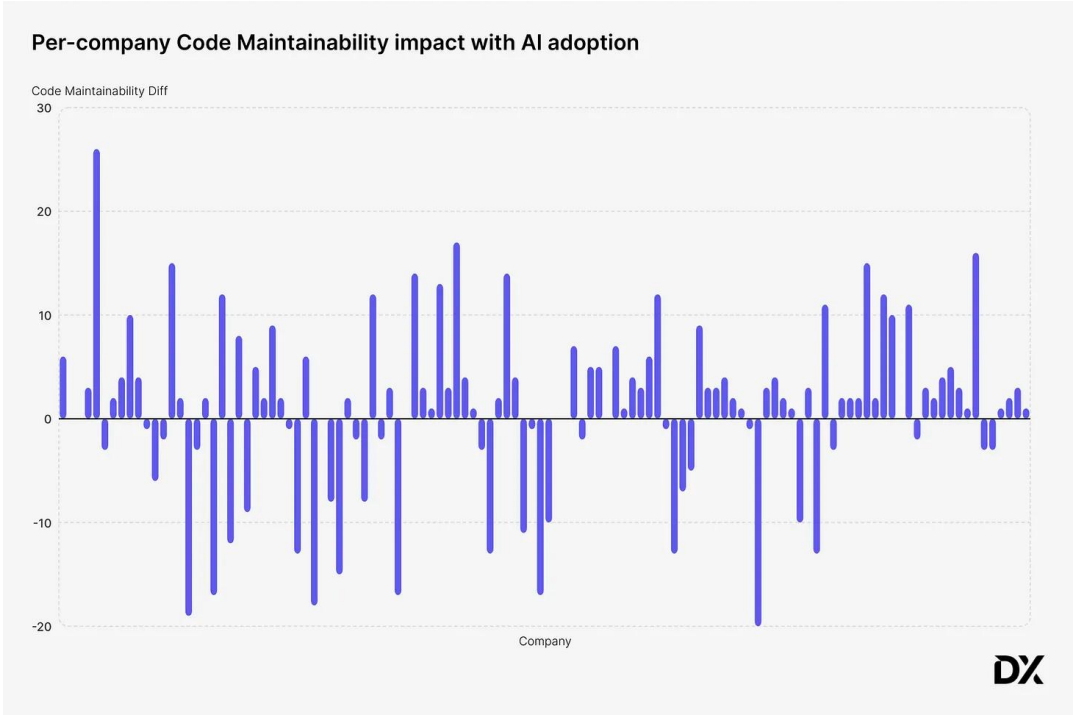
Data from 19,251 developers



Industry averages hide the big picture of quality impact

Per-company Code Maintainability DXI impact for AI Users vs. Non-AI Users

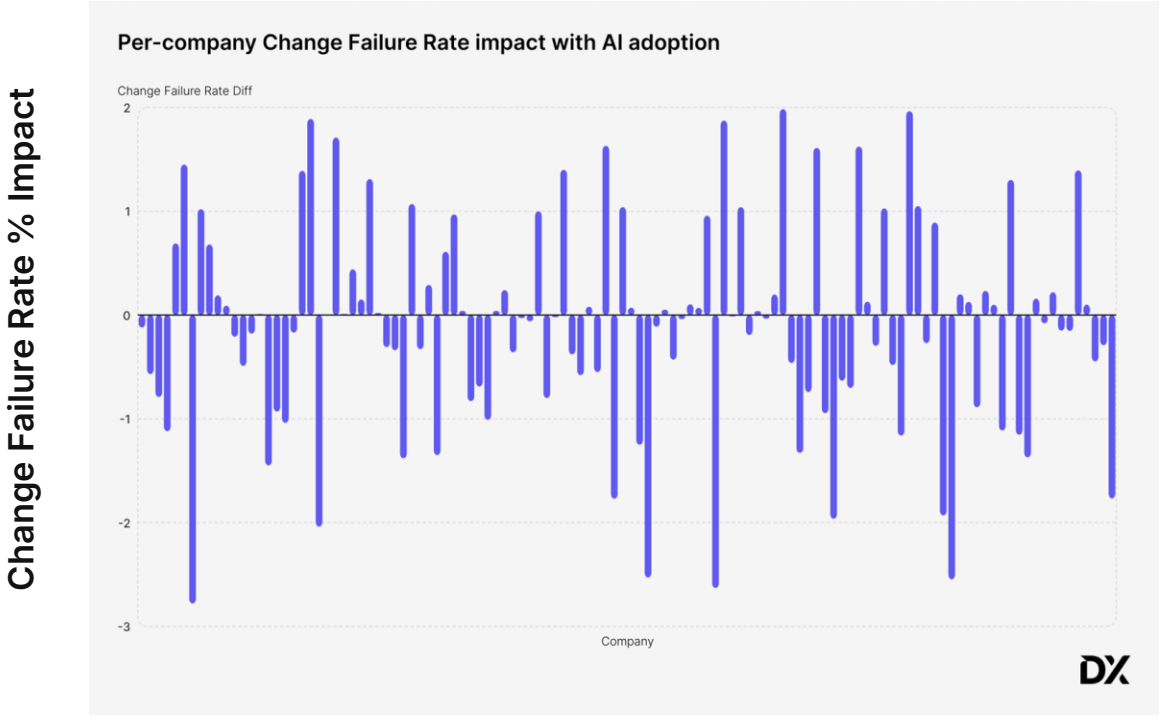
Data from 20,012 developers



Industry averages hide the big picture of quality impact

Per-company Change Failure Rate% impact for AI Users vs. Non-AI Users

Data from 61 companies



GenAI ROI not evenly distributed

Many organizations are **seeing positive impacts to KPIs**

Others are **struggling with adoption and even seeing negative impacts**

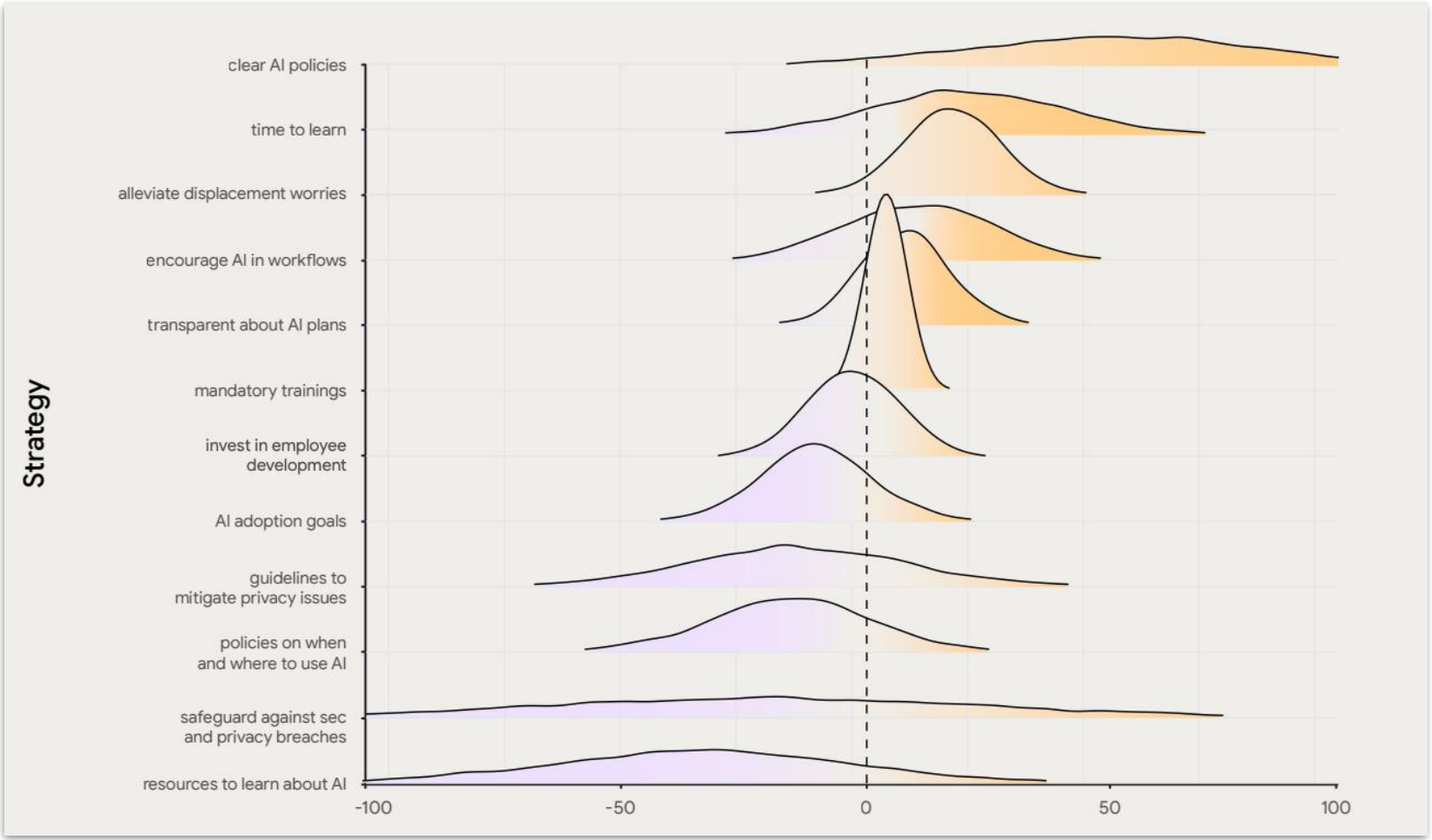
Top down **mandates aren't working**, and are decreasing psychological safety

GenAI ROI not evenly distributed

Lack of education and enablement on best practices and use cases.

Organizations often just “turn on” the tech and expect users to be proficient.

Difficulty measuring the impact and knowing what to measure.



Bayesian Posterior Distributions of AI Adoption Stategies: https://dora.dev/research/ai/gen-ai-report/

Top Distribution Analysis

High Likelihood (80-95%)

Clear AI policies: ~90% - Very strong positive distribution with most mass well to the right

Time to learn: ~85% - Strong positive distribution with clear rightward skew

Alleviate displacement worries: ~80% - Solid positive distribution, though with some uncertainty

Moderate-High Likelihood (60-75%)

Encourage AI in workflows: ~70% - Positive distribution but with more spread

Transparent about AI plans: ~65% - Moderately positive with reasonable uncertainty

Mandatory trainings: ~65% - Similar positive trend but wider distribution

As leaders, we should drive...

Integration across SDLC

The biggest payoffs are rewarded to companies that think beyond code generation, and integrate across multiple areas of the SDLC, including always-on code reviews, automated incident management, refactor, and documentation

Open metrics discussions

Measure GenAI impact using tools such as DX's GenAI impact reporting. Advertise and evangelize these metrics to teams. Showcase teams with particularly good adoption and velocity improvements to drive healthy competition.

Compliance and trust

Validate AI-generated code through human oversight and verification. Ensure proper testing gates exist to limit change failures. If your culture already embraces test-driven design, then continue that practice and continue to run your battery of linting and testing against output code.

Unblocking usage

Make sure that there are no impediments to using code assistants for the use cases called out in this document. Proactively seek ways to limit barriers to adoption, including running models on-premise and training locally on code repositories.

Reducing fear of AI

Frame AI adoption as a force multiplier for performance, unlocking organizational capabilities. Remind engineers that these tools are meant to augment capabilities and transcend what was possible before, not replace jobs.

Employee success

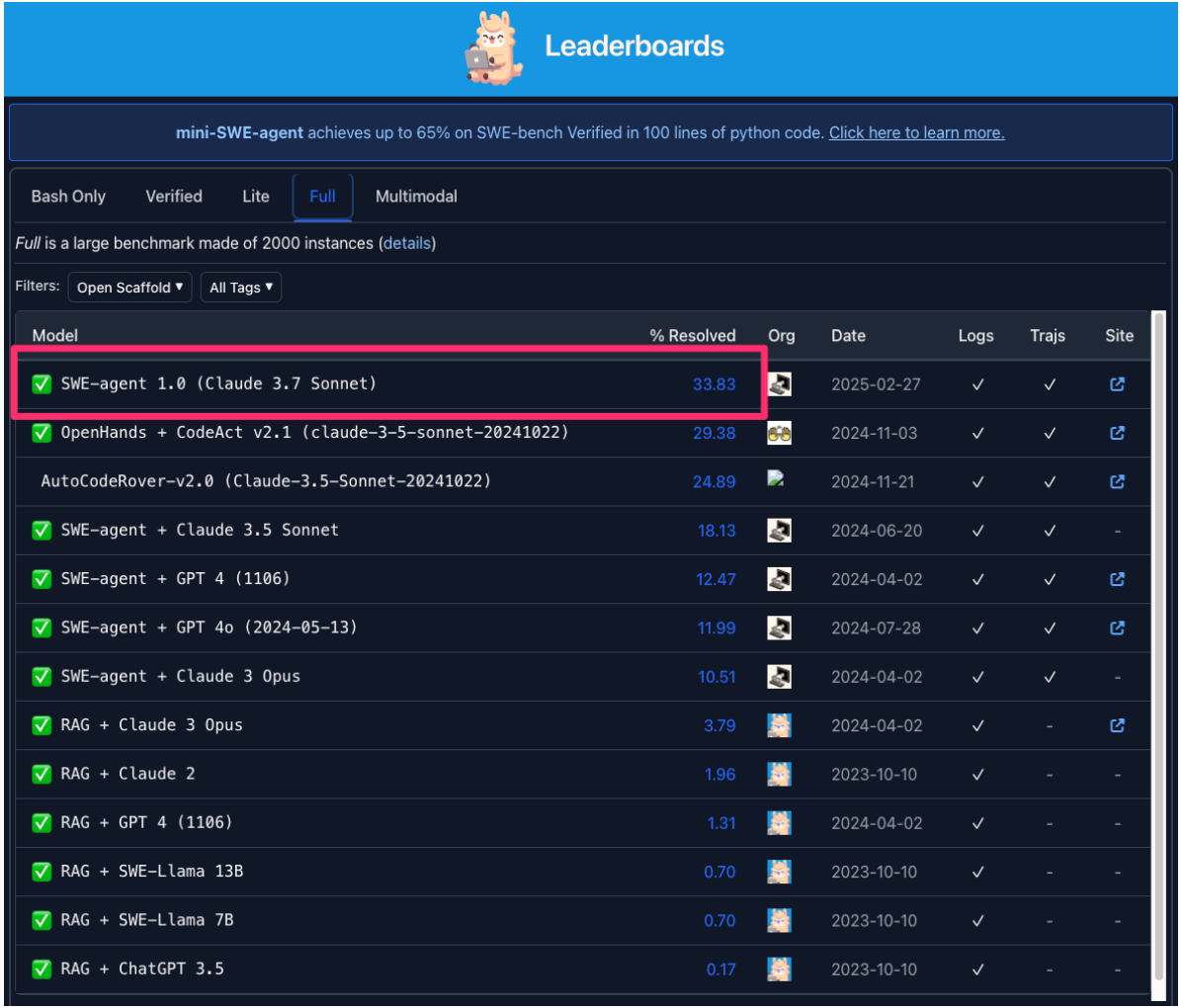
Developers who leverage AI will outperform those who resist adoption. Remind engineers that this is an opportunity to learn about techniques that are likely to benefit them for the remainder of their careers.

Reducing fear of AI

Frame AI adoption as a **force multiplier for performance**, unlocking organizational capabilities. Remind engineers that these tools are meant to augment capabilities and transcend what was possible before, not replace jobs.

Google’s Project Aristotle

- Research project undertaken by Google to understand what increases performance and makes teams successful.
- “The whole is greater than the sum of its parts”
- Assumed that the recipe for a successful team would be a combination of high performers, an experienced manager and unlimited free resources.
- They were wrong. The high performing teams were the ones with greatest psychological safety.


Leaderboards
mini-SWE-agent achieves up to 65% on SWE-bench Verified in 100 lines of python code. Click here to learn more.
Bash Only
Verified
Lite
Full
Multimodal
Full is a large benchmark made of 2000 instances (details)
Filters:
Open Scaffold
All Tags
Model
% Resolved
Org
Date
Logs
Trajs
Site
SWE-agent 1.0 (Claude 3.7 Sonnet) 33.83 2025-02-27 ✓ ✓
OpenHands + CodeAct v2.1 (claude-3-5-sonnet-20241022) 29.38 2024-11-03 ✓ ✓
AutoCodeRover-v2.0 (Claude-3.5-Sonnet-20241022) 24.89 2024-11-21 ✓ ✓
SWE-agent + Claude 3.5 Sonnet 18.13 2024-06-20 ✓ ✓ -
SWE-agent + GPT 4 (1106) 12.47 2024-04-02 ✓ ✓
SWE-agent + GPT 4o (2024-05-13) 11.99 2024-07-28 ✓ ✓
SWE-agent + Claude 3 Opus 10.51 2024-04-02 ✓ ✓ -
RAG + Claude 3 Opus 3.79 2024-04-02 ✓ -
RAG + Claude 2 1.96 2023-10-10 ✓ - -
RAG + GPT 4 (1106) 1.31 2024-04-02 ✓ - -
RAG + SWE-Llama 13B 0.70 2023-10-10 ✓ - -
RAG + SWE-Llama 7B 0.70 2023-10-10 ✓ - -
RAG + ChatGPT 3.5 0.17 2023-10-10 ✓ - -

Transparency – Set Clear Intent and Expectations

Explain the “why”: Frame GenAI adoption as a way to augment, not replace, engineers’ skills. Show data and case studies illustrating benefits and pitfalls.

Address fears directly: Proactively acknowledge concerns about job security, performance monitoring, or AI errors, clarify boundaries on how AI outputs will be evaluated and used.

Open metrics discussion: If you’ll measure AI usage or impact, share exactly what’s being tracked and why, avoiding “black box” surveillance.

Open metrics discussions

Measure GenAI impact using tools such as DX's GenAI impact reporting. Advertise and evangelize these metrics to teams. Showcase teams with particularly good adoption and velocity improvements. Make people feel comfortable about what's being tracked.

Balance short-term speed gains with longer-term maintainability and quality

Speed	Quality and Maintainability
● PR throughput ● Time savings per developer ● PR cycle time	● Change failure rate ● Perception of quality ● Change confidence ● Maintainability ● % of time allocated to bugs

Measuring GenAI impact

Metric Type	Good For	Not Good For	Challenges
Telemetry metrics	Measuring impact on developer output	Quantifying ROI Understanding how tools are being used	Limited, possibly inaccurate insight Incomplete story
Experience sampling	Quantifying ROI Identifying best use cases	Collecting large amounts of data at once	Difficult to set up Must be run over period of time
Self-reported	Measuring adoption, developer satisfaction, productivity	Quantifying ROI	Can only be run periodically Participation rates

Collect workflow and self-reported data with a mixed-methods approach

System data: Admin APIs to track usage, spending, token consumption, and code suggestion acceptance. System metrics from your development stack.

Periodic surveys: Quarterly or regular surveys capture trends in developer experience that system data alone can miss. Measure perceptions like developer satisfaction, confidence in changes, or how maintainable code feels.

Experience Sampling: Gather targeted, in-the-moment feedback by asking brief questions during key workflows. Eg: after submitting a pull request, ask if AI was used to write the code or whether AI-generated code felt easier or harder to understand.

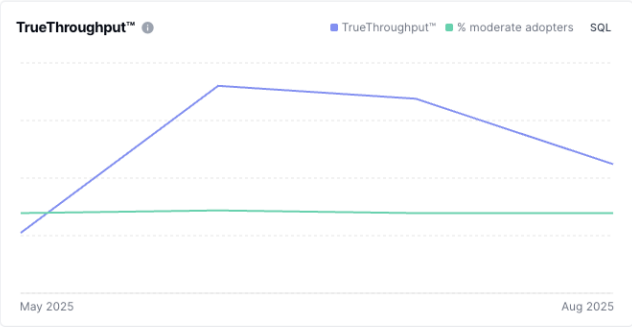
Table 1: DX AI Measurement Framework

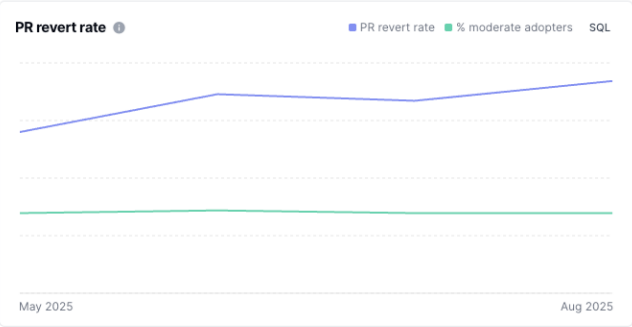
* Metrics for autonomous AI agents

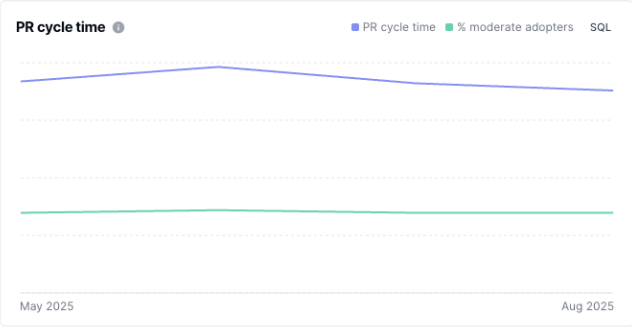
Utilization How much are developers adopting and utilizing AI tools?	**Impact** How is AI impacting engineering productivity?	**Cost** Is our AI spend and return on investment optimal?

• AI tool usage (DAUs/WAUs)
• Percentage of PRs that are AI-assisted
• Percentage of committed code that is AI-generated
• Tasks assigned to agents *
	• AI-driven time savings (dev hours/week)
	• Developer satisfaction
	• DX Core 4 metrics, including:
	  • PR throughput
	  • Perceived rate of delivery
	  • Developer Experience Index (DXI)
	  • Code maintainability
	  • Change confidence
	  • Change fail percentage
	• Human-equivalent hours (HEH) of work completed by agents *
		• AI spend (both total and per developer)
		• Net time gain per developer (time savings - AI spend)
		• Agent hourly rate (HEH / AI spend) *

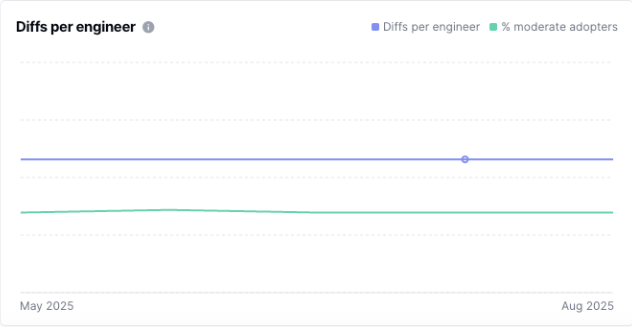
Foundational DevEx and DevProd metrics still matter the most.

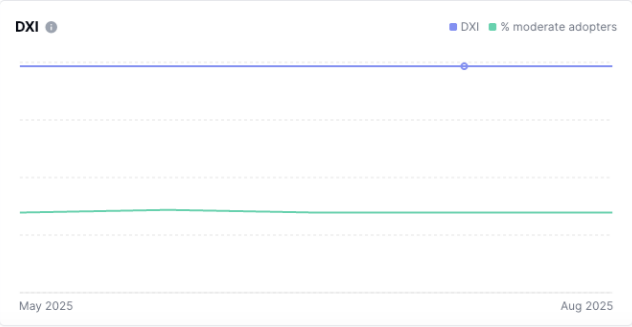

TrueThroughput™
TrueThroughput™
% moderate adopters
SQL
May 2025
Aug 2025


PR revert rate
PR revert rate
% moderate adopters
SQL
May 2025
Aug 2025


PR cycle time
PR cycle time
% moderate adopters
SQL
May 2025
Aug 2025


% feature development
% feature development
% moderate adopters
SQL
May 2025
Aug 2025


Diffs per engineer
Diffs per engineer
% moderate adopters
May 2025
Aug 2025


DXI
DXI
% moderate adopters
May 2025
Aug 2025

What top companies are currently measuring

Microsoft	Dropbox	Booking.com
Microsoft measures the impact of AI with their Engineering Thrive program, using: ● Adoption/usage of AI tools and agentic workflows ● System Velocity ● Developer Satisfaction ● Change Failure Rate ● Bad Developer Days (a telemetry-based measure of toil and disruption)	Metrics based around DX Core 4 and DX AI Measurement Framework: ● Adoption/Engagement - WAUs and DAUs ● Developer sentiment - CSAT (“Using AI tools has improved my overall productivity as a software developer.”) ● Velocity - PR throughput at company level ● Engineering hours saved - User-reported time savings and AI lines added (proxy) ● % AI code - AI lines added / Total lines added ● Quality - Change fail percentage (self-reported)	Metrics around utilization, velocity, quality, and sentiment: ● DAU/WAU ● Time saved per dev ● % of PRs that are AI-assisted ● PR Throughput ● Developer CSAT ● Change failure rate

Compliance and trust

Validate AI-generated code through human oversight and verification. Ensure proper testing gates exist to limit change failures. If your culture already embraces test-driven design, then continue that practice and continue to run your battery of linting and testing against output code.

System prompt update loop

Most enterprise AI solutions will allow you to update the “system prompt” that underlies every prompt sent to the assistant. This is almost like templating, in that the rules contained in this prompt will be applied to every prompt sent into the assistant.

When the model creates inaccurate or suboptimal output, in many cases that output can be corrected going forward by changing the system prompt.

Example usage

[Include Simple System Prompt]

You have been observed making the following errors:

1. Proposing outdated Spring Boot versions (older than 2.6).
2. Suggesting or using deprecated methods in the java.* libraries.
3. Returning code snippets containing syntax errors such as missing parentheses or braces.

Your new rules:
- Always provide code snippets that use Spring Boot 3.x or newer, unless the user explicitly requests otherwise.
- Verify that you are not using deprecated methods from the Java standard library (check the current Java LTS for deprecation).
- Double-check that any code snippets are syntactically valid (balanced parentheses, braces, etc.).

When responding:
- Provide relevant explanations for your code choices.
- If uncertain, indicate possible methods or approaches rather than returning an incorrect snippet.
- Do not include references to internal or proprietary APIs beyond standard library or Spring Boot dependencies.

Determinism and non-determinism

Generative AI models operate on the principle of **token-based probability distributions**, where each generated token (i.e. a word, sub-word, or even a single character) is influenced by prior context and model parameters.

Determinism can be **controlled using temperature**, a system parameter that adjusts the randomness of token selection.

A **lower temperature (i.e., “0.1” as opposed to “0.9”) makes the model more deterministic** by favoring high-probability tokens, leading to consistent and repeatable outputs. Conversely, a **higher temperature (i.e., “0.9” or above) increases randomness.**

Example usage

When we set temperature to a very low value, say “0.0001” and set a prompt like:

Create a Javascript method to render a gradient of colors from blue to red.

... there is little to no variation in the output. **In this case, there’s no difference at all:**

Example 1

```
<!DOCTYPE html>
<html>
<head>
  <title>Blue to Red Gradient</title>
  <style>
    canvas {
      display: block;
    }
  </style>
</head>
<body>
  <canvas id="gradientCanvas"></canvas>
```

Example 2

```
<!DOCTYPE html>
<html>
<head>
  <title>Blue to Red Gradient</title>
  <style>
    canvas {
      display: block;
    }
  </style>
</head>
<body>
  <canvas id="gradientCanvas"></canvas>
```

Determinism and non-determinism

However, look at the extreme difference in approaches when we set the temperature to 0.9 and use the same prompt:

> Create a Javascript method to render a gradient of colors from blue to red.

Output A

In the first attempt, the model creates an HTML document with embedded code that uses CSS <style> tags to manage drawing the gradient.

```
<!DOCTYPE html>
<html>
<head>
   <style>
       .gradient-container {
           width: 600px;
           height: 100px;
           padding: 20px;
           position: relative;
           background: repeating-linear-gradient(
               to right,
               var(--from-blue),
               var(--to-red)
           );
           transition: background linear-gradient(95%);
```

Output B

But look what happens when we execute the exact same prompt a second time. The approach is entirely different, creating a Javascript function without any HTML, and using the HTML5 canvas convention to draw the gradient!

```
function renderGradient() {
   // Create a new blank canvas element with size 400x400
   const canvas = document.createElement('canvas');
   canvas.width = 400;
   canvas.height = 400;

   // Create an image element that will hold the gradient
   const img = document.createElement('img');
   img.src = '';
   img.style.width = canvas.width;
   img.style.height = canvas.height;
```

Guardrails – Protect Quality and Trust

Clear validation steps: Require human review of AI outputs in critical areas to ensure code quality and compliance.

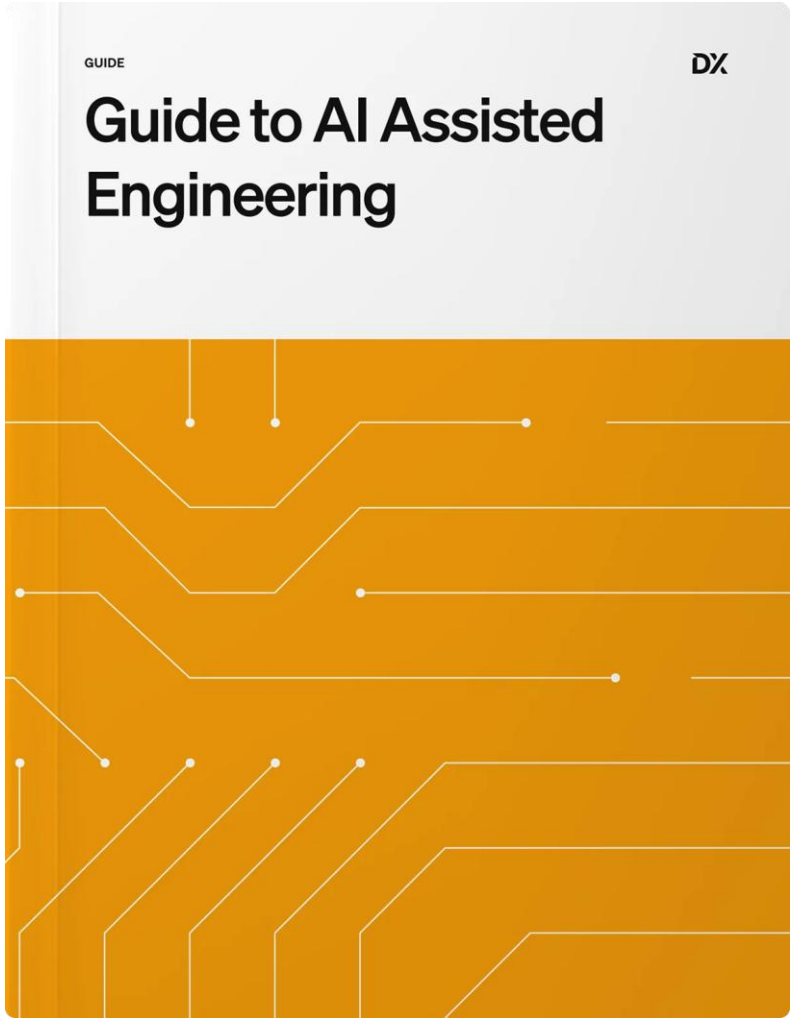
Bias and security training: Teach engineers how to spot hallucinations, data leaks, or biased outputs in AI-generated code or documentation.

Feedback loops: Maintain open channels (Slack channels, retros, office hours) for reporting AI-related issues without fear of blame.

Employee Success

Developers who leverage AI will outperform those who resist adoption. Remind engineers that this is an opportunity to learn about techniques that are likely to benefit them for the remainder of their careers.

Provide both education and adequate time to learn.


GUIDE
DX
Guide to AI Assisted Engineering

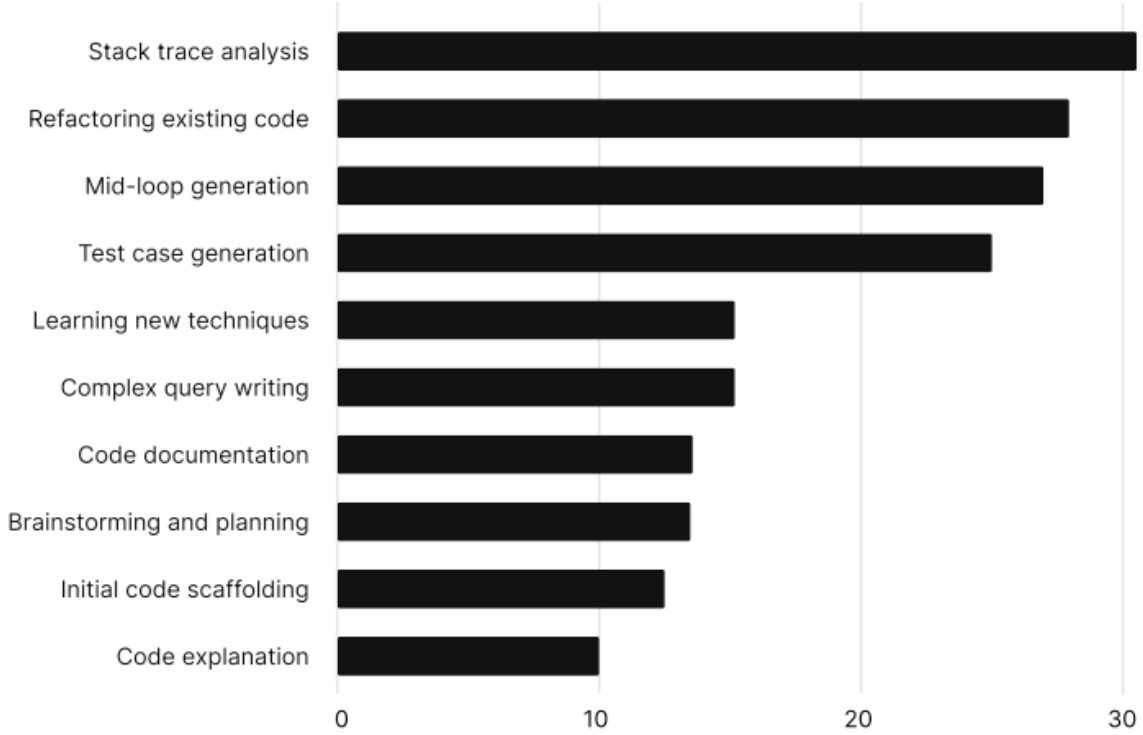
DX GenAI study overview

Purpose: Discover the highest value prompting practices and use cases

Method: Interview S-level+ leaders who have successfully rolled out coding assistants, survey developers who are reporting 1+ hr week savings

Discovery: LinkedIn Polls to target participants, 1:1 interviews with leaders, broad survey of developers stack ranking most valuable use cases

Study outcome



Unblock usage

Make sure that there are **no impediments** to using code assistants and agents. Proactively seek ways to **limit barriers to adoption**, including running models on-premise and training locally on code repositories.

AI Value Starts Where Engineers Can Apply It Safely

Identify & Prioritize High-Impact Workflows: Focus on areas like code reviews, documentation, incident analysis, and architecture exploration where GenAI can deliver measurable gains.

Remove Bottlenecks to Experimentation: Give engineers access to safe, compliant sandboxes so they can test and refine AI-assisted processes.

Champion Innovation Culture: Model curiosity and support “safe-to-fail” pilots to encourage creative exploration without risk aversion.

Security & Compliance Should Enable, Not Block

Leverage Self-Hosted & Private Models: Keep sensitive IP and customer data inside your trust boundary while enabling advanced GenAI capabilities.

Partner with Compliance Early: Co-design workflows that satisfy regulatory obligations without neutering innovation.

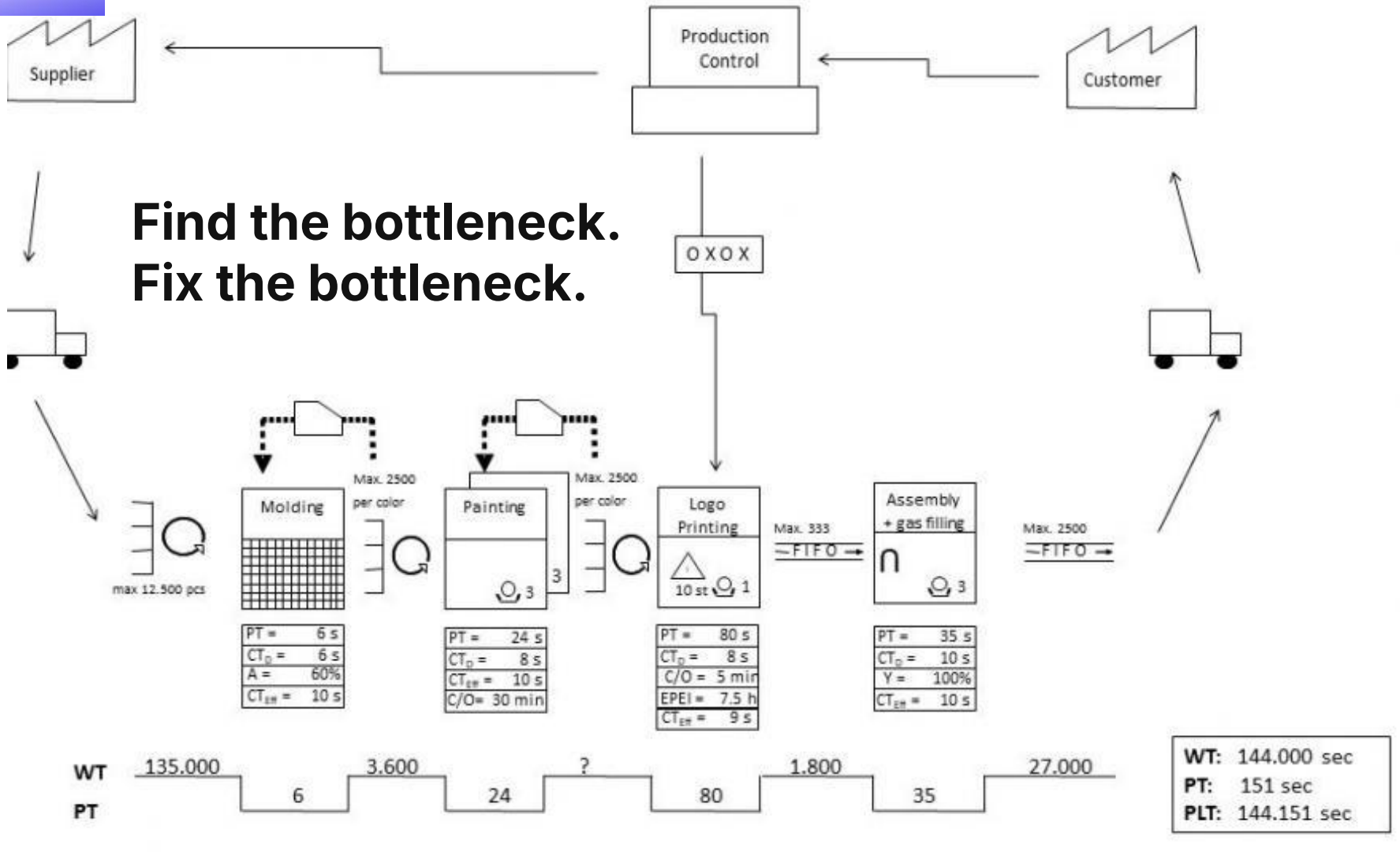
Think Creatively Around Barriers: Use synthetic datasets, anonymization, and prompt engineering to sidestep privacy or regulatory blockers.

Integrate across SDLC

The biggest payoffs are awarded to companies that think beyond code generation, and integrate across multiple areas of the SDLC, including always-on code reviews, automated incident management, refactor, and documentation

SDLC Agent use cases

“An hour saved on something that isn’t the bottleneck is worthless” - Eli Goldratt


Supplier
Production Control
Customer
Find the bottleneck.
Fix the bottleneck.
OXOX
max 12.500 pcs
Molding
Max. 2500 per color
PT = 6 s
CT = 6 s
A = 60%
CT = 10 s
Painting
3
Max. 2500 per color
PT = 24 s
CT = 8 s
CT = 10 s
C/O = 30 min
Logo Printing
10 st
1
PT = 80 s
CT = 8 s
C/O = 5 min
EPEI = 7.5 h
CT = 9 s
Max. 333
FIFO
Assembly + gas filling
3
PT = 35 s
CT = 10 s
Y = 100%
CT = 10 s
Max. 2500
FIFO
WT
PT
135.000
6
3.600
24
?
80
1.800
35
27.000
WT: 144.000 sec
PT: 151 sec
PLT: 144.151 sec

Morgan Stanley - DevGen.AI

Morgan Stanley built a legacy code refactor solution

Agents read legacy code and create developer specs

Saves over 280,000 hours annually by eliminating reverse engineering

Faire - Automated Code Review

“Fairey” triggers off of GitHub PR and provides always-on review

Solution pulls context, surrounding code, and documentation, and provides PR comments

Completes roughly 3,000 reviews a week

Canva - PRD Generation

Internal PRD generator develops epics, stories, and even mockups from prompts from PMs

MCP servers expose context and documentation, and connects to Jira and Figma

Significantly streamlines PRD process, generates developer-friendly specs

Spotify - Incident Management

Internal incident management agent platform drafts remediation steps directly into SRE channels

Monitors logs, correlates alerts, and suggests runbook steps

Currently handling 90% of incidents at Spotify

Next Steps

Distribute the AI guide as a reference for integrating AI into your development workflows

Determine a method for measuring and evaluating GenAI impact

Track and measure AI adoption and iterate on best practices and use cases

LIVE WEBINAR

Running data-driven evaluations of AI engineering tools

Nov 13, 9:00am PT / 6:00pm CET

AI tools are one of the biggest bets engineering leaders are making, but choosing the right ones requires more than a trial run. A successful POC depends on clear goals, structured evaluation, and the right success metrics. Join DX CEO Abi Noda and CTO Laura Tacho as they share how to run AI tool evaluations that deliver actionable insights and lasting impact.

In this session, you'll learn:

- How to choose the right AI tools to evaluate
- Which developers should be included in the evaluation
- What to measure in order to accurately evaluate tool performance

Speakers

Abi Noda
CEO, DX

Laura Tacho
CTO, DX

Register Here!

Q+A