DEVDAY.org
Inspiring
IT Movement

AXON ACTIVE
Focusing on your agility

The Last Junior Developer

The Ticking Time Bomb in AI-Powered Teams

Sebastian Sussmann
CIO, Axon Active Vietnam Co., Ltd.

AXON ACTIVE
Focusing on your agility

Think of the best developer on your team.

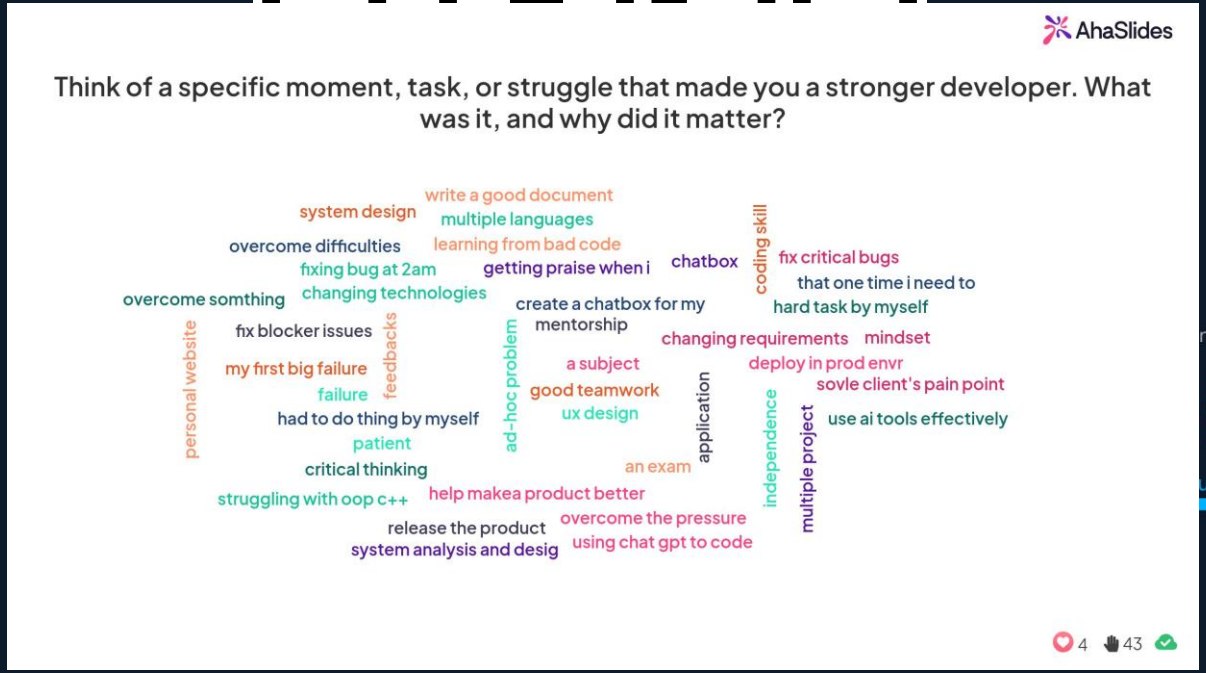
How did they get good?

> ***None of it involved AI.***

AXON ACTIVE
Focusing on your agility

https://ahaslides.com/EAYHV

Think of a specific moment, task, or struggle that made you a stronger developer. What was it, and why did it matter?



remaining

00

ur answer now

AXON ACTIVE
Focusing on your agility

The industry is celebrating.

15%→7%

New-graduate share of Big Tech hires — more than halved since 2022

SignalFire State of Talent Report 2025

85%

Of developers regularly use AI tools for coding and development

JetBrains Developer Ecosystem Survey 2025 · 24,534 developers

15–50%

Output gain observed in AI-augmented teams

Axon Active pilot · 14 teams, 93 devs

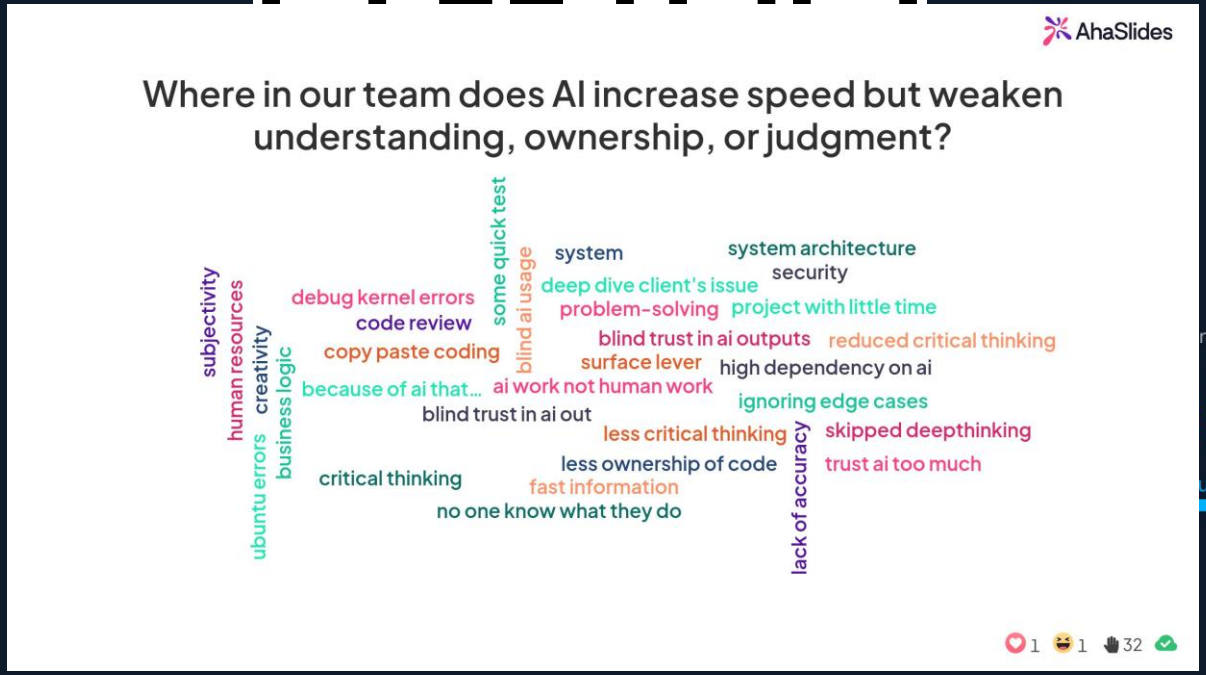
Companies are calling this transformation.

Nobody is asking the next question.

AXON ACTIVE
Focusing on your agility

https://ahaslides.com/BCSAF

Where in our team does AI increase speed but weaken understanding, ownership, or judgment?



remaining

00

ur answer now

AXON ACTIVE
Focusing on your agility

The industry is celebrating.

15%→7%

New-graduate share of Big Tech hires — more than halved since 2022

SignalFire State of Talent Report 2025

85%

Of developers regularly use AI tools for coding and development

JetBrains Developer Ecosystem Survey 2025 · 24,534 developers

15–50%

Output gain observed in AI-augmented teams

Axon Active pilot · 14 teams, 93 devs

Companies are calling this transformation.

Nobody is asking the next question.

But nobody is asking

the devastating question:

When the juniors disappear —
where do the next seniors come from?

The expertise pipeline isn't evolving.

It's collapsing.

AXON ACTIVE
Focusing on your agility

How expertise actually forms

Junior

Year 1–3

Writes code
Breaks things
Learns why
Learns the domain

Mid

Year 3–6

Owns features
Mentors others
Reviews code
Grasps business context

Senior

Year 6+

Architects systems
Judges AI output
Guards quality
Drives business decisions

This is not a career ladder. It is a knowledge forge.

Every senior's judgment was built on struggle. AI is automating the struggle away.

THE PARADOX

AXON ACTIVE
Focusing on your agility

If you survive AI, you need more judgment, not less.

AI hallucinates

It produces confident, plausible, wrong code. You need experience to know when to distrust it.

AI fails silently

No error. No warning. Subtly broken logic ships. Only you catch this — if you have the experience to recognize it.

AI has no stakes

It doesn't care about your production system. The human in the loop carries all the judgment.

That judgment requires exactly the experience we're eliminating the path to.

Internal observation, Axon Active delivery 2025–2026:

"AI output looked correct to those who lacked the foundation to see why it was wrong.
The increase in throughput makes it harder to verify, not easier."

AXON ACTIVE
Focusing on your agility

The science is clear

Decades of cognitive science say you cannot learn by reading alone

01

ACT-R Theory

Anderson, Carnegie Mellon (1982)

- All knowledge begins as declarative — facts you can recite but can't apply
- Only active practice converts it into procedural skill (Stage 2)
- Without practice, you can describe what code does but cannot produce, debug, or architect
- Reading AI output keeps you permanently in Stage 1

02

Dreyfus Model

Dreyfus & Dreyfus (1980/1986)

- Five stages: Novice → Advanced Beginner → Competent → Proficient → Expert
- Advancing past Competent requires emotional engagement with real outcomes
- Over-reliance on rules and AI stalls learners at Advanced Beginner
- True expertise is "knowing how" — tacit skill that only comes from doing

03

Procedural Knowledge

Healy & Bourne / Braithwaite (1995–2021)

- Procedural knowledge is not transferable from reading — it must be built through practice
- It is context-dependent and often inaccessible to conscious awareness
- Practice strengthens declarative knowledge too — but the bridge is doing, not watching
- Instruction without practice produces developers who know about code but cannot write it

You cannot learn to code by only reading AI-generated code, just as you cannot learn to swim by watching YouTube videos.

AXON ACTIVE
Focusing on your agility

Your brain on AI

MIT Media Lab: EEG brain scans reveal what AI does to cognitive function

47% collapse in brain activity

ChatGPT users vs. brain-only control group (EEG measurement)

Study Design

- 54 participants from 5 Boston-area universities
- 3 groups: ChatGPT vs. Google Search vs. brain-only
- 3 essay-writing sessions measured via EEG brain scans
- Kosmyna et al., MIT Media Lab (2025)

What the teachers said

"Two English teachers who assessed the essays called the ChatGPT-produced work largely soulless — all essays were extremely similar, lacking original thought."

Kosmyna et al., arXiv:2506.08872 (2025)

83% Memory failure

AI users unable to remember a passage they had just written

↓ Progressive laziness

Users shifted from structural questions to simple copy-paste over time

⚠ Cognitive debt persists

Even after stopping AI use, weaker brain connectivity and lower memory retention lingered

Programming study underway

A follow-up study on coding with AI is in progress; published results are not yet available — Kosmyna et al.

FRONT LINE EVIDENCE

AXON ACTIVE
Focusing on your agility

What we see at Axon Active

Hiring & team signals

- Clients now require AI competence as a must-have for new hires and team substitutions
- 85% of developers regularly use AI tools for coding (JetBrains 2025, 24,534 devs) — the baseline has shifted
- AI skills take 2–4 weeks to learn.
  Domain knowledge takes years.
  Axon Active internal observation
- New-graduate share of Big Tech hires halved: 15%→7% (SignalFire 2025) — we are investing counter-cyclically
- AI-assisted code shows ~8× growth in duplicate code blocks (GitClear, 211M lines, 2025)

Percentages show relative time distribution, not absolute effort.

Activity	Before	With AI		Comparison (bars = % of total work)
Writing code *AI generates boilerplate, completions & scaffolding*	50%	**20%**	↓	was 50% / now **20%**
Code review *More review needed — AI produces more code, faster*	20%	**35%**	↑	was 20% / now **35%**
Testing *Both AI test generation AND human verification increase*	15%	**25%**	↑	was 15% / now **25%**
Docs & Planning *AI drafts docs, but alignment and refinement increases*	15%	**20%**	↑	was 15% / now **20%**

↓ Decreasing | ↑ Increasing | Blue bar = AI | Green bar = Human

Illustrative internal time-distribution snapshot, Q1 2026 — not industry benchmark

WARNING SIGNS

You may already be seeing these in your team.

AXON ACTIVE
Focusing on your agility

Your junior developers report high confidence — but show gaps in fundamentals

AI masking skill development. They can use the tool. They can't explain the output.

Your team cannot explain AI-generated code in review — they accepted it without understanding

The most dangerous failure mode. Code ships that nobody truly owns.

Your PR review times are shrinking even as AI usage increases

Review may be getting compressed rather than quality actually improving. Speed is hiding risk.

Your velocity is spiking — but tested, deployed features aren't increasing proportionally

AI producing unvalidated code. Output volume is not delivery value.

Your team feels faster than they are — 39 percentage-point gap between perceived and measured productivity

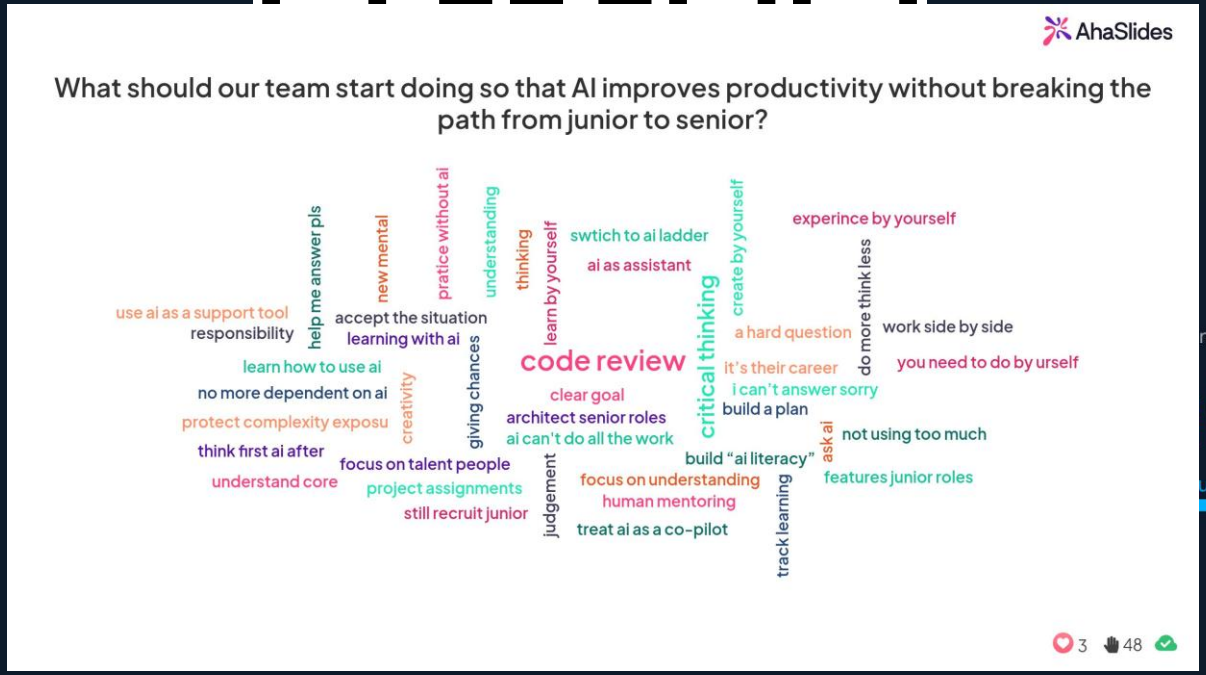
METR 2025 RCT (experienced open-source developers, 246 tasks): predicted 24% faster, reported 20% faster, measured 19% slower.

Source: Axon Active AI Leadership Handbook v1.3 · Chapter 9: Measuring What Matters

AXON ACTIVE
Focusing on your agility

https://ahaslides.com/VENJV

What should our team start doing so that AI improves productivity without breaking the path from junior to senior?



remaining
00
ur answer now

THE REAL PROBLEM

AXON ACTIVE
Focusing on your agility

AI is the junior that never grows up.

What AI does

Writes the code. Fixes the bug. Builds the feature. Does the work juniors used to do — faster, cheaper, at scale.

The output is there.

What AI can't do

Build judgment. Understand consequences. Grow. AI does the junior work but never becomes a mid. It needs a senior permanently holding its hand.

The growth is not there.

What breaks

The senior who holds the hand was grown the old way — through years of struggle. When that generation retires, nobody is coming up behind them.

The pipeline is empty.

No market incentive fixes this. It has to be a values decision.

Anthropic Agentic Coding Report, Jan 2026: engineers use AI in 60% of work but can fully delegate only 0–20% of tasks.

THE SOLUTION

There is no clean answer. But three approaches can work.

AXON ACTIVE
Focusing on your agility

01 For leaders

Design the struggle back in

Junior roles built around learning milestones, not output. No-AI tasks by design. Mandatory code ownership at small scale. Deliberate pairing: junior + AI + senior — not junior replaced by AI.

02 For juniors

Create the AI failure specialist

A new entry point: learn by breaking AI output. Red-team it. Find hallucinations, edge cases, silent failures. This still builds deep judgment — just through a different door. Could produce strong future seniors through a different path.

03 For everyone

Treat senior time as finite and irreplaceable

Your senior developers are a non-renewable resource. Extract and document their judgment now. Build mentorship structures. Measure expertise debt. Ask: if they left tomorrow, who carries this?

The uncomfortable truth:
There is no market incentive to solve this. Cutting juniors saves money now. The consequences arrive in 10–15 years, under different leadership. The organizations that fix it will do so as a values decision — not a financial one.

THE SOLUTION

There is no clean answer. But what we can find out?

AXON ACTIVE
Focusing on your agility

You are analyzing workshop results from software development teams in Vietnam. The workshop explored how AI adoption affects junior developer growth, software quality, and security.

This workshop was held with software development teams in Vietnam. The participants come from different companies and experience levels across the Vietnamese software industry.

The goal of this workshop is not to fight AI or slow it down. AI is a gift — it makes us faster, sometimes 50% or more. The goal is to use AI right: be more efficient, more secure, and stay in control.

The reality we see: AI-assisted and vibe coding can produce working apps fast, but many cannot scale and carry serious security risks. Juniors ship code faster with AI but may not understand what they ship. Without IT security knowledge, architecture understanding, and debugging experience, developers cannot set the right boundaries for AI. AI should not control us — we should control AI. The human in the loop is not optional, it is the safety net. If juniors never earn the basics — debugging, architecture thinking, security awareness, reading code critically — then in 5 years we have senior titles with junior judgment, and nobody left who can catch what AI gets wrong.

The core question: if AI takes over the tasks that juniors used to learn from, how do juniors build the experience and judgment needed to guide AI toward good, secure, scalable software?

I am attaching 3 word cloud images. Each image represents the combined answers from all workshop teams. Image 1 is answers to "Think of a specific moment, task, or struggle that made you a stronger developer. What was it, and why did it matter?" Image 2 is answers to "Where in our team does AI increase speed but weaken understanding, ownership, or judgment?" Image 3 is answers to "What should our team start doing so that AI improves productivity without breaking the path from junior to senior?" The larger the word, the more frequently teams mentioned it.

Synthesize into three sections:

SECTION A — For companies and team leads: One sentence naming the core tension. The top 5 risks to junior developer growth and software quality from AI adoption. The top 5 root causes behind those risks. 5 concrete team practices to introduce — specific enough to start next week, each stating what to do, who owns it, and how it balances AI speed with human learning. 3 mistakes to avoid when introducing AI into development teams. A 30-day action plan with weekly milestones and owners (tech lead, senior dev, manager).

SECTION B — For junior developers: The 5 skills and experiences you must actively seek out, even when AI can do the task for you. A personal learning path for months 1–6, 6–12, and 12–24. How to use AI as a learning tool instead of a shortcut — with specific daily habits. The security and architecture knowledge you need before you can trust and guide AI output.

SECTION C — When to use what: Where AI adds the most value with the least risk (use freely). Where AI helps but human review is critical (use with controls). Where AI is dangerous without deep experience (use with caution or avoid).

Base everything on what the word clouds show. Clearly mark anything you add beyond what is visible in the images.

THE SOLUTION

There is no clean answer. But what we can find out?

You are analyzing workshop results from software development teams in Vietnam. The workshop explored how AI adoption affects junior developer growth, software quality, and security.

This workshop was held with software development teams in Vietnam. The participants come from different companies and experience levels across the Vietnamese software industry.

The goal of this workshop is not
secure, and stay in control.

The reality we see: AI-assisted
understand what they ship. Witho
control us — we should control A
awareness, reading code critical

The core question: if AI takes o
software?

I am attaching 3 word cloud imag
made you a stronger developer. W
judgment?" Image 3 is answers to
frequently teams mentioned it.

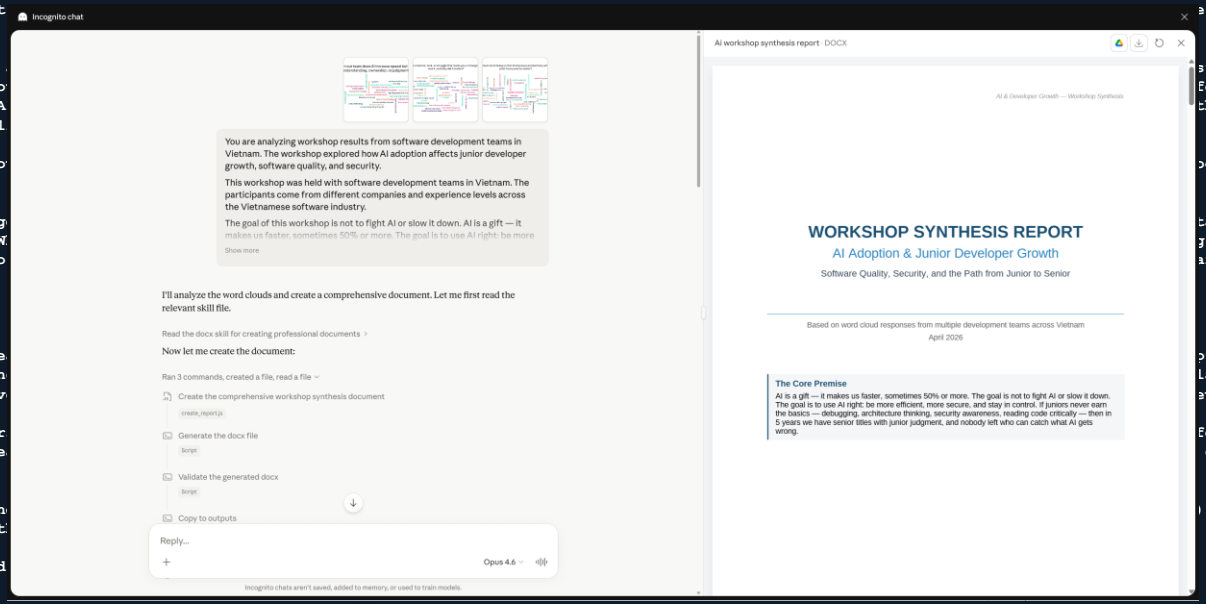
Synthesize into three sections:

SECTION A — For companies and te
causes behind those risks. 5 con
human learning. 3 mistakes to av

SECTION B — For junior developer
and 12–24. How to use AI as a le
output.

SECTION C — When to use what: Wh
without deep experience (use wit

Base everything on what the word



THE VISION

AXON ACTIVE
Focusing on your agility

The perfect AI developer, end of 2026

01 Systems thinker

Sees architecture, not just syntax. Designs for scale, resilience, and change.

02 AI force multiplier

Uses AI to accelerate every phase — but validates every output like a senior.

03 Domain expert

Understands the business deeply enough to say no to technically correct but wrong solutions.

04 Quality guardian

Reviews AI output with the skepticism of a seasoned engineer. Owns every line shipped.

05 Perpetual learner

Adapts faster than the tools change. Learns from failure, not just documentation.

Not a superhero. A disciplined professional who treats AI as a tool, not a crutch.

AXON ACTIVE
Focusing on your agility

What a junior must do to become the senior AI cannot replace

01

Build real things

Stop following tutorials. Ship features to real users. Break production and learn to fix it.

02

Learn the business

Sit with stakeholders. Understand why the software exists, not just how it works.

03

Review like a senior

Read every line AI writes. Question it. Rewrite it. Build the judgment muscle.

04

Own your failures

Bugs you caused and fixed teach more than features AI wrote for you. Embrace the struggle.

05

Pair with seniors now

Their instincts were forged in decades of trial. Absorb it before AI makes the path invisible.

The window is closing. The juniors who start this journey today will be the seniors AI can never replace.

AXON ACTIVE
Focusing on your agility

We should see new developments as a gift
— a second chance to build a better world,
and use AI to save it, not destroy it faster

Use AI — but use it wisely!

Sebastian Sussmann (SEBI)
CIO at Axon Active Vietnam Co., Ltd.

Most images in this presentation were created with AI tools including Gamma, AIEASE, and Claude, except where otherwise credited.

APPENDIX

AXON ACTIVE
Focusing on your agility

Selected sources & references

SignalFire
New-graduate hiring trends in Big Tech. Source for 15% → 7% new-grad share decline since 2022.

JetBrains Developer Ecosystem Survey (2025)
24,534 developers, 194 countries. Source for 85% regular AI usage figure.

GitClear (2025)
Analysis of 211M changed lines of code. Source for ~8× growth in duplicate code blocks in AI-assisted codebases.

Anderson, J. R. (1982)
ACT-R cognitive architecture. Acquisition of cognitive skill. Psychological Review, 89(4), 369–406. Three-stage model of skill acquisition.

Dreyfus & Dreyfus (1980/1986); Dreyfus, S. E. (2004)
Five-Stage Model of Adult Skill Acquisition. Original model 1980/1986; 2004 paper is Dreyfus's summary in Bulletin of Science, Technology & Society, 24(3), 177–181.

METR (2025)
RCT with experienced open-source developers, 246 tasks. Predicted 24% faster, reported 20% faster, measured 19% slower. arXiv preprint.

Anthropic RCT (January 2026)
52 developers learning a new Python library. AI users scored 17% lower on comprehension. Published on anthropic.com.

Anthropic Agentic Coding Report (January 2026)
Internal survey of Anthropic engineers. Source for 60% AI usage / 0–20% full delegation figure. resources.anthropic.com.

Kosmyna et al., MIT Media Lab (2025)
Your Brain on ChatGPT: 54 participants, EEG brain scans. 47% brain activity collapse, 83% memory failure. arXiv:2506.08872.

Healy & Bourne (1995) / Braithwaite et al. (2021)
Procedural knowledge characteristics, transfer, and relationship to declarative knowledge. Cognitive Science.

Axon Active internal data
Hiring signals, AI skills timelines, delivery observations. Internal pilot data 2025–2026. Not externally validated.