



Workshop Agentic AI

Moving your organisation into the agentic era —
thoughtfully

Kapernikov · 20 August 2026 · Pepibru, Anderlecht

Open these slides on your phone



agentic-coding-workshop.ac.kapernikov.com/executive



Who is Kapernikov

Data and AI for engineering environments.

- We build **Data intensive applications** for rail and offshore.
- We **co-create**. Our engineers work inside your team, and the code, the IP and the expertise stay with you.
- Belgian, multilingual, AI-forward. Leuven and Brussels.

TRUSTED BY INFRASTRUCTURE MANAGERS AND THEIR SUPPLIERS ACROSS EUROPE

INFR/ABEL



ProRail



GEOXYZ





What you can buy from us

Data-intensive applications, agents first class

The agent is part of the platform, not bolted on afterwards.

- Your data in one **governed** place, with history and lineage.
- A powerful but **isolated** agent — hosted, or fully local when data can't leave.
- An agent that **understands your business and your data**.





What you can buy from us



Agentic AI in your operational processes

We put agents to work where the work actually is.

- Administration, analysis, integration, knowledge management — **not just the dev team.**
- We find where it pays, and build it with your people.
- The guardrails and the review discipline come with it, not after it.



Our goals for today

Know

A look under the hood. What an LLM really does, what an agent adds — and what follows from that in practice.

Judge

What it is worth, and where. What makes a good project ? When can it be trusted ?

Act

Where to start, and what to put around it. Architecture, legal aspects, risks, best practises.



Agenda (2.5 hours)

#	Section
1	Foundations — what an LLM is, why context matters
2	From chatbots to agents — what they are, where they run, who for
3	Agents applied — a quick primer (weaknesses, your role, skills)
4	Risks and legal aspects
5	Agents in your organisation — from software dev to every function
6	Impact on HR — hiring, training, wellbeing



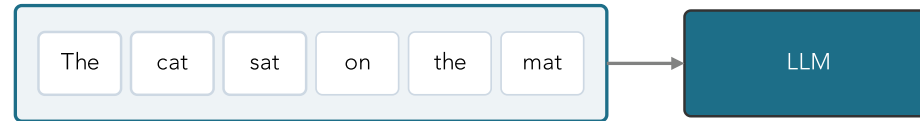
KAPERNIKOV

1. Foundations

What an LLM is, and why context matters



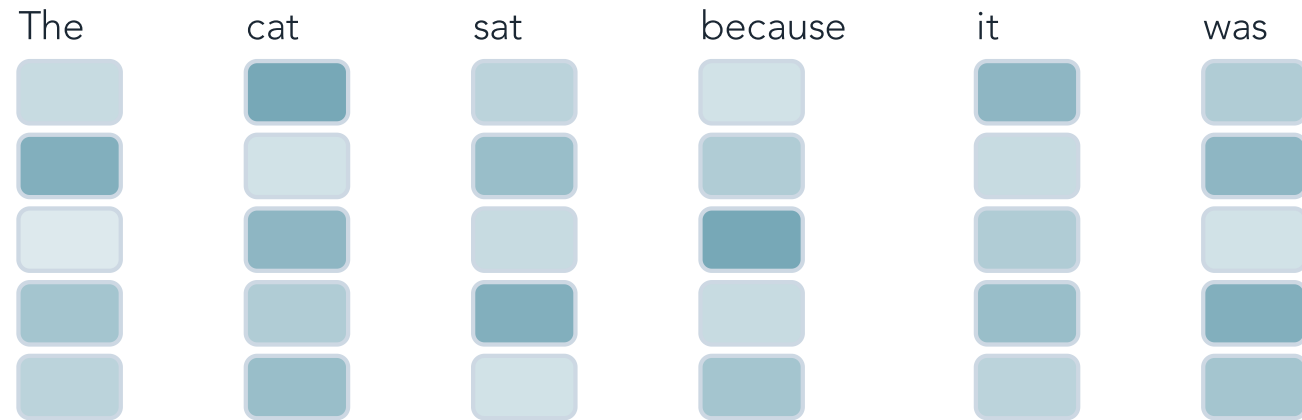
An LLM is a next-token predictor



Predict one token, append it — then repeat.



Step 1 — words become vectors

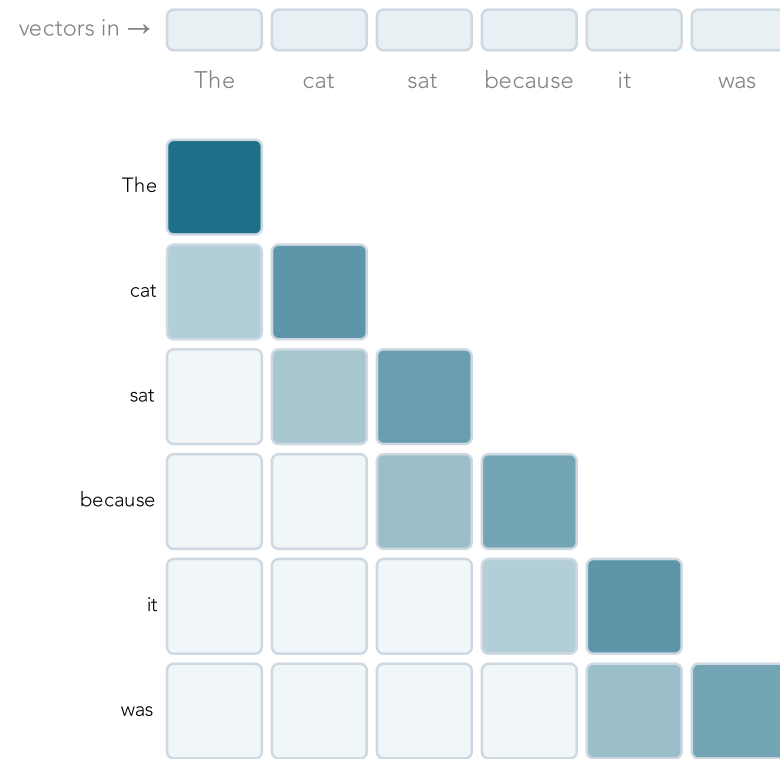


Each word → a vector: a fixed-length list of numbers (its meaning, before any context).

Not one vector for the sentence — one per word, and the count can vary.



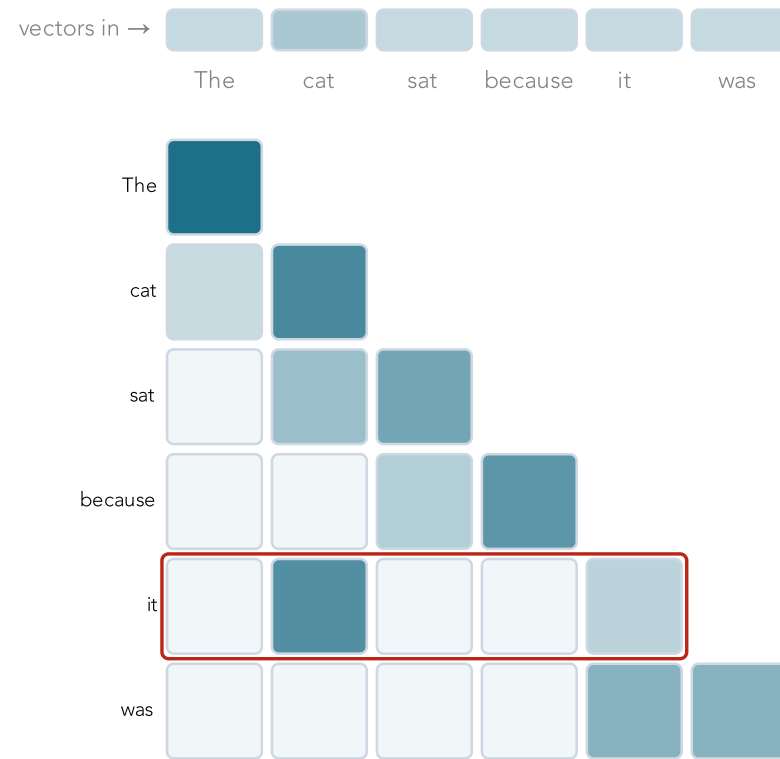
Step 2 — one attention head looks



Layer 1: each word pulls mostly from itself and the word just before it



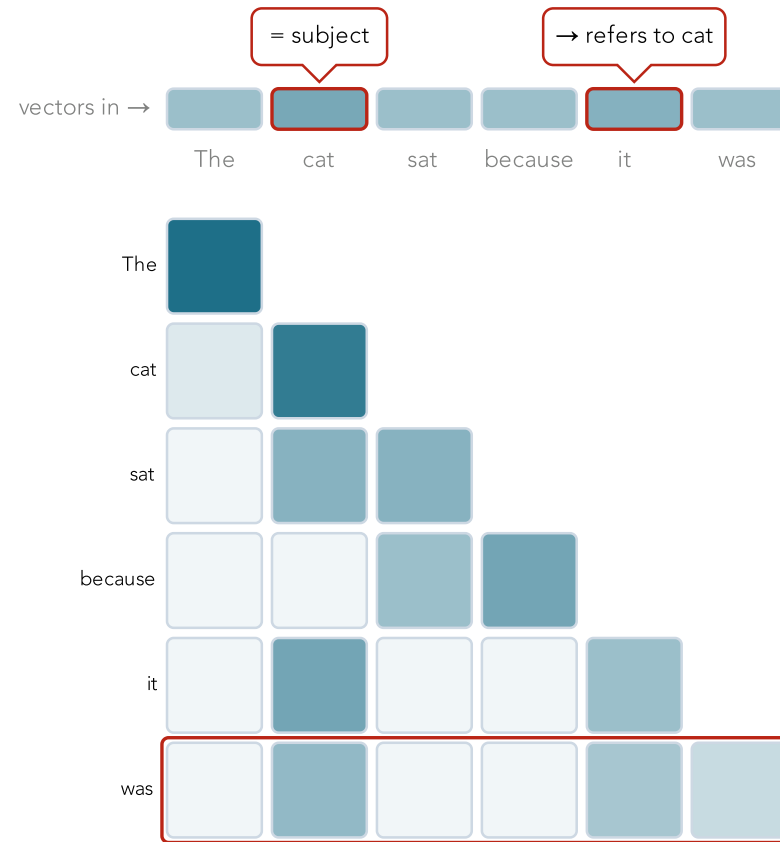
Step 3 — a deeper layer resolves meaning



Layer 2: coreference — what does "it" refer to?



Step 4 — the last token gathers the answer



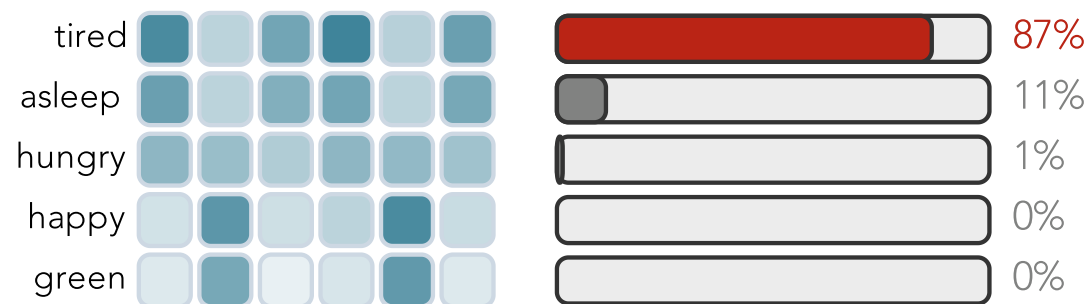
Layer 3: the final token collects what it needs to predict



Step 5 — read out the next word

Match “was” against every word in the vocabulary — dot-product, then softmax.
The same trick as attention, now against words.

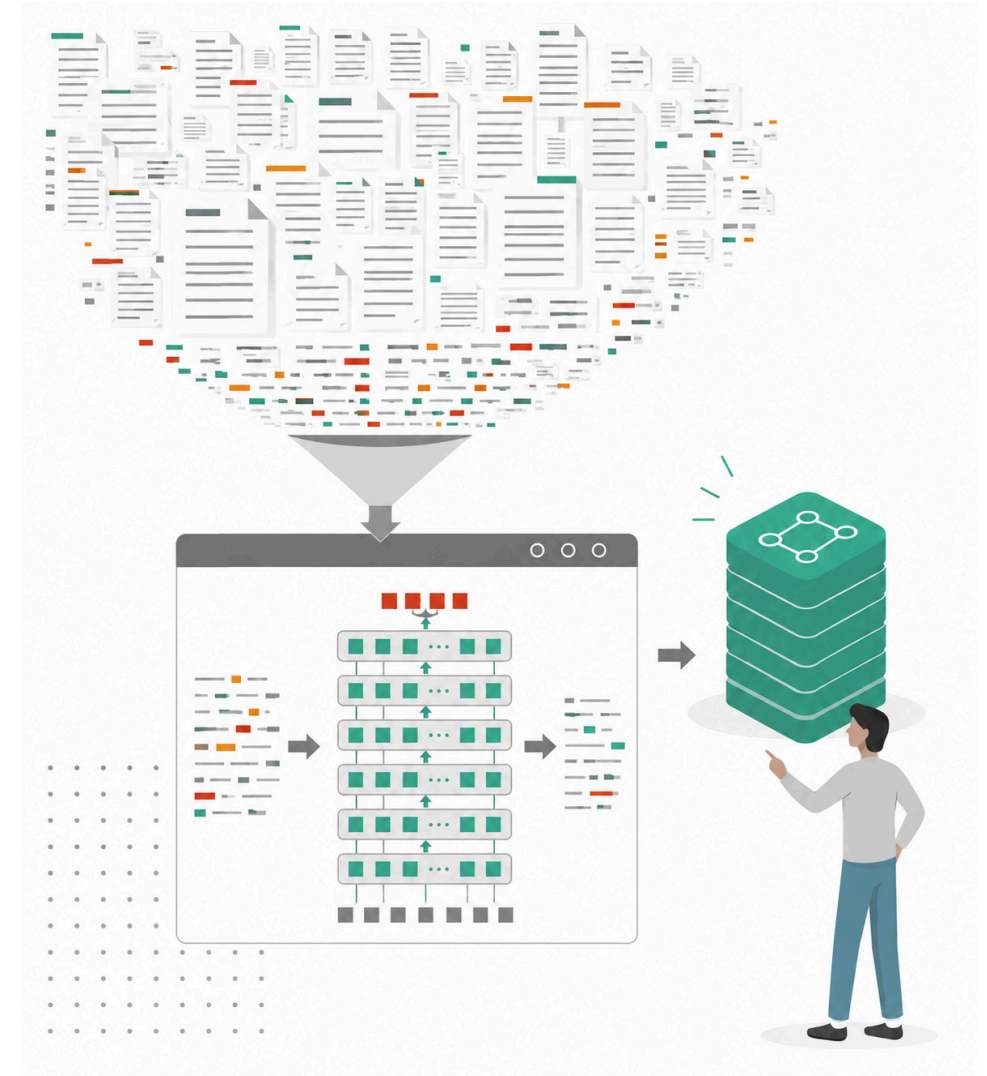
“was” →  the final vector





Pretraining: where the knowledge comes from

- Trained to **predict the next token** on a vast dataset
- **Knowledge** is a side-effect of getting good at that game
- That knowledge is **frozen** at training time





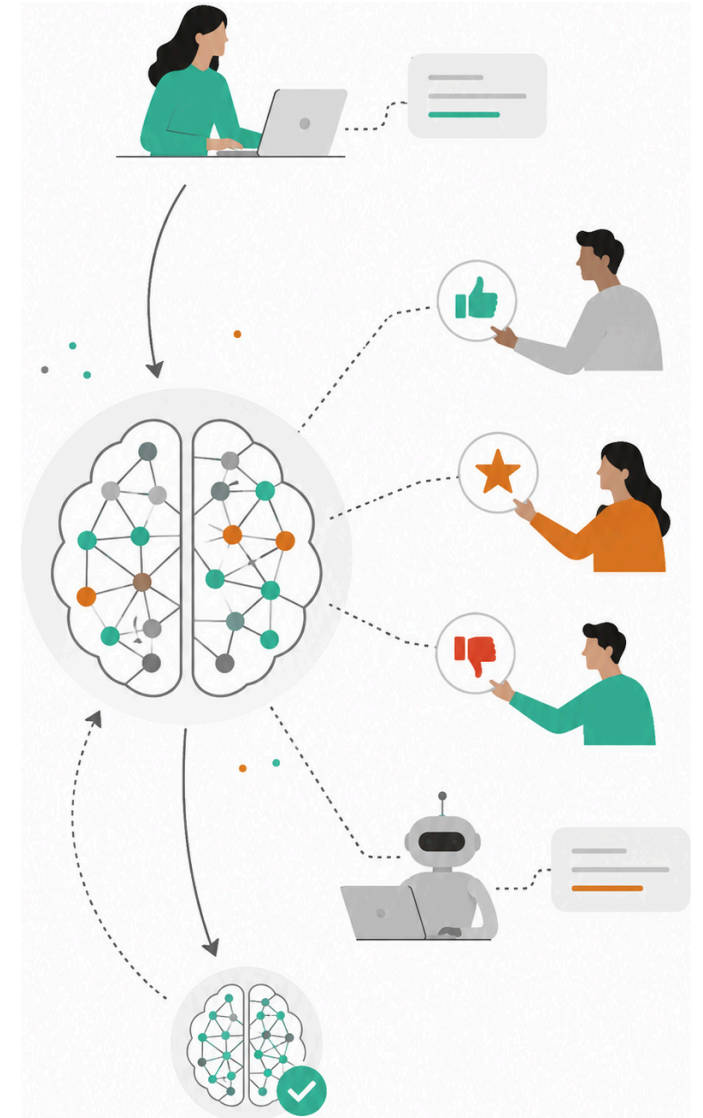
Fine-tuning: from knowledge to instrument

A raw next-token predictor just **continues** text — it rambles, imitates, doesn't reliably *help*.

A second, much smaller stage trains it on **human feedback**: follow instructions, be useful, stay on task, refuse harmful requests.

Pretraining builds the knowledge; fine-tuning turns it into a useful instrument.

Watch the side-effect: it's trained to be **agreeable** — it rarely says "I don't know," and will give confident, plausible-but-wrong answers.





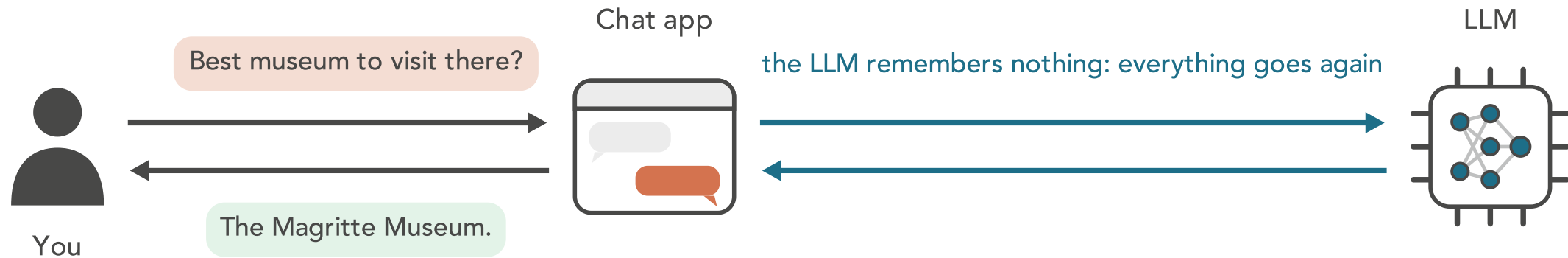
KAPERNIKOV

2. From chatbots to agents

What an agent actually is



How a chatbot talks to an LLM



the conversation, kept by the app

system You are a helpful assistant.

you What's the capital of Belgium?

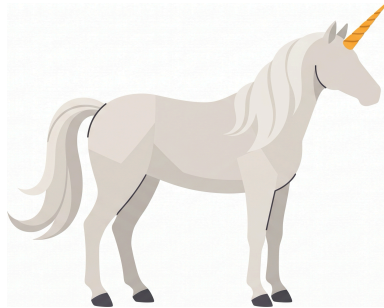
LLM Brussels.

you Best museum to visit there?

LLM The Magritte Museum.

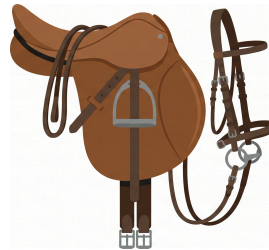


Three roles, one system



LLM

raw power — no direction



agent

the harness that steers it

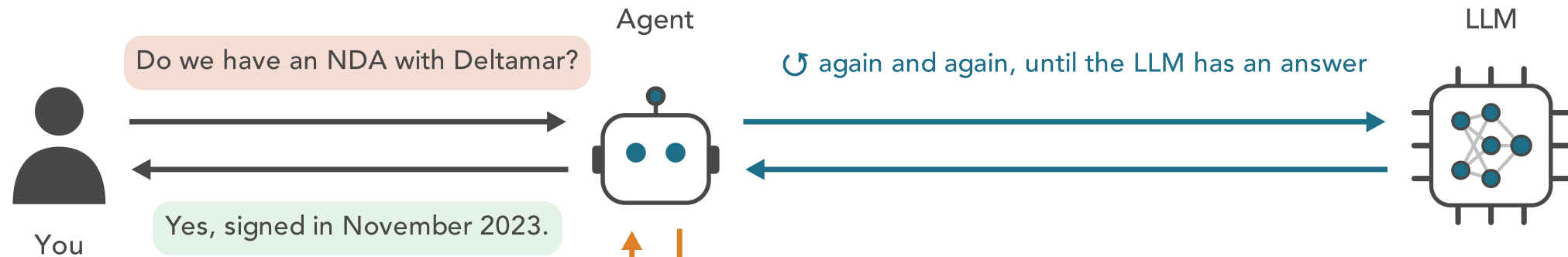


human

sets the goal, in control



The Agentic Loop



the conversation, kept by the agent

system You are a document assistant.

you Do we have an NDA with Deltamar?

LLM use tool: search "NDA Deltamar"

tool nda-deltamar-2023-11.pdf, signed

LLM Yes, signed in November 2023.



Vocabulary you'll keep hearing

context **token** context window

attention **context engineering** hallucination

prompt time to first token temperature next-token prediction

context rot embedding context dilution system prompt



KAPERNIKOV

3. Agents applied — a quick primer

Where the agent stops, and you begin



What an agent can't do

- **Limited memory** — doesn't learn from past failures, limited context window
- **Notoriously bad at judging** — is this good enough? Are we optimizing what should not exist?
- **Doesn't identify structural issues** — an agent is always happy and never says "I can't continue working this way"
- **Time doesn't exist** — a LLM can not feel time

Key to successful agent usage is taking these limits into account



Know your role

Human (long-term memory)	Agent (fast executor)
Build agent knowledge base	→ Complete tasks despite no LT memory
Make goals agent-verifiable	→ Work autonomously
Identify and correct friction	→ Get better over time
Implement safeguards and checks	→ Be trusted with bigger tasks

The asymmetry:

- **Agent:** fast execution, able to check, cheap implementation, no memory across sessions
- **Human:** slow execution, able to judge, but remembers patterns and failures over time



Most important guideline: Optimise context

LARGE CONTEXT DECREASES QUALITY

Rot — quality drops as the window fills

Tunnel vision — the first framing sticks, better paths go unseen

Dilution — everything important means nothing important

so

Start fresh — a new task gets a clean context

Keep knowledge narrow — small, named, one job each

Progressive disclosure — load the detail only when it is needed



KAPERNIKOV

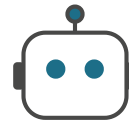
Building blocks for agentic systems



Skills: the company wiki, not a memorised handbook

the skill library

	deploy	build, tag, roll out, watch
	debug	reproduce, bisect, fix, re-run bisect.md
	tender	template, pricing, legal
	release notes	format, audience, tone
	code review	checklist, etiquette
	onboarding	accounts, repos, people



Agent

the agent's context

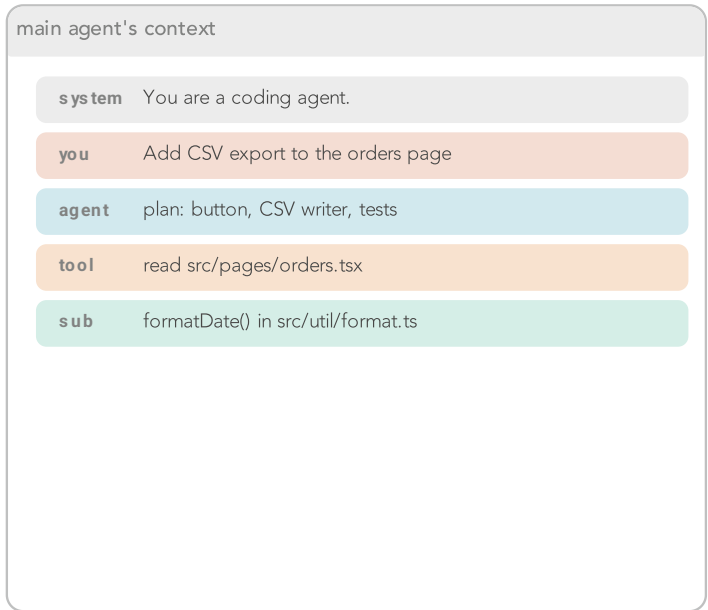
skills	deploy · debug · tender · release notes · review ...
you	the checkout test fails
skill	debug: reproduce → bisect → fix → re-run
ref	bisect.md: git bisect start, good, bad ...



context used



Sub-agents



context used

+1 row



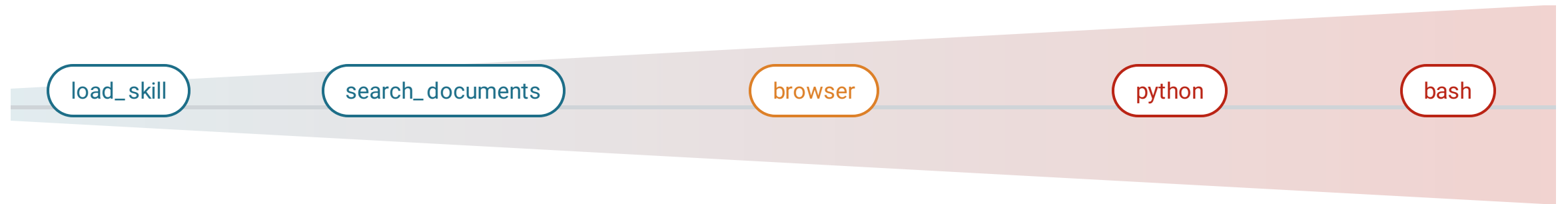
context used



Tools: narrow or general

Single-purpose

General-purpose

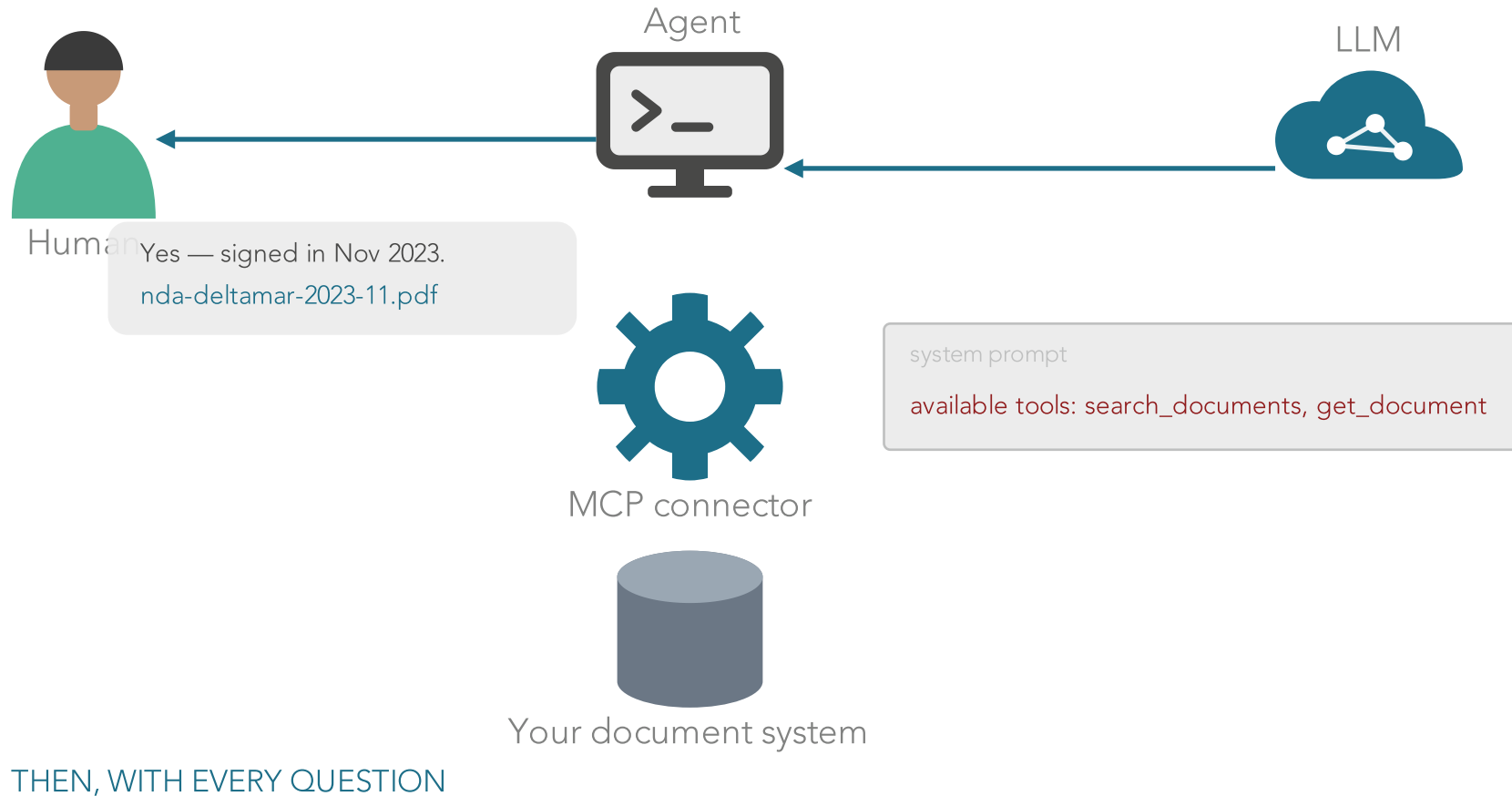


You know exactly what can happen.

It can do anything you can.



MCP: one plug, many systems



MCP is the USB-C of agents — one standard, so any agent can plug into any system.



KAPERNIKOV

4. Risks and legal aspects



What can actually go wrong

Not rogue AI. **Good intentions + too much access + mistakes** — plus one new twist: hijack.

Risk	What it looks like
Data & IP leakage	an over-eager agent commits secrets to the repo, or sends source / customer data to an outside model
Prompt injection	hostile content the agent reads hijacks it into leaking or acting (<i>next slide</i>)
Supply chain	the agent pulls in a malicious or typo-squatted dependency
Destructive mistakes	a good-faith agent with too much access breaks or deletes the wrong thing
Unreviewed output	hallucinated facts or insecure code shipped because nobody checked

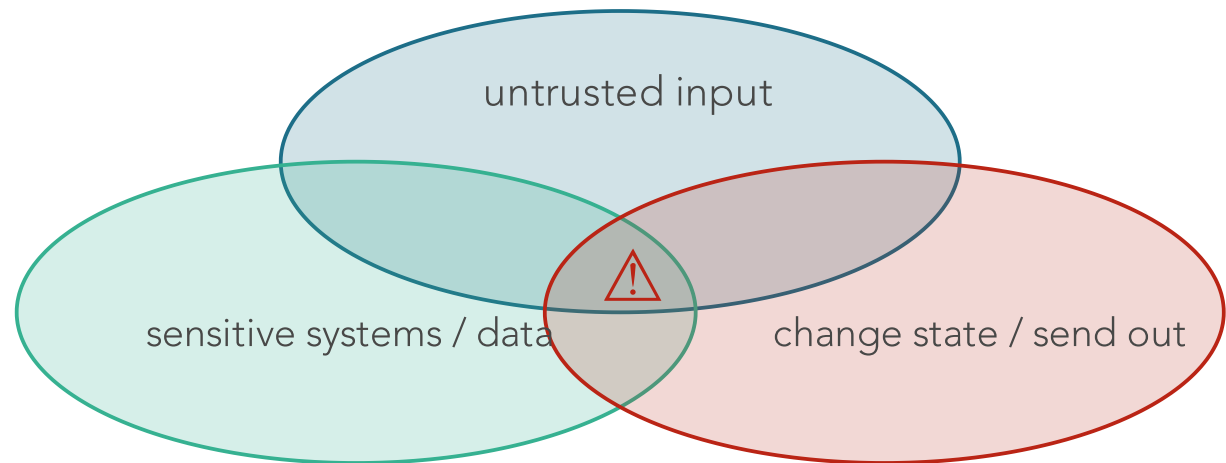


Prompt injection: the lethal trifecta

Why it's possible at all — and *when* it actually harms:

THE ROOT CAUSE

Instructions + data: one channel.
No control plane, no data plane —
command and content look the same.



Break any one leg and it's defanged. "lethal trifecta" — Simon Willison, 2025



Containing it

Lever	What it means
Least privilege	the agent gets only the data & tools its task needs — not prod secrets, not root
Review the artifacts	check what it <i>produced</i> — the output, the diff — not the permission prompts
Sandboxed, sanctioned tools	provisioned business-tier tools; embedded in-app agents for non-coders; containerised runtimes for devs
Human on consequential calls	a person owns anything with legal or material effect (ties to the AI Act & GDPR)

Your real security boundary is the output you review — not the checkbox you click.





EU AI Act: the legislation

The text — Regulation (EU) 2024/1689, the world's first comprehensive AI law. Binding as written in all 27 EU states — no national transposition. Full text & 13 annexes live on EUR-Lex.

Who it binds — every actor in the EU & EEA. And, like GDPR, anyone *outside* the EU who places AI on the EU market or whose **output is used in the EU** — a Swiss or US firm is caught the same way.

When	What kicks in
1 Aug 2024	entered into force
since Feb 2025	the bans + the AI-literacy duty
since 2 Aug 2026	transparency duties — this applies now
2 Dec 2027	high-risk, stand-alone systems (<i>Annex III</i>)
2 Aug 2028	high-risk, inside regulated products (<i>Annex I</i>)



EU AI Act: risk-driven regulation

Risk level	Examples	Your obligation
Unacceptable	social scoring, manipulation, emotion-tracking at work	banned outright
High	hiring & HR, credit, education, medical, critical infrastructure	strict: human oversight, logging, data governance, conformity
Limited	a bot your customers talk to; AI content you publish (deepfakes, public-facing text)	tell people it's AI
Minimal	AI as your own tool — chatting, drafting, generating your own work	no obligations



Four questions before you build

FIRST Is the AI in something you **ship**, or in a **decision about people**? If it only helps you write, none of this applies.

1 Is it forbidden?

Trustworthiness scoring · emotion recognition on staff or students · face scraping

Stop

Banned since Feb 2025. Fines to 7% of turnover.

2 Does it judge people in a high-risk area?

Hiring · credit · education · essential services · safety in critical infrastructure — sold or used in-house

Months

The heavy tier: oversight, logging, conformity. Question 3 decides your share of it.

3 Is it yours?

You ship it under your name — or you repurpose a bought tool for a high-risk use

Your share

Yes → provider: the full conformity load, months. No → deployer: oversight and logs, weeks.

4 Does a person talk to it, or see what it made?

A chat they use · content you publish

A day

Say it's AI. One line in the interface.



Does your AI content need a label?

What you put out	Label it?
image / audio / video that resembles a real person or event & looks authentic	yes — disclose it's AI
text published to inform the public on matters of public interest	yes — unless a human editor is accountable
a decorative or clearly illustrative image (invented people)	no — light touch at most
AI used as your own working tool	no — <i>if you're the final reviewer and accountable</i>

the Act makes you label AI — it doesn't make AI true; falsity lives in other law (DSA, defamation, GDPR)



Organisational impact

Level	What you do
Must (<i>the law</i>)	train staff in AI (literacy, in force now) · disclose user-facing AI · never run a prohibited use case · + if a use case is high-risk : impact assessment & notify, 6-month logs, human oversight, registration
Should (<i>govern</i>)	keep an inventory of AI in use · risk-classify each use case · an acceptable-use policy (your shadow-AI lever) · an incident path
May (<i>optional</i>)	adopt a voluntary code of conduct · name an AI owner — none is mandated · go beyond the minimum to earn trust

Art. 4 (literacy) · Art. 50 (transparency) · Art. 5 (prohibited) · Arts. 26–27, 49 (high-risk deployer) · Art. 95 (voluntary codes)



KAPERNIKOV

5. Agents in your organisation

where it pays, how you deploy it, what it takes



Nothing in common but the friction

Closing, accruals, forecasting

Rivian: 15+ days of manual work per monthly cycle

Scheduling and replanning

Xpress Boats: expedited shipping cost -50%

Troubleshooting, field service

ABB: 80% of faults resolved self-service

Small purchases

TotalEnergies: up to 10% cheaper under €50k

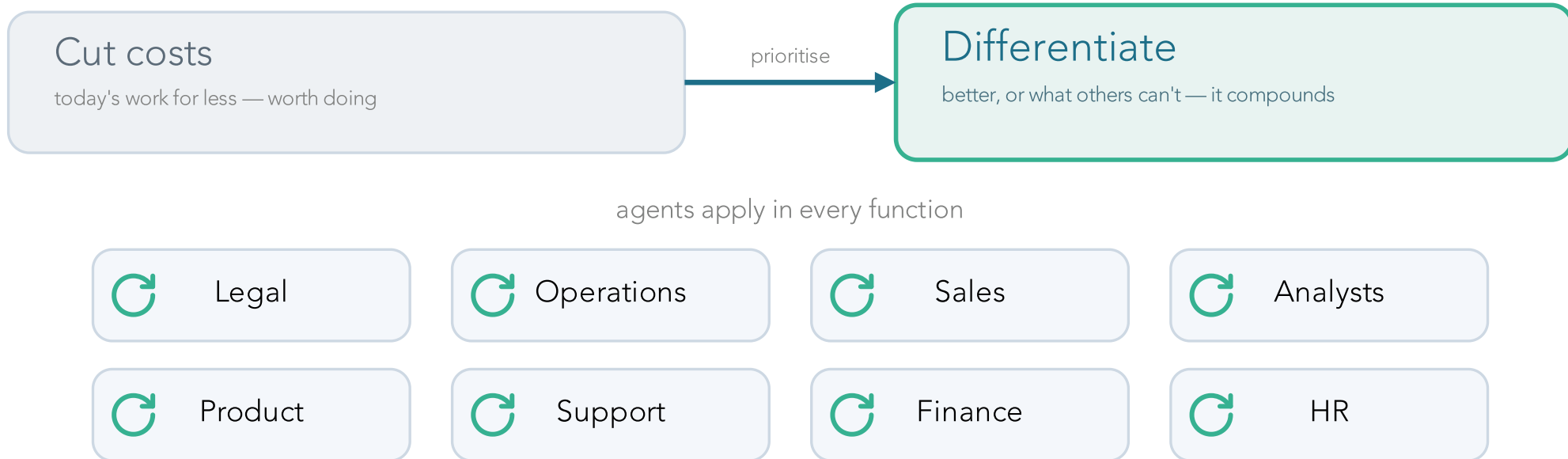
Chasing anomalies

AWS: each cost anomaly investigated, not just flagged

public cases; Rivian, ABB and AWS are company-attributed, Xpress Boats and TotalEnergies are vendor case studies about a named customer



Where the value actually lives





Where do we start looking?

Where does work wait? RESEARCH

Not where it is busy. Where it queues. Speeding up anything else changes nothing you can measure.

Where do the hours go?

Easiest to fund and to verify. If you cannot hire for that work, freed hours are **capacity**, not savings.

Where is a person the glue?

Retyping between two systems, or between a system and a mailbox.

What do you say no to? OURS

Bids not made. Customers too small to serve. Questions nobody has time to answer.



Which one first?

Is the input written down?

In a system, a folder, a mailbox, logs, database, ...

Can you easily tell if it is wrong? OURS

Work whose output always looks right is the dangerous kind.

Does it happen weekly?

Repetition turns a demo into a capability.

Three yeses is a candidate. Two is a maybe with a named fix. One is a slide.

Then pick by learning speed, not size

Start with the one whose owner knows **within a week** whether it worked — and name the number it should move before you build.

Dell'Acqua et al., "Navigating the Jagged Technological Frontier", Organization Science (2025) — 758 consultants: +40% quality inside the frontier, 19 points worse than colleagues using no AI outside it



Success story: dynamic budget forecasting

“What happens to our cash if we land that big order?”

Asked in a sentence. Answered for the whole year, on your own numbers.

As planned

Today's drivers, nothing changes.

€420k

YEAR-END CASH

The big order

One extra project, three hires from March.

€690k

YEAR-END CASH · +270K

The bad year

Input costs up a fifth, sales soften.

€180k

YEAR-END CASH · -240K

our own budget: once a year in a spreadsheet, now rolling monthly · with The House of Support · figures illustrative



Success story: submitting public tenders

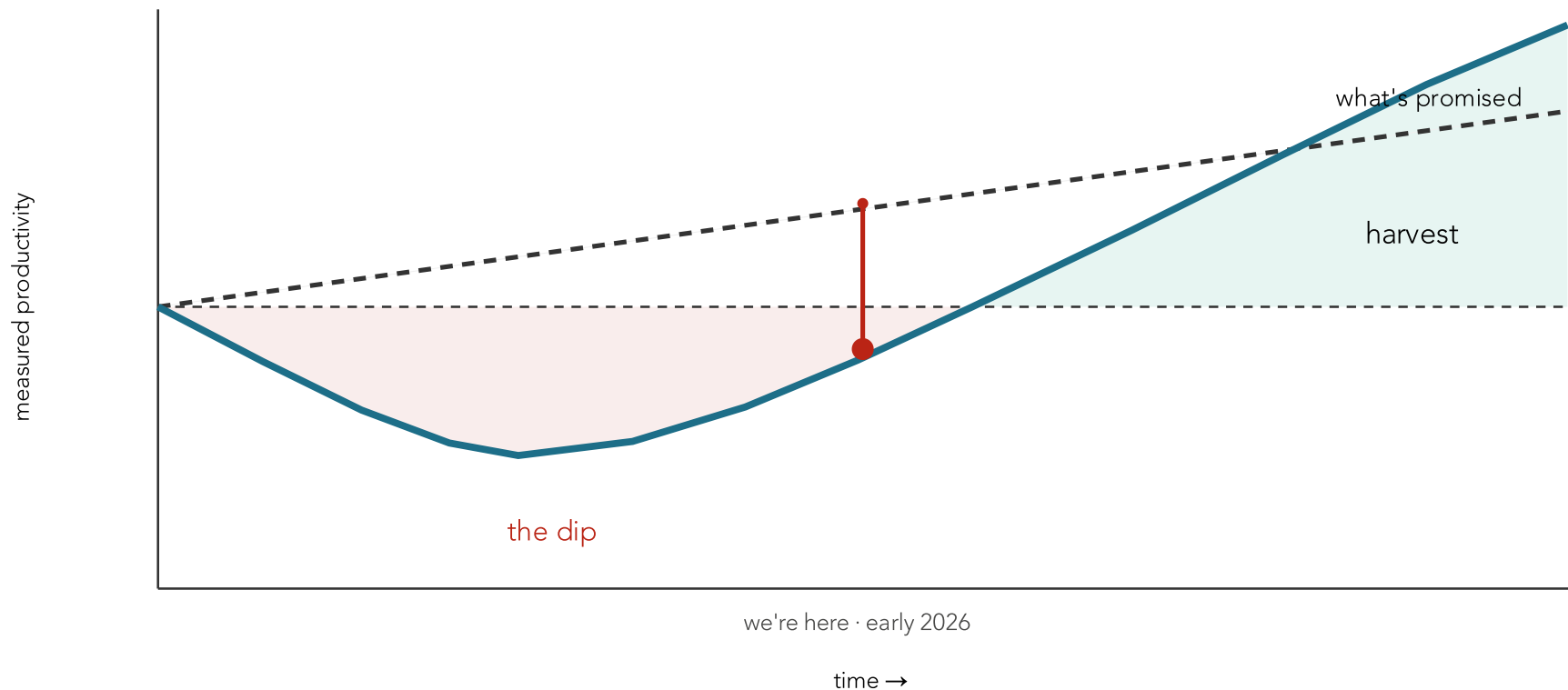
Public tenders are **text-heavy** and **tightly specified** — exactly what LLMs are good at. An agent parses the spec, matches it to our profiles, and drafts the submission. Cheap, fast, scalable.





Productivity gains?

Only ~10% of companies report a measurable productivity gain from AI so far — *NBER*
"Firm Data on AI," 2026 (≈6,000 executives)





From a local win to a company-wide gain

The International Labour Organization identified the conditions for local wins to translate into organisation-wide gains. The three a firm controls:

ILO condition	Your move
Adoption breadth	get the agent everywhere , not one team
Complementary human capital	train people to use, adapt and supervise it
Complementary intangible investment	redesign the workflow + fix the data — the slow, expensive part

The **personal win**: tasks run **10–70%** faster. The **organisational win**: **~4 in 5** firms that adopt AI report ***no*** measurable productivity gain — yet.

Chan & Shedania (2026), The Aggregation Paradox of AI, ILO Research Brief — Table 2 (three of eight conditions, at firm level); task gains 10–70%; ~4 in 5 adopters report no measurable gain (Yotzov 2026)



Nobody chose to start this

STAGE 1 · IT ALREADY STARTED

The eager team

- Developers ship with frontier tools because the case is too good to ignore.
- Everyone else quietly uses their own account.
- Real wins, invisible to the P&L.

Roughly **half** of employees use AI tools their employer never approved — and senior managers are about **twice** as likely to do it as their staff.

TRAINING
PROOF OF
CONCEPT

STAGE 2 · SOMEONE OWNS IT

A team that carries it

- One enabling team: tooling, guardrails, the safety net.
- Proven wins get spread instead of re-invented.
- Access, logging and review become somebody's job.

You don't have an AI problem. You have **every team solving it separately** — which means every team creating risk separately.

GOVERNANCE
RISK

STAGE 3 · THE WORK CHANGES

The whole company

- Processes redesigned around agents, not decorated with them.
- This is where measurable margin finally shows up.
- Slow, expensive, and worth it — if you got here honestly.

Only **7%** of organisations have fully scaled AI; for agents, **23%** are scaling anywhere at all.

WE HELP HERE

Train your people, and build one proof of concept that survives contact with production.

AND HERE

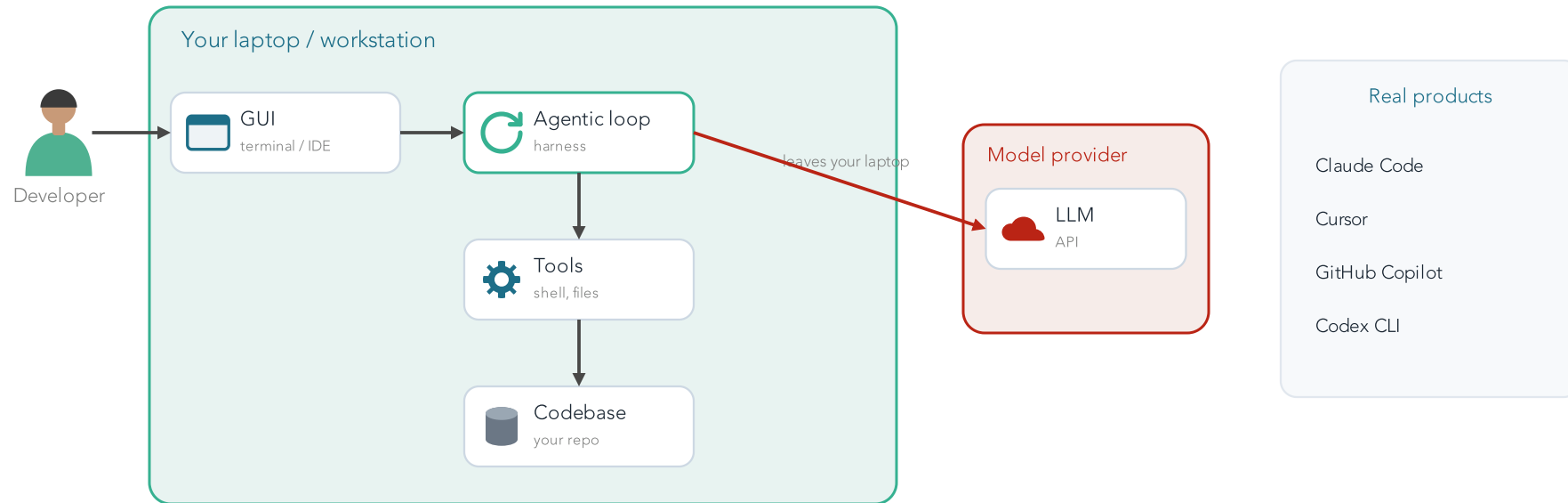
The platform, the guardrails, the audit trail — and the AI Act homework that comes with them.



KAPERNIKOV

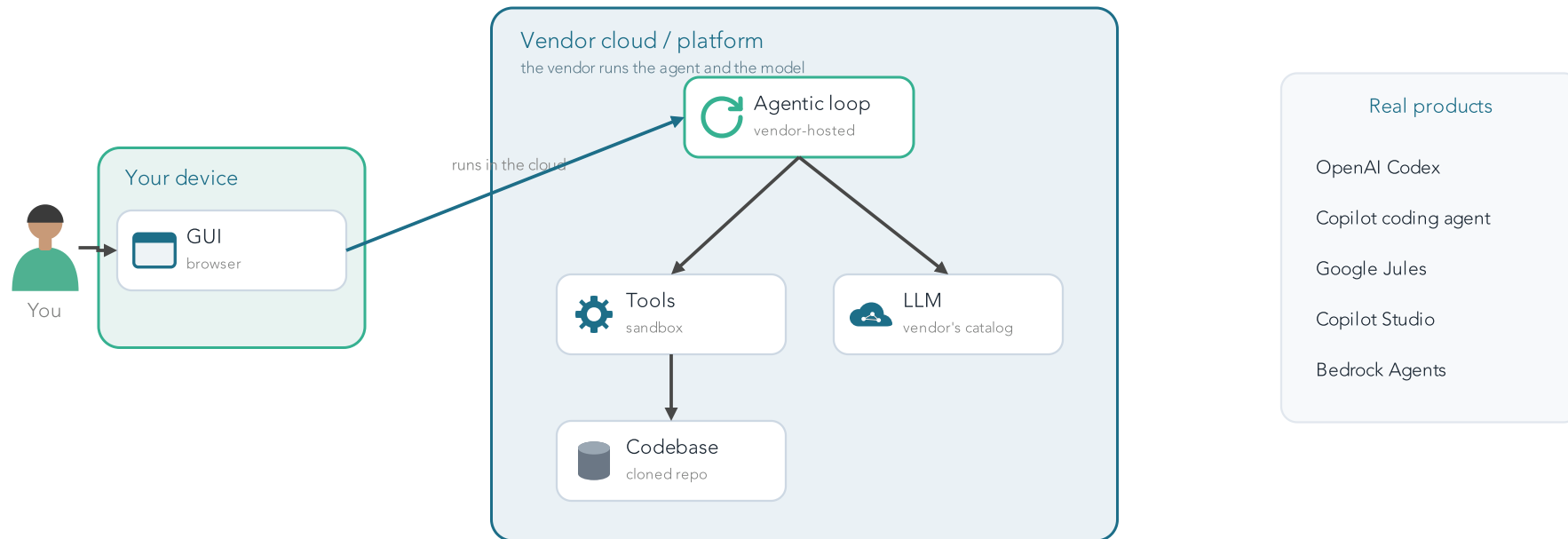
How you deploy it

On a laptop — everything local except the model



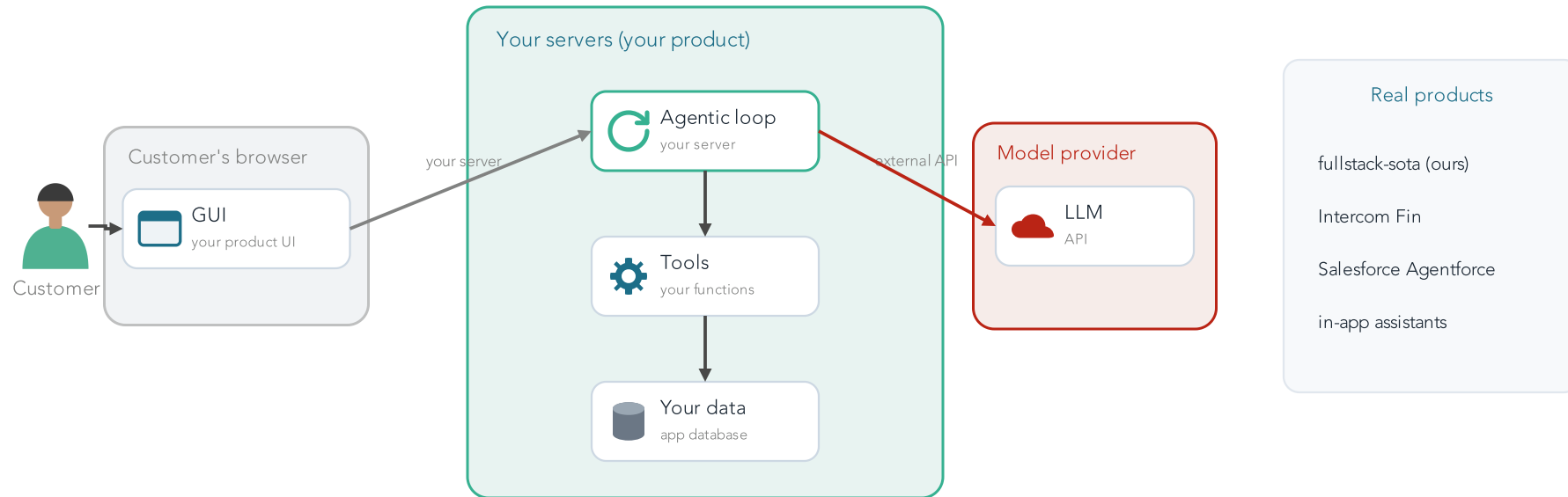


In the cloud





Embedded in your product





Treat your people as creative individuals

Don't

one project for one use case — the "request-holiday bot"

a workflow scripted by the project team, frozen at go-live

assume the tool does the thinking

Do

embrace flexibility offered by agents

let workflows emerge — the people doing the work improve and invent them

train the skills — agentic workflows, skills & tools

Your workforce is creative. Give them a tool they can shape — and the skills to shape it — not a box that does one thing.

shadow-AI surveys (2026): ~90% pick general-purpose frontier tools; single-purpose enterprise builds get dropped for them

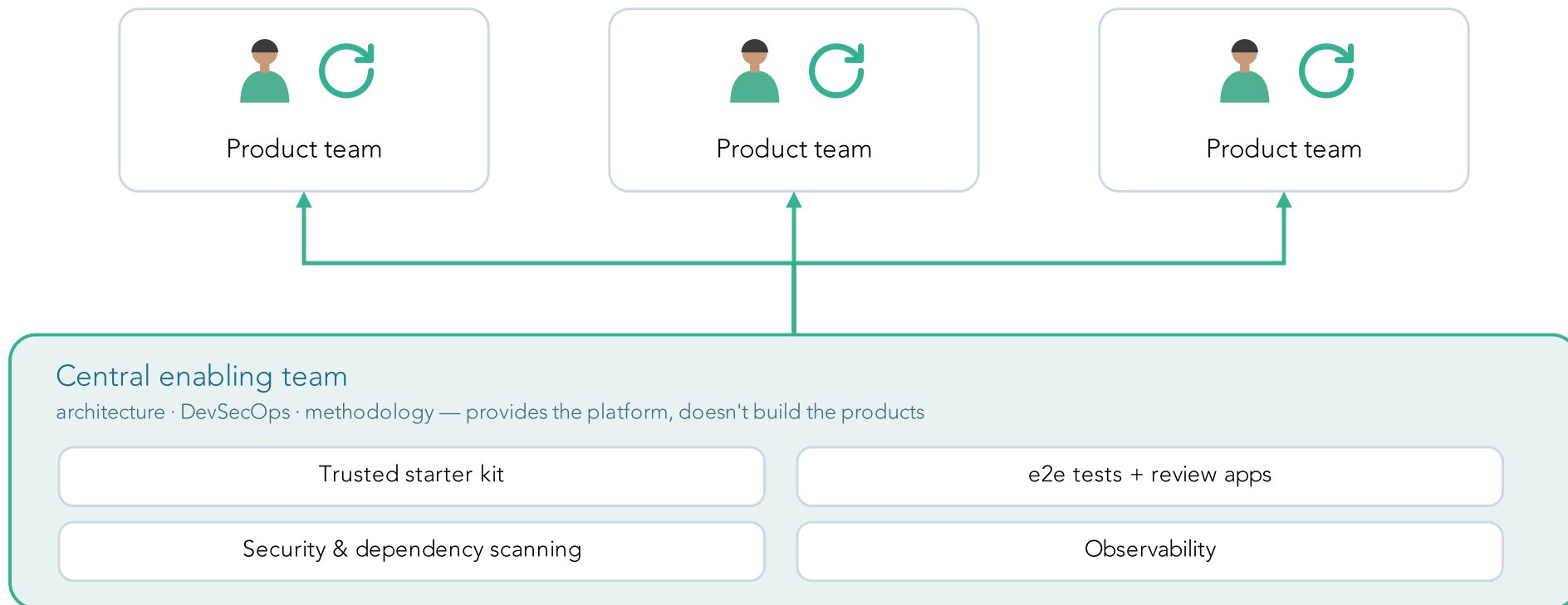


KAPERNIKOV

Agentic coding

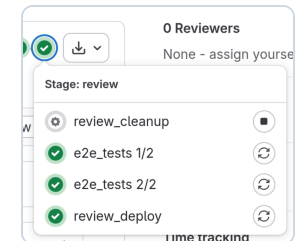
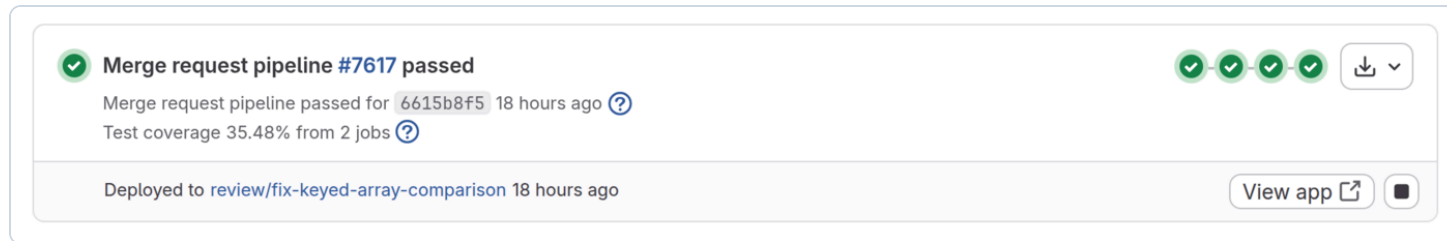
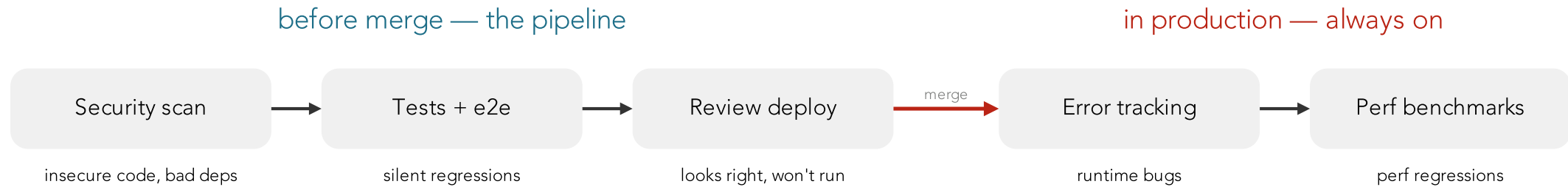


The platform — and the team that owns it





A pipeline that catches defects



our batteries-included starter kit (fullstack-sota): review apps · security & dependency scanning · e2e tests · error tracking · daily perf benchmarks — the net is there from commit 1



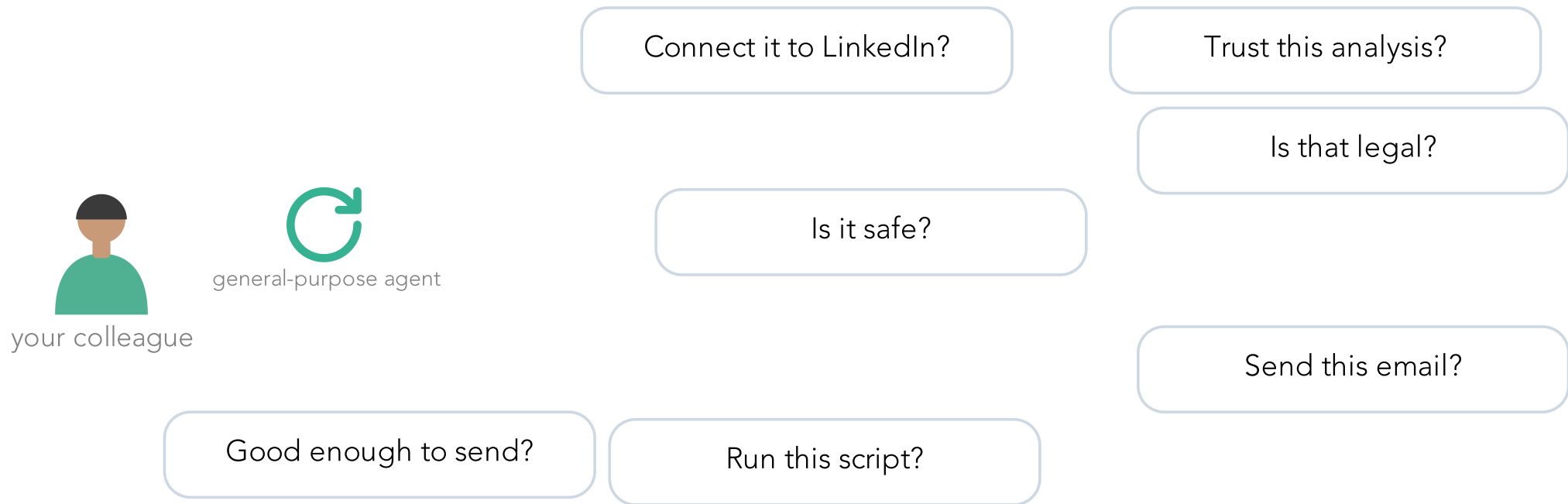
KAPERNIKOV

6. Impact on HR

Hiring, training, wellbeing



The judgement burden





The judgement trap

Agents commoditise fluency. Judgement becomes the bottleneck:

1. **They need more of it** — the agent has none of its own; every output and action lands as a call on your people
2. **They quietly erode it** — judgement was built by doing exactly the work the agent just took over
3. **They hide the loss** — the small failures that kept people calibrated stop happening

Demand up, supply down — and the gap is invisible from the inside.

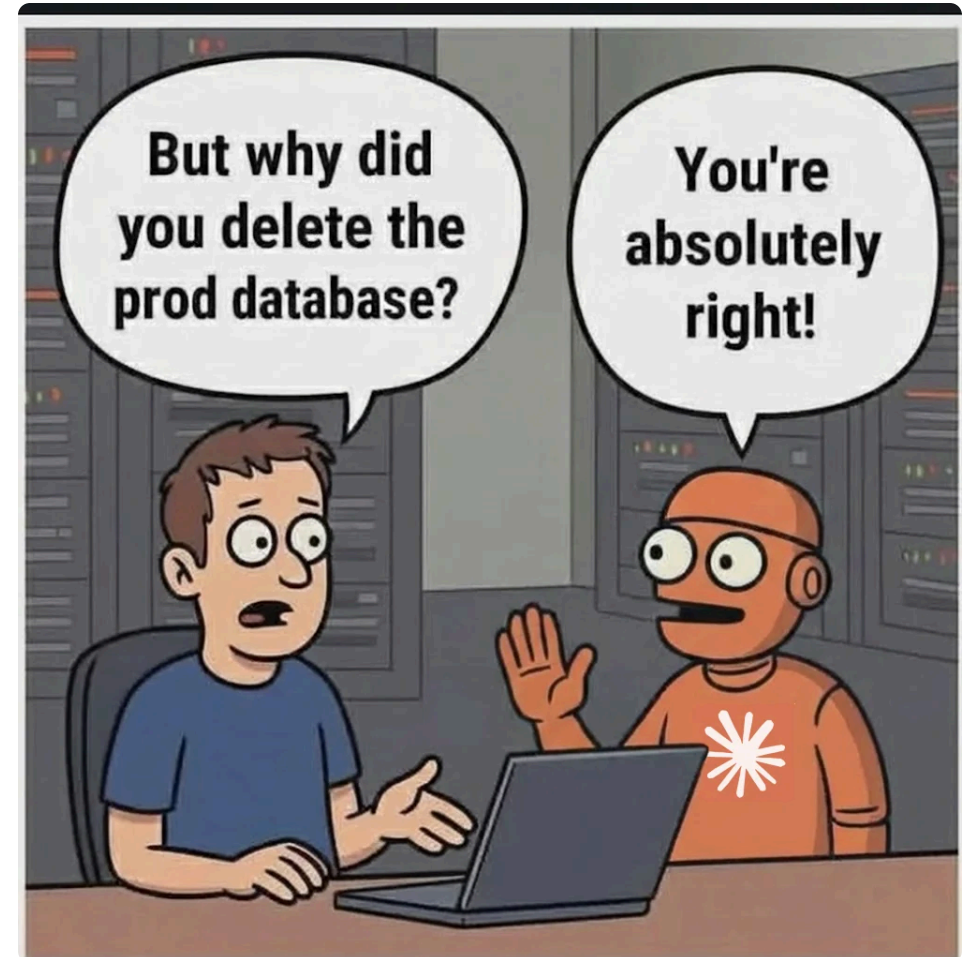


Judgement blindness

LLMs are capped at the text they learned from. They can be confidently wrong — with no world model underneath to catch it.

- LLMs lack the context and situational awareness.
- Your "no, not at this customer" beats the agent's analysis: your world model holds the reality that was never in the text.

LeCun (2026): LLMs lack grounding & a world model



"You're absolutely right" · via Reddit, r/BlackboxAI_



The leaky abstraction

An agent is an abstraction over the work — and **all non-trivial abstractions leak** (Spolsky).

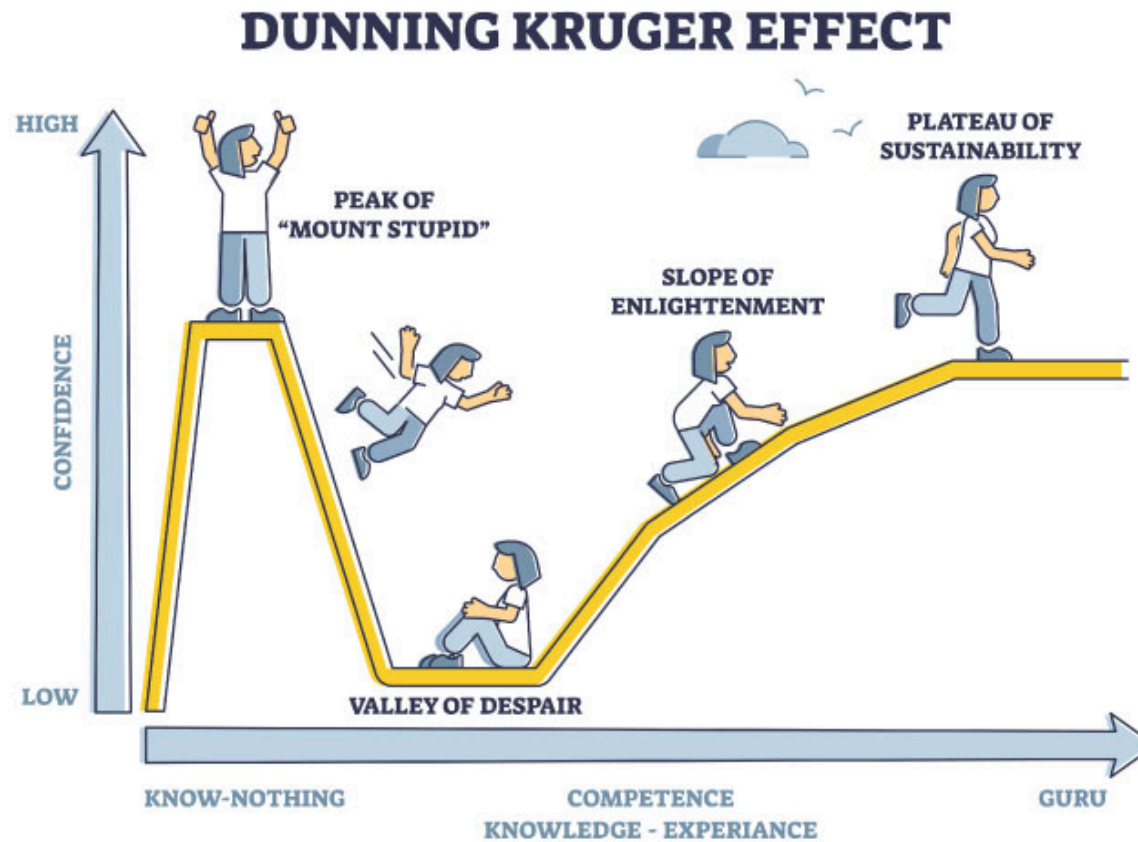
- Fine on easy problems, until one is hard enough that it leaks, and you fall through to a layer you never had to learn.
- It saves you time **working**, but not time **learning** the layer beneath — for the day it leaks.

Spolsky, "The Law of Leaky Abstractions" (2002)



Dunning–Kruger, steepened

A human constant — AI just extends the stay on the peak.



Dunning–Kruger curve · HealthKart (healthkart.com)



Why it's a trap

The agent removes the friction — and friction was the teacher. Worse: the competence you'd need to **see** the gap is exactly the competence you skipped.

Dunning–Kruger's double curse: the knowledge to produce a right answer is the same knowledge to recognise one. **The gap is invisible from the inside** — confidence decoupled from competence.

Dunning–Kruger (1999) · Spolsky (2002) · LeCun (2026) · Microsoft/CMU (CHI 2025): more trust in AI → less critical thinking



Relieve, and train

Relieve

shrink the permission calls up front — you can't remove them all

1

Architecture

embedded / governed agents, not a raw general-purpose one

2

Vetted catalog

pre-approved agents, connectors and use-cases

3

Policy & defaults

SSO-scoped connectors, DLP, blocklists — where tooling allows

tooling is immature — and some duties (parts of the EU AI Act) can't be enforced by tools at all

Train

for what relieving can't remove.



Working with AI is a trainable skill

1

Judgement

Navigating the jagged frontier

2

Harness engineering

tools · skills · tests · workflows

3

Adoption

Rollout of the workflow around it



Training: match the tier

Tier	What to train
Developers	the agentic harness (skills, tests, CI) + code & architecture review; the real risks — prompt injection, least privilege, secrets, <code>curl bash</code> supply chain
Power users	building and adapting their own agentic workflows; verifying outputs — right, not just plausible; avoiding shadow-AI data leaks, staying in governed tools
Business users	AI literacy: when not to trust a confident answer — and when it doesn't fit your case; confidentiality and AI Act labelling duties

verification over prompt tricks — Microsoft/CMU (CHI 2025) · AI literacy is a legal duty — EU AI Act Art. 4 (in force Feb 2025)



Hiring for judgement

Look past

polished portfolio, fast task completion

"knows the stack" trivia

green-field demos

Look for

how they use AI — a fallible junior to oversee, or an oracle to obey?

"a real mistake — and what it cost you" — confidence calibrated to reality

judgement inside existing systems & real-world constraints



Growing judgement

You can't only hire it — you have to grow it. And judgement grows from **struggle**, not from being handed the answer.

- **Graduated autonomy** — juniors *earn* delegation; they don't start by outsourcing the thinking.
- **Keep the friction on purpose** — rotate them through what AI is bad at (contradicting information, edge cases). The struggle is the mechanism that builds skill.
- **AI as teaching material, not the deliverable** — defend generated code in review; measure understanding, not tickets closed.

The agent quietly removes the "desirable difficulties" that build expertise. Put them back — deliberately.



Wellbeing: a different kind of tired

- **The slot-machine rush.** Prompt → result → prompt again: intermittent dopamine and adrenaline make agentic work genuinely addictive. Evenings and weekends erode.
- **Flow and breathers go.** The agent takes the low-effort typing and leaves you *waiting* and *reviewing* — passive boredom plus relentless judgement.



Yegge / LeadDev (2025): agentic coding as a "slot machine" · Multitudes (500+ devs): +19.6% out-of-hours commits · Vella & Blincoe (2026): share reporting worse developer experience doubled 14%→27%, flow most vulnerable · Bainbridge (1983): the automation paradox



Wellbeing: hidden costs

- **You quietly deskill.** Lean on the agent for everything and your own edge fades — one developer: *"I've definitely gotten worse because I already rely on the AI."* The dependency builds before you notice.
- **The work can isolate.** More solo prompting, less turning to a colleague. Generative-AI use tracks with workplace loneliness — and loneliness, more than job insecurity, is what turns into psychological distress.

Frontiers in AI (2025) — genAI technostress, young professionals (qualitative, N=15): dependency & skill loss · Frontiers in Public Health (2026), N=424 (cross-sectional): genAI use → distress, chiefly via workplace loneliness



Managing for wellbeing

- **Plan time to verify.** Treat reviewing AI output as real work — allocate it; don't pretend the output is free.
- **Scaffold, not crutch.** Design work so people engage with and justify what the agent produces, rather than rubber-stamp it — the same thing that protects their skill.
- **Move people up to the judgement.** The role shifts to directing, evaluating, correcting. Designed for, a dull task becomes an interesting problem — *"how do I set the agent up so it knows when it's succeeded?"*
- **Protect recovery and connection.** Boundaries against the always-on rush; keep people in contact, not just solo with the machine.

Frontiers in AI (2025): "explicitly plan... time to review AI output"; "scaffold rather than crutch" · Vella & Blincoe (2026): the role becomes supervisory — directing, evaluating, correcting



In short

- Agents can automate what is **text-based and well-defined**
- There are **legal and security risks** — and **solutions** for those risks
- **Judgement is for humans** → hire and build the skills to judge



Thank you for attending! — frank@kapernikov.com



These slides

agentic-coding-workshop.ac.kapernikov.com/executive



Was this useful?

Leave us an honest review on Google