Toward Evidence-Based Agentic Software Engineering

Prof. Dr. Sebastian Baltes

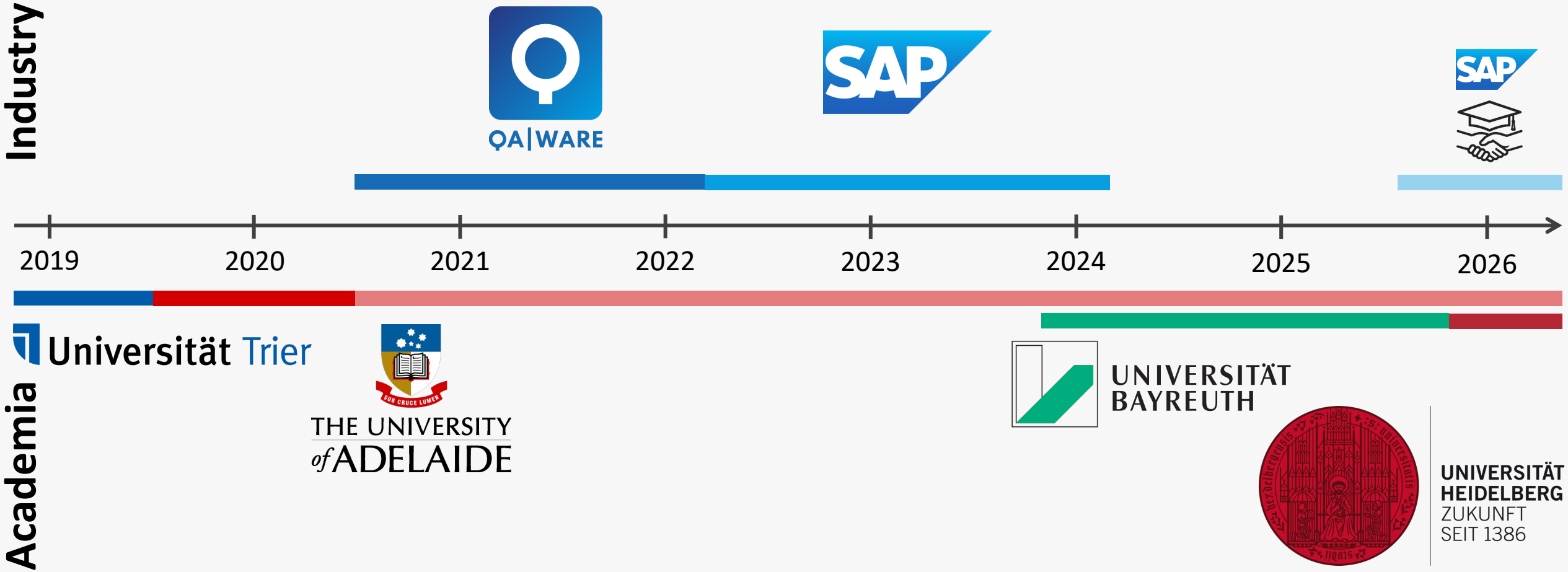


UNIVERSITÄT
HEIDELBERG
ZUKUNFT
SEIT 1386

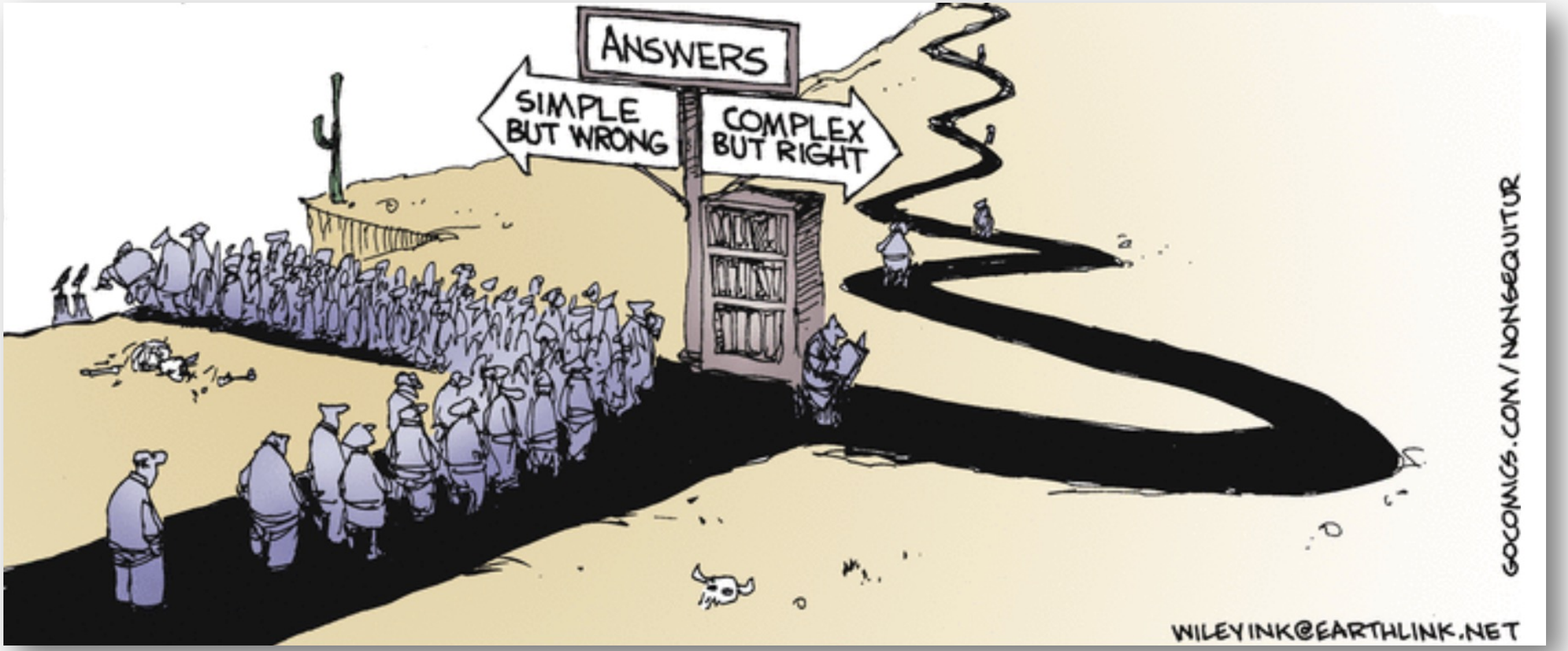
SE@UHD
Software Engineering

“Hi, my name is Sebastian and I’m a pracademic”

https://en.wikipedia.org/wiki/Pracademic



Current Discourse on AI for SE



2022–2024: The Fear That AI Is Eating Software

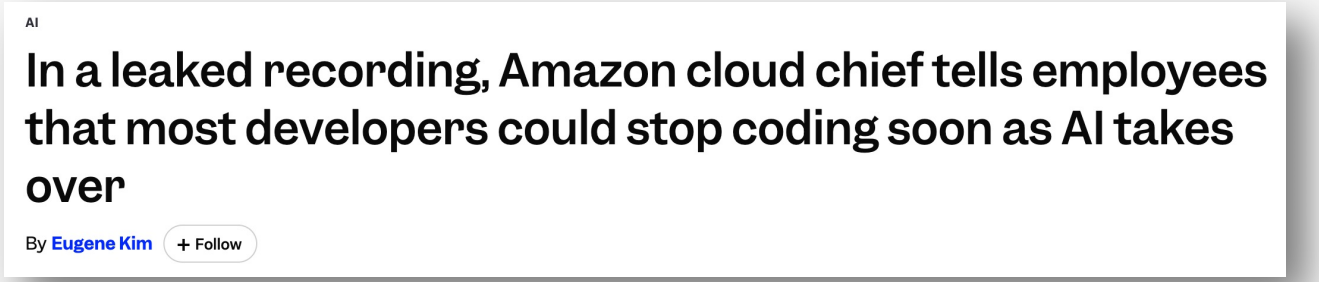

DOI:10.1145/3570220 Matt Welsh

Viewpoint

The End of Programming

The end of classical computer science is coming, and most of us are dinosaurs waiting for the meteor to hit.

https://dl.acm.org/doi/10.1145/3570220


AI

In a leaked recording, Amazon cloud chief tells employees that most developers could stop coding soon as AI takes over

By Eugene Kim + Follow

https://www.businessinsider.com/aws-ceo-developers-stop-coding-ai-takes-over-2024-8


Don't learn to code: Nvidia's founder Jensen Huang advises a different career path

MICHAEL PETRAEUS • 23 FEB 2024

https://vulcanpost.com/853029/dont-learn-to-code-jensen-huang-on-career/


Salesforce Will Hire No More Software Engineers in 2025, Says Marc Benioff

By Henry Martin

Published December 18, 2024

Updated December 23, 2024

https://www.salesforceben.com/salesforce-will-hire-no-more-software-engineers-in-2025-says-marc-benioff/

2022–2024: The Fear That AI Is Eating Software

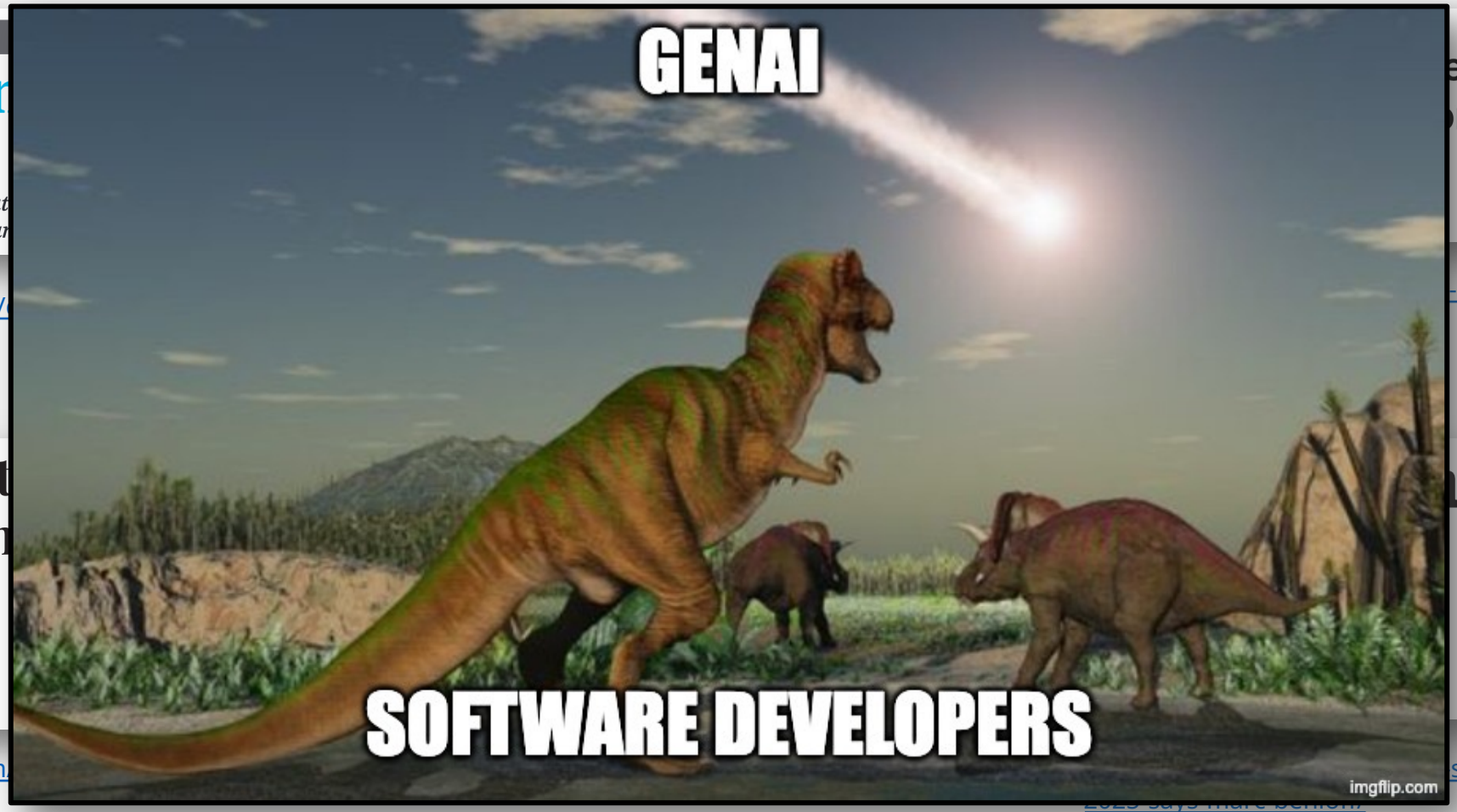
DOI:10.1145/3570220

Viewpoi

The End

The end of classical comput
and most of us are dinosaur

https://



lls employees
n as AI takes

-ai-takes-over-2024-8

Don't learn t
Jensen Huan
path

MICHAEL PETRAEUS • 23 FEB 2024

https://vulcanpost.com/

are Engineers

software-engineers-in-

Since 2025: Discourse Shifts From Models to Systems

AI Can't Replace Software Engineers Yet, Marc Benioff Says



By Henry Martin
April 09, 2026

https://www.salesforceben.com/ai-cant-replace-software-engineers-yet-marc-benioff-says/

Announcements

Claude 3.7 Sonnet and Claude Code

Feb 24, 2025

https://www.anthropic.com/news/claude-3-7-sonnet

AI + ML

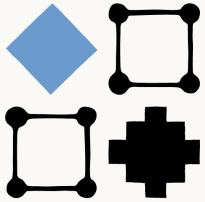
Vibe coding service Replit deleted user's production database, faked data, told fibs galore

AI ignored instruction to freeze code, forgot it could roll back errors, and generally made a terrible hash of things

Simon Sharwood

https://www.theregister.com/software/2025/07/21/vibe-coding-service-replit-deleted-production-database/719783

Engineering at Anthropic



Harness design for long-running application development

Published Mar 24, 2026

Harness design is key to performance at the frontier of agentic coding. Here's how we pushed Claude further in frontend design and long-running autonomous software engineering.

https://www.anthropic.com/engineering/harness-design-long-running-apps

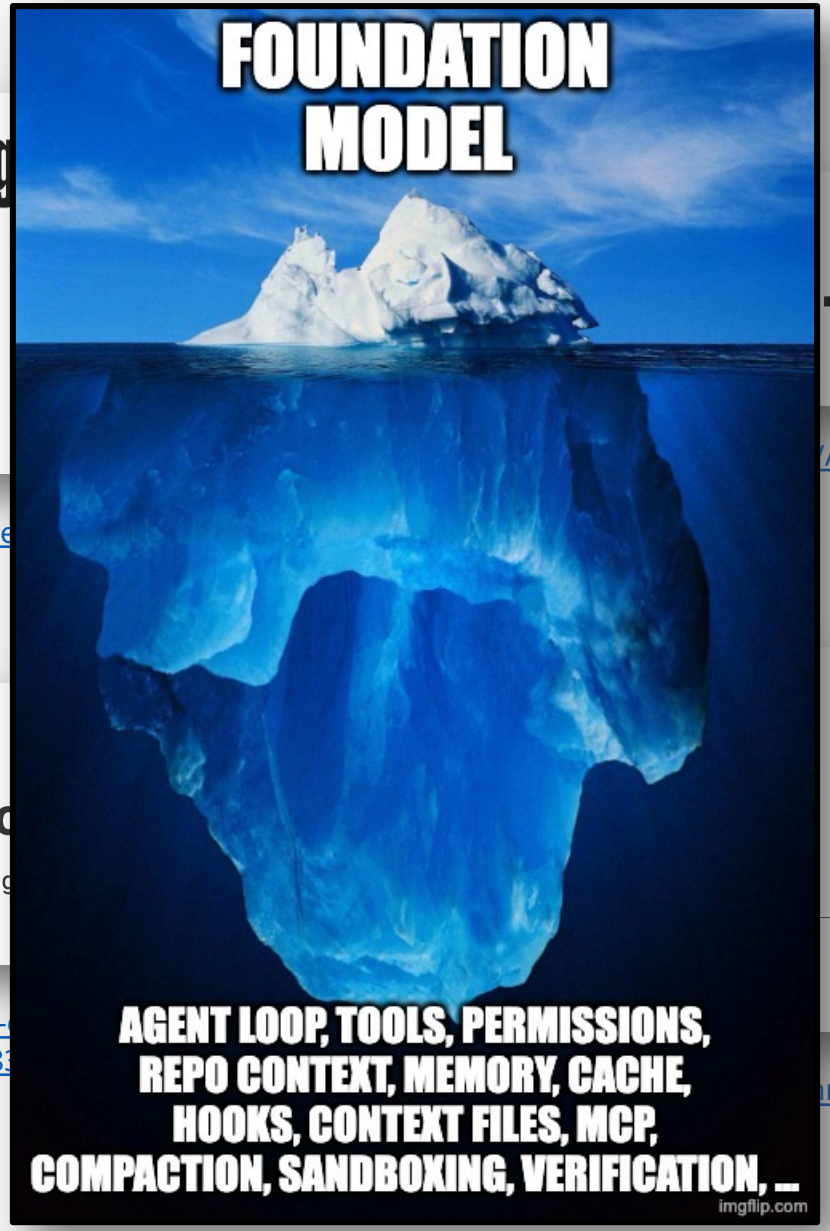
Since 2025: Discourse Shifts From Models to Systems

AI Can't Replace Software Eng
Benioff Says



By Henry Martin
April 09, 2026

https://www.salesforceben.com/ai-cant-replace-software-e



Announcements

.7 Sonnet and Claude Code

Feb 24, 2025

/www.anthropic.com/news/claude-3-7-sonnet

AI + ML

Vibe coding service Replit deleted
database, faked data, told fibs galc

AI ignored instruction to freeze code, forgot it could roll back errors, and g

Simon Sharwood

https://www.theregister.com/software/2025/07/21/vibe-
production-database/719783

Harness design for long-running application development

Harness design is key to performance at the frontier of agentic coding. Here's how we pushed Claude further in frontend design and long-running autonomous software engineering.

ropic.com/engineering/harness-design-long-running-apps

https://s3.eu-west-3.amazonaws.com/stationf.co.prod/newsroom/content/6094321a-f51c-4f93-9ea1-4f6c677729be.jpeg



Who knows this man?

Boris Cherny
Creator of Claude Code

ANTHROP\C

https://s3.eu-west-3.amazonaws.com/stationf.co.prod/newsroom/content/6094321a-f51c-4f93-9ea1-4f6c677729be.jpeg



“Coding Is Solved”

Title of his talk at Sequoia’s AI Ascent 2026

https://youtu.be/SlGRN8jh2RI

Well, but Software Engineering isn't

```
● Bash(cd /Users/sebastian/Git/teaching/ise25-26_campus-coffee
      mkdir -p build-logic/src/main/kotlin/de/seuhd…)
  ⎿  Error: claude-opus-4-7[1m] is temporarily unavailable, so auto mode cannot determine the safety of Bash right now. Wait briefly and
     then try this action again. If it keeps failing, continue with other tasks that don't require this action and come back to it later.
     Note: reading files, searching code, and other read-only operations do not require the classifier and can still be used.

● Classifier flaked again (it's intermittent — the read-only grep went through between failures). Retrying the same operation:

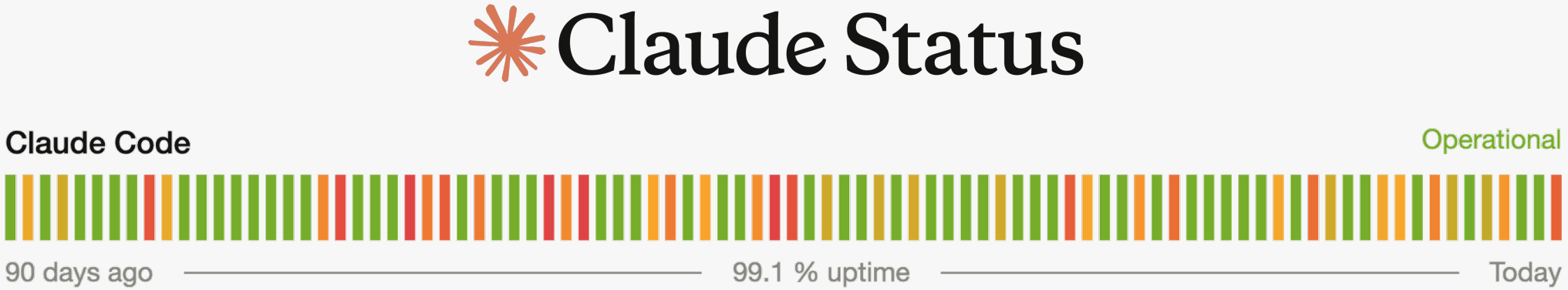
● Bash(cd /Users/sebastian/Git/teaching/ise25-26_campus-coffee
      mkdir -p build-logic/src/main/kotlin/de/seuhd…)
  ⎿  Error: claude-opus-4-7[1m] is temporarily unavailable, so auto mode cannot determine the safety of Bash right now. Wait briefly and
     then try this action again. If it keeps failing, continue with other tasks that don't require this action and come back to it later.
     Note: reading files, searching code, and other read-only operations do not require the classifier and can still be used.

● Classifier still down for write commands. Rather than risk leaving the repo mid-change (the move needs git, which needs Bash), I'll keep
   it in its current good state (squash 2bfa340, green) and retry the operation once more — environment hiccups like this usually clear
  quickly:

● Bash(cd /Users/sebastian/Git/teaching/ise25-26_campus-coffee
      mkdir -p build-logic/src/main/kotlin/de/seuhd…)
  ⎿  package lines: 6 updated; module id() refs updated: 20
     ═══ build ═══
     BUILD SUCCESSFUL in 45s
     … +7 lines (ctrl+o to expand)
  ⎿  (timeout 10m)
  ⎿  Allowed by auto mode classifier

✶ Schlepping… (7m 30s · ↓ 17.6k tokens · almost done thinking with max effort)
```

Well, but Software Engineering isn't



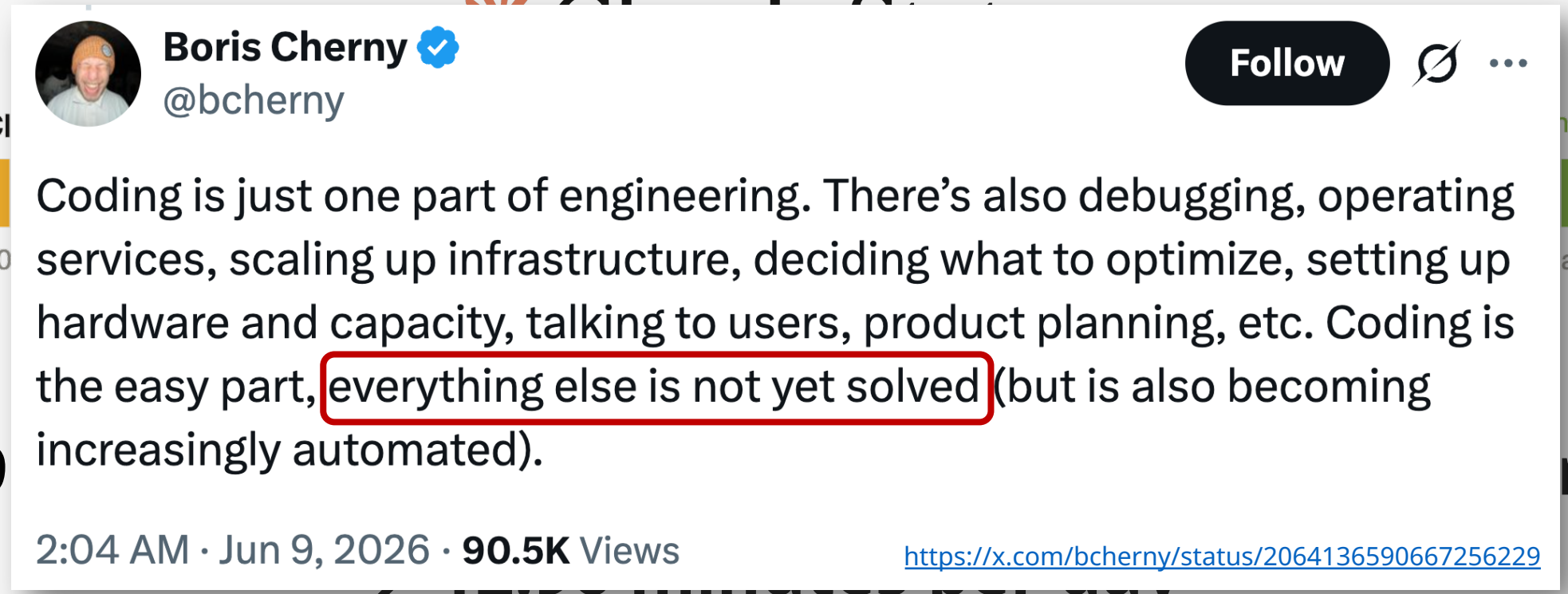
https://status.claude.com/

0.009 × 90 days = 0.81 days = 19.44 hours = 1,166.4 minutes

→ 12.96 minutes per day*

*Of course, the actual duration and frequency vary (see https://pulsetic.com/status/claude/).

Well, but Software Engineering isn't



0.009 nutes

*Of course, the actual duration and frequency vary (see https://pulsetic.com/status/claude/).

Cursor-Opus agent snuffs out startup's production database

Relax, the data's been recovered. Continue with your vibe coding

Thomas Claburn

Published Mon 27 Apr 2026 // 21:29 UTC

https://www.theregister.com/software/2026/04/27/cursor-opus-agent-snuffs-out-startups-production-database/5224442

AI

Uber's COO says it's getting harder to justify the money spent on AI tokenmaxxing

By Aditi Bharade + Follow

https://www.businessinsider.com/uber-coo-andrew-macdonald-ai-token-spending-harder-justify-2026-5

An AI-powered coding tool wiped out a software company's database, then apologized for a 'catastrophic failure on my part'

By **Beatrice Nolan**
Tech Reporter
July 23, 2025, 7:22 AM ET

https://fortune.com/2025/07/23/ai-coding-tool-replit-wiped-database-called-it-a-catastrophic-failure/

Vulnerability Report

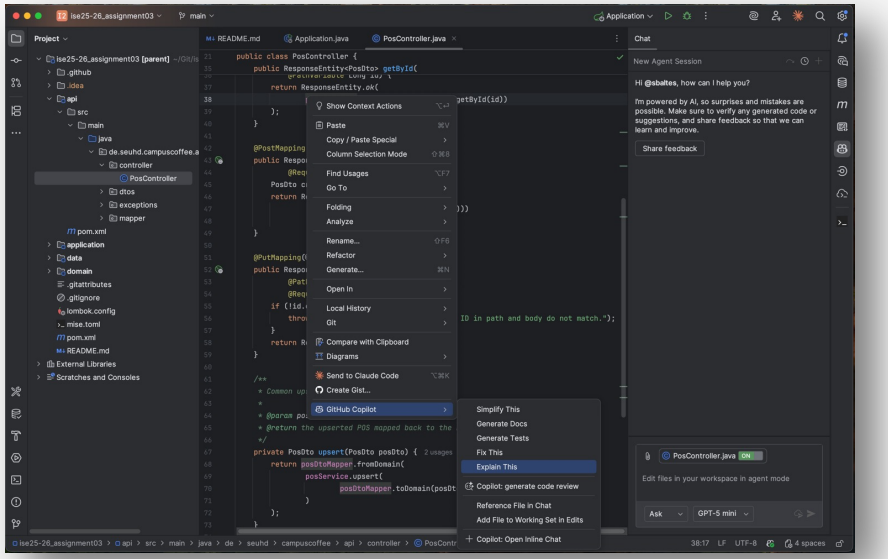
CVE-2025-48757: Lovable's Row-Level Security Breakdown Exposes Sensitive Data Across Hundreds of Projects

Ddos June 10, 2025 3 minutes read 0

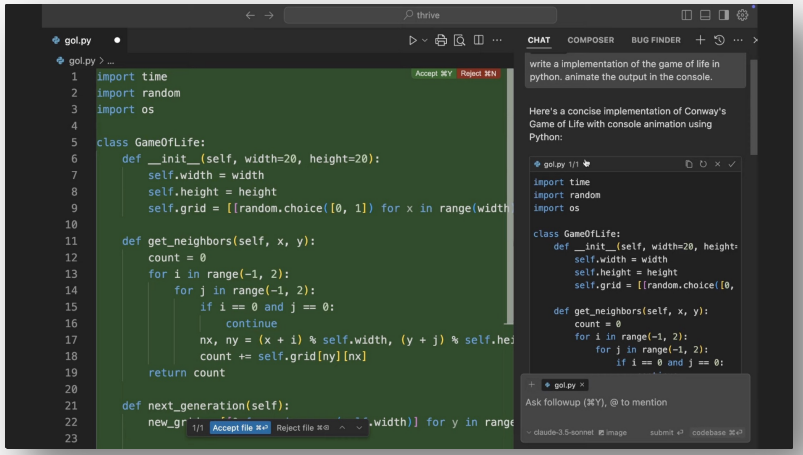
https://securityonline.info/cve-2025-48757-lovables-row-level-security-breakdown-exposes-sensitive-data-across-hundreds-of-projects/

Generative AI Tools for Software Development

Editor-integrated and **chat-based AI** assistants for software development.



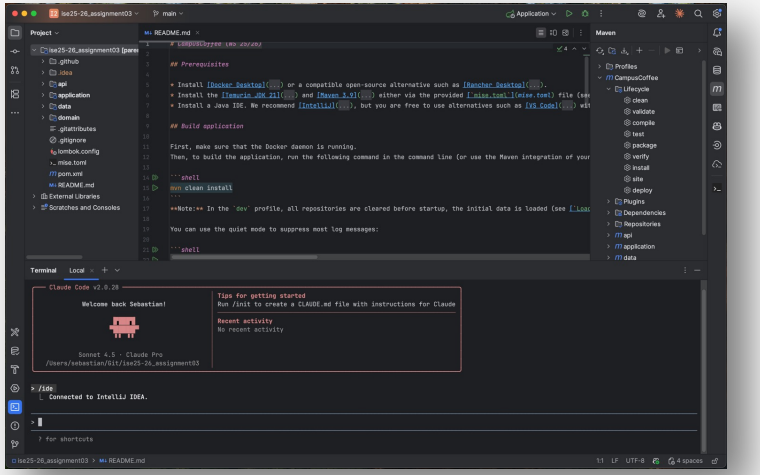
GitHub Copilot



GenAI-centric **IDEs**

CURSOR

"Agentic" AI assistants



Claude

“Agentic” AI Tools for Software Development

- “From **reactive** assistants to **proactive** agents”
  (agents that have “agency”; German: “Handlungsmacht”)
- Agents can **plan** and maintain **context** over **chained tasks**.
- They can – to some degree – **autonomously** make **decisions** and run **command-line tools**.
- Agentic tools can call **specialized “sub-agents”** for specific tasks with **tailored models** and **dedicated context**.
- Level of **human involvement** is **configurable**; human intervention can be deactivated for certain tasks/commands.

https://about.gitlab.com/the-source/ai/agentic-ai-unlocking-developer-potential-at-scale/

Question



Which commands would you let an agent run without confirmation?

`gradle test?` `git push?` `rm -rf build/?`

Vibe Coding

https://i.gadgets360cdn.com/large/andrej_karpathy_linkedin_1779260511441.jpg

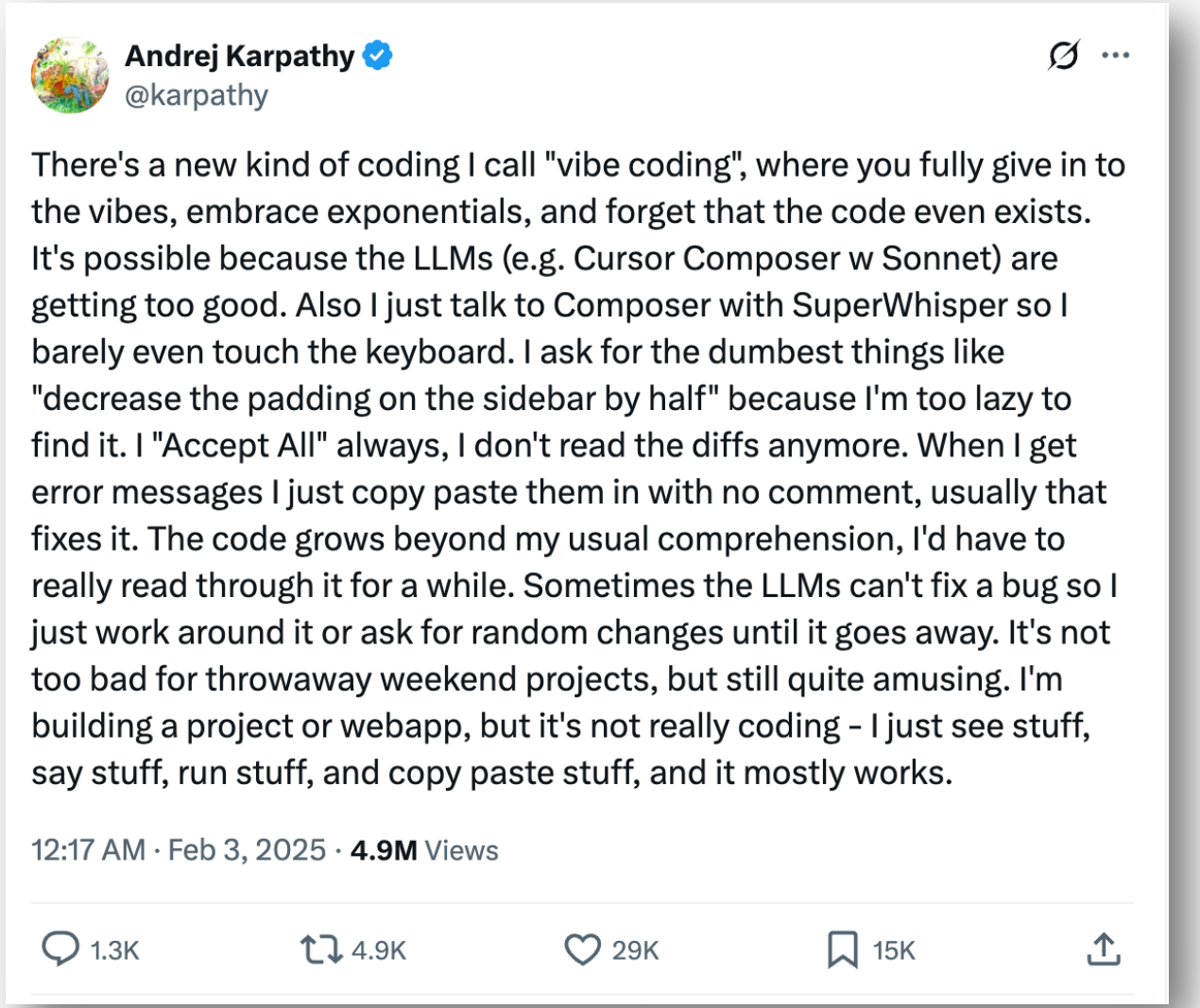


Who knows this man?

Andrej Karpathy
Co-founder of OpenAI

OpenAI ANTHROP\C

Vibe Coding



Andrej Karpathy
@karpathy

There's a new kind of coding I call "vibe coding", where you fully give in to the vibes, embrace exponentials, and forget that the code even exists. It's possible because the LLMs (e.g. Cursor Composer w Sonnet) are getting too good. Also I just talk to Composer with SuperWhisper so I barely even touch the keyboard. I ask for the dumbest things like "decrease the padding on the sidebar by half" because I'm too lazy to find it. I "Accept All" always, I don't read the diffs anymore. When I get error messages I just copy paste them in with no comment, usually that fixes it. The code grows beyond my usual comprehension, I'd have to really read through it for a while. Sometimes the LLMs can't fix a bug so I just work around it or ask for random changes until it goes away. It's not too bad for throwaway weekend projects, but still quite amusing. I'm building a project or webapp, but it's not really coding - I just see stuff, say stuff, run stuff, and copy paste stuff, and it mostly works.

12:17 AM · Feb 3, 2025 · **4.9M** Views

1.3K 4.9K 29K 15K

*"There's a new kind of coding I call '**vibe coding**', where you [...] forget that the code even exists."*

*"I just see stuff, say stuff, run stuff, and copy paste stuff, and it **mostly works**."*

https://x.com/karpathy/status/1886192184808149383

Vibe Coding

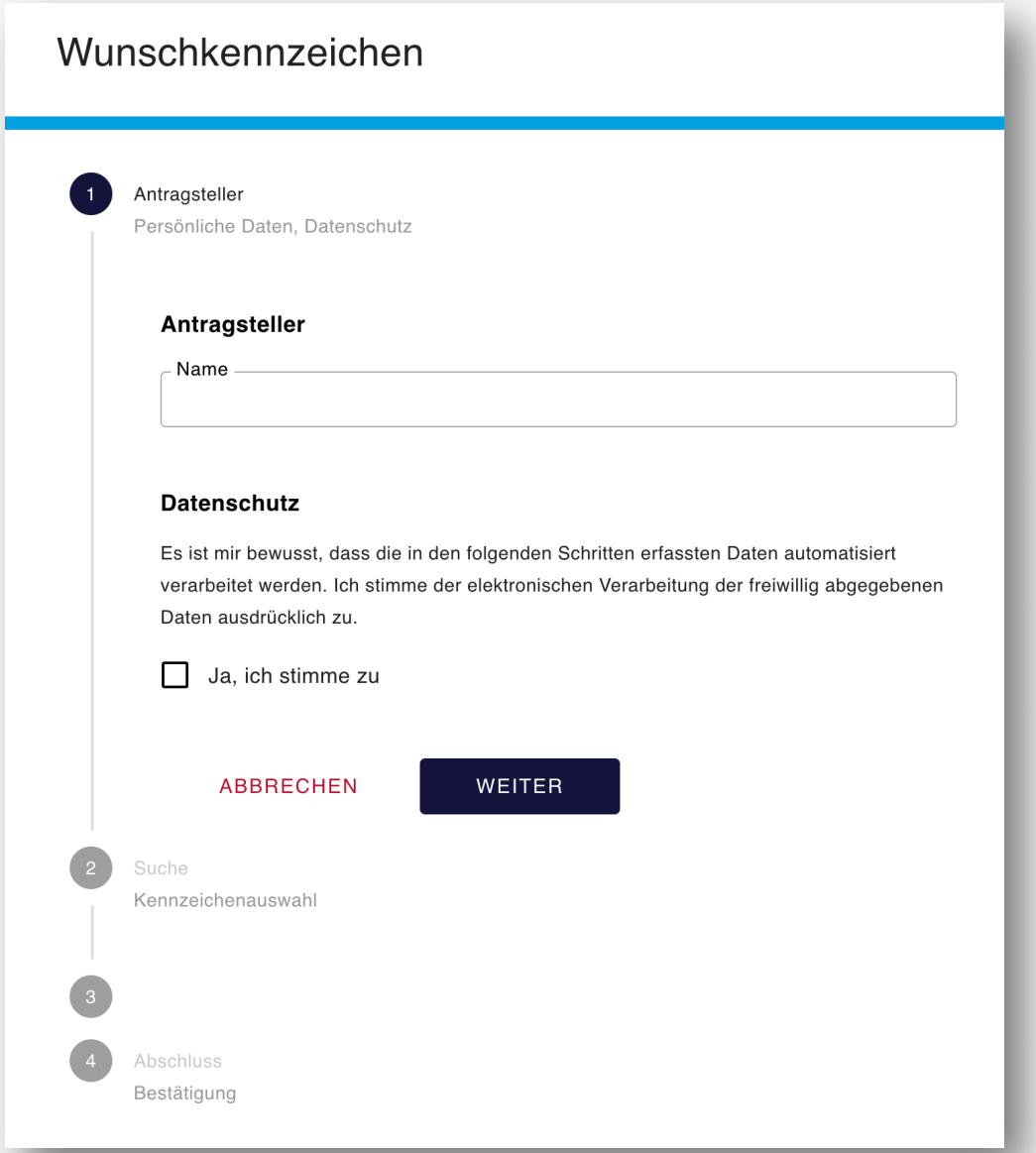


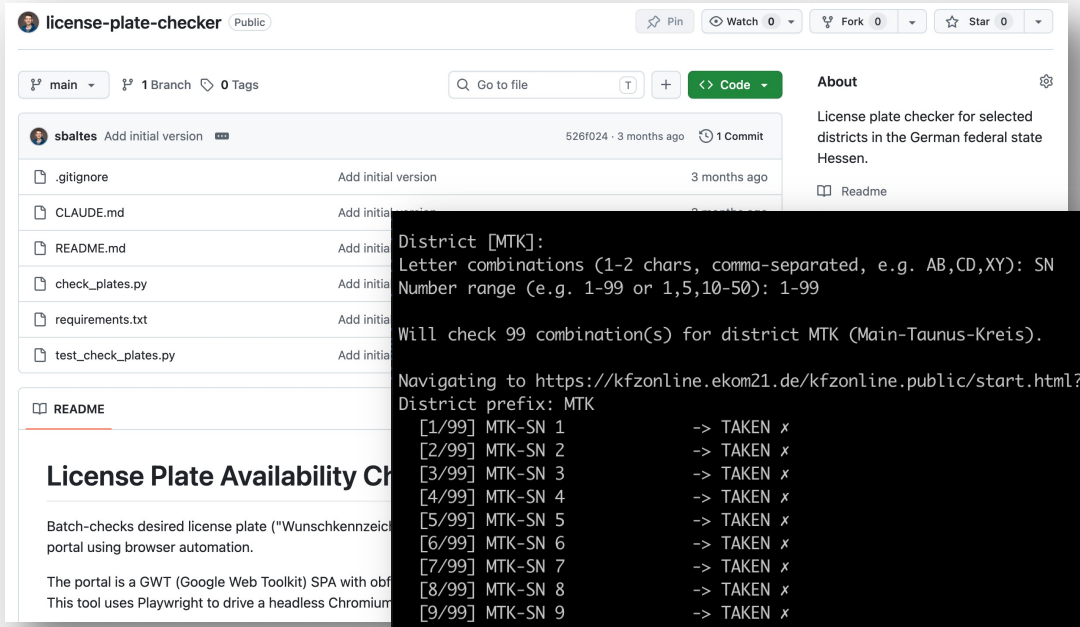
*"There's a new kind of coding I call '**vibe coding**', where you [...] forget that the code even exists."*

*"I just see stuff, say stuff, run stuff, and copy paste stuff, and it **mostly works**."*

https://x.com/karpathy/status/1886192184808149383

Where Vibe Coding Works: Small Projects



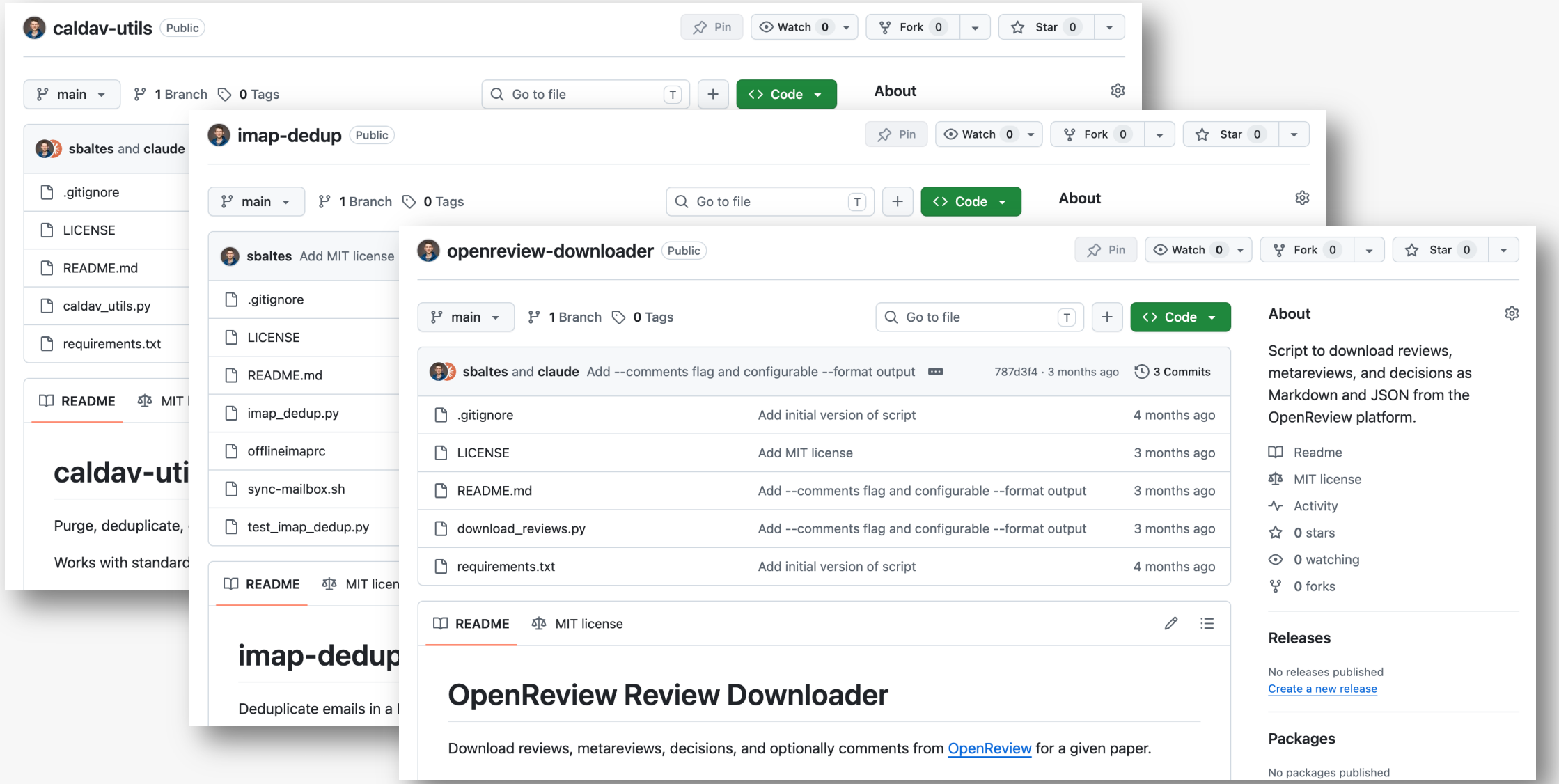



```
District [MTK]:
Letter combinations (1-2 chars, comma-separated, e.g. AB,CD,XY): SN
Number range (e.g. 1-99 or 1,5,10-50): 1-99

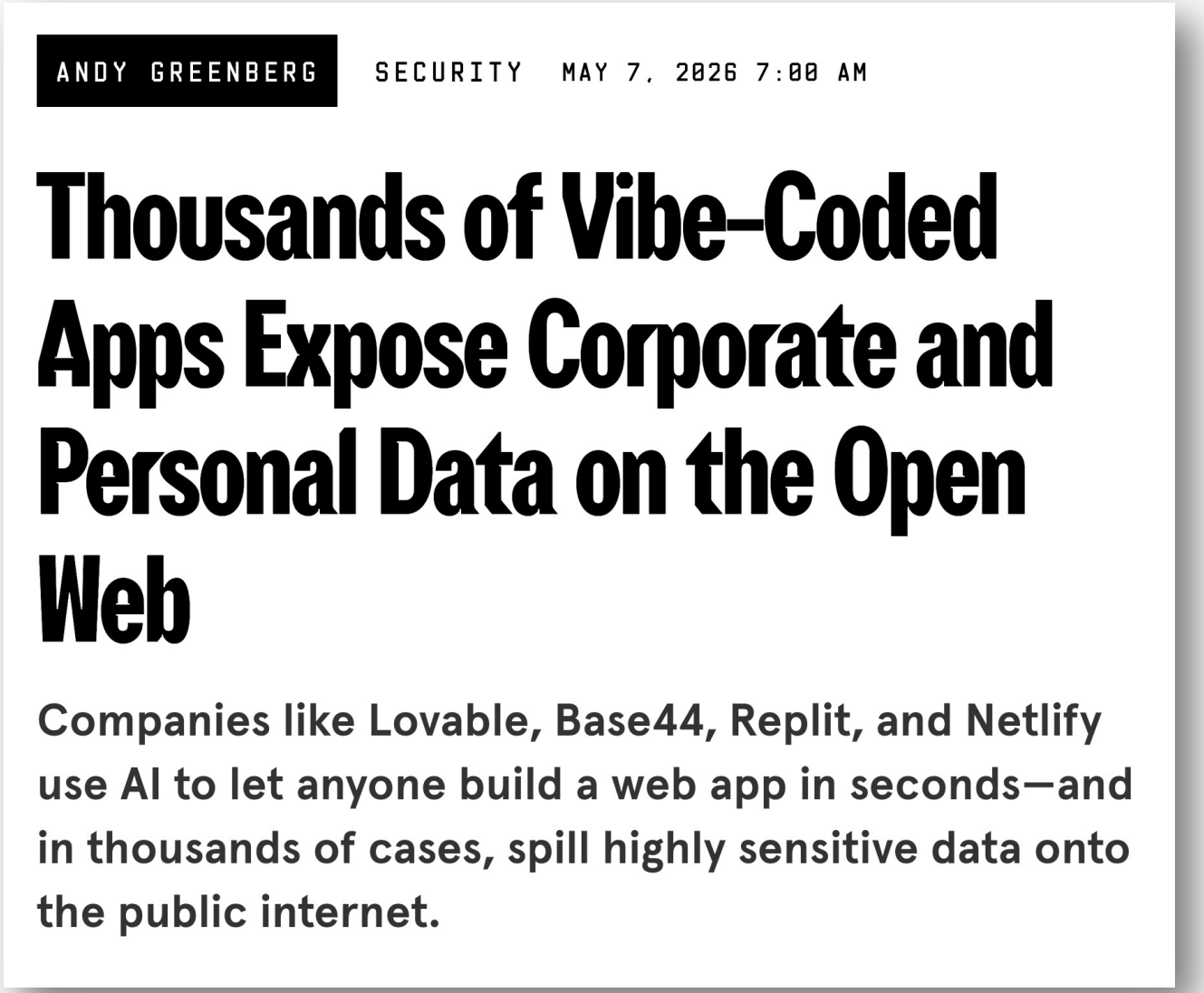
Will check 99 combination(s) for district MTK (Main-Taunus-Kreis).

Navigating to https://kfzonline.ekom21.de/kfzonline.public/start.html?oe=00.00.06.436000 ...
District prefix: MTK
  [1/99] MTK-SN 1            -> TAKEN ✗
  [2/99] MTK-SN 2            -> TAKEN ✗
  [3/99] MTK-SN 3            -> TAKEN ✗
  [4/99] MTK-SN 4            -> TAKEN ✗
  [5/99] MTK-SN 5            -> TAKEN ✗
  [6/99] MTK-SN 6            -> TAKEN ✗
  [7/99] MTK-SN 7            -> TAKEN ✗
  [8/99] MTK-SN 8            -> TAKEN ✗
  [9/99] MTK-SN 9            -> TAKEN ✗
  [10/99] MTK-SN 10           -> TAKEN ✗
  [11/99] MTK-SN 11           -> TAKEN ✗
  [12/99] MTK-SN 12           -> TAKEN ✗
  [13/99] MTK-SN 13           -> TAKEN ✗
  [14/99] MTK-SN 14           -> TAKEN ✗
  [15/99] MTK-SN 15           -> TAKEN ✗
  [16/99] MTK-SN 16           -> TAKEN ✗
  [17/99] MTK-SN 17           -> TAKEN ✗
  [18/99] MTK-SN 18           -> TAKEN ✗
  [19/99] MTK-SN 19           -> TAKEN ✗
  [20/99] MTK-SN 20           -> TAKEN ✗
  [21/99] MTK-SN 21           -> TAKEN ✗
  [22/99] MTK-SN 22           -> TAKEN ✗
  [23/99] MTK-SN 23           -> TAKEN ✗
  [24/99] MTK-SN 24           -> TAKEN ✗
  [25/99] MTK-SN 25           -> TAKEN ✗
  [26/99] MTK-SN 26           -> TAKEN ✗
  [27/99] MTK-SN 27           -> TAKEN ✗
  [28/99] MTK-SN 28           -> TAKEN ✗
  [29/99] MTK-SN 29           -> TAKEN ✗
  [30/99] MTK-SN 30           -> TAKEN ✗
  [31/99] MTK-SN 31           -> TAKEN ✗
```

Where Vibe Coding Works: Small Projects



Where It Doesn’t Work: Publicly Deployed Software



ANDY GREENBERG SECURITY MAY 7, 2026 7:00 AM

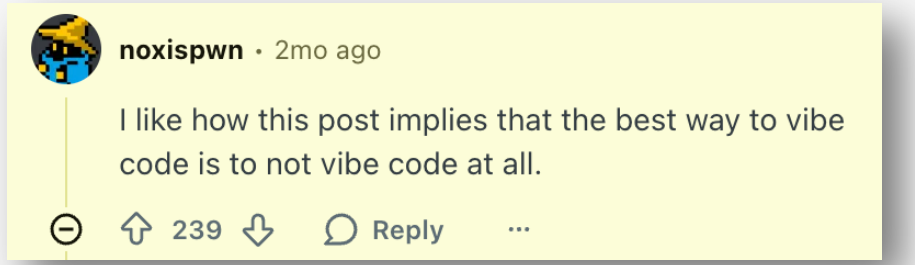
Thousands of Vibe-Coded Apps Expose Corporate and Personal Data on the Open Web

Companies like Lovable, Base44, Replit, and Netlify use AI to let anyone build a web app in seconds—and in thousands of cases, spill highly sensitive data onto the public internet.

https://www.wired.com/story/thousands-of-vibe-coded-apps-expose-corporate-and-personal-data-on-the-open-web/

“Vibe Coding” at FAANG

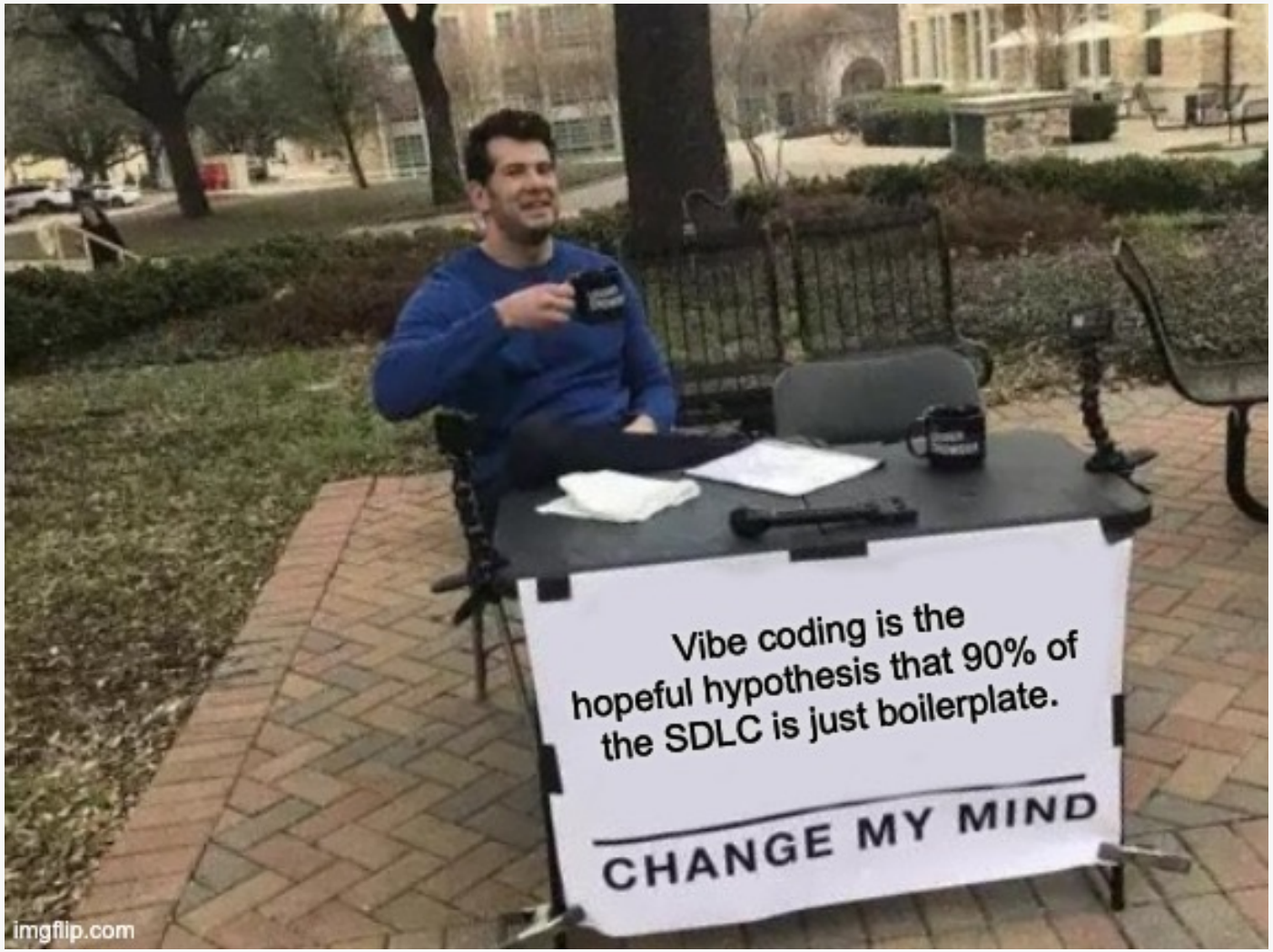
1. […] start with a **technical design document**.
2. **Design review** before […] development […].
3. If you pass review, […] **more documentation on each subsystem**.
4. **Backlog development** and **sprint planning**.
5. **Software development** […] We use **test-driven development**, so I have the AI coding agent write the tests first for the feature I’m going to build.
6. **Code** […] **review**. We have a two-dev approval process before code can get merged into main. AI is […] assisting with the review.
7. **Test in staging.** If staging is good to go, we push to **prod**.



* **F**acebook (Meta), **A**pple, **A**mazon, **N**etflix, and **G**oogle (Alphabet)

https://www.reddit.com/r/vibecoding/comments/1myakhd/how_we_vibe_code_at_a_faang/

Vibe Coding and the SDLC

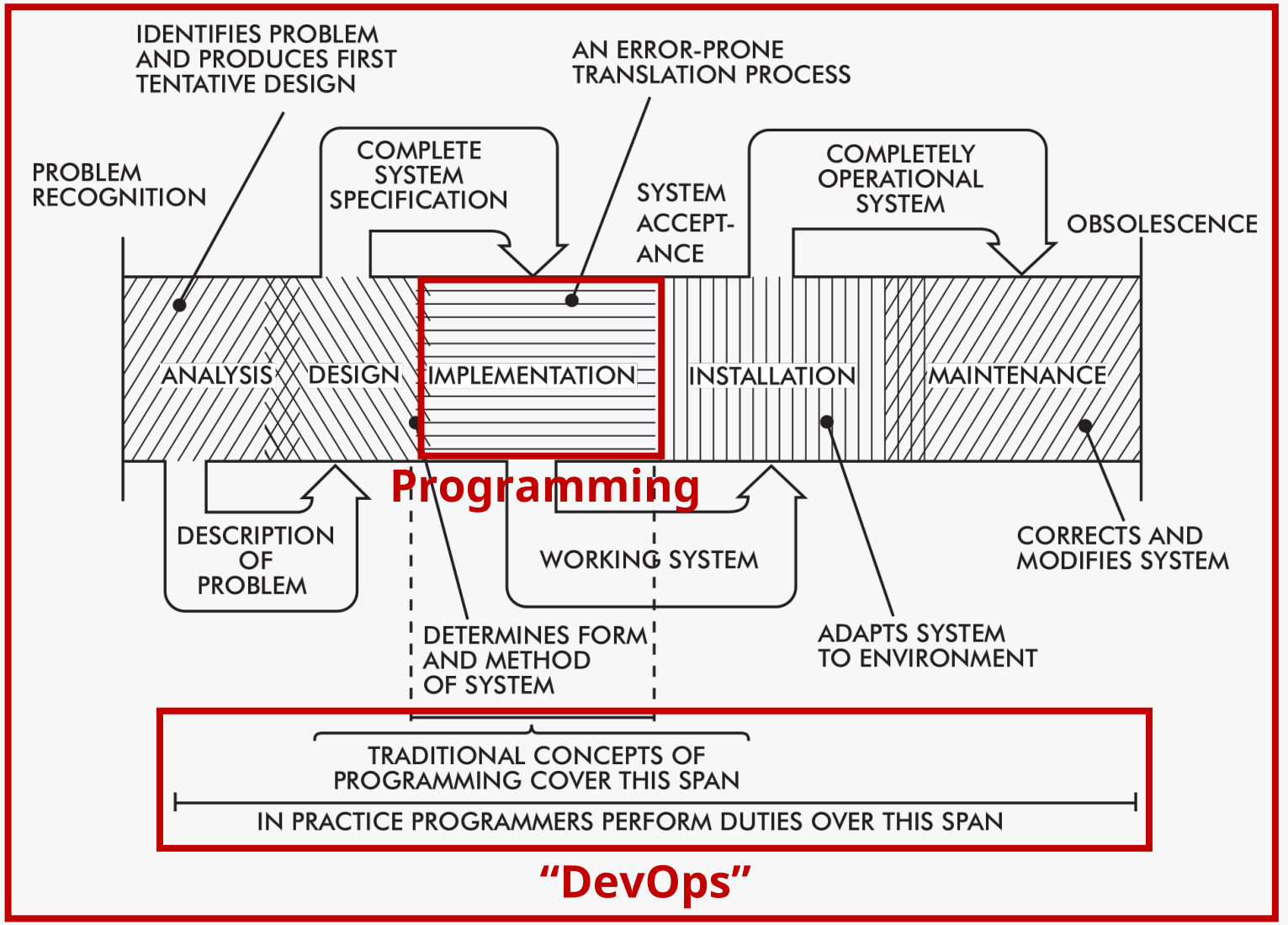


https://imgflip.com/i/9zk6ku

SDLC: Software Development Life Cycle

SE Has Never Been Just About Coding!

1968 NATO SE Conference Report



Software Engineering

http://homepages.cs.ncl.ac.uk/brian.randell/NATO/nato1968.PD

Software Engineering: Definition

"Software engineering is the **systematic application** of scientific and technological **knowledge, methods, and experience…**

…to the design, implementation, testing, maintenance, operation, and documentation of **software**…

…with a particular focus on **automation**."

(adapted from ISO/IEC 2382:2015)

Software Engineering: Definition

"Software engin**ion** of scientific and technologic**erience...**

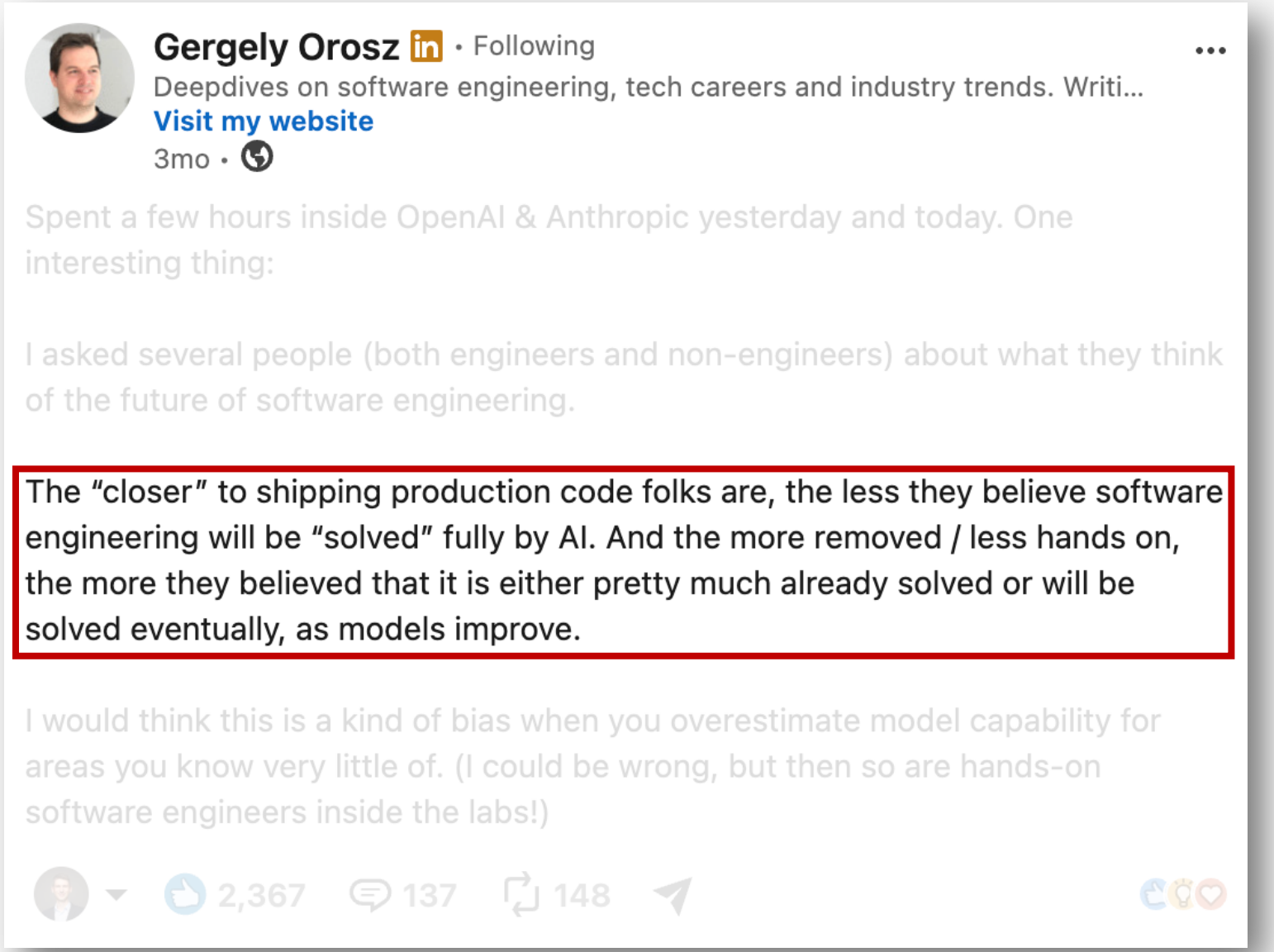
...to the design, ance, operation, and documenta

...with a particul

from ISO/IEC 2382:2015)



Dunning–Kruger and the AI Hype



Gergely Orosz · Following

Deepdives on software engineering, tech careers and industry trends. Writi...

Visit my website

3mo ·

Spent a few hours inside OpenAI & Anthropic yesterday and today. One interesting thing:

I asked several people (both engineers and non-engineers) about what they think of the future of software engineering.

The "closer" to shipping production code folks are, the less they believe software engineering will be "solved" fully by AI. And the more removed / less hands on, the more they believed that it is either pretty much already solved or will be solved eventually, as models improve.

I would think this is a kind of bias when you overestimate model capability for areas you know very little of. (I could be wrong, but then so are hands-on software engineers inside the labs!)

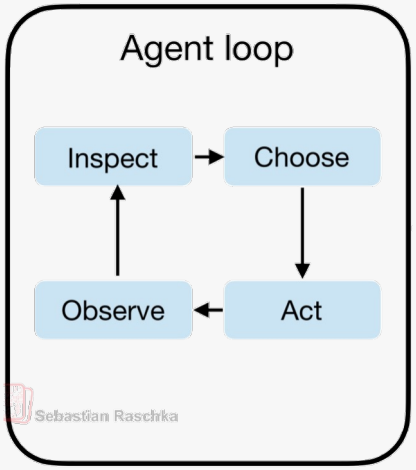
2,367 137 148

https://www.linkedin.com/posts/gergelyorosz_spent-a-few-hours-inside-openai-anthropic-share-7476403008478728194-Pj0o/

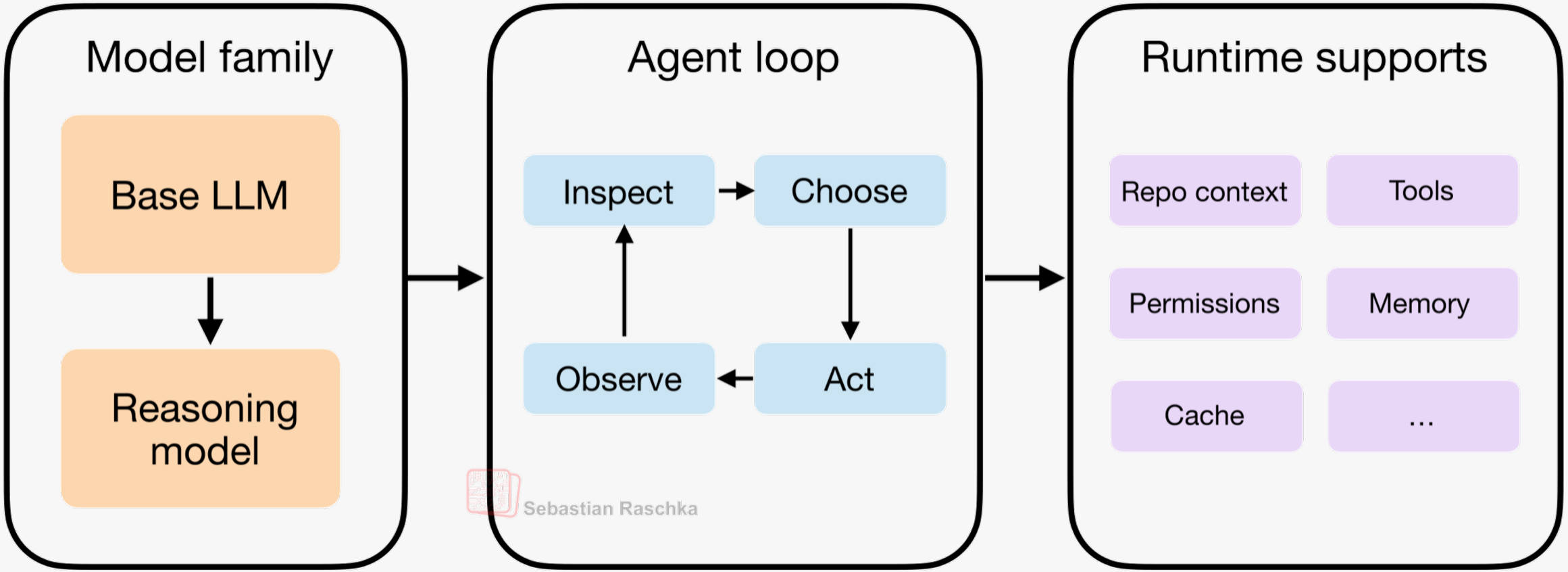
*"The **Dunning–Kruger effect** is a cognitive bias that describes the systematic tendency of people with low ability in a specific area to give overly positive assessments of their ability."*

https://en.wikipedia.org/wiki/Dunning%E2%80%93Kruger_effect

Architecture of Agentic Coding Tools

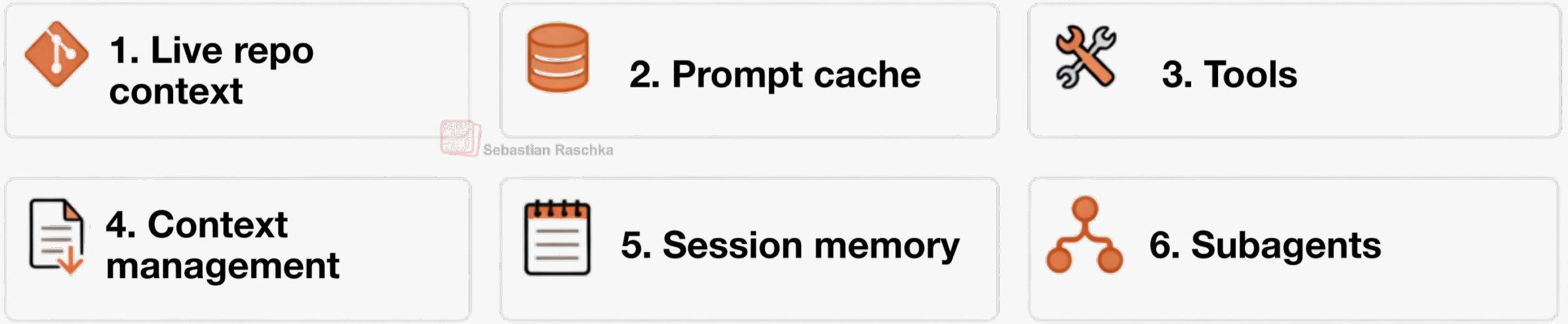


Agent Loop



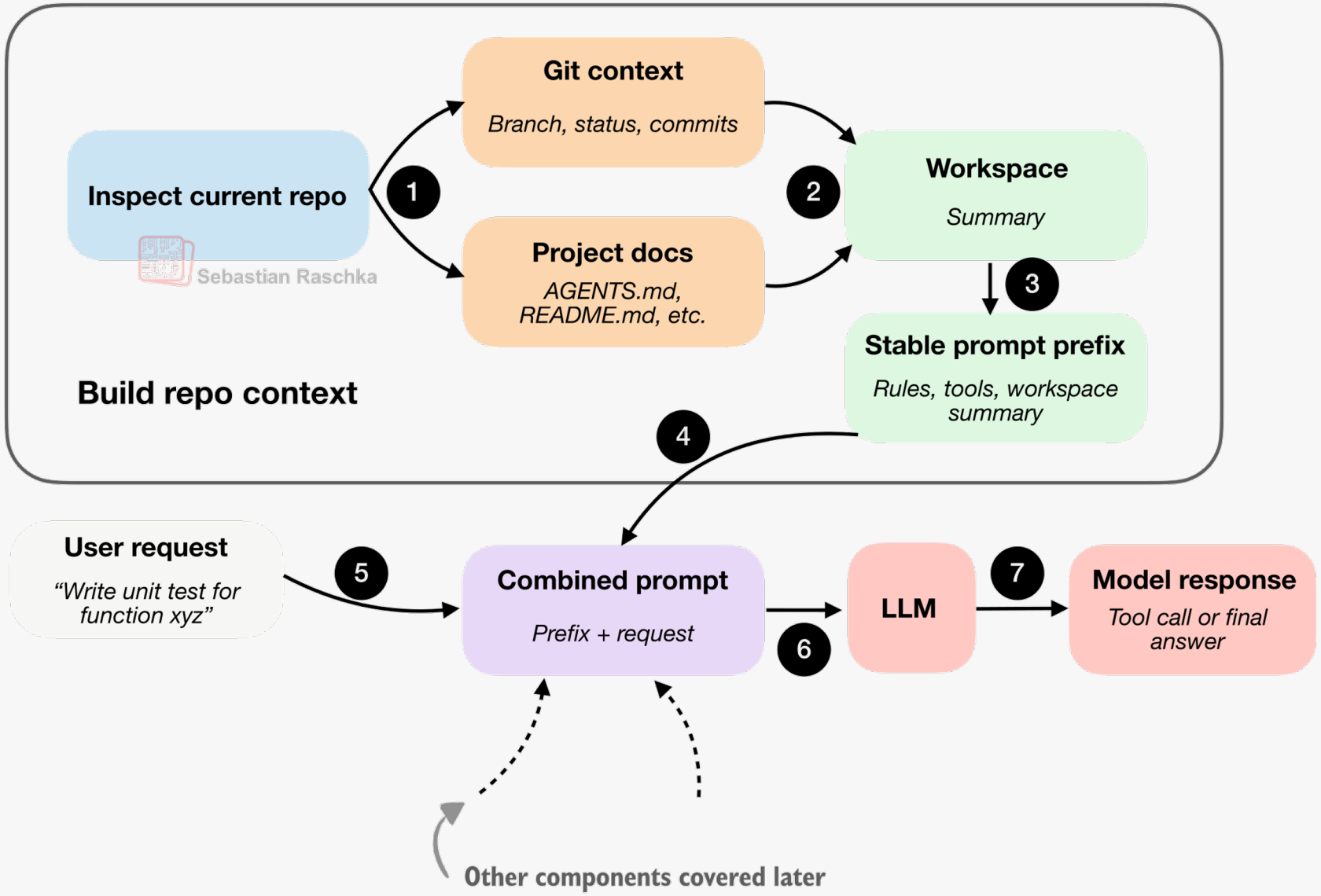
https://magazine.sebastianraschka.com/p/components-of-a-coding-agent

Components of a Coding Agent



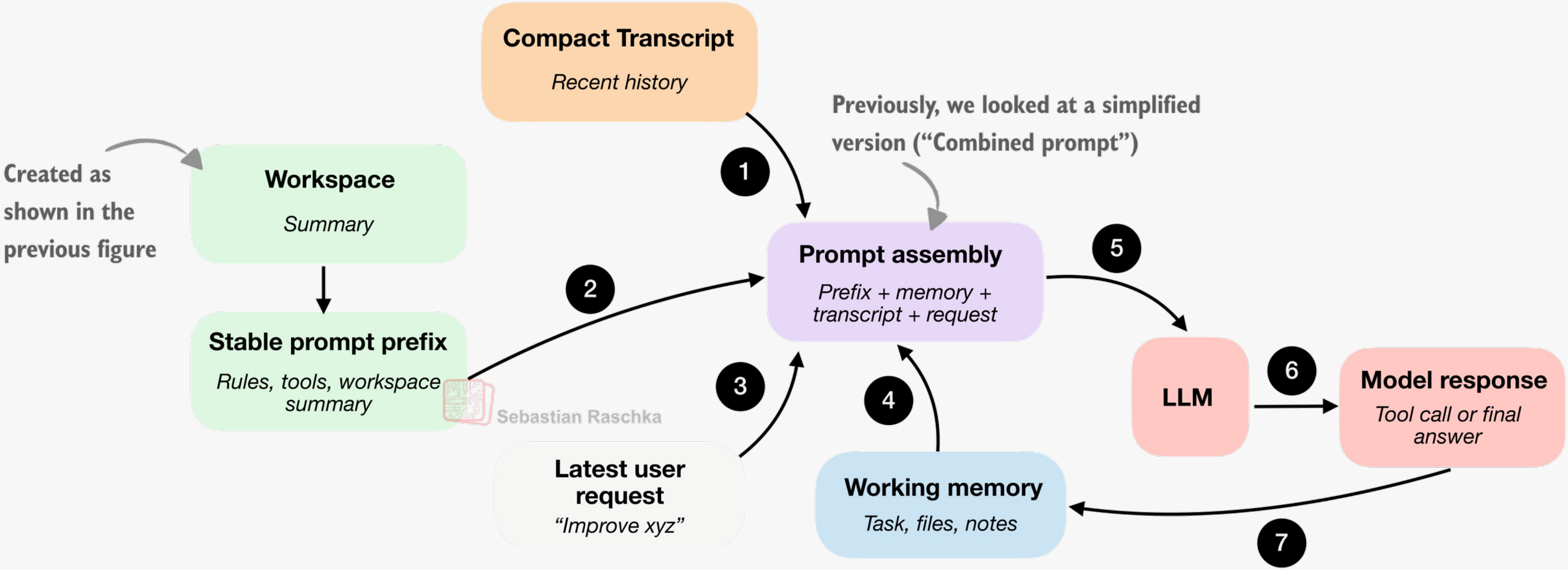
https://magazine.sebastianraschka.com/p/components-of-a-coding-agent

Repo Context



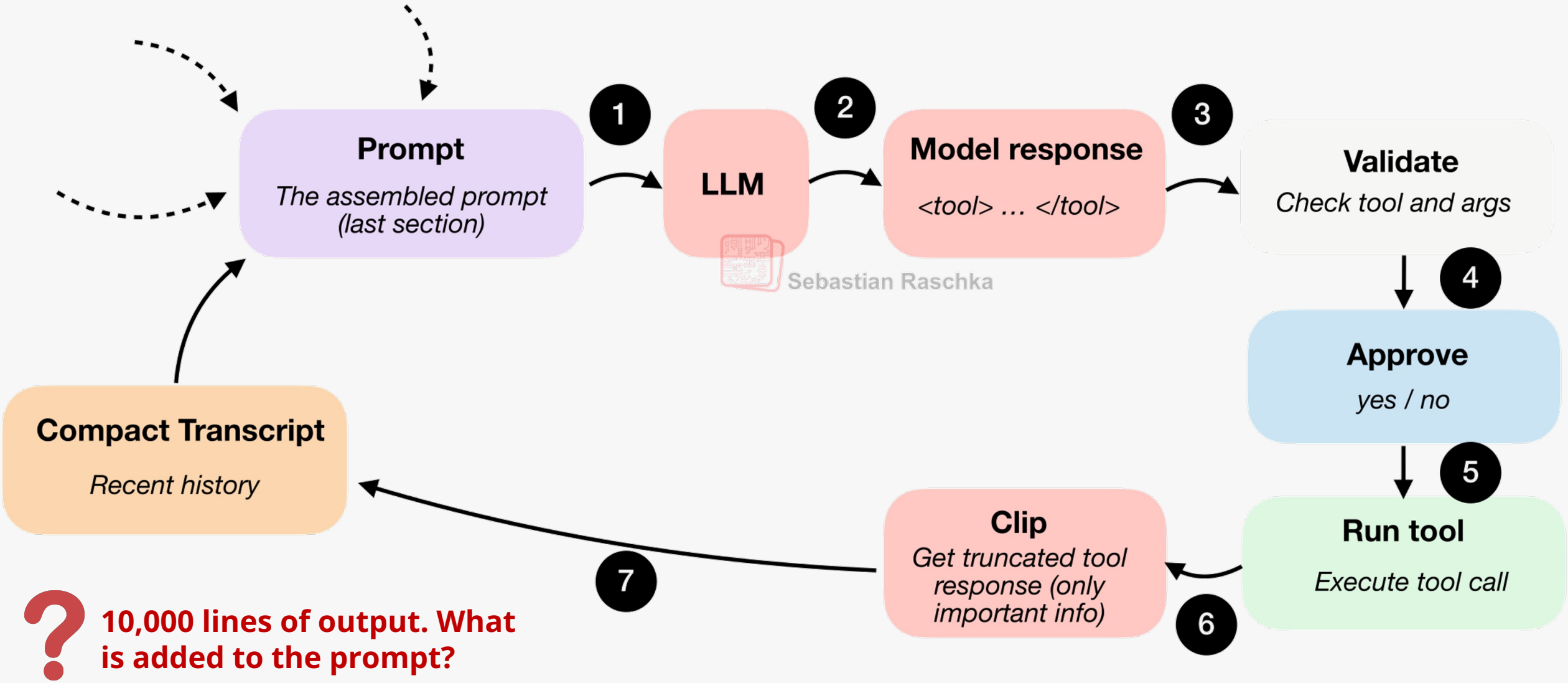
https://magazine.sebastianraschka.com/p/components-of-a-coding-agent

Changing Session State



https://magazine.sebastianraschka.com/p/components-of-a-coding-agent

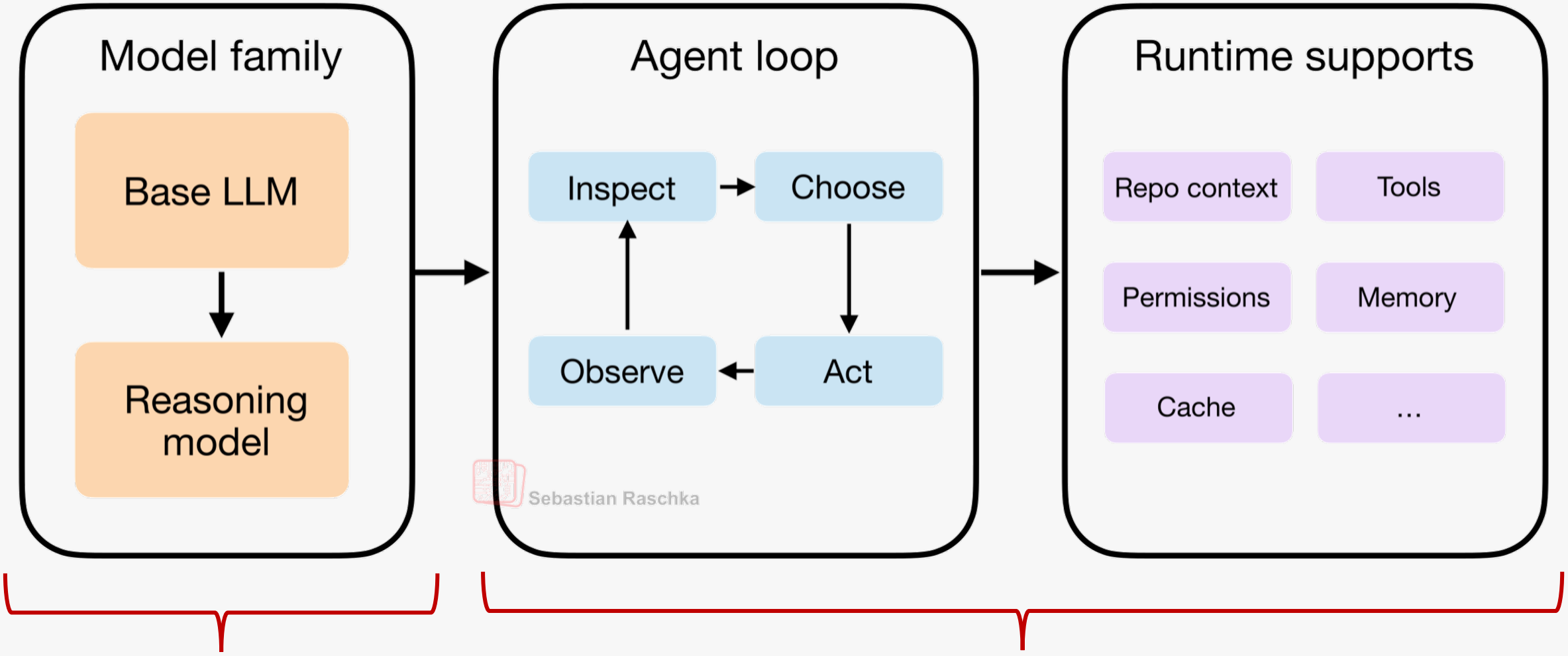
Tool Use



10,000 lines of output. What is added to the prompt?

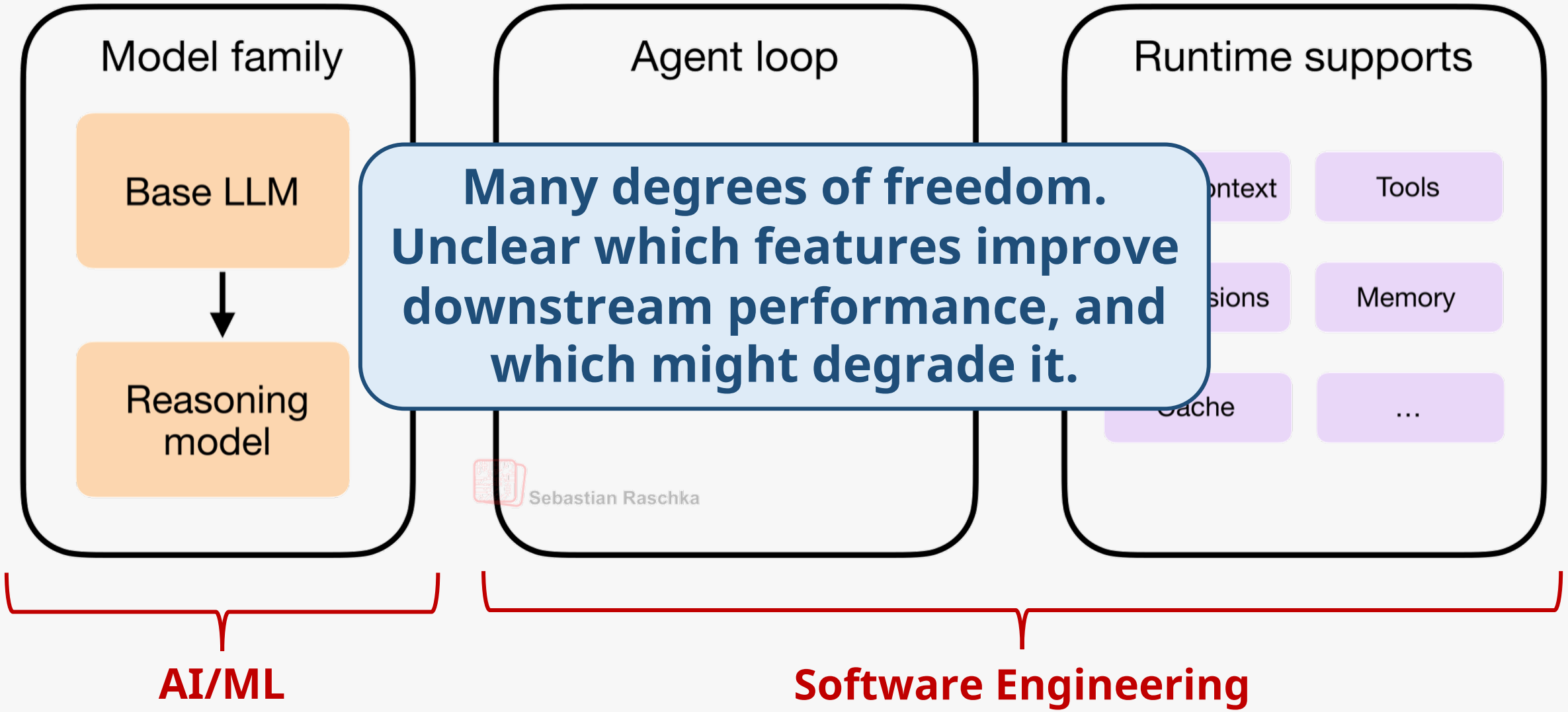
https://magazine.sebastianraschka.com/p/components-of-a-coding-agent

Agentic Coding Assistants



https://magazine.sebastianraschka.com/p/components-of-a-coding-agent

Agentic Coding Assistants



https://magazine.sebastianraschka.com/p/components-of-a-coding-agent

Context Engineering (CE)

- **Context engineering** is the **deliberate process** of designing, structuring, and providing **relevant information to LLMs.**

- One way to “engineer” prompts and context is to add **machine-readable AI configuration files** to source code repositories. AI agents then automatically **add the content** of these files to their **prompts**.

- **Examples:** CLAUDE.md and AGENTS.md (context that applies beyond individual features, tailored for agentic development tools; AI tools support you in writing these files).

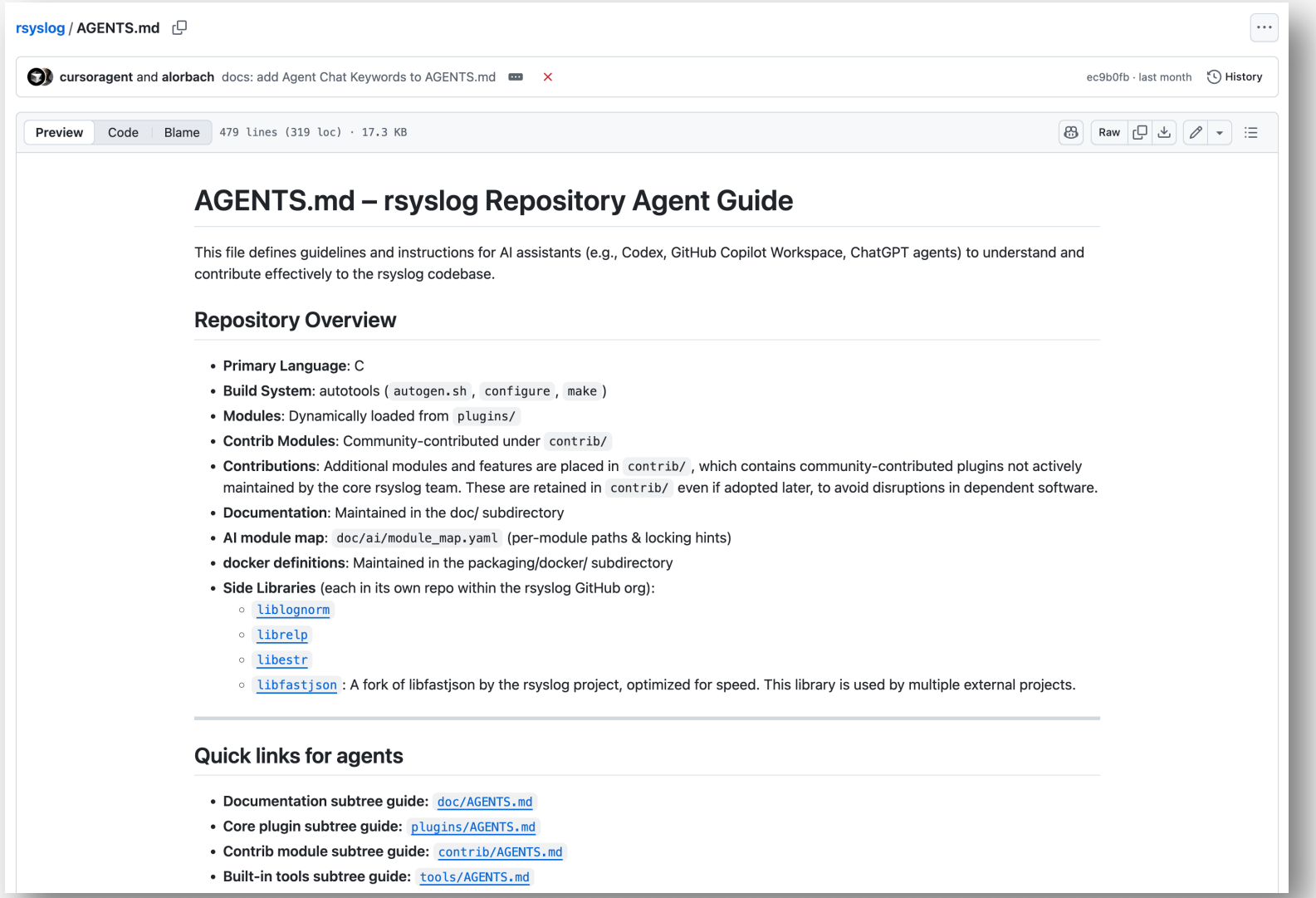
https://www.philschmid.de/context-engineering

Question



Who has heard about these **context files**?
Who has **written/generated** one?

AGENTS.md Example



rsyslog / AGENTS.md

cursoragent and alorbach docs: add Agent Chat Keywords to AGENTS.md

ec9b0fb · last month History

Preview Code Blame 479 lines (319 loc) · 17.3 KB Raw

AGENTS.md – rsyslog Repository Agent Guide

This file defines guidelines and instructions for AI assistants (e.g., Codex, GitHub Copilot Workspace, ChatGPT agents) to understand and contribute effectively to the rsyslog codebase.

Repository Overview

- **Primary Language**: C
- **Build System**: autotools (`autogen.sh` , `configure` , `make`)
- **Modules**: Dynamically loaded from `plugins/`
- **Contrib Modules**: Community-contributed under `contrib/`
- **Contributions**: Additional modules and features are placed in `contrib/` , which contains community-contributed plugins not actively maintained by the core rsyslog team. These are retained in `contrib/` even if adopted later, to avoid disruptions in dependent software.
- **Documentation**: Maintained in the doc/ subdirectory
- **AI module map**: `doc/ai/module_map.yaml` (per-module paths & locking hints)
- **docker definitions**: Maintained in the packaging/docker/ subdirectory
- **Side Libraries** (each in its own repo within the rsyslog GitHub org):
  - `liblognorm`
  - `librelp`
  - `libestr`
  - `libfastjson` : A fork of libfastjson by the rsyslog project, optimized for speed. This library is used by multiple external projects.

Quick links for agents

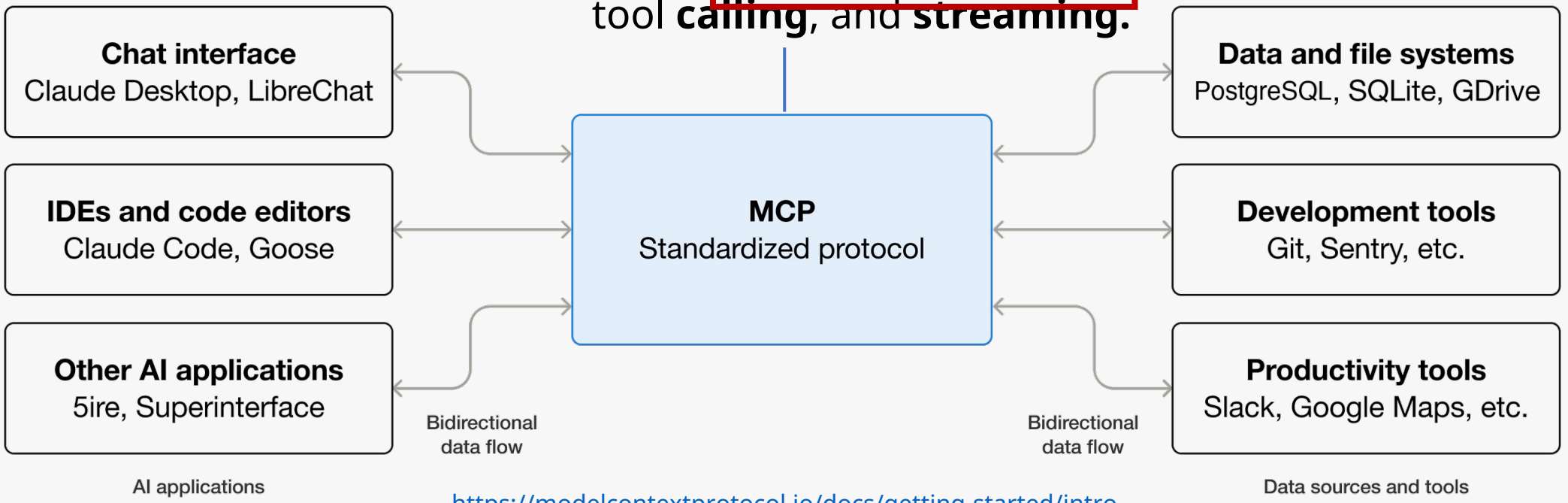
- **Documentation subtree guide:** `doc/AGENTS.md`
- **Core plugin subtree guide:** `plugins/AGENTS.md`
- **Contrib module subtree guide:** `contrib/AGENTS.md`
- **Built-in tools subtree guide:** `tools/AGENTS.md`

https://github.com/rsyslog/rsyslog/blob/main/AGENTS.md

Context on Demand: Model Context Protocol (MCP)

"MCP (Model Context Protocol) is an open-source standard **for connecting AI applications** to **external systems**."

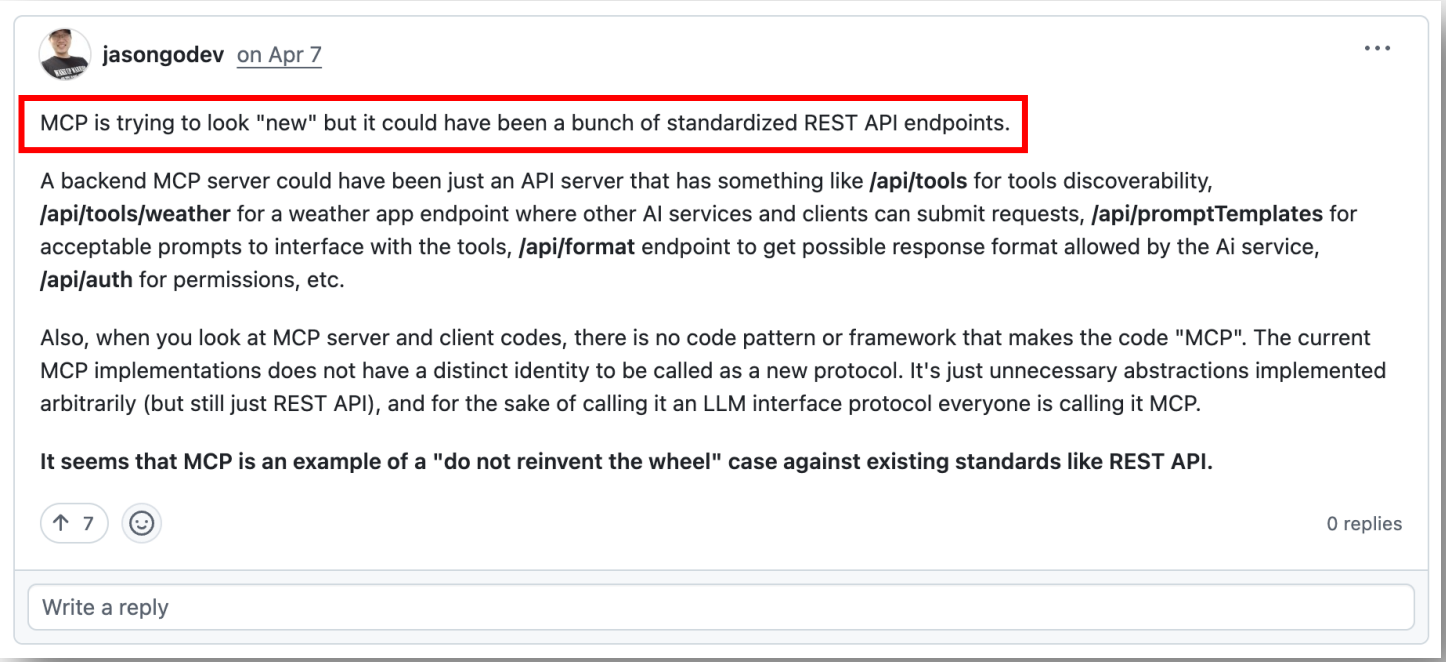
Transport: stdio, Streamable HTTP
Messages: JSON-RPC
Protocol: Allows capability **discovery**, tool **calling**, and **streaming**.



https://modelcontextprotocol.io/docs/getting-started/intro

Model Context Protocol (MCP)

"MCP (Model Context Protocol) is an open-source standard **for connecting AI applications** to **external systems**."



jasongodev on Apr 7

MCP is trying to look "new" but it could have been a bunch of standardized REST API endpoints.

A backend MCP server could have been just an API server that has something like **/api/tools** for tools discoverability, **/api/tools/weather** for a weather app endpoint where other AI services and clients can submit requests, **/api/promptTemplates** for acceptable prompts to interface with the tools, **/api/format** endpoint to get possible response format allowed by the Ai service, **/api/auth** for permissions, etc.

Also, when you look at MCP server and client codes, there is no code pattern or framework that makes the code "MCP". The current MCP implementations does not have a distinct identity to be called as a new protocol. It's just unnecessary abstractions implemented arbitrarily (but still just REST API), and for the sake of calling it an LLM interface protocol everyone is calling it MCP.

It seems that MCP is an example of a "do not reinvent the wheel" case against existing standards like REST API.

↑ 7 0 replies

Write a reply



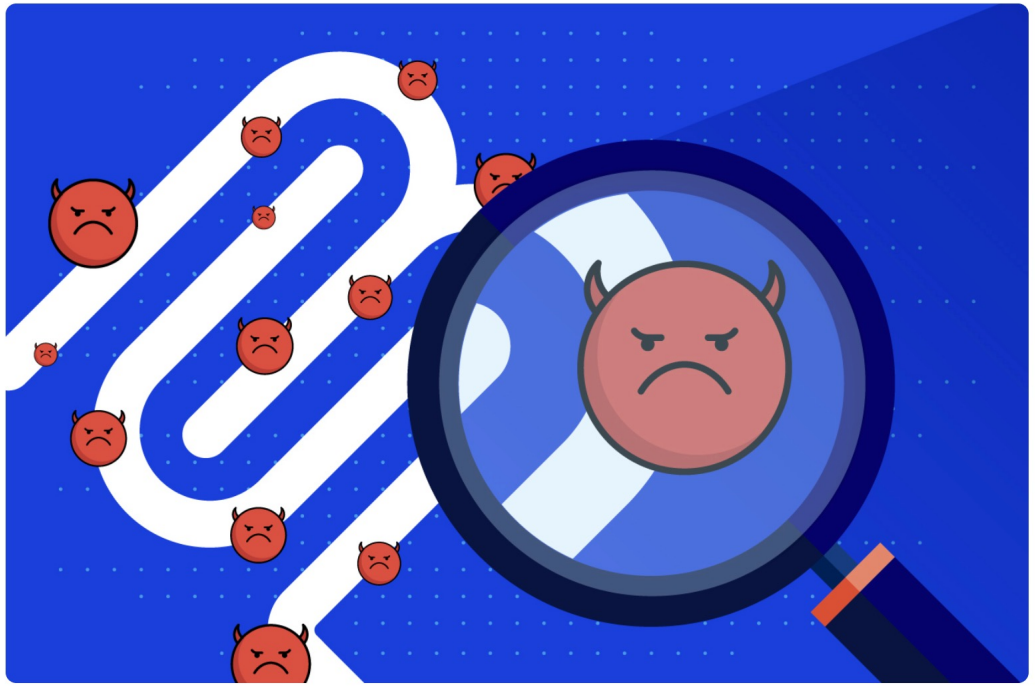
Your agent needs the open issues from GitHub.
REST API, MCP server, or the `gh` command-line tool?
Tradeoffs?

https://github.com/modelcontextprotocol/modelcontextprotocol/discussions/1093#discussioncomment-12744895

Model Context Protocol (MCP)

Threat Research: Hundreds of MCP Servers Vulnerable to Abuse

Mustafa Naamnih and Jonathan Ginzburg | Backslash Research Team - June 25, 2025



Base Protocol

Authorization

Copy page

Protocol Revision: draft

https://modelcontextprotocol.io/specification/draft/basic/authorization

https://www.backslash.security/blog/hundreds-of-mcp-servers-vulnerable-to-abuse
https://live.paloaltonetworks.com/t5/community-blogs/mcp-security-exposed-what-you-need-to-know-now/ba-p/1227143

Harness Engineering (CE for Agents)

*"Agentic harness engineering is the **use of structured artifacts to guide** the creation and evolution of software by coding agents."*

Descriptive

Guides

Architecture documentation
Domain documentation
Requirements specification

Context files `AGENTS.md, CLAUDE.md`
Skills `.claude/skills/`
Commands `.claude/commands/`

Sensors

Browser automation (e.g., Playwright)
Observability (e.g., logs and traces)
Performance tests

MCP servers `.mcp.json`
Subagents `.claude/agents/` (e.g., a reviewer)

Feedforward (before generation) — **Feedback** (after generation)

Guards

ADR checked by an LLM judge
Data schemas, e.g., JSON Schema

Settings permissions in `settings.json`

Checks

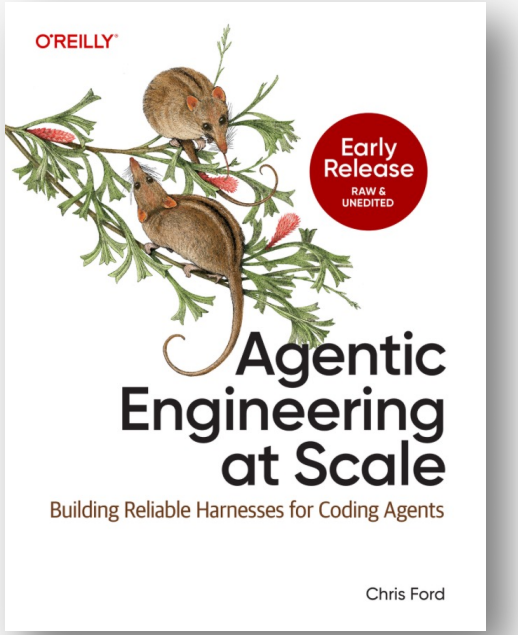
Unit, integration, and system tests
Static analysis tools
Linters

Hooks `settings.json`, e.g., ktlint after each edit

Normative

Axes and artifacts after Ford (2026). Blue entries: the configuration mechanisms of Galster et al. (2026), paths/files as in Claude Code.

Which artifacts would you use?



Generated using Claude Fable 5.1

Configuration Mechanism: Context Files

Generated using Claude Fable 5.1

AGENTS.md

```
# Web shop

## Build and test
gradle build      # runs ktlint, detekt, and tests

## Conventions
- Kotlin, no wildcard imports, camelCase names
- Domain terms: docs/domain.md (expired != invalid)

## Architecture
- Modules: orders, catalog, checkout
- Decisions: docs/adr/ (read before changing a module)
```

CLAUDE.md

```
# Claude Code

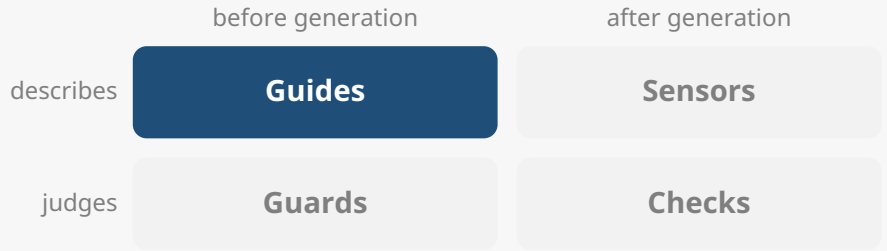
Read @AGENTS.md first. It is the shared configuration
for every coding agent used in this repository.

- Run gradle check before you report a task as done.
```

before generation **describes**

The agent loads the file in every session. It reads the file into the context before the first prompt. The file describes build commands, conventions, and the architecture, like a README for agents.

Most tools also read their own file. `CLAUDE.md`, `copilot-instructions.md`, and `GEMINI.md` are tool-specific. `AGENTS.md` is read by almost all tools, by Claude Code only if there is no `CLAUDE.md`.



Keep one shared `AGENTS.md` and let the tool-specific files point to it.

https://arxiv.org/abs/2602.14690

Guide Example: Domain Glossary

Generated using Claude Fable 5.1

docs/domain.md

```
# Web shop: domain glossary

Order
  A customer's basket at checkout. Has line items,
  a shipping fee, and at most one discount code.

Discount code
  Reduces the item subtotal, never the shipping fee.
  A code can be expired or invalid.
  Expired is not the same as invalid.

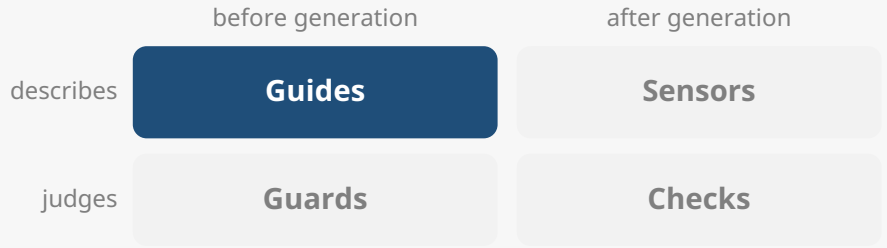
Total
  Item subtotal minus discount, plus shipping.
  Expired code: say so and charge the full total.
```

before generation **describes**

An agent reads the glossary before coding. It loads the file into its context, e.g., because `AGENTS.md` points to it.

The glossary defines the domain terms. Example: It explains what an order, a discount code, and the total are, but not how to implement them.

The agent uses these terms in its code.



If the agent ignores a Guide, nothing fails. Only a review by you or an agent finds it.

Configuration Mechanism: Skills

.claude/skills/release-notes/

```
SKILL.md
scripts/collect_changes.py
references/changelog-format.md
```

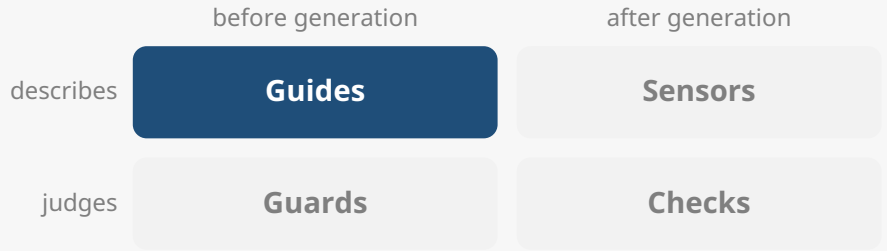
.claude/skills/release-notes/SKILL.md

```
---
name: release-notes
description: Draft release notes from the pull requests
  merged since the last tag. Use when the user asks for
  a changelog or release notes.
---
1. Run scripts/collect_changes.py <last-tag> to list
   the merged pull requests.
2. Group the entries: features, fixes, breaking changes.
3. Apply references/changelog-format.md.
4. Show the draft; do not create a release.
```

before generation describes

Skills are loaded on demand. At first, only the name and description are in the context. If the model judges that a task fits the description, it reads `SKILL.md`. Example: Steps 1 and 3 of `SKILL.md` tell it to run the script and read the format file.

Most AI tools use the same format. A Skill is a directory with `SKILL.md` (YAML front matter plus instructions) and optional `scripts/`, `references/`, and `assets/`. `SKILL.md` should have fewer than 500 lines.



Most agent skills are structured instructions, not executable extensions.

https://arxiv.org/abs/2602.14690

Generated using Claude Fable 5.1

Configuration Mechanism: Subagents

Generated using Claude Fable 5.1

.claude/agents/reviewer.md

```
---
name: reviewer
description: Reviews a change for bugs and violations
  of the project conventions. Use after a feature is
  implemented and before it is committed.
tools: Read, Grep, Bash
model: sonnet
---
You are a code reviewer. Read the diff, run
gradle check, and report your findings ordered by
severity. Each finding names the file, the line, and
a fix. Do not edit any files.
```

after generation | describes

A subagent runs in its own context window. Example: The main agent gets back the list of findings, not the files that the subagent read or the output of `gradle check`.

Like a Skill, the front matter has `name` and `description`. `tools` lists the tools that the subagent may use (here: no Edit or Write), and `model` configures the model for the subagent. Claude Code stores the files in `.claude/agents/`.

	before generation	after generation
describes	Guides	Sensors
judges	Guards	Checks

A subagent returns results without sharing the whole session transcript to not bloat the main context.

https://arxiv.org/abs/2602.14690

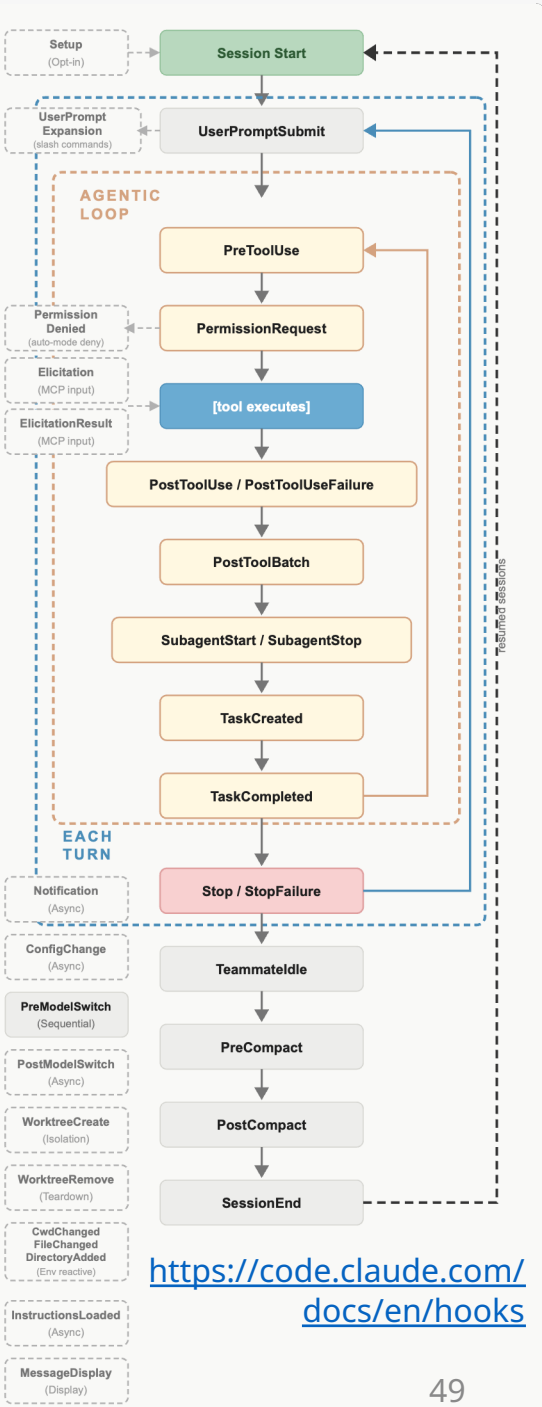
Hooks for Coding Agents

- **Coding agents also support hooks**
  (i.e., commands that run at fixed points of the agent loop).
- **Example:** After each file edit (`PostToolUse`), Claude Code runs `gradle ktlintFormat`, which fixes the formatting.
- What `ktlint` cannot fix, e.g., a function named `Sum_Values`, goes back to the agent (exit code 2) → Agent can then rename the function.
- No guarantee that the agent doesn't ignore or work around a check!

Generated using Claude Opus 5.5

.claude/settings.json

```
{
  "hooks": {
    "PostToolUse": [{
      "matcher": "Edit|Write",
      "hooks": [{
        "type": "command",
        "command": "gradle ktlintFormat >&2 || exit 2"
      }]
    }]
  }
}
```



https://code.claude.com/docs/en/hooks

LLMs as Judges

- In the judge role, **“LLMs rate or rank artifacts on quality criteria”**. The prompt contains the criteria and asks for a **score** or a **yes/no** verdict.
- **A prompt hook** can check plans against ADRs before Claude leaves plan mode. The check is a **separate model call** that sees the prompt and the hook input with the plan, but not the whole interaction log.
- If the model answers `ok: false`, Claude Code denies the `ExitPlanMode` call and returns the reason to the model.

.claude/settings.json (excerpt)

```
"PreToolUse": [{
  "matcher": "ExitPlanMode",
  "hooks": [{
    "type": "prompt",
    "prompt": "…",
    "continueOnBlock": true
  }]
}]

prompt (the "…" above):
  ADR 4: amounts are Int cents, never Double
  or Float. Answer ok: false if a step of the
  plan violates ADR 4. Input: $ARGUMENTS

answer of the model:
  {"ok": false, "reason": "Step 2 stores the
   10% discount as a Double, which ADR 4
   forbids."}
```

Generated using Claude Opus 5.5

It is not guaranteed that an LLM judge decides like humans would.

https://llm-guidelines.org/study-types/judges/ https://code.claude.com/docs/en/hooks

Guard Example: ADR plus LLM Judge

Generated using Claude Fable 5.1

docs/adr/0004-money.md

```
# ADR 4: Represent money as integer cents

Status: accepted, 2026-10-13

Context
  Totals were computed with Double and rounded
  differently in the cart, the invoice, and the API.

Decision
  Money is an Int amount in cents plus a currency.
  Never Double or Float. Only the Money class
  rounds, formats, or converts amounts.

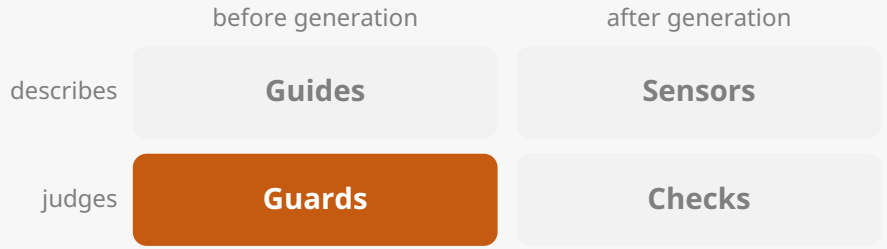
Consequences
  Prices are parsed once, at the API boundary.
```

before generation **judges**

On its own, ADR 4 is context, like a context file. The agent reads it before coding but can still compute a total as a `Double`.

A model that checks the plan turns ADR 4 into a Guard. A prompt hook sends the plan to a model before Claude leaves plan mode (previous slide). The model answers `ok: false` for a plan that uses `Double` or `Float` for amounts.

A model that compares the diff with ADR 4 is a Check. It runs after generation, as a unit test does.



ADR 4 becomes a Guard only when a model checks the plan against it before Claude writes code.

Sensor Example: Browser Automation

Generated using Claude Fable 5.1

agent session, Playwright tool

```
> browser.goto("http://localhost:8080/checkout")
> browser.screenshot("checkout.png")

title:     Checkout
text:      "Total: 0.00 EUR"
           "Code SPRING10 applied"

network:   GET /api/orders/1042  200  (total: 5000)

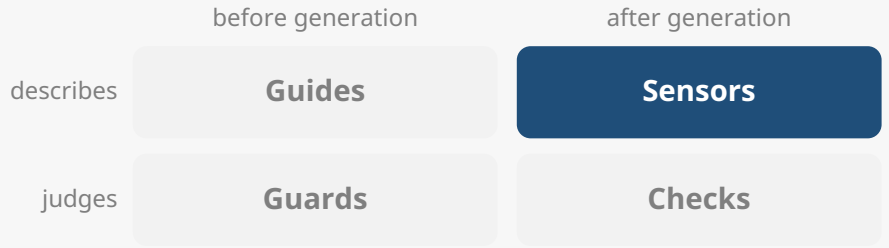
console:
  [error] TypeError: discount is undefined
          at renderTotal (checkout.js:42)
```

after generation | describes

The agent opens the checkout page in a browser. It can load pages with Playwright and read the text, the requests, and the console output.

Playwright cannot decide whether the page is correct. Example: The page shows 0.00 EUR, the API returned 5000 cents, and the console shows a `TypeError`. The agent must conclude that its change broke the page.

Logs, metrics, and performance measurements are also Sensors. They report values but do not compare them with a defined limit or target.



With a limit/target, such as “max. 500 ms response time”, a measurement becomes a Check.

Check Example: Unit Test

Generated using Claude Fable 5.1

src/test/kotlin/.../OrderTest.kt

```kotlin
@Test
fun `expired code does not reduce the total`() {
    val order = Order(items = 5000, code = Code.EXPIRED)
    assertEquals(5000, order.total())
}
```

bash

```
$ gradle test --console=plain

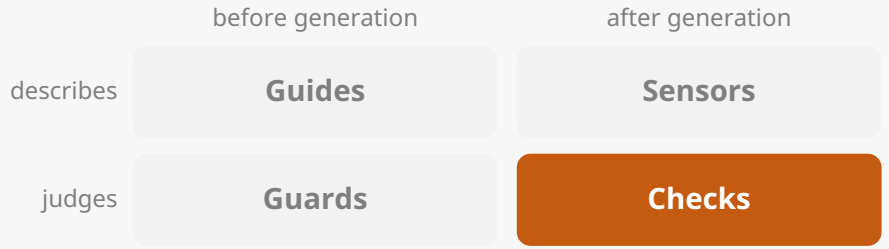
> Task :test FAILED
OrderTest > expired code does not reduce
  the total() FAILED
    AssertionFailedError:
      expected: <5000> but was: <4500>

FAILURE: Build failed with an exception.
```

after generation **judges**

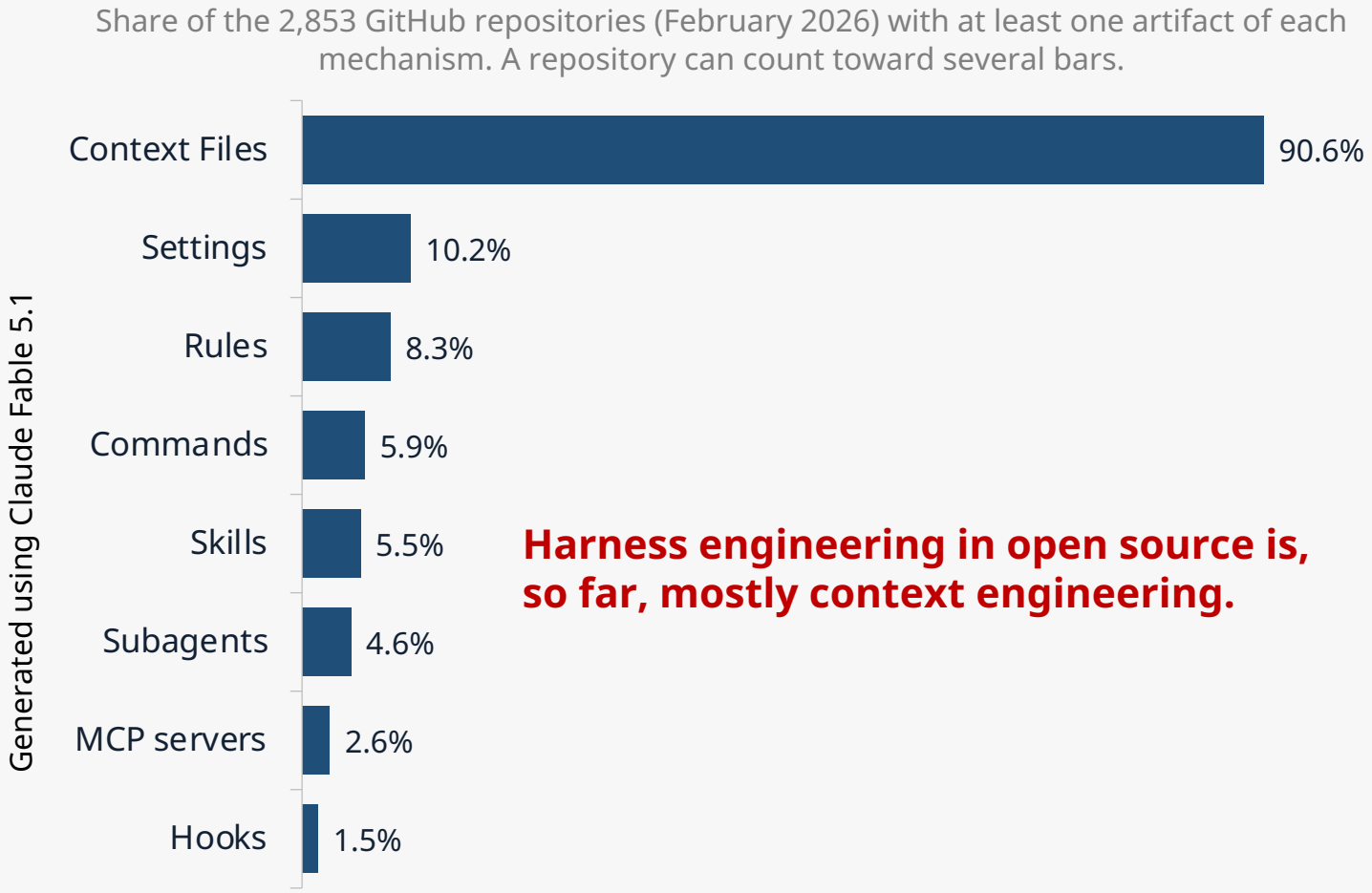
The test returns a verdict on the generated code. Example: It fails because the agent's code applies the 10% discount of an expired code: the total is 4500 cents (45.00 EUR) instead of 5000 (50.00 EUR).

The agent can run the test itself. It reads the assertion error in the gradle test output (expected 5000, but was 4500), changes the code, and runs the test again until it passes.



When all Checks pass, the agent can stop. Humans then review what the Checks do not test.

Adoption of AI Configuration Mechanisms



Harness engineering in open source is, so far, mostly context engineering.

open-source repositories

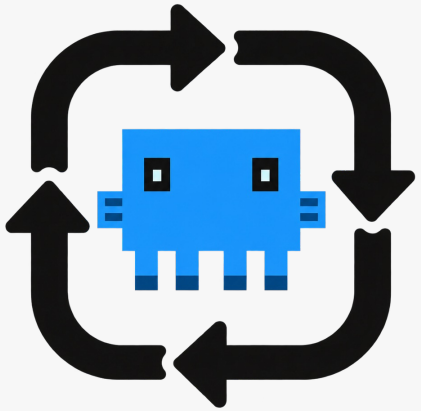
Sample: 2,853 GitHub repositories. They are repositories of engineered software projects with at least one configuration artifact for Claude Code, Copilot, Cursor, Gemini, or Codex.

Context Files dominate. Nine of ten repositories have a context file, and most repos have nothing else. 493 repositories use only `AGENTS.md`, without any tool-specific file.

Users of different tools configure different mechanisms. Claude Code repositories use the broadest range of mechanisms, and Copilot repositories rarely go beyond Context Files.

https://arxiv.org/abs/2602.14690

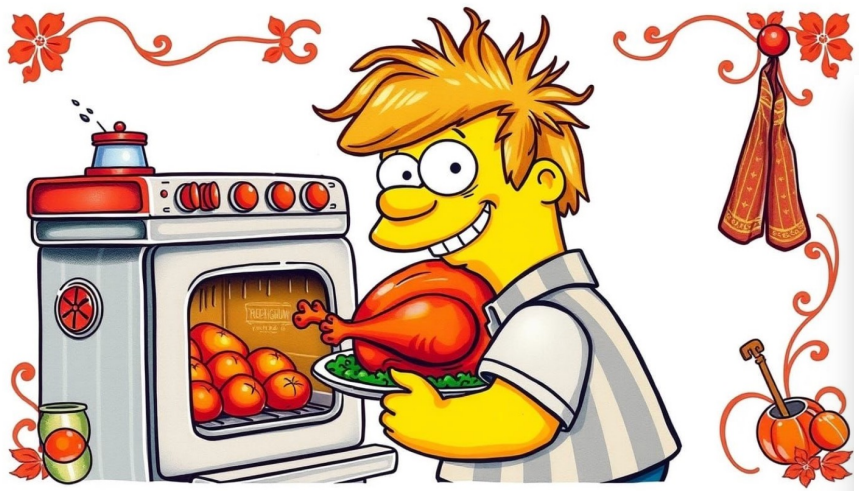
Loop Engineering



“Ralph Loop”

BY **GEOFFREY HUNTLEY** IN **AI** — 14 JUL 2025

Ralph Wiggum as a "software engineer"



If you've seen my socials lately, you might have seen me talking about Ralph and wondering what Ralph is. Ralph is a technique. In its purest form, Ralph is a Bash loop.

```
while :; do cat PROMPT.md | claude-code ; done
```

Ralph can replace the majority of outsourcing at most companies for greenfield projects. It has defects, but these are identifiable and resolvable through various styles of prompts.

> **That's the beauty of Ralph - the technique is deterministically bad in an undeterministic world.**

Ralph can be done with any tool that does not cap tool calls and usage.

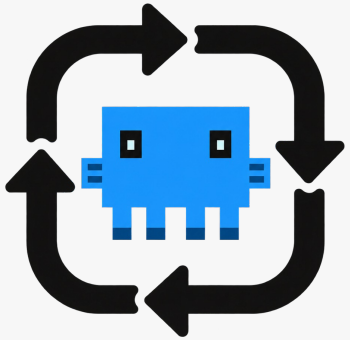
https://ghuntley.com/ralph/

Prompt → Context → Harness → Loop

Layer	Unit of Concern	Example Artifacts
Prompt eng.	One instruction	Prompts, prompt templates
Context eng.	One model call	`AGENTS.md`, `CLAUDE.md`, rules
Harness eng.	One agent loop	Skills, subagents, hooks, MCP config.
Loop eng.	Repeated agent loops	Schedules, stop conditions, state files, run logs, budgets, verifier agents

https://arxiv.org/abs/2608.21884

Loop Engineering



Generated using ChatGPT (GPT-6)

“Loop engineering is the practice of **designing automated control structures that repeatedly invoke coding agents**, triggered on a **schedule** or by **events**, with each agent loop bounded by a machine-checkable **stop condition.”** (Lulla et al., 2026)

`/goal`	The previous turn finishes, or, in an interactive session, an **idle check-in** or an **automatic retry** comes due	A model confirms the condition is met or judges it impossible, or a turn fails on **an error you have to fix**, or you run `/goal clear`

https://arxiv.org/abs/2608.21884 https://code.claude.com/docs/en/goal

Goals: Machine-Checkable Stop Conditions

- The Ralph loop never stops. **A goal defines a stop condition:** `/goal <condition>` keeps Claude starting new turns until the condition holds.
- After every turn, a **separate evaluator,** Haiku by default, reads the conversation and answers “not yet met”, “met”, or “impossible”.
- **The evaluator runs no commands**. Here: It judges the condition from the `gradle test` output, and the turn count that the main Claude session reports.
- **Auto mode** skips permission prompts (using an LLM judge).

```
claude

> /goal all tests in test/auth pass and ktlint is clean,
  or stop after 20 turns

◎ /goal active
  turn 1: gradle test, 3 failures in test/auth
  evaluator: not yet met (3 tests fail)
  turn 2: fix TokenParser, rerun, 1 failure
  evaluator: not yet met (1 test fails, ktlint not run)
  turn 3: fix expiry check, rerun, 14 pass,
          gradle ktlintCheck, clean
  evaluator: met (14 tests pass, 0 ktlint findings)

> /goal
Goal achieved after 3 turns (11 min, 184k tokens)
```

Without /goal, the agent decides when it is done. With /goal, the evaluator decides based on the transcript.

https://code.claude.com/docs/en/goal

Generated using Claude Fable 5.1

Scheduled and Event Triggers

- Within a session, `/loop 5m <prompt>` **reruns a prompt** every five minutes while Claude Code is open. The interval is at least one minute, and the loop ends after seven days.
- Without a prompt, `/loop` runs **a built-in maintenance prompt**: it continues unfinished work, handles review comments and failed CI runs on the current pull request, and runs cleanup passes when nothing else is pending.
- Cloud tasks can also start on an HTTP POST, e.g., from an alerting system, or on GitHub events such as `pull_request.opened`.

```
claude

> /loop 5m check whether CI passed and address review comments
> /schedule daily PR review at 9am
> /schedule in 2 weeks, open a cleanup PR that removes the flag

cron     0 9 * * 1-5    weekdays at 9:00 UTC
         0 */2 * * *    every two hours
GitHub   pull_request.opened, release.published
API      POST …/routines/<id>/fire  {"text": "alert …"}
```

	Cloud task	Desktop task	`/loop`
Runs on	Anthropic cloud (default)	your machine	your machine
Needs open session	no	no	yes
Local files	no (fresh clone)	yes	yes
Minimum interval	1 hour	1 minute (form: 1 hour)	1 minute
Other triggers	API, GitHub events	none	none

Event-based triggers and scheduled invocations of an agent.

https://code.claude.com/docs/en/scheduled-tasks https://code.claude.com/docs/en/web-scheduled-tasks

Generated using Claude Fable 5.1

Studying Loop Engineering

Loop Engineering: Building Blocks, Adoption, and Impact

Jai Lal Lulla
Singapore Management University
Singapore, Singapore
jailal.l.2025@phdcs.smu.edu.sg

Vahram Nersesyan
Heidelberg University
Heidelberg, Germany
vahram.nersesyan@stud.uni-heidelberg.de

Seyedmoein Mohsenimofidi
Heidelberg University
Heidelberg, Germany
s.mohsenimofidi@uni-heidelberg.de

Christoph Treude
Singapore Management University
Singapore, Singapore
ctreude@smu.edu.sg

Sebastian Baltes
Heidelberg University
Heidelberg, Germany
sebastian.baltes@uni-heidelberg.de

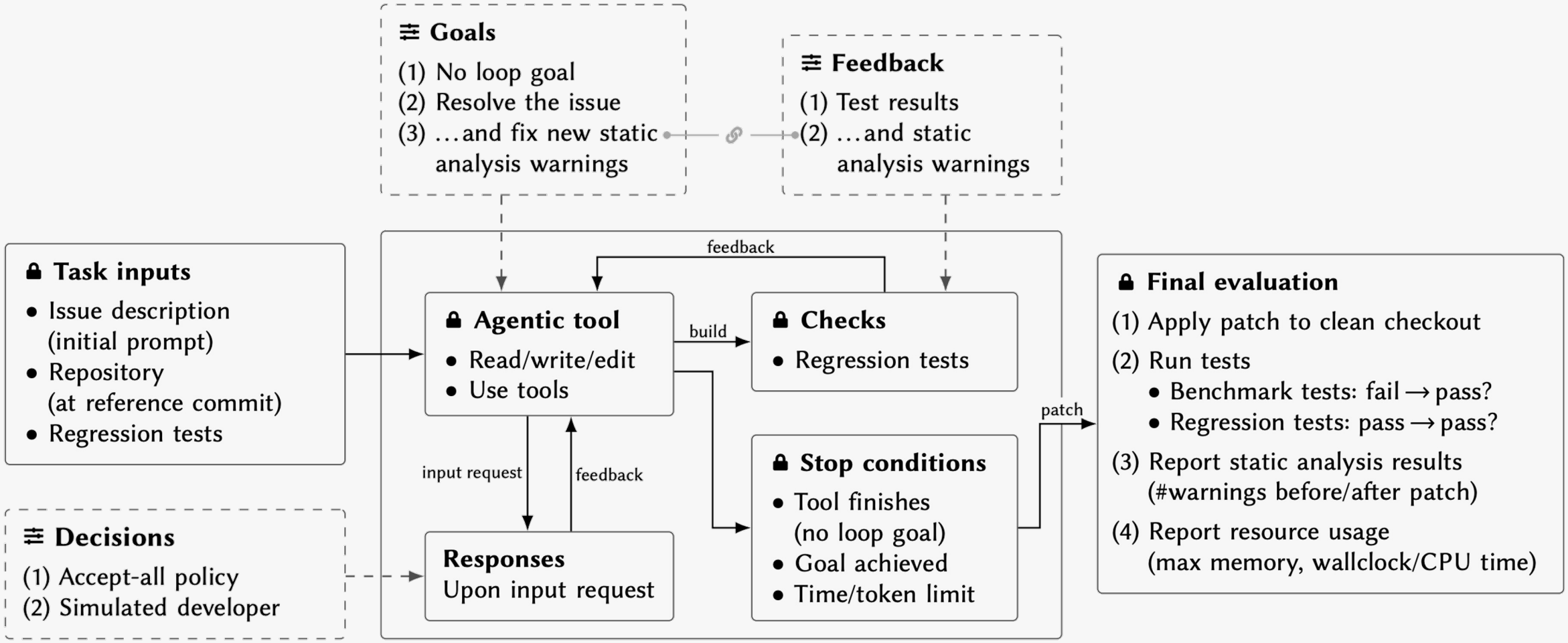
RQ1 *How is loop engineering defined in gray literature, and how does it relate to prompt, context, and harness engineering?*

RQ2 *Which aspects of loop engineering are traceable from repository data, and which practices are adopted in open-source projects?*

RQ3 *How do loop goals, static analysis checks, and decision policies affect task success, code quality, resource usage, and token consumption of agent loops?*

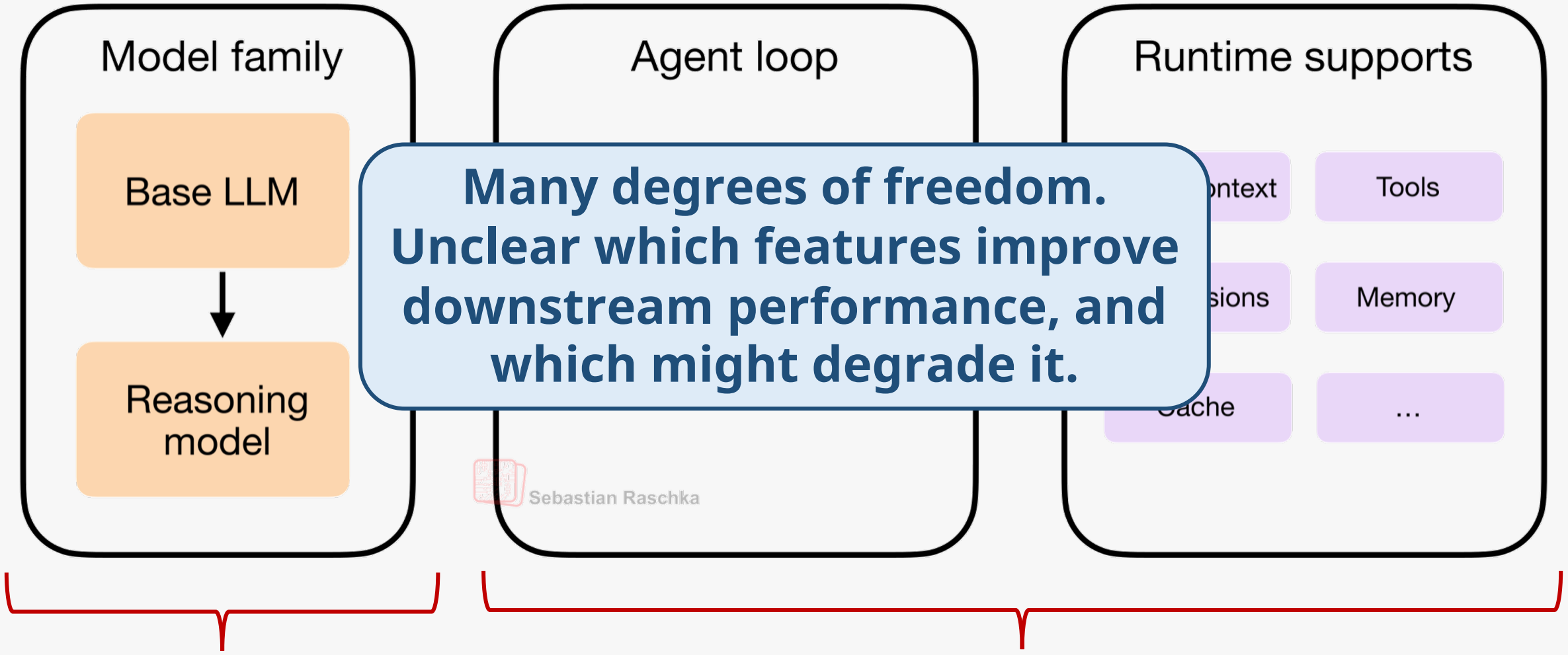
https://arxiv.org/abs/2608.21884

Studying Loop Engineering



https://arxiv.org/abs/2608.21884

Recap: Agentic Coding Assistants



https://magazine.sebastianraschka.com/p/components-of-a-coding-agent

Degrees of Freedom in Agentic SE Experiments

OH **Outer harness**
Configuration that users add, such as context files, settings, hooks, and MCP servers

IH **Inner harness**
Harness code and release, including the built-in system prompt and tool schemas

MO **Model**
Model version and its sampling and reasoning settings, such as reasoning effort

TA **Tasks**
Task sets and versions, selected instances, and the task descriptions given to the agent

IN **User interaction**
Who answers agent requests during a session: a human, a model, or no one

325 experiments

Sample. 146 peer-reviewed full papers from 2025 and 2026, selected from 21,654 DBLP and arXiv records, report 325 experiments with agentic coding tools.

Contrast. One pattern of changed degrees of freedom within an experiment. The 325 experiments contain 782 contrasts.

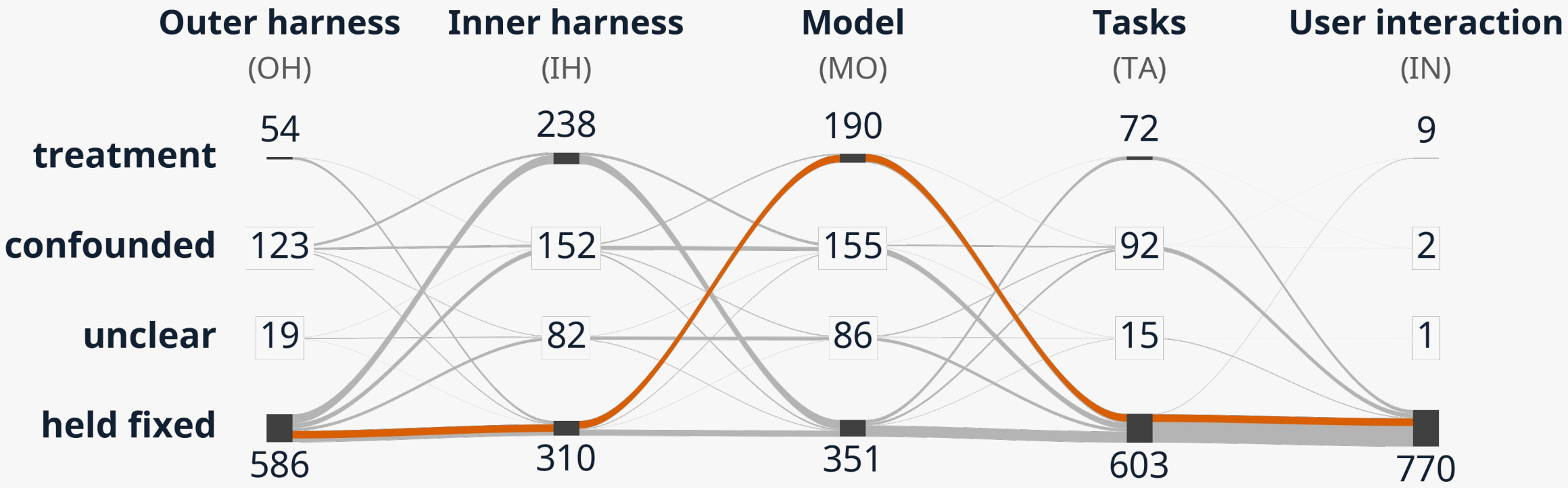
Codes. In each contrast, a degree of freedom is the treatment (the change being tested), confounded (changed along with it), unclear, or held fixed.

A comparison that changes model and harness together cannot isolate the effect of either.

https://arxiv.org/abs/2610.35348

Generated using Claude Opus 5.5

Five Degrees of Freedom in 782 Contrasts



Generated using Claude Opus 5.5

One path per contrast. Numbers and line widths count contrasts, and orange marks the most frequent combination.

The most frequent contrast changes only the model and holds everything else fixed (160 of 782).

https://arxiv.org/abs/2610.35348



empirical-software.engineering

Prof. Dr. Sebastian Baltes