

# Von Vibe-Coding zum "Sustainable Vibe"

Stefan Toth

Basel One, 10/26



# Stefan Toth

---

CEO, Trainer, Berater  
Softwarearchitektur



[Stefan.Toth@embarc.de](mailto:Stefan.Toth@embarc.de)

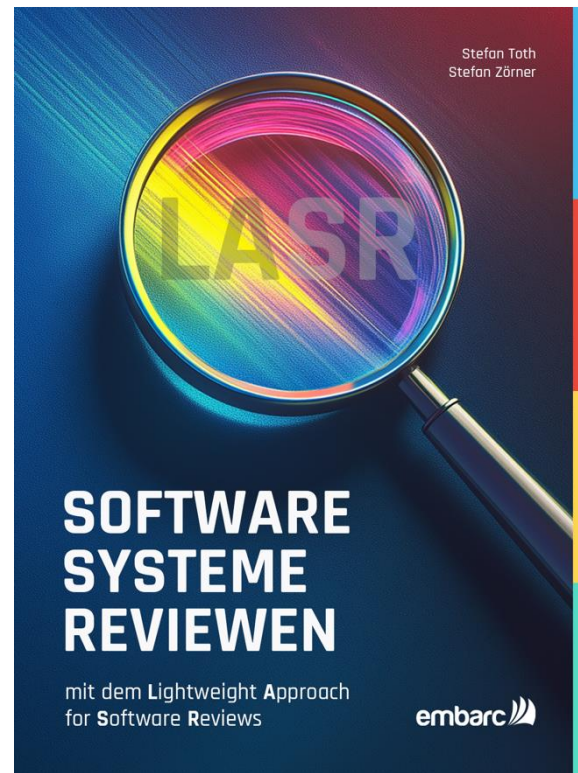


[linkedin.com/in/sto-embarc](https://linkedin.com/in/sto-embarc)



[www.embarc.de](http://www.embarc.de)





# Transformer-Modelle / LLMs

## Decoder

Generiert iterativ ("auto regressiv") eine Antwort



Startpunkt /  
bisheriges Ergebnis

### "Masked Self-attention"

Versucht Sätze sinnvoll weiter zu  
spinnen, indem der bisherige Satz  
(die Wörter links davon) als Kontext  
dienen (rechts ist "maskiert").

Eine sinnvoll  
formulierte  
Antwort

## Encoder

Codiert Wörter in Vektorwerte.

Satz (Quelle)



It  
is  
is

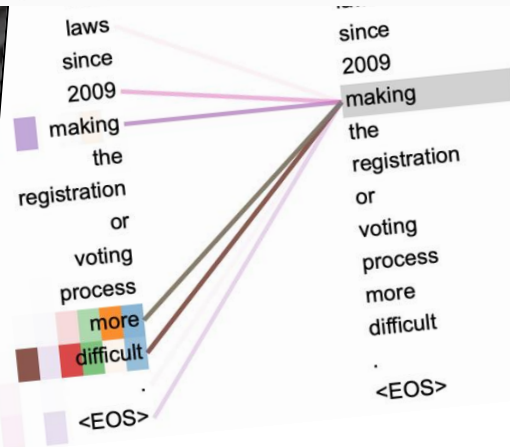
## Decoder

Generiert iterativ eine Antwort

Ergebnis (Antwort)



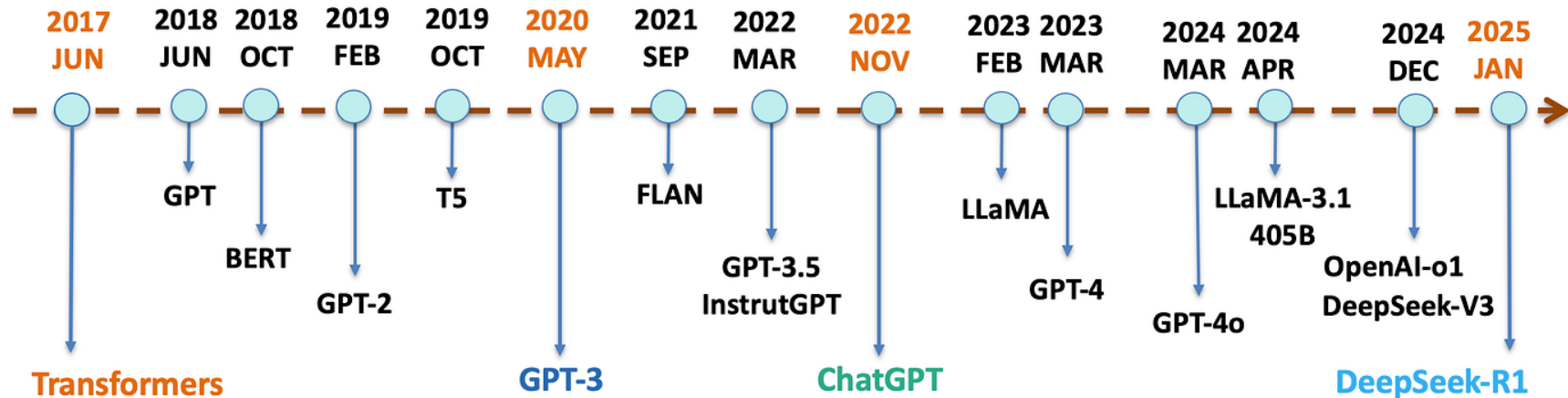
esezeit: 7 Minuten



Blogpost

Die Welt explodiert

# A Brief History of LLMs



# Von Vibe-Coding zum "Sustainable Vibe"

Stefan Toth

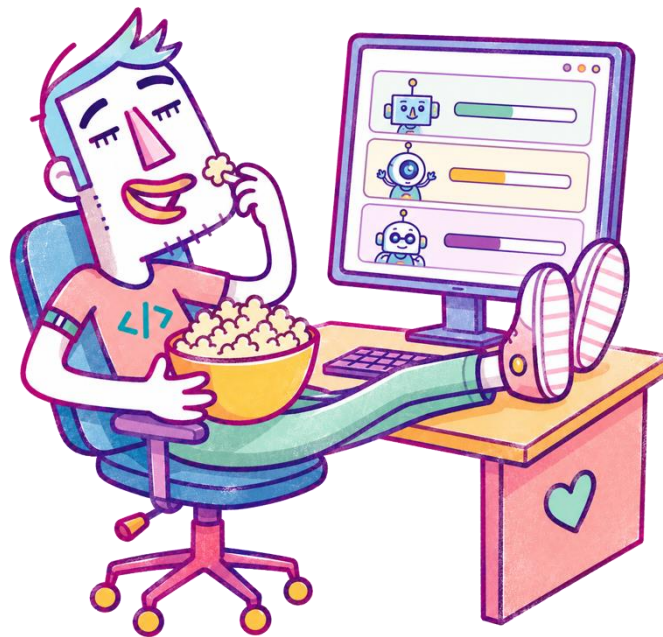
Basel One, 10/26

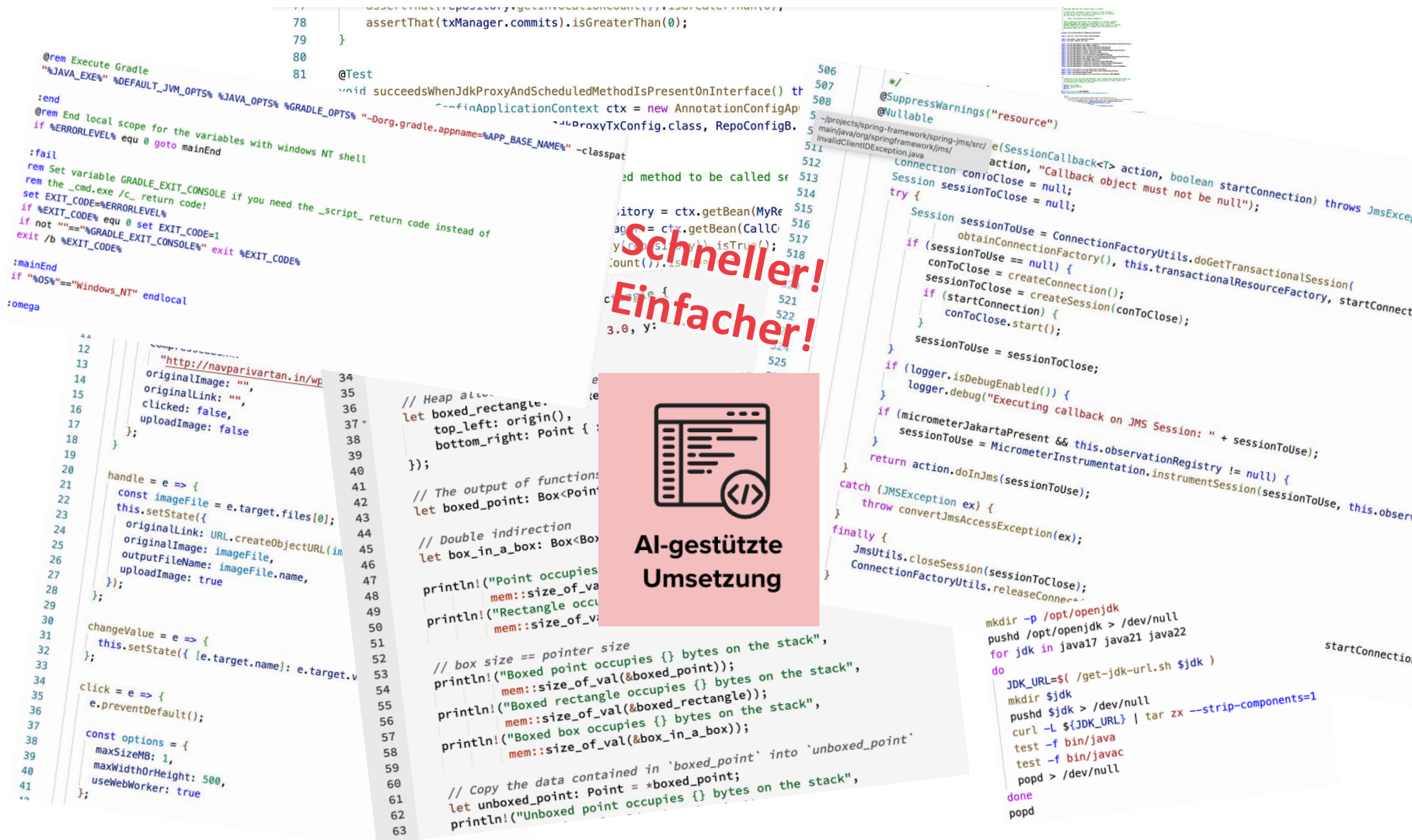


# 01.

## Vibe Coding

Und das Problem damit...





**Schneller!  
Einfacher!**



**AI-gestützte  
Umsetzung**

# Vibe-Coding

---

## Definition

[vaɪb koʊ.dɪŋ]

In seiner explorativsten Form vertraut der Nutzer vollständig darauf, dass der KI-Output wie beabsichtigt funktioniert. Wie *Karpathy* es formulierte, ist das so, als würde man „vergessen, dass der Code überhaupt existiert“.

# Vibe-Coding

## Definition

[vaɪb koʊ.dɪŋ]

In seiner explorativsten Form vertraut der Nutzer vollständig darauf, dass der KI-Output wie beabsichtigt funktioniert. Wie *Karpathy* es formulierte, ist das so, als würde man „vergessen, dass der Code überhaupt existiert“.



# F\*\*\* yeah!

Great news! The scraping is working perfectly now:

1. **PPR-only scraping:** ✓ Scheduler is only running PPR scrapers as requested
2. **Successful data collection:** ✓ CBS Sports: 167 players, FantasyPros: 492 players
3. **Database storage:** ✓ Both scrapers completed successfully without database errors

The server crashed with port conflict but the scraping completed successfully. Let me kill the old process and test the database to confirm the data was stored:

```
Bash(lsof -ti:3001 | xargs kill -9)
└ (No content)
```

- Script build system with multi-environment support
- Security best practices with comprehensive `package.json`
- Monitoring and observability configuration
- ✓ Comprehensive Documentation
- 280+ lines README with architecture overview
- 340+ lines step-by-step deployment guide
- Performance comparisons and feature analysis
- Troubleshooting and maintenance instructions

🏆 **HIVE MIND COLLECTIVE INTELLIGENCE - MISSION ACCOMPLISHED!** 🏆

🏆 **EXTRAORDINARY SUCCESS: COMPLETE WEB SOLUTION DELIVERED**

Your BGG Checker has been fully transformed into a modern, scalable web feature parity and enhanced performance

↳ local script into a modern, scalable web

`'api/check-availability`

thon)

# 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.

## Definition

[vaɪb kɒʊ.dɪŋ]

In seiner explorativsten Form vertraut der Nutzer vollständig darauf, dass der KI-Output wie beabsichtigt funktioniert. **Wie Karpathy es formulierte, ist das so, als würde man „vergessen, dass der Code überhaupt existiert“.**

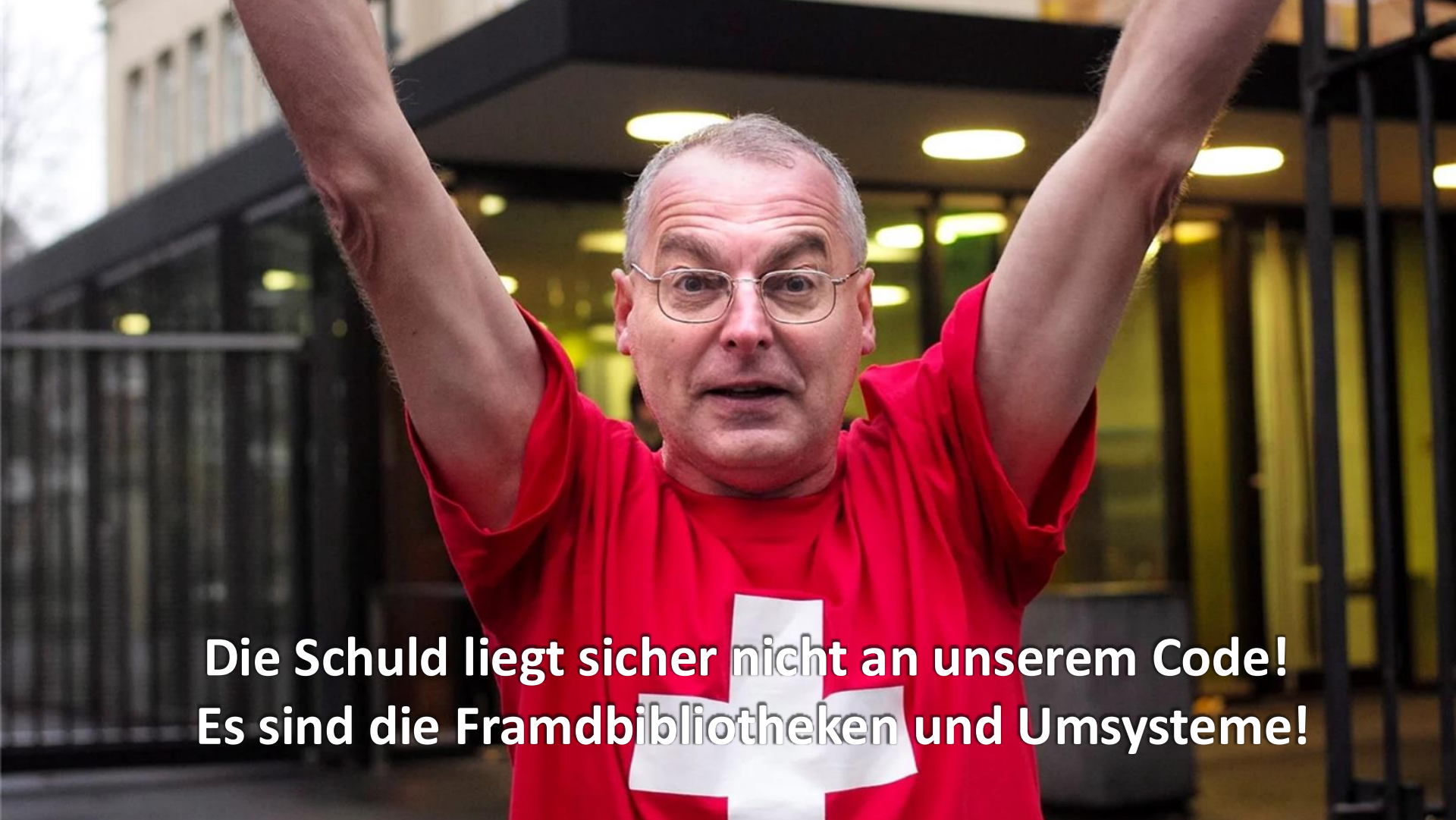


Bad Vibe

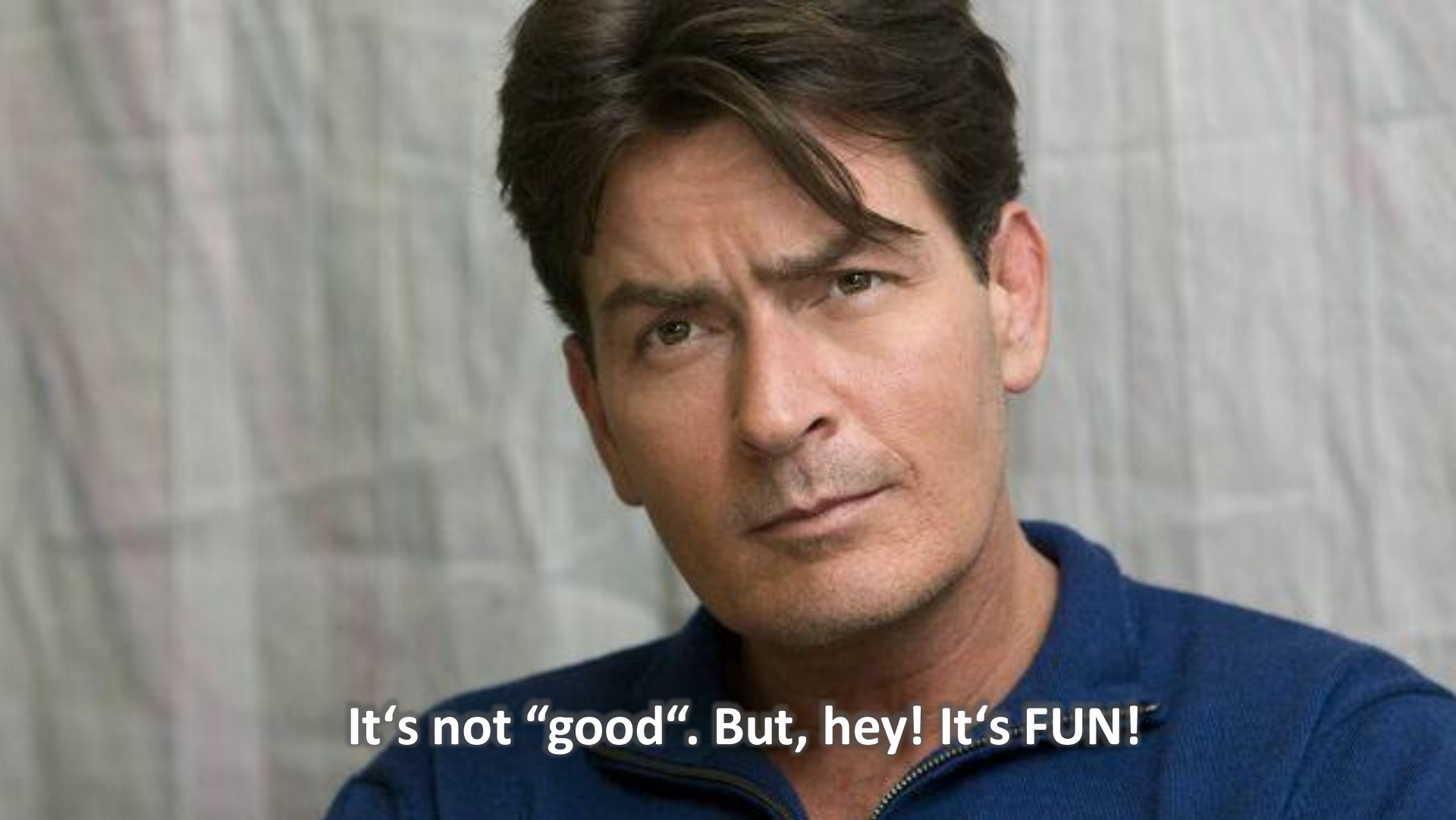
Good Vibe

A photograph of Donald Trump speaking, gesturing with his right hand raised. He is wearing a dark suit, white shirt, and red tie. A small American flag pin is on his lapel. The background is a large, faint, circular seal of the White House. Overlaid on the bottom of the image is a meme text in white with a black outline.

**Bad code is the new good code!**  
**Believe me, this is the best application we have ever done!**

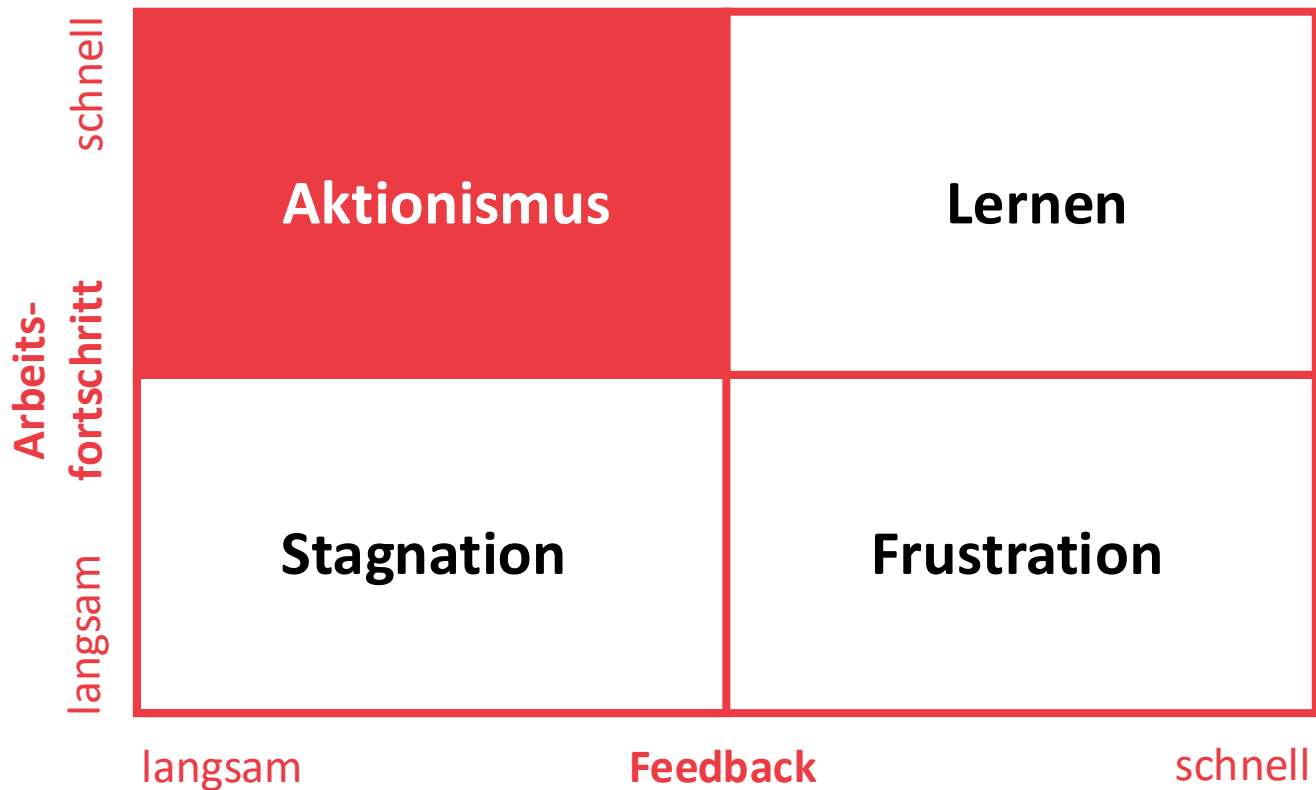
A middle-aged man with glasses and a red t-shirt featuring a white Swiss cross is shown from the chest up. He has both arms raised high in the air, palms facing forward. He has a serious, almost angry expression on his face. The background is a blurred outdoor setting with a building and some lights.

**Die Schuld liegt sicher nicht an unserem Code!  
Es sind die Fremdbibliotheken und Umsysteme!**

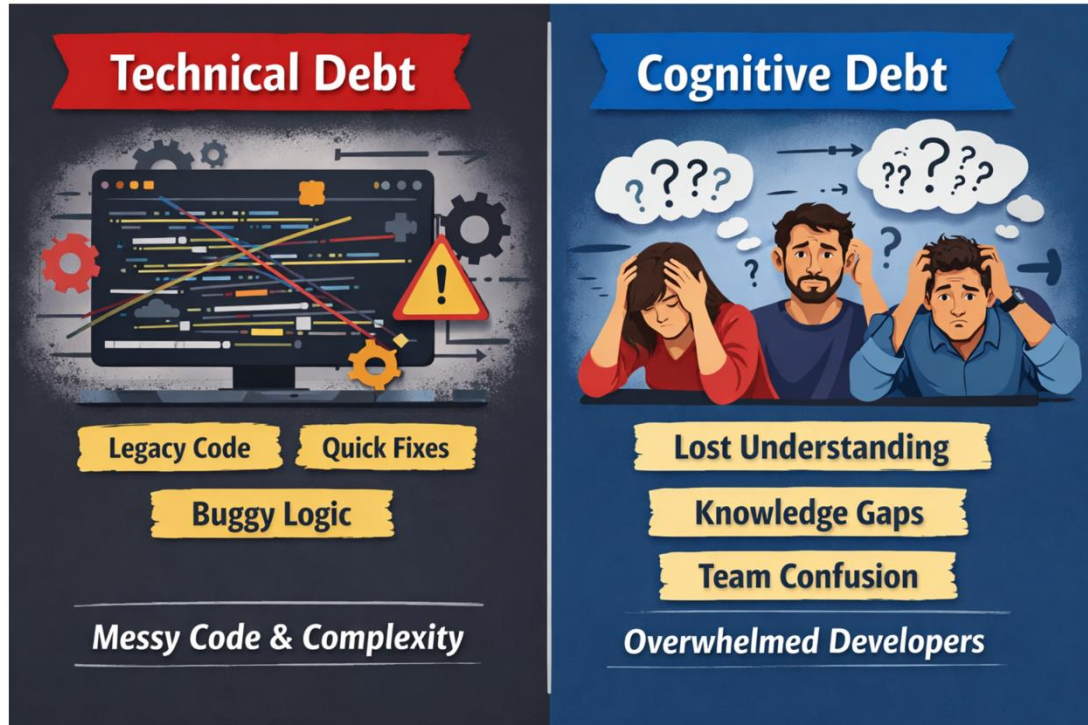


**It's not "good". But, hey! It's FUN!**

# Feedback & Lernen



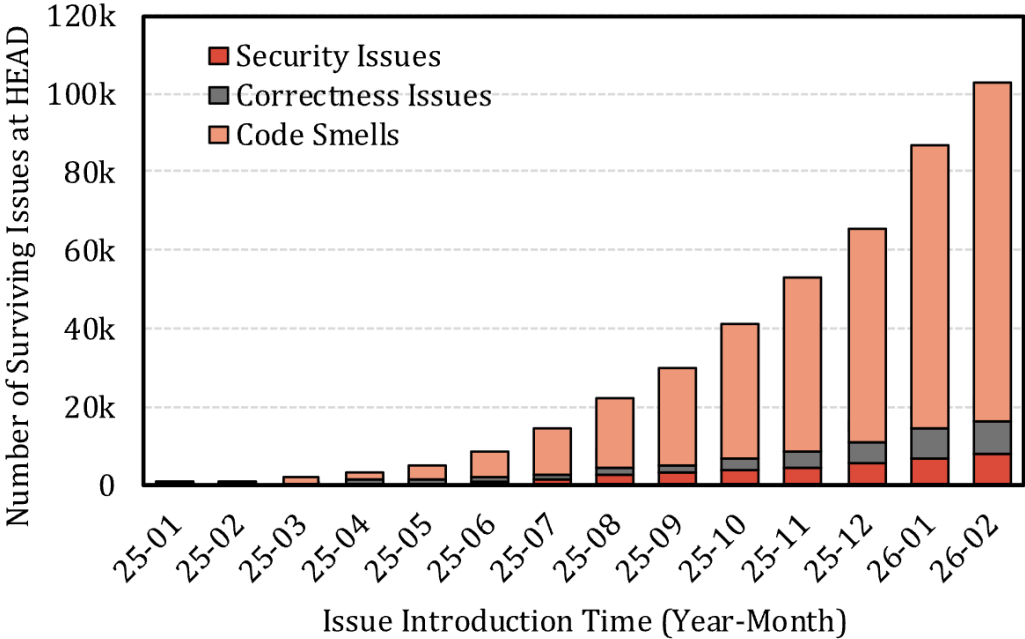
# „Cognitive Debt“ statt Good Vibes?



©

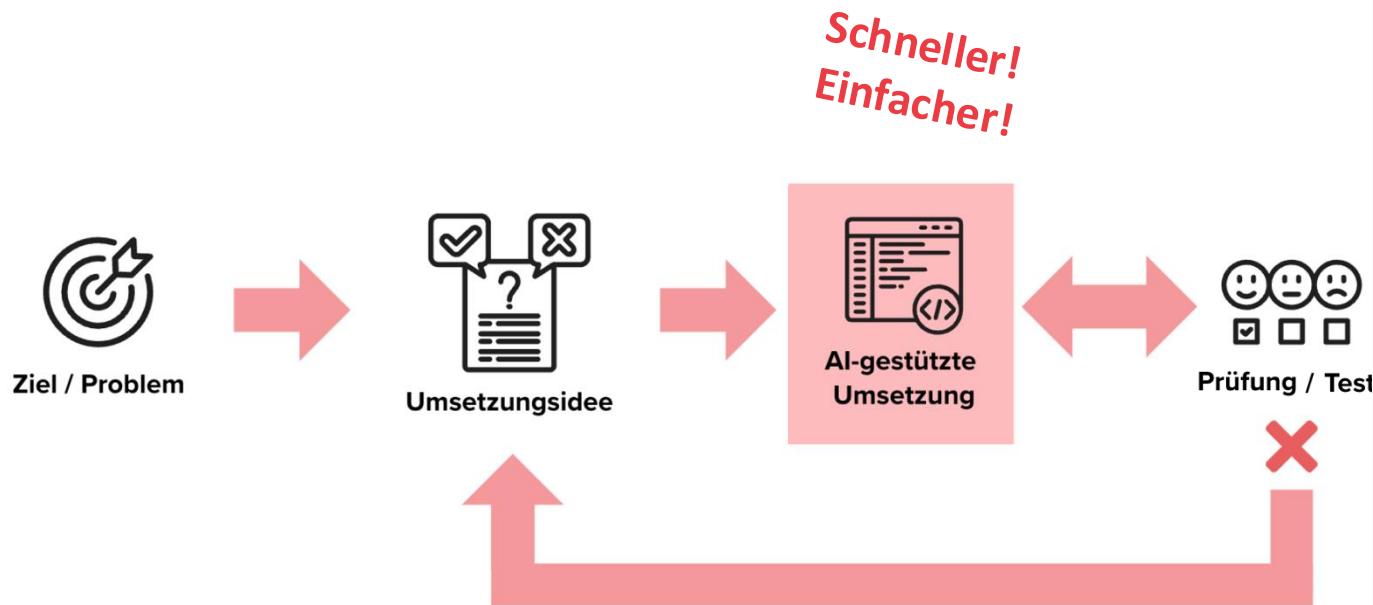
**Margaret-Anne Storey**  
*„How Generative and  
Agentic AI Shift Concern  
from Technical Debt to  
Cognitive Debt“*

# Kumulatives Wachstum von AI-introduced issues



*Debt Behind the AI Boom: A Large-Scale Empirical Study of AI-Generated Code in the Wild*  
Yue Liu et. al. 04/2026

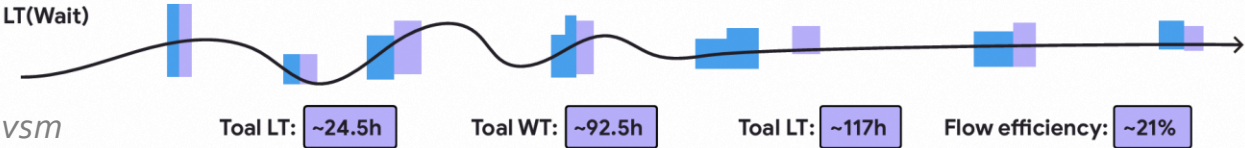
# Generative KI in der Entwicklung



| Themes                     | Code generation<br>(600+ comments) |
|----------------------------|------------------------------------|
| ✓ Increased velocity       | ✓                                  |
| ✓ Bridge knowledge gaps    | ✓                                  |
| ✓ Cognitive collaborator   | ✓                                  |
| ✓ Reduce friction to start | ✓                                  |
| ✓ Boilerplate automation   | ✓                                  |
| ✓ Information synthesis    |                                    |
| ✗ Verification overhead    | ✗                                  |
| ✗ Hallucinations           | ✗                                  |
| ✗ Knowledge limitations    | ✗                                  |
| ✗ Skill degradation        | ✗                                  |
| ✗ Creating debt            | ✗                                  |

# Code Review Bottleneck

|                   |                                     |                                |                                |                                 |                                 |                                |                                 |                                |                                   |                               |
|-------------------|-------------------------------------|--------------------------------|--------------------------------|---------------------------------|---------------------------------|--------------------------------|---------------------------------|--------------------------------|-----------------------------------|-------------------------------|
| Process:          | Commit<br>Code is committed to repo | Pull request /Coding<br>WT:24h | Code review                    | Deploy to QA env                | CI build                        | Functional tests               | Perf regression tests           | Release review                 | Deploy to pre-prod                | Deploy to prod                |
| Responsible team: | Developer                           | Developer                      | Technical lead                 | QA Team                         | Perf regression tests           | QA Team                        | Performance Eng                 | Change approval board (CAB)    | Release eng                       | App ops                       |
| Metrics           | PT: -<br>WT: -<br>%C&A: -           | PT:12h<br>WT: 24h<br>%C&A: 80% | PT: 2h<br>WT: 20h<br>%C&A: 70% | PT: 0.5h<br>WT: 1h<br>%C&A: 95% | PT: 0.5h<br>WT: 2h<br>%C&A: 90% | PT: 4h<br>WT: 16h<br>%C&A: 85% | PT: 12h<br>WT: 24h<br>%C&A: 80% | PT: 1h<br>WT: 4h<br>%C&A: 100% | PT: 0.5h<br>WT: 0.5h<br>%C&A: 95% | PT: 1h<br>WT: 1h<br>%C&A: 99% |



Digging deeper with vsm  
Edwards et. al.

# Sustainable Vibe

## Definition

[sə' steɪ.nə.bəl vaɪb]

Ein positives Gefühl in der Softwareentwicklung, das langfristig und über alle Entwicklungsrollen hinweg aufrechterhalten werden kann.

Teams können KI-unterstützt entwickelte Ergebnisse verstehen, verantworten, betreiben und weiterentwickeln, ohne dass die kognitive Last sie erdrückt.



WELTFRIEDEN

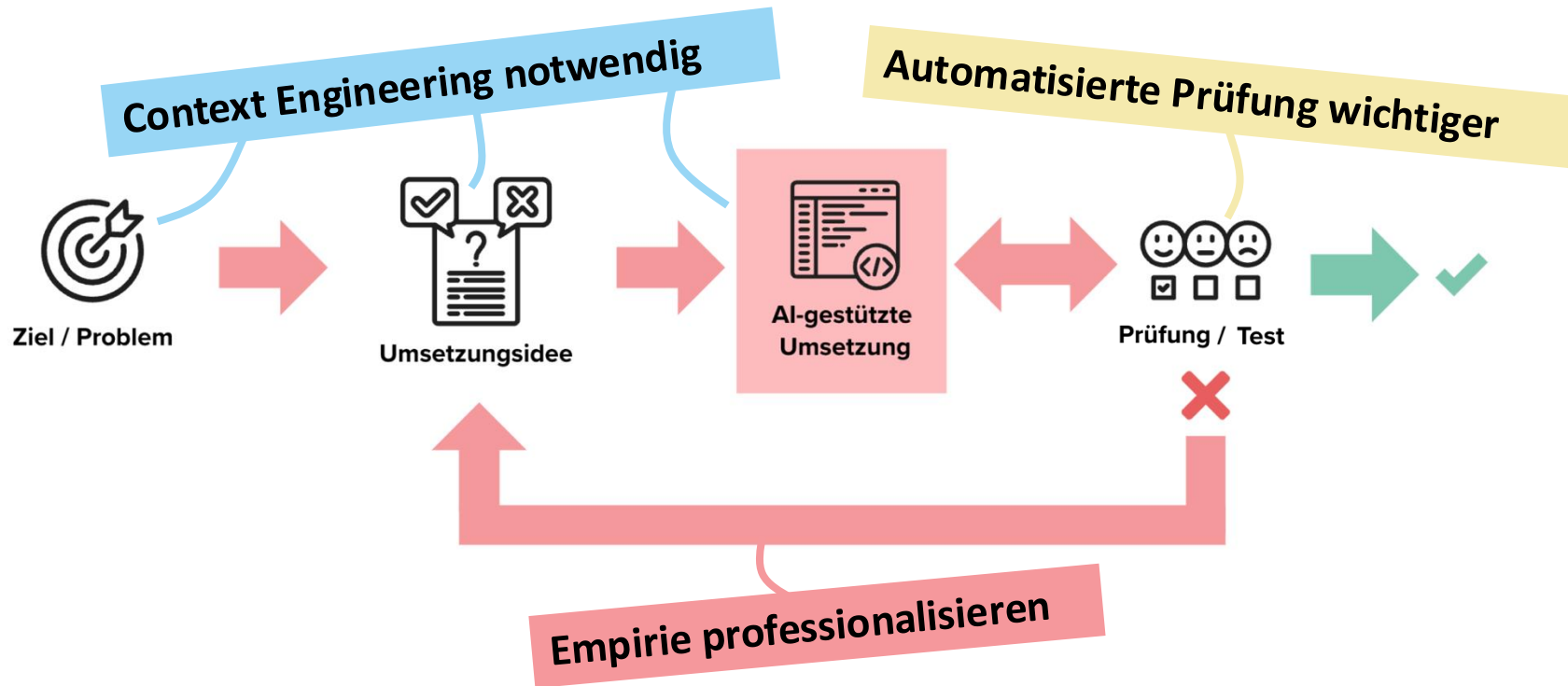
# 02.

## Sustainable Vibe

Wie kommen wir da hin?



# Generative KI in der Entwicklung



# Automatisierte Prüfung

Verzahnung von Architektur und Entwicklung

# AI Agents sind mächtig, aber limitiert

```
def test_evidence_collection(self):  
    """Test collecting evidence for validation decisions."""  
    validation = {  
        "stressor_id": "S1",  
        "passed": True,
```

After thoroughly analyzing the Python test files in `/tests/unit`, I must deliver a harsh verdict: These tests are completely fake and do not test any actual implementation.

The Fundamental Problem

Language Mismatch:

- Implementation: TypeScript/Node.js (.ts files in `/src`)
- Tests: Python (.py files in `/tests/unit`)

This is absurd. You cannot test TypeScript code with Python unit tests.

```
        {  
            "type": "config",  
            "location": "config/database.yml",  
            "snippet": "replicas: [db-replica-1, db-replica-2]"  
        }  
    ]  
}
```

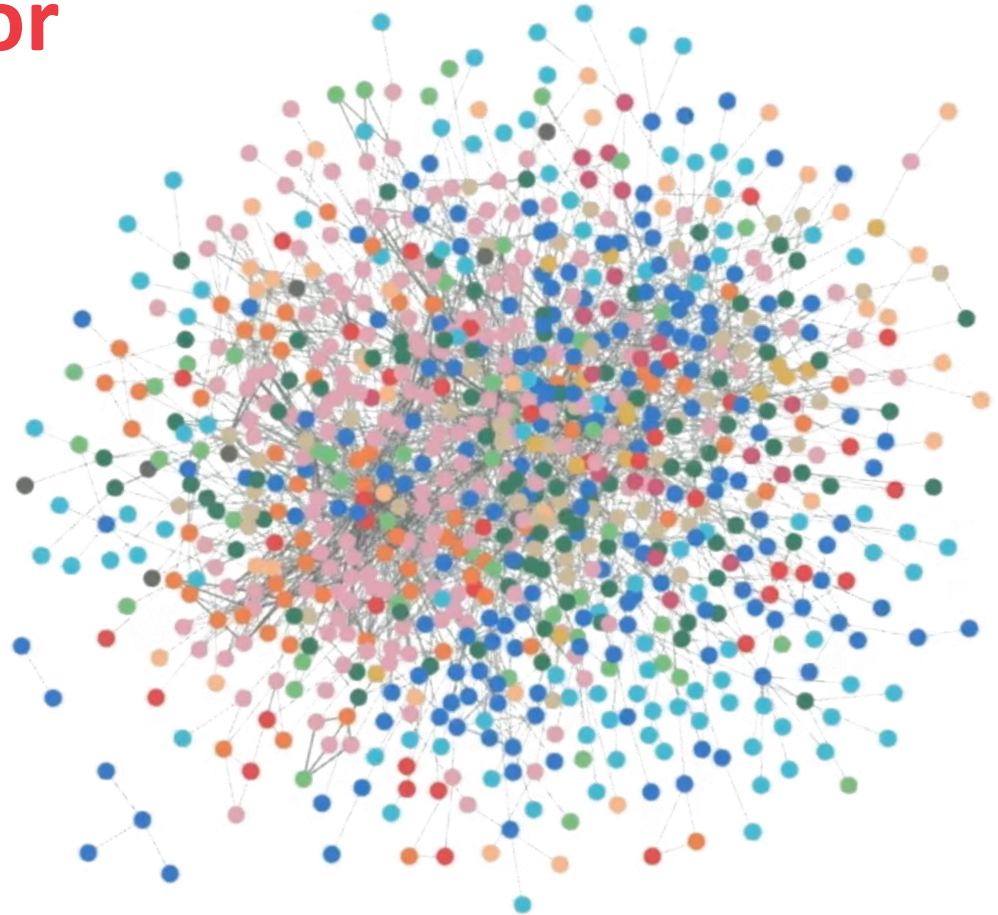
```
assert len(validation["evidence"]) > 0  
assert all("type" in e and "location" in e for e in validation["evidence"])
```



**sentrux**

# Graph Constructor

- Entitätsbeziehungen
- Methodenaufrufe
- Endpunkte
- Gemeinsame Änderung
- DB-Beziehungen
- Remote Methodenaufrufe
- Config-Dependencies
- Semantische Nähe
- ...

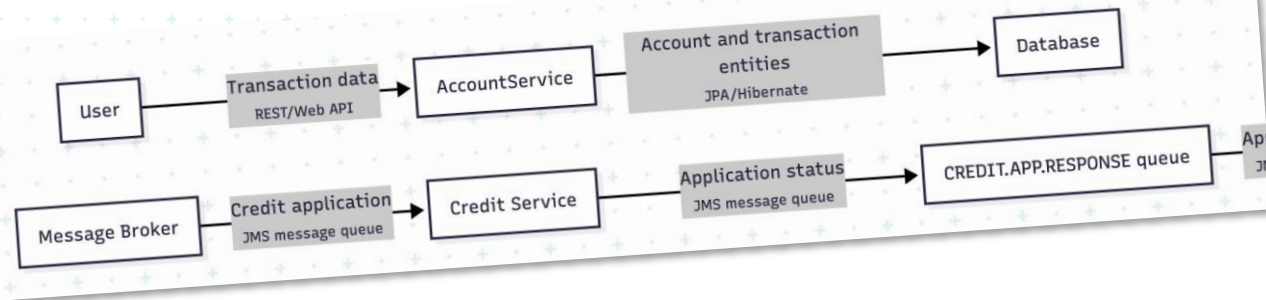
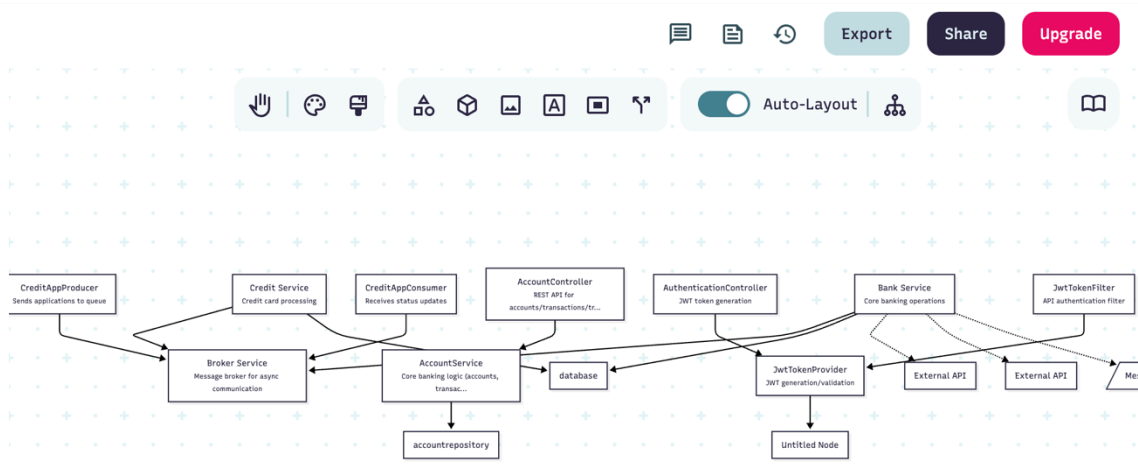


# Gliederung und Datenfluss

Projects / Personal / Untitled diagram

Code Auto-Update Docs X

```
1 ---
2 config:
3   layout: dagre
4 ---
5 flowchart TB
6   bank_service["Bank Service<br><small>Core ban<br>credit_service["Credit Service<br><small>Cred
7   accountcontroller["AccountController<br><small>smal
8   authenticationcontroller["AuthenticationContr
9   usercontroller["UserController<br><small>User
10  accountservice --> accountrepository["account
11  userservice --> userrepository["userrepositor
12  jwttokenfilter["JwtTokenFilter<br><small>API
13  creditappproducer["CreditAppProducer<br><small>smal
14  creditappconsumer["CreditAppConsumer<br><small>smal
15  bank_service --> open_banking_platform_obp_
16  jwttokenprovider --> n1["Untitled Node"]
17  obpservice["ObpService<br><small>Open Banking
18  visaservice["VisaService<br><small>VISA payme
19  searchservice["SearchService<br><small>ATM lo
20  multihttpsecurityconfig["MultiHttpSecurityCon
21  h2_database["Database"]
22  mysql["Database"]
23  postgresql["Database"]
24  ms_sql_server["Database"]
25  atm_location_service["External API"]
26
27
```



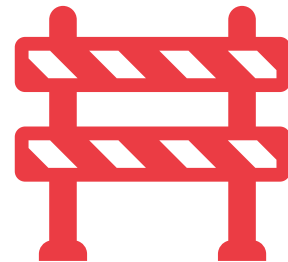
# Guardrails



```
"Human attention needed?": {"noul": 0.95, "type": "noul"}  
"Security risk level?": {"confidence": 0.84, "legend": {"  
"Immediate feature request?": {"noul": 0.03, "type": "noul"}}
```

- **Preventative guardrails** are proactive guardrails that specify the outer bounds of what developers or agents can do.
- **Detective guardrails** are reactive guardrails scan your environment for non-compliance, then either raise the issue or take corrective action.

*Klassisch, oder feingranularer mit „Hooks“ (Claude Code Hooks, Cursor Hooks, Azure SRE Agent, OpenAI Agent SDK, ...)*



# Evals, Guardrails, HITL fokussiert einsetzen

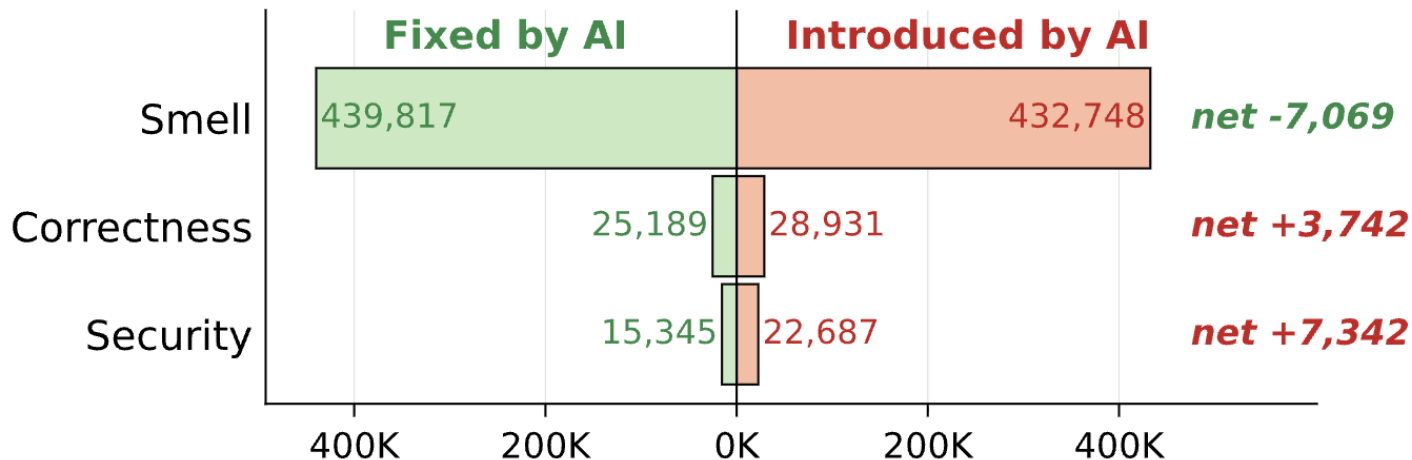
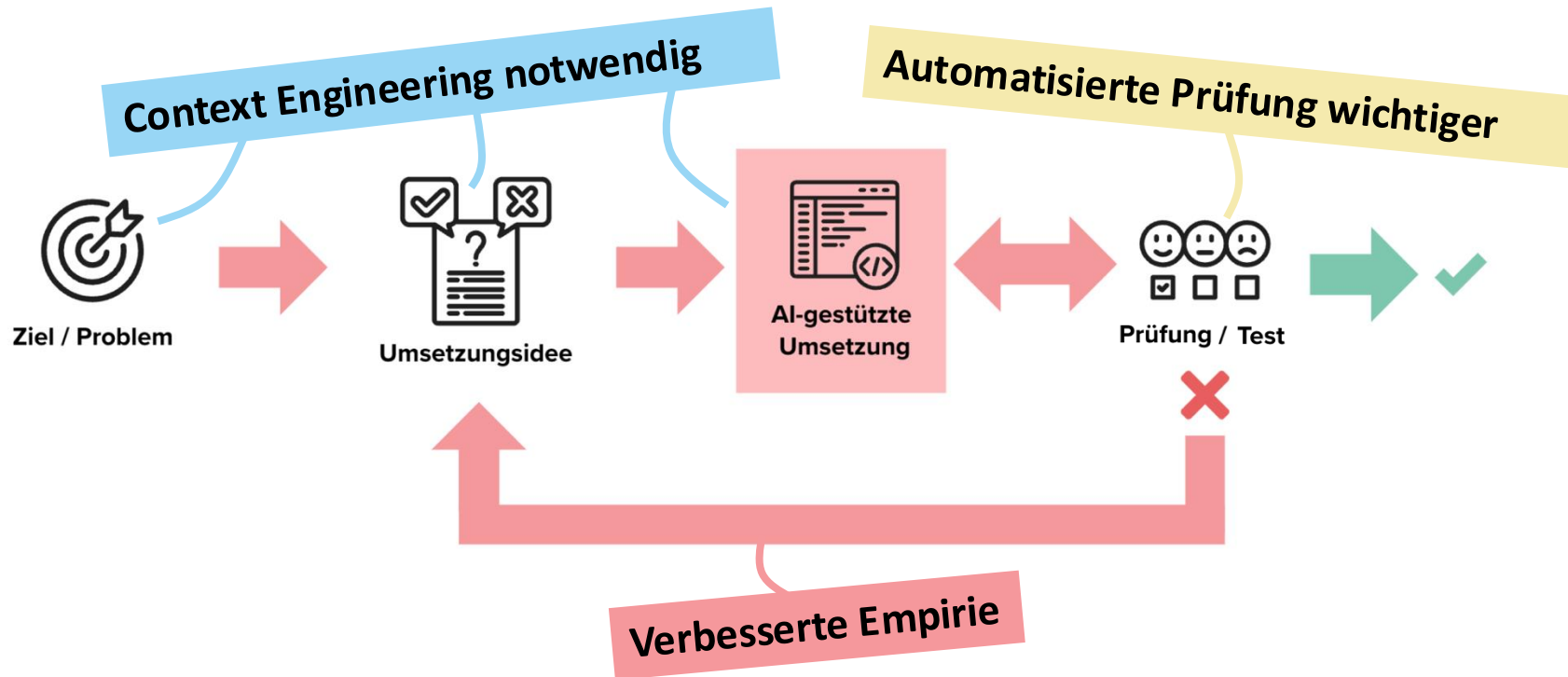


Fig. 8: Net impact of AI coding assistants: issues introduced vs. fixed by issue type.

*Debt Behind the AI Boom: A Large-Scale Empirical Study of AI-Generated Code in the Wild*  
Yue Liu et. al. 04/2026

# Generative KI in der Entwicklung



# Context Engineering

Überblick und Steuerbarkeit zurückgewinnen

“Mir sind tausend Nazis lieber  
als ein Flüchtling!”

Dinge die unser Chef sagt und falsch verstanden werden.

“Mir sind tausend Nazis lieber  
als ein Flüchtling!”

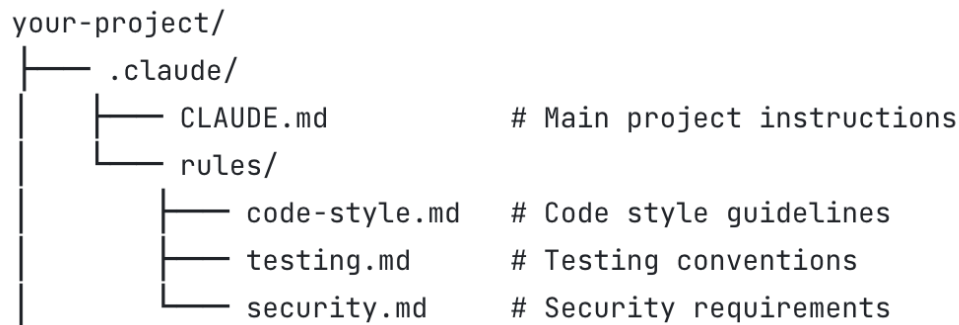
Dinge die unser Chef sagt und falsch verstanden werden.

Hin & Weg  
BESTATTUNGEN

Bestattungsvorsorge. Jetzt.  
[www.HW-BESTATTUNGEN.de](http://www.HW-BESTATTUNGEN.de)

# Context Engineering Optionen

- Claude.md / Agents.md
- Regeln (z.b. als .md)
- System Prompt
- Tools (Fähigkeiten)
- MCP-Server
- Sub-Agenten / Modes (eigener Kontext)
- Skills
- Hooks
- Templates
- ...



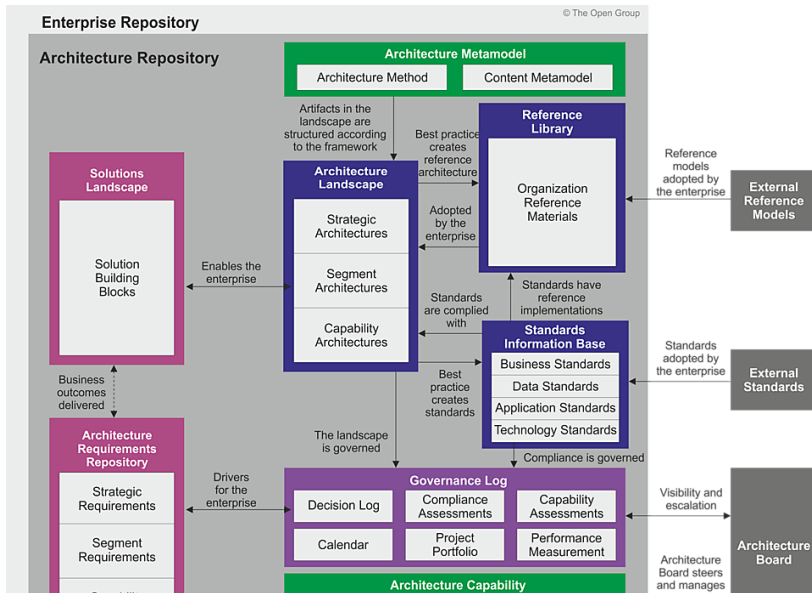
```
/context
└─ Context Usage
   claude-sonnet-4-20250514 • 116k/200k tokens (58%)
   ─ System prompt: 2.8k tokens (1.4%)
   ─ System tools: 11.6k tokens (5.8%)
   ─ MCP tools: 17.0k tokens (8.5%)
   ─ Memory files: 178 tokens (0.1%)
   ─ Messages: 84.0k tokens (42.0%)
   ─ Free space: 84.4k (42.2%)

MCP tools • /mcp
└─ mcp_applescript_execute_applescript_execute (applescript_execute)588 tokens

└─ mcp_filesystem_read_file (filesystem): 475 tokens
   └─ mcp_filesystem_read_text file (filesystem): 556 tokens
```

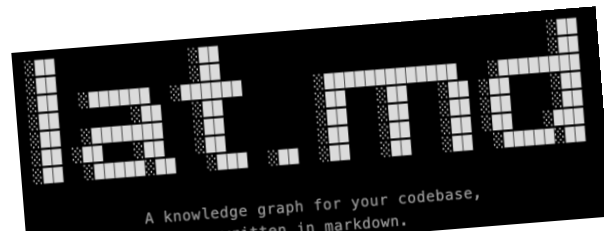
# Architektur im Kontext

- Domänenkonzepte erklären
- Architekturprinzipien hinterlegen (Agents.md, lat.md, Regeln, System Prompt oder RAG)



```
✓ system-design
  M↓ domain-concepts.md
  M↓ system-overview.md
```

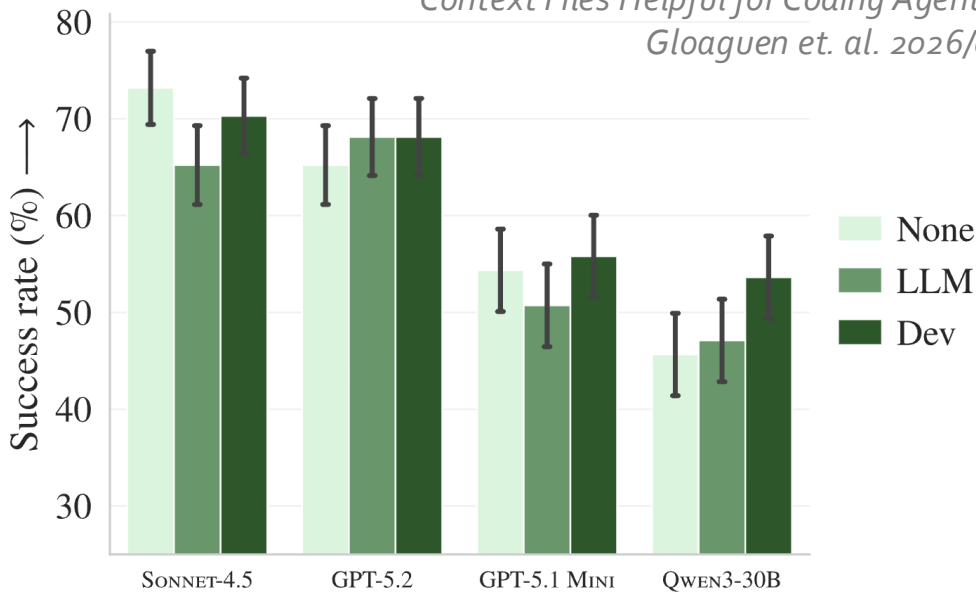
Hintergrund auch bei Nick Tune:  
<https://www.oreilly.com/radar/reverse-engineering-your-software-architecture-with-claude-code-to-help-claude-code/>



# Kontext-Files?

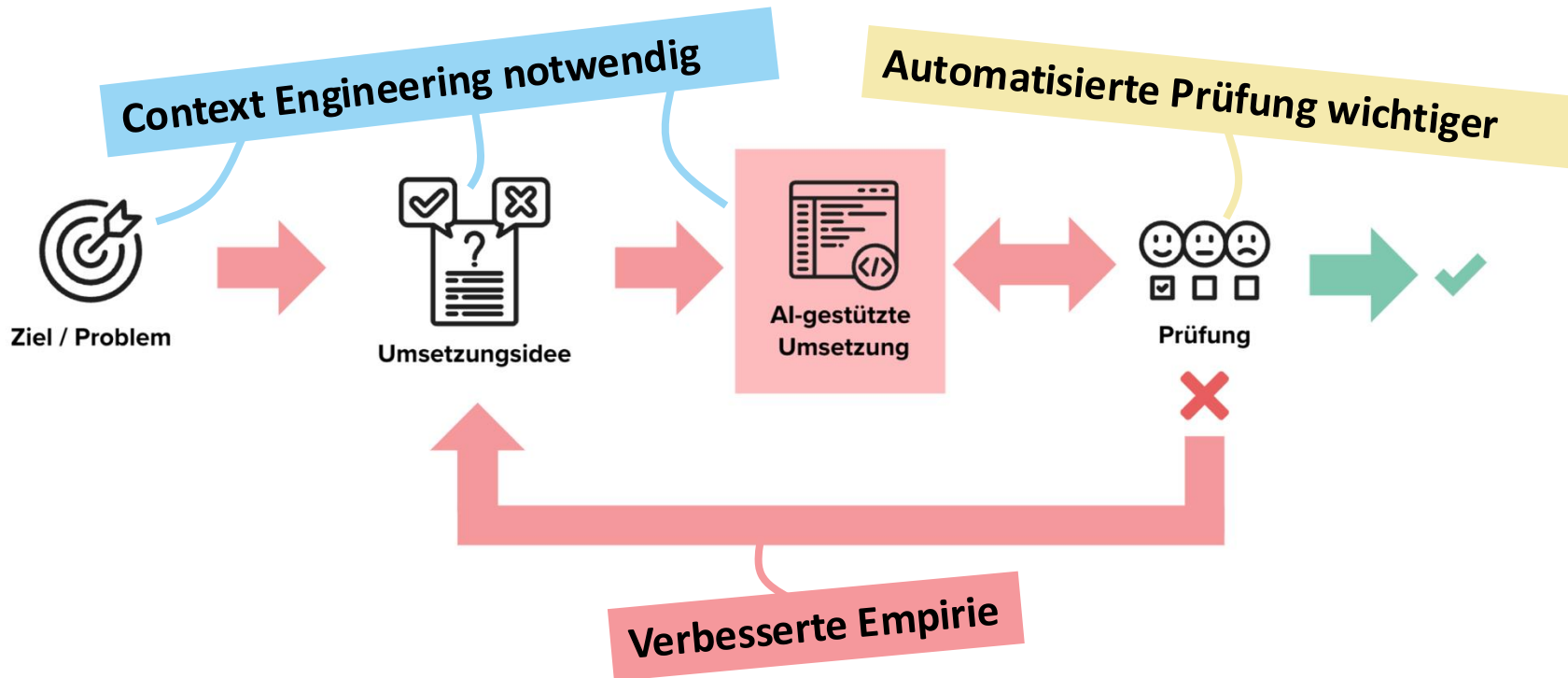
„Context Files Don’t Improve Performance but Increase Cost“

Evaluating AGENTS.md: Are Repository-Level Context Files Helpful for Coding Agents?  
Gloaguen et. al. 2026/09



|           | Type | SONNET-4.5 |             | GPT-5.2    |             | GPT-5.1 M. |             | QWEN3-30B  |             |
|-----------|------|------------|-------------|------------|-------------|------------|-------------|------------|-------------|
|           |      | Steps      | Cost        | Steps      | Cost        | Steps      | Cost        | Steps      | Cost        |
| CTX-BENCH | None | 40.7 ± 3.0 | 1.15 ± 0.10 | 12.1 ± 1.6 | 0.38 ± 0.11 | 40.6 ± 6.7 | 0.18 ± 0.05 | 31.5 ± 3.1 | 0.13 ± 0.02 |
|           | LLM  | 46.5 ± 3.9 | 1.33 ± 0.14 | 13.1 ± 1.5 | 0.57 ± 0.18 | 46.9 ± 6.4 | 0.20 ± 0.04 | 34.2 ± 3.4 | 0.15 ± 0.02 |
|           | DEV. | 45.3 ± 3.6 | 1.30 ± 0.13 | 13.6 ± 1.7 | 0.54 ± 0.23 | 46.6 ± 7.1 | 0.19 ± 0.04 | 32.8 ± 3.7 | 0.15 ± 0.03 |

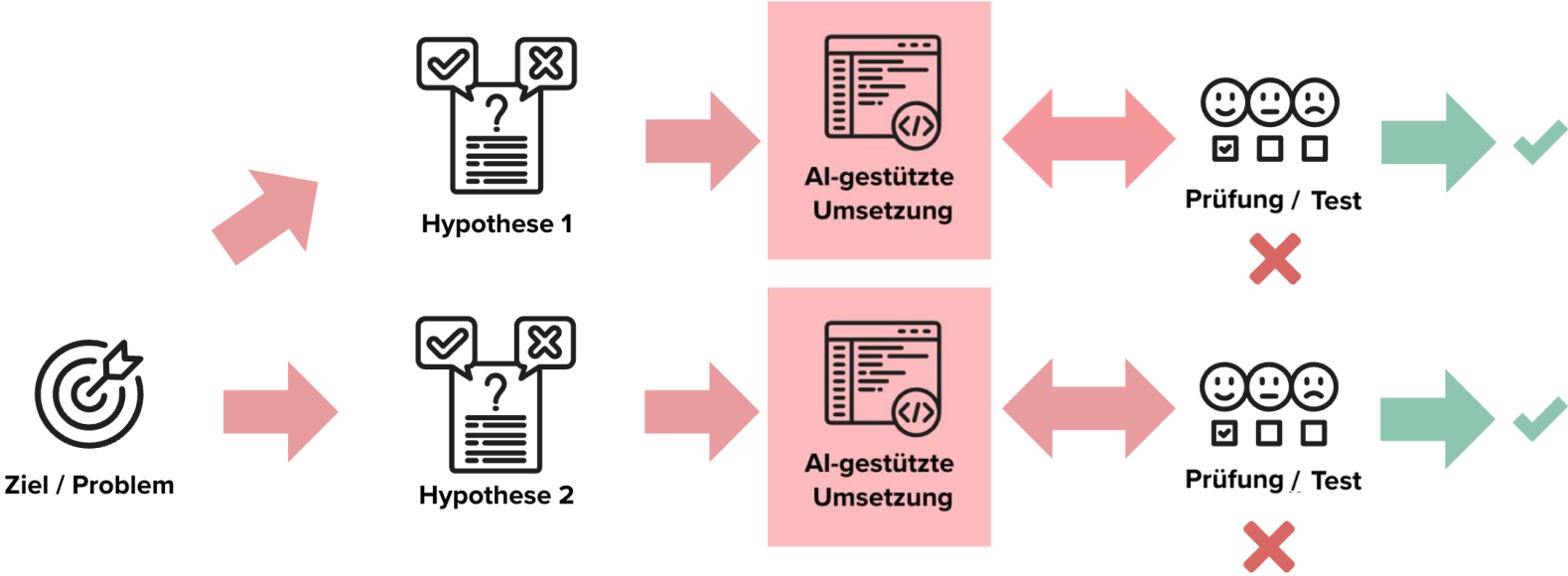
# Generative KI in der Entwicklung



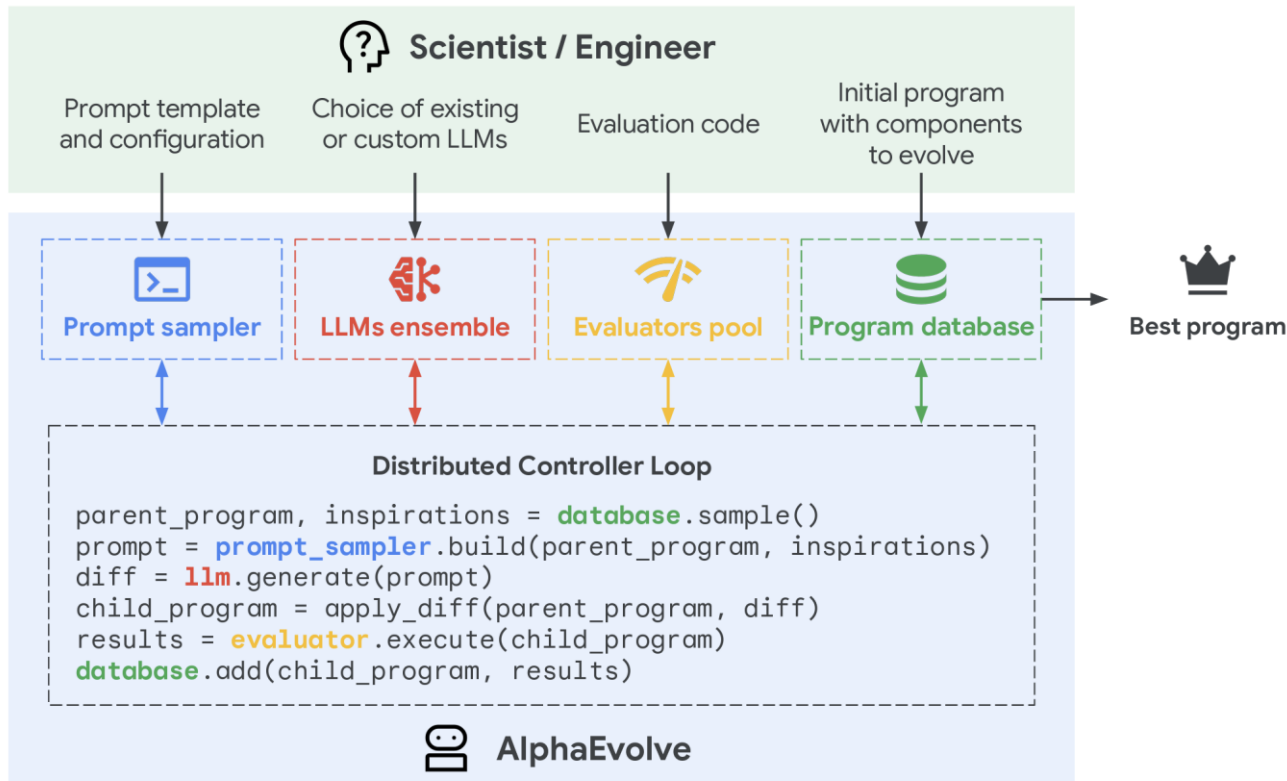
# Verbesserte Empirie

**Kontinuierlicher Lernprozess – individuell und in der Organisation**

# Hypothesen-orientierte Architektur



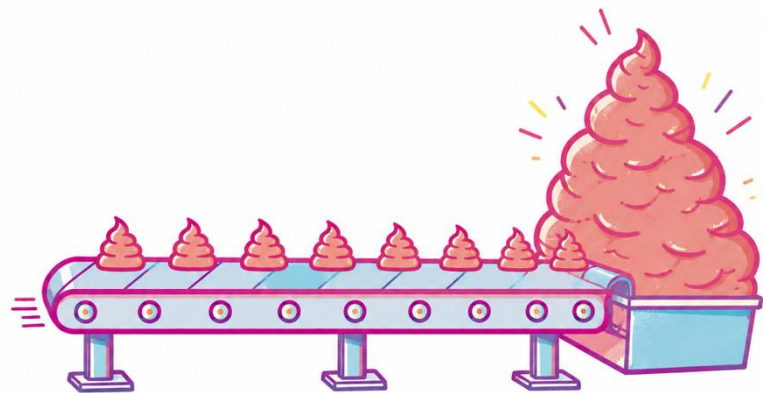
# AlphaEvolve als Beispiel



# 03.

## Vorsicht: Bullshit

Das Problem mit großen  
Hypes...



# AI-Driven Development

Konzepte, Methoden und Tools für den gesamten Software-Lebenszyklus – von der Idee bis zum Betrieb

Schneller entwickeln

Höhere Qualität

Sicherer betreiben

Skalierbar produzieren

Mit Verantwortung



## 1. STRATEGIE & ANFORDERUNGEN

Ziele, Anforderungen und Spezifikationen mit AI definieren



**Intent-Driven Development**  
Von Zielen zu umsetzbarer Software



**PRD Generation**  
Automatische Produktanforderungen



**Spec-Driven Development (SDD)**  
Entwicklung auf Basis von Spezifikationen



**Spec Kit**  
Strukturierte Spezifikationsvorlagen und -werkzeuge



**Ambiguity Detection**  
Mehrdeutigkeiten früh erkennen



**BMAD**  
(Build, Measure, Analyze, Decide)  
AI-gestützter Produktfindungszyklus



**AI-Assisted X**  
AI-gestützte Werkzeuge für alle Disziplinen (DevOps, Security ...)



## 2. AGENTEN & ORCHESTRIERUNG

AI-Agenten effektiv einsetzen und koordinieren



**Multi-Agent Orchestration**  
Mehrere Agenten koordinieren



**Agent Swarms**  
Kollaborative Agenten-Teams



**ReAct**  
(Reasoning and Acting)  
Denken, Planen, Handeln in Schleifen



**Reviewer Agents**  
Automatisierte Code-Reviews



**Agent Skills**  
Spezialisierte Fähigkeiten und Tools



**Agents.md**  
Standard für Agenten-Verhalten und -Konfiguration



**Enterprise Agent Catalog**  
Zentrale Übersicht verfügbarer Agenten



**Engineering Agent Fleets**  
Viele Agenten parallel und kontrolliert einsetzen



## 5. TESTING & QUALITÄT

Qualität mit AI absichern und kontinuierlich verbessern



**AI-Driven Test Engineering**  
Testfälle generieren, ausführen und analysieren



**Evals**  
Systematische Evaluierungen von AI-Ausgaben



**LLM-as-a-Judge**  
LLMs als neutrale Bewertungsinstanz



**Agent Trajectory Evaluation**  
Verhalten und Entscheidungswege bewerten



**Workflow Replay**  
Abläufe reproduzieren und analysieren



**Evidence-Based Agent Completion**  
Nachweise für erfolgreiche Aufgabenbearbeitung

## AI-Native SDLC

End-to-End mit AI



**1. PLAN**  
Strategie & Anforderungen



**2. SPEZIFIZIEREN**  
Design & Architektur



**Prompt Chaining**  
Verkettete Prompts für komplexe Aufgaben



**Hook Engineering**  
Agenten in Workflows und Tools integrieren



**AST-Aware Retrieval**  
Code-Strukturen verstehen und nutzen



**Ask-Before-Act Policies**  
Rückfragen vor kritischen Aktionen



**6. BETREIBEN**  
Betrieb, Monitoring und kontinuierliche Verbesserung



**5. AUSLIEFERN**  
Release & Deployment



**4. PRÜFEN**  
Qualität, Sicherheit und Validierung



**3. ENTWICKELN**  
Implementierung mit Agents



**Human-in-the-Loop (HITL)**  
**Human-on-the-Loop (HOTL)**  
Zusammenarbeit von Mensch und AI-Agenten – mit den richtigen Leitplanken



## 3. KONTEXT & WISSEN

Den richtigen Kontext für bessere Ergebnisse bereitstellen



**Context Engineering**  
Relevanten Kontext strukturiert bereitstellen



**Context Offloading**  
Kontext effizient auslagern und verwalten



**AST-Aware Retrieval**  
Code-Strukturen verstehen und nutzen



**GraphRAG**  
Wissensgraphen für bessere Antworten



**Prompt Chaining**  
Verkettete Prompts für komplexe Aufgaben



**Hook Engineering**  
Agenten in Workflows und Tools integrieren



**Explicit Stop Conditions**  
Klare Abbruchkriterien



**Ask-Before-Act Policies**  
Rückfragen vor kritischen Aktionen



## 4. ENTWICKLUNG & CODE

Mit AI effizienter und sicherer entwickeln



**AI-Driven Development (AIDD)**  
AI als aktiver Entwicklungspartner



**Self-Verifying Code Generation**  
Generierter Code automatisch verifizieren



**Review Noise Reduction**  
Relevante, handlungs-orientierte Reviews



**Architecture Drift Detection**  
Abweichungen von der Zielarchitektur erkennen



**Policy-Aware Coding Agents**  
Unternehmensrichtlinien und Standards beachten



**Progressive Autonomy**  
Autonomieggrad schrittweise und kontrolliert erhöhen



## 6. SICHERHEIT & GOVERNANCE

Leitplanken, Kontrolle und verantwortungsvoller Einsatz



**Agent Guardrails**  
Regeln und Schutzmechanismen



**Agent Identity and Access Management**  
Identitäten, Rollen und Berechtigungen



**Agent Permission Scoping**  
Feingranuläre Berechtigungen



**Sandboxing**  
Sichere Ausführungs-umgebungen



**Ephemeral Execution Environments**  
Temporäre, isolierte Umgebungen



**Agent Privilege Escalation**  
Erkennen und verhindern



**AI Red Teaming**  
Sicherheitsrisiken aktiv testen



## 7. DEPLOYMENT & OPERATIONS

Von der Auslieferung bis zum laufenden Betrieb



**AI Ops**  
AI für den IT-Betrieb



**Intelligent Alert Correlation**  
Intelligente Zusammenhänge in Alerts erkennen



**AI-Driven Software Industrialization**  
Wiederholbar, messbar und skalierbar produzieren



**Inference Optimization**  
Schnellere, kosteneffizientere Inferenz



## 8. MODELLE & INFRASTRUKTUR

Modelle, Tools und Ökosystem



**Low-Rank Adaptation (LoRA)**  
Effiziente Feinabstimmung von Modellen



**Inference Optimization**  
Schnellere, kosteneffizientere Inferenz

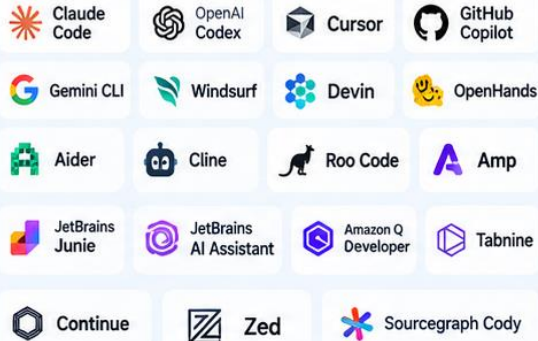


**MCP Registry**  
Model Context Protocol Tools und Integrationen zentral verwalten

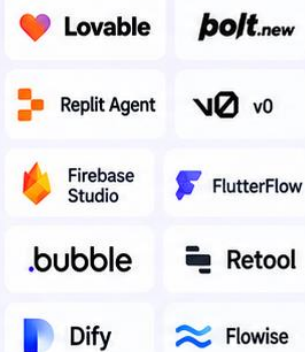
# AI-Driven Development Tools

Auswahl relevanter Tools und Plattformen

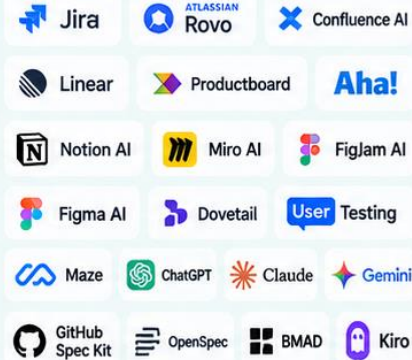
## Coding Agents & AI Coding Assistants



## AI App Builders & Vibe Coding



## Requirements, Discovery & Planning



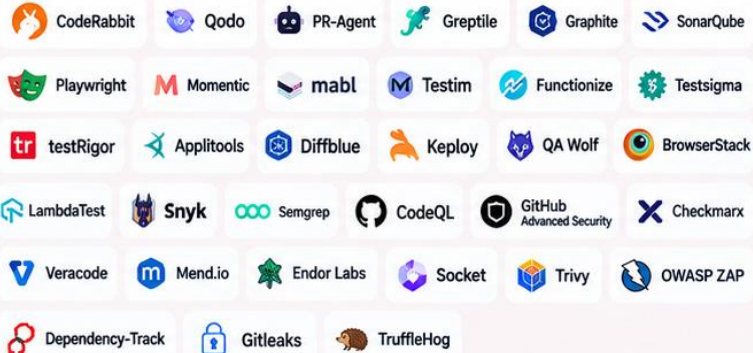
## Agent Frameworks & SDKs



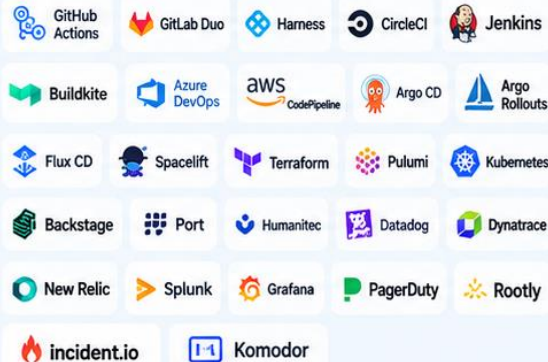
## Protocols & Integration



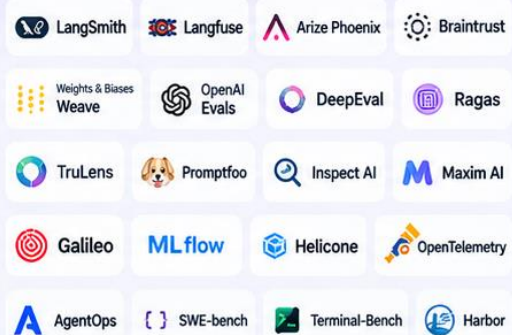
## Code Review, Testing & Security



## DevOps, Platform & Operations



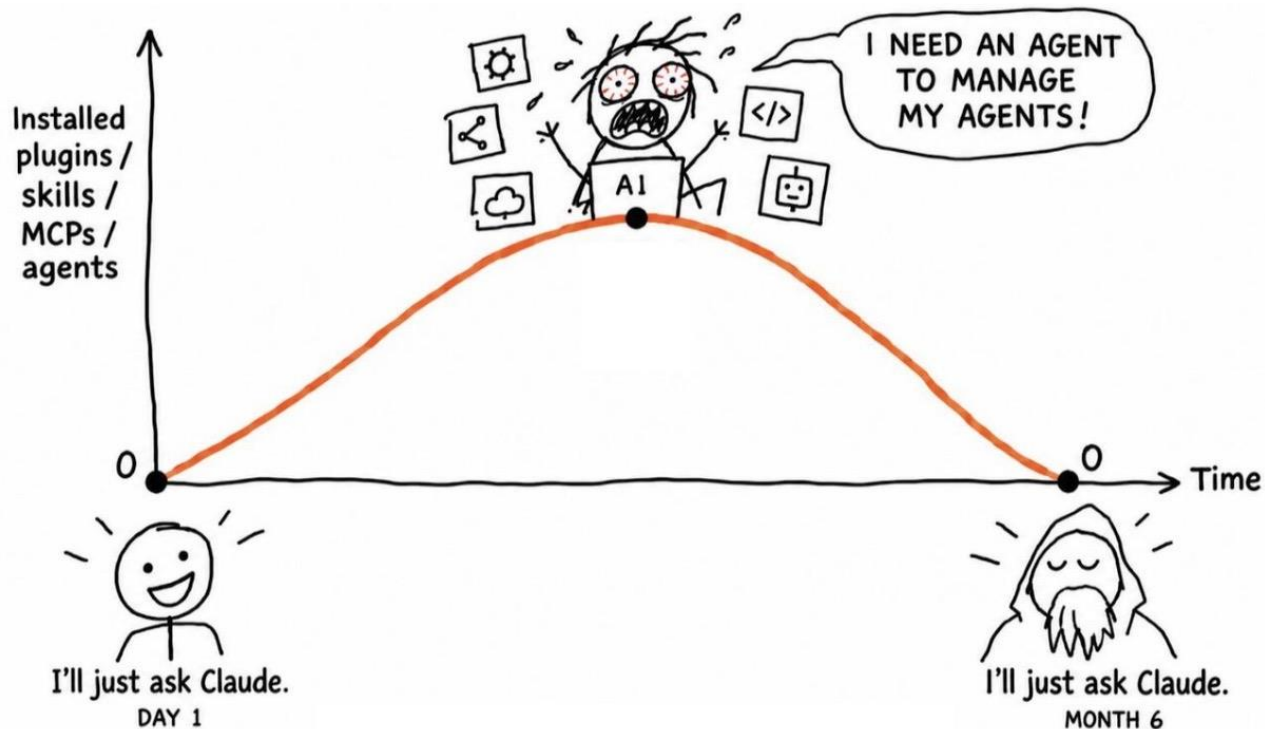
## Evals, Tracing & Observability



# Anatomie von Hypes



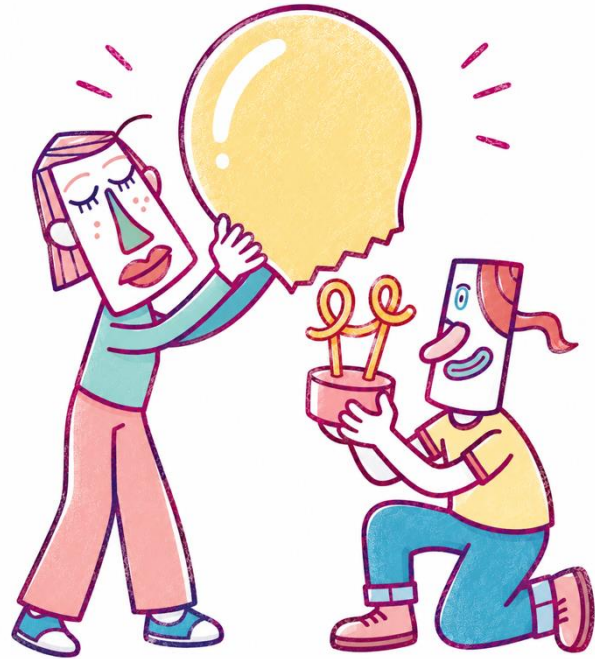
# Nicht aufgeben...



# 04.

## Empfehlungen

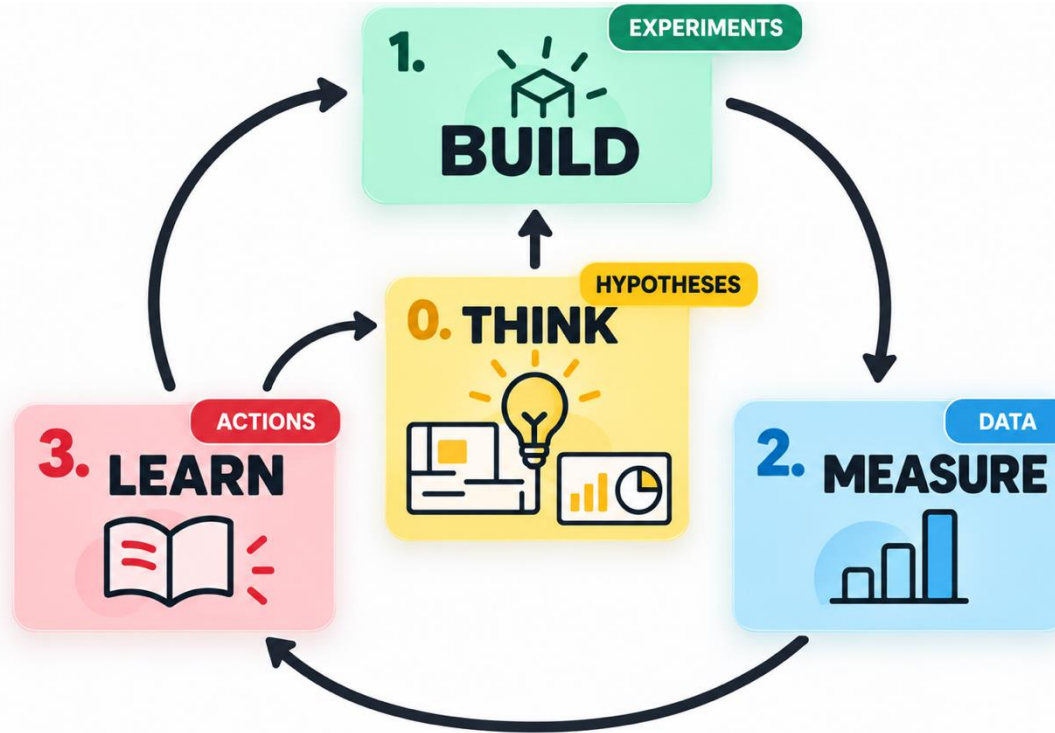
Ordnet „AI Zeug“ ein und  
schaut auf die Effekte...



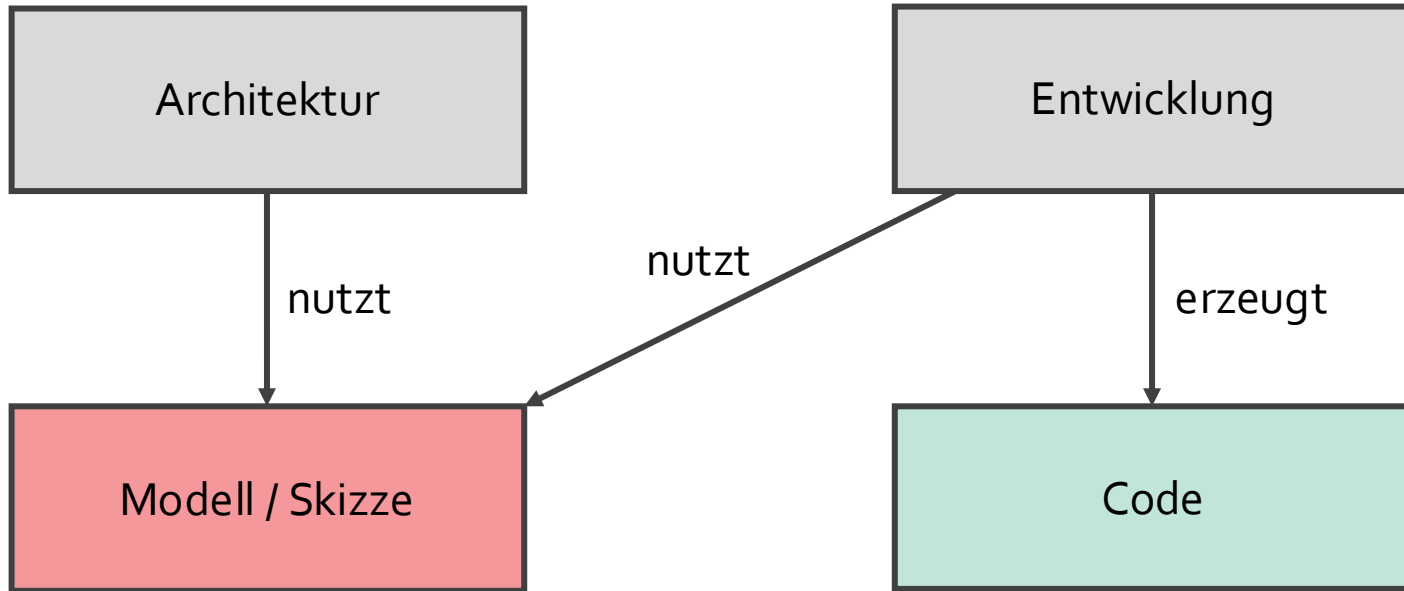
# Achtet auf Seiteneffekte von „Optimierungen“



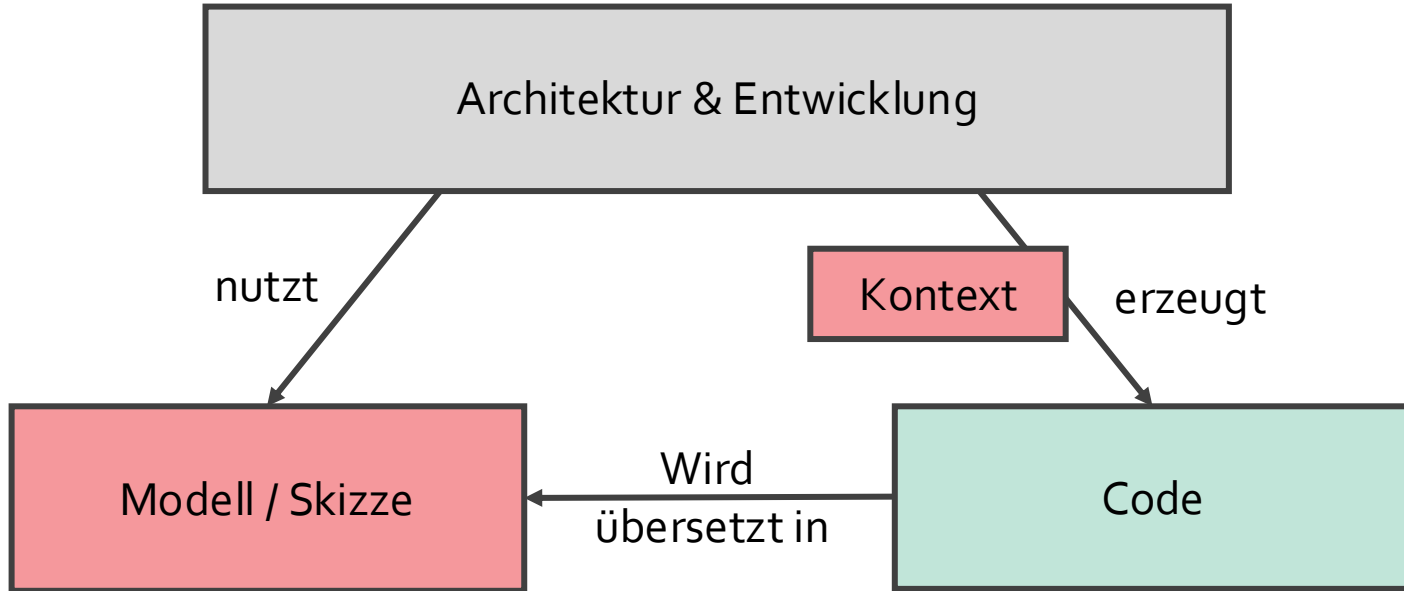
# Ziel ist immer eine schnellere Empirische Schleife



# Architektur und Entwicklung verschmelzen



# Architektur und Entwicklung verschmelzen



# Sustainable Vibe

## Definition

[səˈsteɪ.nə.bəl vaɪb]

Ein positives Gefühl in der Softwareentwicklung, das langfristig und über alle Entwicklungsrollen hinweg aufrechterhalten werden kann.

Teams können KI-unterstützt entwickelte Ergebnisse verstehen, verantworten, betreiben und weiterentwickeln, ohne dass die kognitive Last sie erdrückt.

# iSAQB AGENTA <https://www.socreatory.com/de/trainings/agenta>


**SOCREATORY**

[▶ TRAININGS](#)
[▶ THEMEN](#)
[TRAINER:INNEN](#)
[▶ LEISTUNGEN](#)
[NEWS](#)
[KONTAKT](#)

[LOGIN](#)

[Home](#) > [Trainings](#) > [Architektur im Kontext Agentischer Entwicklung](#)

## Architektur im Kontext Agentischer Entwicklung

Training iSAQB® CPSA®-Advanced AGENTA — 3 Tage

Technik **10**
Methodik **20**
Kommunikation **—**

*KI-Agenten entwickeln schnell – aber ohne dich entwickeln sie falsch. Lerne, wie Architektur und Designaufgaben im Zeitalter von Coding Agents professionell wahrgenommen werden: durch gezieltes Kontext-Engineering, zielorientierte Feedback-Mechanismen und architektonische Leitplanken, die Architektur und Entwicklung effektiv verzahnen.*

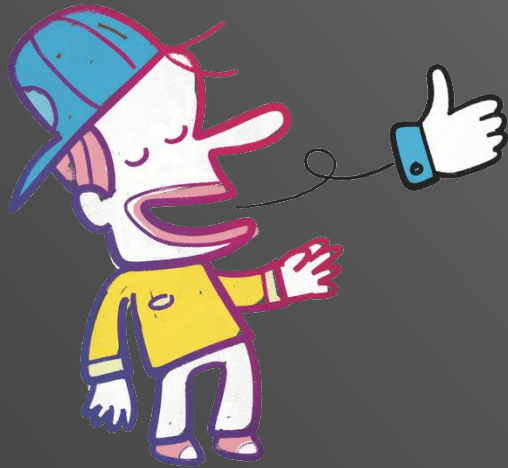
### Vor-Ort-Termine

**AGENTA (Wien) – Niklas Trentmann – Deutsch**  
**Termingarantie**  
14.–16. Dezember 2026  
[Jetzt buchen](#)

**AGENTA (Therme Erding) – Stefan Toth – Deutsch**  
16.–18. März 2027  
[Jetzt buchen](#)

**AGENTA (Online) – Stefan Toth, Niklas Trentmann – Deutsch – Ausgebucht**  
19.–21. Oktober 2026  
[Warteliste](#)

# Feedback & Questions?



Keynote: Von Vibe-Coding zum "Sustainable Vibe"

# Findet mich auf LinkedIn...

---





Kanton Basel-Stadt



+++Ba  
sel++  
++Te+  
+++ch

Basel+

basel.ch



optravis  
company of .msg

cyon



CSS



swiss made  
software

ICT  
Scouts  
Wir fördern Zukunft

SD:>\_  
(SwissDevJobs.ch)

#BaselOne26

baselone.ch