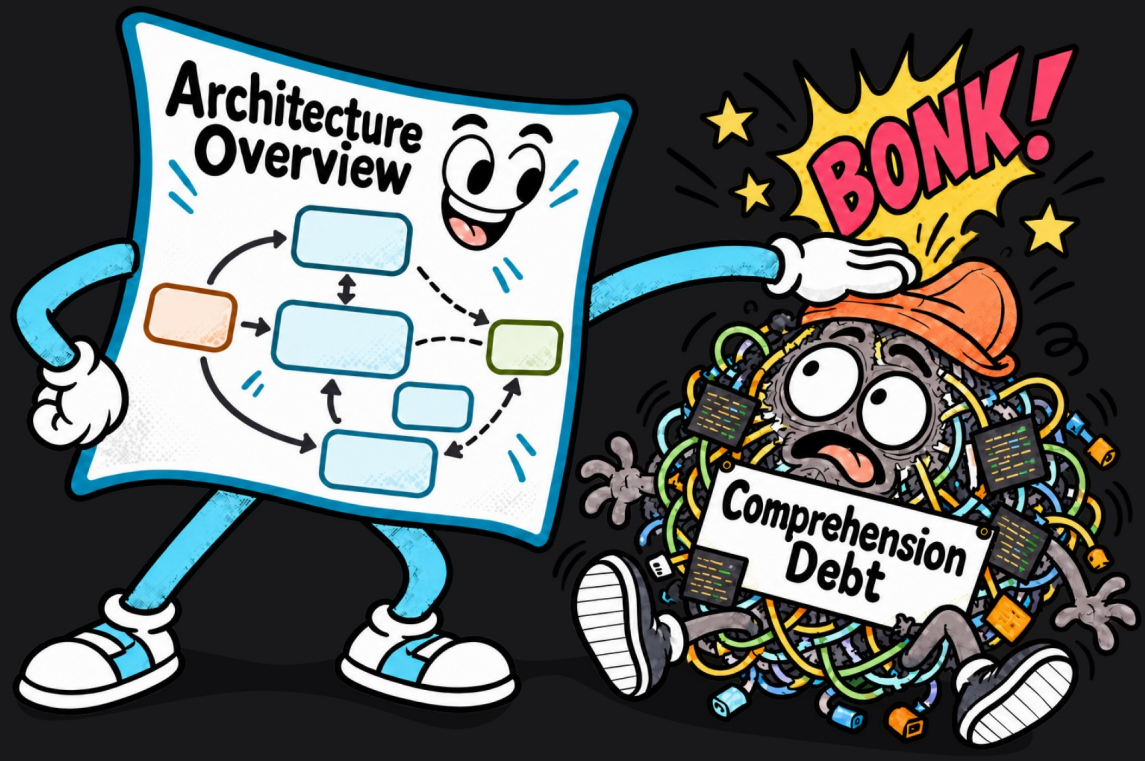
CODE.TALKS 2026 · HAMBURG · 5.11.

Architecture Overview gegen Comprehension Debt

Ein Erfahrungsbericht aus meinem Labor



Nils Eckelt · ecke.lt

WER HIER SPRICHT

Nils Eckelt

Berater und Trainer für Agentic Coding aus Hamburg



Wozu?

Entwickler → UX → Cloud → DevOps → Entwicklungsleitung

Als Entwickler wollte ich wissen, was der Nutzer davon hat, und wurde UX-Designer. Diese Frage hat mich seitdem überallhin gebracht. Dann baute Copilot meinen kleinen Go-Server nach: halb so viele Zeilen, sauberer.

Code skaliert. Verstehen nicht.

WAS EUCH ERWARTET

Die Evolution meines Vorgehens

2025
Vibe Coding

→

AUGUST
Software Factory

→

SEIT SEPTEMBER
Spec, Delta und Karte

An meinen eigenen Tools, neben der Arbeit. Was funktioniert, was nicht, und was ihr morgen ausprobieren könnt.
Work in Progress. Kein Tool zum Herunterladen, keine Studie.

DAS PROBLEM

Lesen wir jede Zeile, verschenken wir den Speed. Lesen wir sie nicht, verstehen wir unser System nicht mehr.

Addy Osmani nennt das **Comprehension Debt**.

Quelle: Addy Osmani, „Comprehension Debt“
https://addyosmani.com/blog/comprehension-debt/

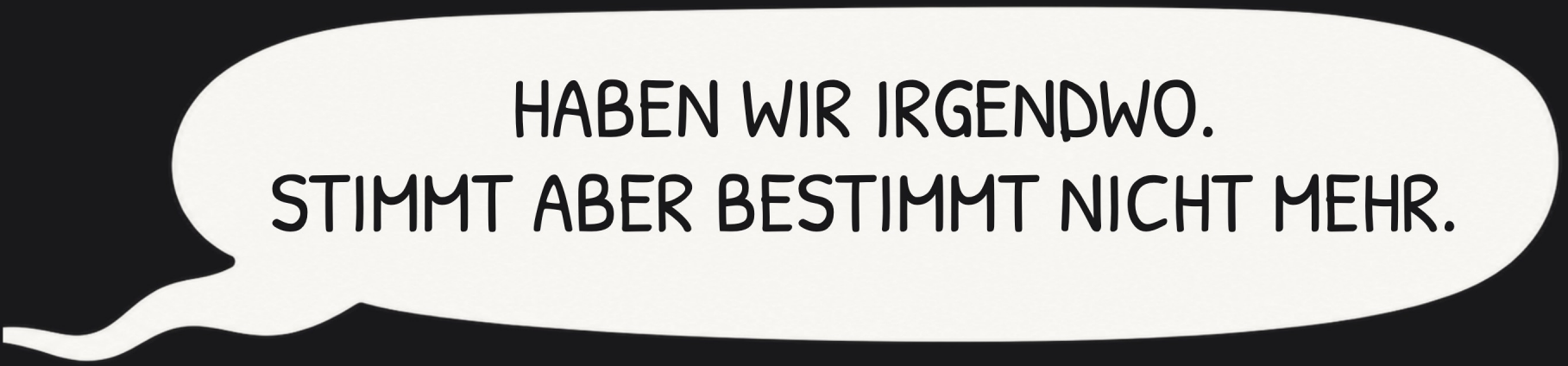
WAS PASSIERT, WENN NIEMAND MEHR LIEST

Lights off: vier Monate, dann abgeschaltet

Dex Horthy (HumanLayer) ließ ab Juli 2025 eine Factory laufen, deren Code kein Mensch las. Nach rund vier Monaten war Schluss: ein Fehler in Produktion, den das Modell nicht fand, und Wochen, um sich in ungelesenen Code einzuarbeiten.

Quelle: Dex Horthy bei The Pragmatic Engineer, 15.7.2026
https://newsletter.pragmaticengineer.com/p/context-engineering-with-dex-horthy

DIE ALTE ANTWORT

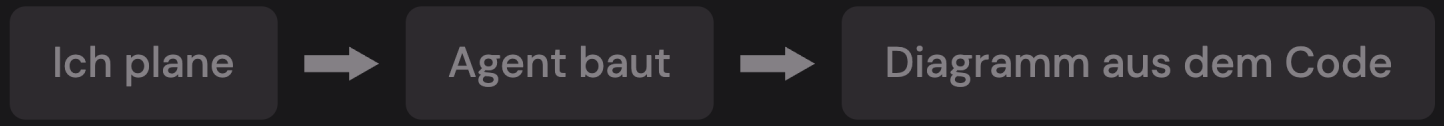


Aus dem Off: Kunden, immer wieder

Architektur-Doku entsteht am Anfang und driftet dann vom Code weg.

DIE WENDE

Aus dem Code generiert zeigt das Diagramm, was der Agent gebaut hat.



Ich will sehen, was ich bauen will, bevor er es baut.



ERST DIE SPEC, DANN DER CODE

Spec im Repo, Änderung als Delta

Ist-Spec
liegt im Repo, beschreibt das heutige Verhalten



Delta-Spec
liegt im Feature-Branch, beschreibt die Änderung



Merge
das Delta wird Teil der Ist-Spec

Quelle: OpenSpec, Spec-Driven Development für KI-Coding-Assistenten
https://github.com/Fission-AI/OpenSpec

Was und Wie, wie ich es trenne

	Was: Spec alles, was von außen sichtbar ist	**Wie: Architektur** Karte und Entscheidungen
IST	Spec › Requirement › Szenario	Manifest je Repo und ADRs
ÄNDERUNG	Delta-Spec	Soll-Manifest und neue ADRs
BEISPIEL	„Buchen sendet eine Bestätigungsmail“	„Worker auf Cloudflare, Mail über Fastmail“
GEPRÜFT DURCH	Tests, je Szenario einer*	Drift auf der Karte**
ICH …	**lasse es mir erklären**	**lasse es mir zeigen**

* So soll es sein. Heute hat noch nicht jedes Szenario einen Test, deshalb das Experiment im leeren Repo.
** Drift: Etwas passt nicht zusammen, etwa ein Vertrag ohne Gegenstück oder ein Szenario ohne Test.

Mit Agenten wandert das Verstehen vom Code in die Spec.

Aber auch Specs liest man irgendwann nicht mehr selbst.

Deshalb: Lass dir das Verhalten erklären und die Struktur zeigen.

NICHT ALLEIN

What-if, bevor gebaut wird

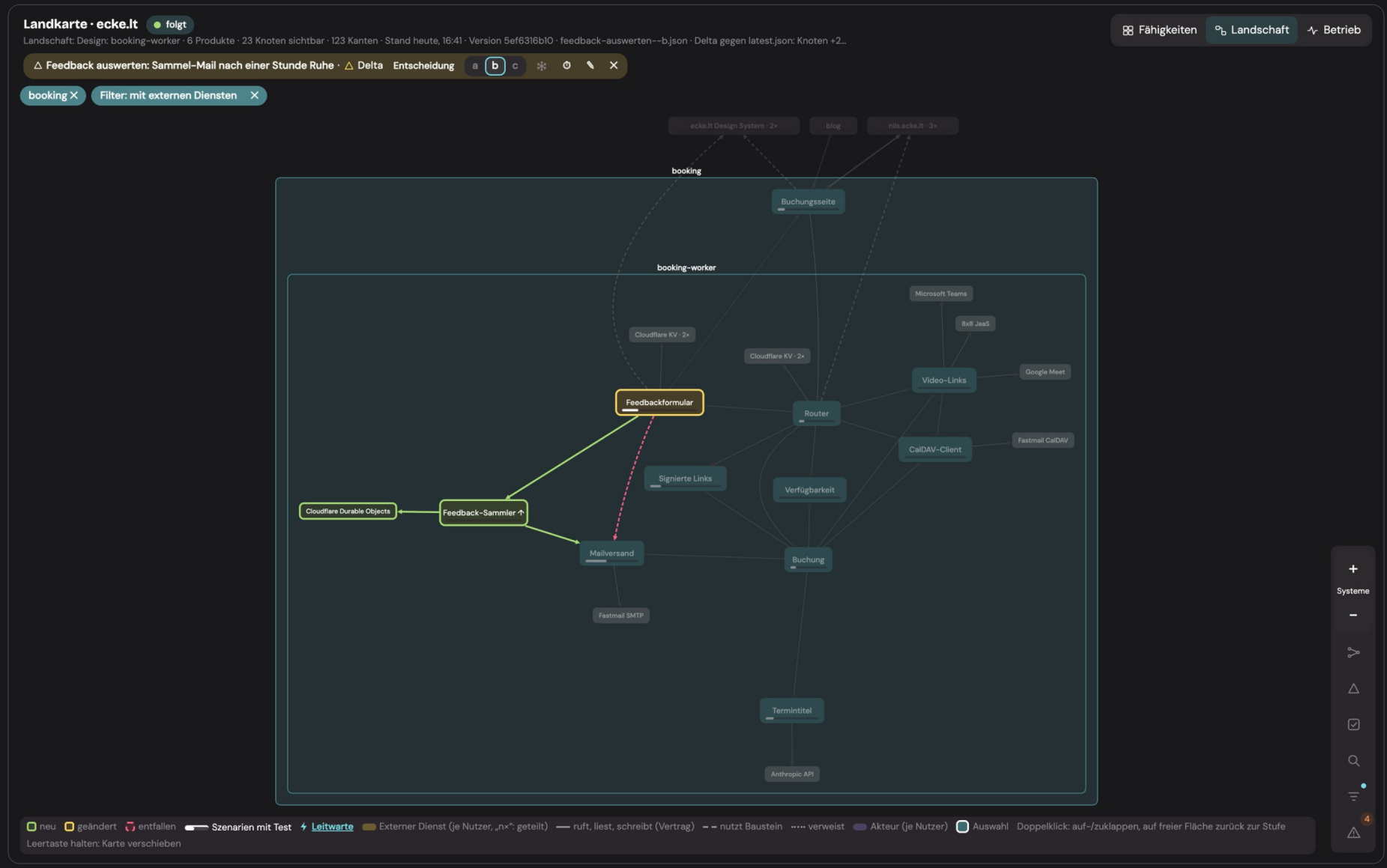
Uncle Bob zeigt ein UML-artiges Modell seines Systems, über das er mit seinem Agenten What-if-Spiele spielt. Der Agent schlägt eine Struktur vor, und erst wenn sie passt, wird daraus Code.

Quelle: Robert C. Martin (Uncle Bob) im Aviator-Podcast
https://www.youtube.com/watch?v=5kCISBJwoZo

DIE LANDKARTE

booking, mit einem Delta darauf

Gelb: geändert
Grün: neu
Rot gestrichelt: entfällt



Nils Eckelt · ecke.lt

Ein echtes Delta aus meinem Alltag

Feedback auswerten: eine Sammel-Mail eine Stunde nach dem letzten Feedback statt einer Mail pro Rückmeldung

1 · Idee

Im Chat: Ich will nicht jede Rückmeldung einzeln lesen.

2 · Delta-Spec

Eine Spec, drei Varianten im Wie:
A Cron
B Durable Object
C mit KI-Zusammenfassung

3 · Karte

In C hängt ein neuer Baustein am Feedbackformular. Und der ruft die Anthropic API.

4 · Aufgefallen

Moment: Dann gehen Rückmeldungen meiner Teilnehmenden an Anthropic.

5 · Zurück im Chat

Neue Anforderung: Datenschutzhinweis am Formular.

6 · Entscheidung: offen

Sicher ist nur: Wird es C, stehen die Datenschutz-Pflichten in der Spec statt im Hinterkopf.

Die Pflicht stand in keinem Ticket. Ich habe sie auf der Karte gesehen.

HAUPTTEIL

So entsteht ein Change auf der Karte

1 · Absicht

ICH + POLIER (AGENT)

Im Pair skizzieren wir den Change.

2 · Übersetzen

ARCHITEKT (AGENT)

Schreibt Soll-Manifest und Delta-Spec in den Change-Branch.

3 · Ableiten

CODE

Auf aktuellen main setzen, verbinden, Soll gegen Ist vergleichen.

4 · Prüfen

MENSCH

Plan auf der Karte ansehen und freigeben.

5 · Umsetzen

GEWERK (AGENT)

Baut Code und Tests nach der Delta-Spec.

6 · Absichern

CODE

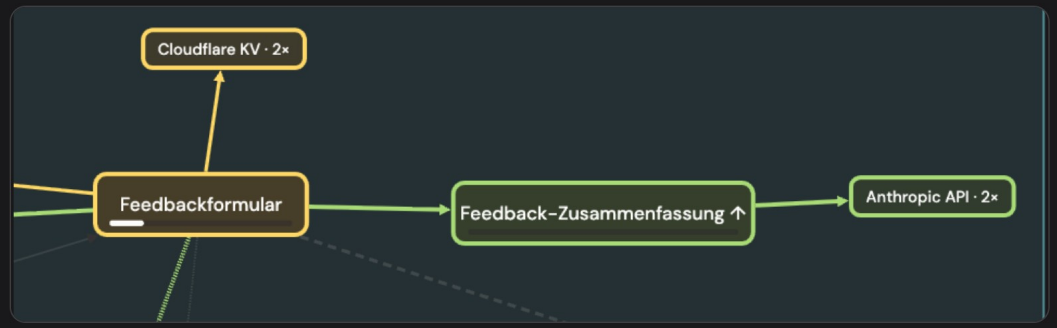
Tests je Szenario, Delta in die Spec mergen.

Jeder Agent-Schritt hat einen Prüfer: 2 prüfe ich in 4, 5 prüfen die Tests in 6.

Mensch und Agent arbeiten auf demselben Modell.

Ich sehe es als Karte, der Agent liest es als Struktur.

ICH SEHE



DER AGENT LIEST

```
{
  "name": "Feedback-Zusammenfassung",
  "interfaces": [
    { "direction": "provides",
      "contractRef": "booking:worker-feedback-summary@1" },
    { "direction": "consumes",
      "contractRef": "anthropic-api:messages@1" }
  ]
}
```

WAS ICH GELERNT HABE

Hilft das gegen Comprehension Debt? — Nicht vollständig. 😖

Ich verstehe viel mehr. Lücken habe ich immer noch, aber ich merke sie jetzt und lasse sie mir vom Agenten erklären.

Diese Woche: Ich dachte, meine Karte entsteht aus der Spec. Erklärt bekommen: Sie entsteht aus dem Manifest.

Kollegentest ohne Erklärung: Die Ebenen hat er verstanden. Was Drift ist, nicht.

Wo das heute endet

Mein Labor

- Ein Ökosystem: Buchung, Blog, Profil
- Eine Person mit Agenten
- Kleine, eigene Systeme

Offen

- Viele Teams, viele Services
- Ob Spec und Karte inhaltlich zusammenpassen, prüft niemand
- Halten Agenten Architekturvorgaben ein?

Als Nächstes: Kopplung im Code prüfen, mit dem modularity-Skill von Vlad Khononov
github.com/vladikk/modularity

AUSBLICK: WAS ICH ALS NÄCHSTES UNTERSUCHE

Spec ins leere Repo, Agent baut ohne Rückfragen

Sechs Läufe

mit der Landkarte selbst als Testobjekt

Specs aus Deltas

entstanden im Gespräch mit dem Agenten und durch eingemergte Deltas

Buchung: nicht nachbaubar

Erster Trockenlauf, Specs aus dem Code. Es fehlen Seitentexte, Mails, Titelregel, Video-Anbieter.

Work in progress: Meine Specs enthielten noch Implementierungsdetails. Im Sinne von Dave Farley zählt eine Spec erst, wenn man sie beweisen kann. Mehr dazu in einem eigenen Talk.

Quelle: Dave Farley, Modern Software Engineering (YouTube, Oktober 2026)
https://www.youtube.com/watch?v=EkH_uoybj8k

ZUM MITNEHMEN

Vier Schritte, ohne mein Tool

1. Ist-Spec ins Repo. Nur das **Was**.
2. Änderungen nur als Delta-Spec. Reviewt wird das Delta, nicht der Code.
3. Das **Wie** getrennt: Manifest und ADRs, daraus eine Übersicht. Mermaid reicht.
4. Die Spec mit einem Rebuild im leeren Repo testen.

Baut euch das Werkzeug, das ihr braucht. Meins ist nur ein Beispiel.

Danke

Feedback, Folien und ein Gespräch mit mir

Nils Eckelt · ecke.lt



feedback.ecke.lt/codetalks