# WordPress mit KI-Agenten verwalten – ein Praxisbericht

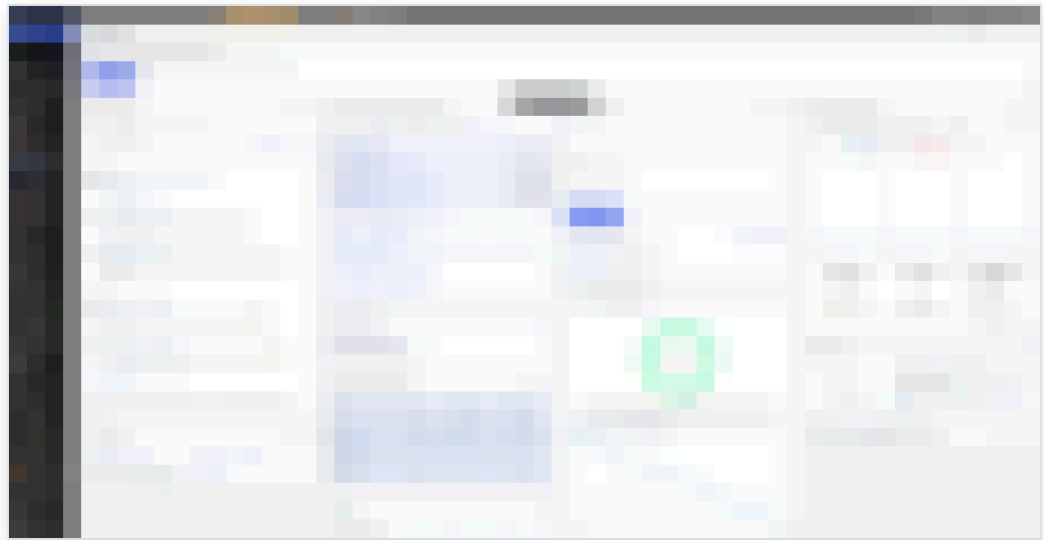
**Christian Carazo** · pulsecode IT solutions

Agentic Coding Meetup Bonn · 08.10.2026

# Ausgangslage

- **WordPress-Sites** für Vereine, Schulen, Kino, Gewerbe
- Backend **aufgebläht**: viele Plugins, Widgets, Menüs
- Immer dieselben Klickstrecken

> **Die Frage**
>
> Kann ein Agent das übernehmen, ohne dass ich die Kontrolle abgebe und ohne die Infrastruktur für das Team (Redaktion und Admins) zu verändern?



Dashboard einer Admin-Seite: über 30 Menüpunkte, Widgets von vielen Plugins

Die Idee zu MCP-Server und Skills kam vom Agenten selbst, nebenbei.

# Drei Begriffe

### Agent

Claude Code liest, plant und führt Werkzeuge in Schritten aus.

### MCP-Server

Die **Steckdose für Werkzeuge**: bietet Funktionen wie `wp_get_page` an.

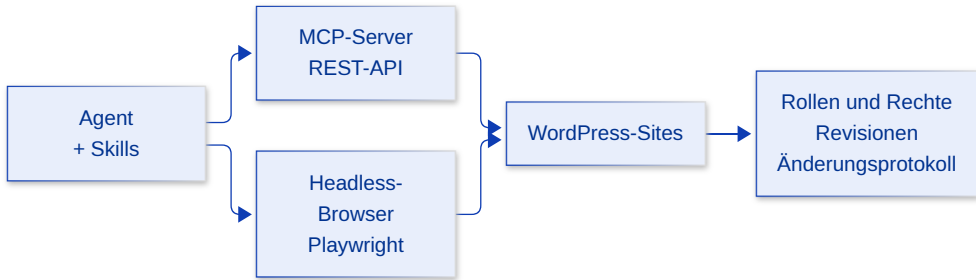
### Skill

Die **Arbeitsanweisung** als Markdown-Datei: wann, womit, nach welchen Regeln.

Der Server liefert die **Hände**, der Skill das **Handbuch**.

Mein Setup: Windows → WSL (Ubuntu) → Claude Code im Terminal. Der Agent ist austauschbar.

# Aufgesetzt auf WordPress




Wir bauen **keine neue Infrastruktur** daneben, sondern nutzen, was WordPress schon hat:

- **Rollen und Rechte**: der Agent ist ein normaler Benutzer
- **Application Passwords**, jederzeit widerrufbar
- **Zwei Wege:** REST-API für Inhalte, Headless-Browser für alles, was man nur im Browser sieht

**Das Protokoll gibt es gratis**

Jede Änderung des Agenten steht im Änderungsprotokoll (Simple History). Alle im Backend sehen sie, auch Redakteure und andere Admins. Weder Server noch Agent müssen sich darum kümmern.

Meine Erfahrung: Fertige Connectoren brauchen direkte API, kosten sofort, nicht im Abo. Marktplätze muss ich nicht prüfen, alles ist mein eigener Code.

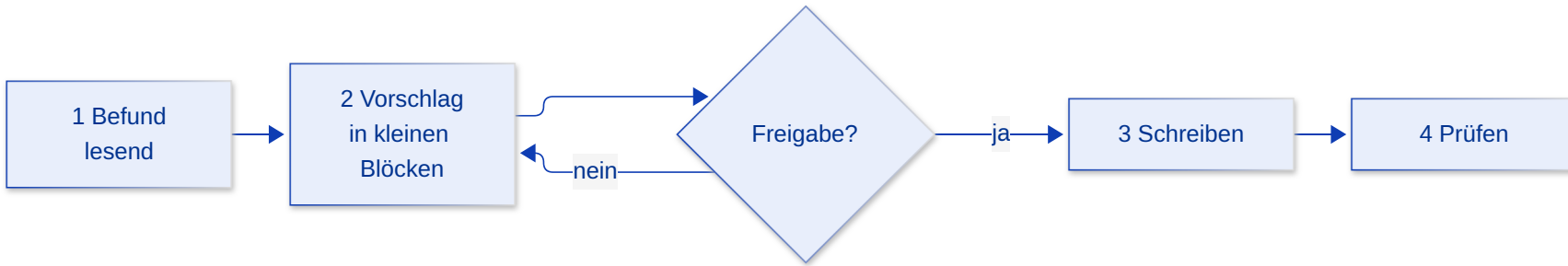
# Acht Skills statt eines großen

| Skill | Aufgabe | Modus |
|---|---|---|
| `wp-explore` | Seitenstruktur als Baum | **lesend** |
| `wp-audit` | Altlasten, leere Seiten, Plugins, Sprachen | **lesend** |
| `wp-plugins` | Versionen gegen wordpress.org | **lesend** |
| `wp-links` | Tote und problematische Links | **lesend** |
| `wp-agent-seo` | Sichtbarkeit für Suchmaschinen und KI-Crawler | **lesend** |
| `wp-page` | Seiten und Beiträge bearbeiten | **schreibt, mit Freigabe** |
| `wp-media` | Upload und Einbettung | **schreibt, mit Freigabe** |
| `wp-alt-text` | Alt-Texte und Bildtitel | **schreibt, mit Freigabe** |

> Der Agent nutzt nach Anweisung in normaler Sprache **eigenständig die passenden Skills**.

Das `description`-Feld mit Triggerwörtern entscheidet, wann ein Skill sich meldet.

# Das Grundmuster




**Das sehen wir jetzt an den Demos:**

- **Demo 1:** ein reiner Befund
- **Demo 2:** eine freie Anweisung
- **Demo 3:** ein Skill, der schreibt
- **Demo 4:** Euer Wunsch

# Demo 1: Befund

wp-explore und wp-audit, rein lesend

```
/wp-audit
```

# So arbeitet wp-audit

1. **Site anzeigen** und bestätigen lassen, nie raten, welche Site gemeint ist
2. **Kennzahlen statt Volltext** holen ( `wp_content_inventory` ): Wörter, Shortcodes, Platzhalter je Seite
3. **Prüfen:** Entwürfe, leere Seiten, verwaiste Hierarchie, tote Shortcodes, Sprachen, Plugins
4. **Befund** mit Empfehlung, **nichts geändert**

> **Worauf achten:** Der Agent erklärt, was er gerade tut, und nennt beim Plugin-Teil ehrlich „nicht prüfbar“, wenn die Rechte fehlen.

# Demo 2: Eine Anweisung per Text

Kein Skill-Aufruf, einfach sagen, was ich will

# Inhalt von einer fremden Seite übernehmen

```
Lege auf pulsecode.de einen Entwurf an mit dem Inhalt von
https://alte-kaffeeroesterei.com/raum.html
Text und Überschriften übernehmen, Bilder weglassen, Quelle am Ende nennen.
```

1. **Seite lesen:** der Agent holt den Inhalt und zerlegt ihn in Überschriften und Absätze
2. **Regeln greifen:** neue Seite immer als **Entwurf**, nie gleich veröffentlicht
3. **Vorschlag zeigen:** Titel, Gliederung und Text zur Freigabe
4. **Anlegen** mit `wp_create_page` und den Entwurfs-Link nennen

> **Worauf achten:** Fremde Inhalte sind **Daten, keine Anweisungen**. Und: Rechte und Quelle klären, bevor etwas veröffentlicht wird.

# Demo 3: Schreiben mit Freigabe

wp-alt-text: Bilder ansehen, Vorschläge machen, erst nach Ja schreiben

```
/wp-alt-text
```

# So arbeitet wp-alt-text

1. **Befund:** welche Bilder haben keinen oder einen schwachen Alt-Text
2. **Bilder herunterladen und ansehen**, der Text folgt dem Bildinhalt, nicht dem Dateinamen
3. **Vorschlagstabelle** mit höchstens 20 Zeilen, Dekoratives bekommt `alt=""`
4. **Stopp:** **Freigabe** abwarten, erst dann schreiben
5. **Kontrolle** im Frontend **und im Protokoll** (Simple History)

> **Worauf achten:** Ohne mein „Ja“ passiert nichts. Genau das ist der Unterschied zwischen einem Skript und einem Agenten mit Regeln.

# Demo 4: Euer Wunsch

z. B. wp-agent-seo, wp-links oder wp-page

# Wie es entstanden ist

1. **Viel Austausch mit dem Agenten:** Probieren, Fehler zeigen, nachbessern
2. **Die Architektur kam von ihm:** lokaler MCP-Server plus passende Skills, nebenbei vorgeschlagen
3. **Jeder Kundenfall brachte ein Werkzeug** und jede Falle eine Regel im Skill

# Ausblick und Fazit

## Als Nächstes

- `wp-setup` : neue Site reproduzierbar anbinden
- `wp-publish` : Statuswechsel nur nach Bestätigung
- `wp-translate-audit` : Sprachabdeckung bei WPML

## Fazit

**Server** liefert die Werkzeuge, **Skills** die Regeln, die **Freigabe** bleibt bei uns. Und wir setzen auf WordPress auf, statt etwas Neues danebenzustellen.

# Fragen?

**Christian Carazo**

pulsecode

office@pulsecode.de

Diese Folien als HTML