Agentic Coding: Schlechte Wahl (mit Beweis und Empfehlung)

Veröffentlicht am 6. März 2026 von Dr. DSGVO · Zuletzt aktualisiert am 10. März 2026 · Lesezeit: 6 Minuten

Kategorien: Datenschutz und Künstliche Intelligenz

Wer „etwas mit KI“ machen will und KI auch für die Programmierung einsetzen möchte, dem fällt oft Agentic Coding als Stichwort ein. Das ist keine gute Idee. Ein Beweis am konkreten Beispiel. KI Coding ist allerdings sinnvoll, wenn es richtig eingesetzt wird.

Agentic Coding ist der Einsatz eines KI-Agenten, um Programmcode zu schreiben. Der Mensch (meist kein Programmierer) tippt dazu seine Anweisung in eine Chatbox und wartet auf das fertige Endergebnis.

Ein Beispiel für eine solche Anweisung, die auch als Prompt bezeichnet wird:

```
Erstelle einen Python Code, der eine Webseite auf Sicherheitslücken testet.
```

Dies ist nur ein simples Beispiel, bitte nicht nachmachen! Der Prompt ist ungenau und sollte so nicht verwendet werden. Es geht in diesem Beitrag nur um die Betrachtung von Ausgaben der KI als künstlicher Programmierer.

Ein KI-Agent arbeitet anders als ein reines Sprachmodell, welches auch als Programmiermodell eingesetzt werden kann. Agenten sind in einem extra Artikel ausführlicher beschrieben. Kurz gesagt, läuft das Agentic Coding in einer autonomen Schleife, während bei der „normalen“ KI-Programmierung der Mensch nach jedem Schritt eingreift, Entscheidungen trifft und neue Anweisungen gibt.

Der Vorteil des Agenten: Der Mensch muss ihn nur mit einer initialen Anweisung starten und kann dann andere Dinge tun. Das Ergebnis ist im Idealfall ein perfekt laufendes Programm. Die Realität sieht anders aus, wie nachfolgend gezeigt wird. Auch in einem vorigen Beitrag wurde bereits auf die Nachteile des Agentic Coding eingegangen.

Ein Zwischenergebnis des Agenten könnte folgende Routine sein, die auf einer Webseite (HTML) prüft, ob ein Eingabefeld in einem Javascript-Code ausgewertet wird:

```
def _input_name_used_in_js(name: str, js_content: str) -> bool:
 e = re.escape(name)
 patterns = [
  rf'getElementById\s*\(\s*["\']{ e }["\']',
  rf'querySelector\s*\(\s*["\']#{ e }["\']',
  rf'querySelector\s*\(\s*["\'][^"\']*#{ e }[^"\']*["\']',
  rf'["\']{ e }["\'].{{0,200}}(?:innerHTML|eval|location|fetch|write)',
  rf'(?:innerHTML|eval|location|fetch|write).{{0,200}}["\']{ e }["\']',
 ]
 return any(re.search(p, js_content, re.DOTALL) for p in patterns)
```

Dieser Code (hier leicht gekürzt) sieht nicht nur kompliziert aus, er ist es auch. Kein Mensch, auch nicht der beste Programmierer der Welt, sieht sofort, was hier passiert. Grund sind sogenannte reguläre Ausdrücke, die zum sehr schnellen und flexiblen Finden von Texten nützlich sind.

Im genannten Code fehlt ein wichtiger Sonderfall, der sehr häufig auftritt: Die Berücksichtigung der populären Javascript-Bibliothek jQuery. Beim manuellen Prompten der KI, mit der Bitte um Berücksichtigung von jQuery, kam dann folgendes heraus:

```
def _input_name_used_in_js(name: str, js_content: str) -> bool:
    e = re.escape(name)
    patterns = [
        rf'getElementById\s*\(\s*["\']{ e }["\']',
        rf'querySelector\s*\(\s*["\']#{ e }["\']',
        rf'querySelector\s*\(\s*["\'][^"\']*#{ e }[^"\']*["\']',
        rf'\$\(\s*["\']#{ e }["\'\s)][^;]{{0,80}}\.val\s*\(',
        rf'#{ e }["\'][^;]{{0,80}}\.val\s*\(',
        rf'\$\s*\(\s*["\']#{ e }["\']',
    ]
    return any(re.search(p, js_content, re.DOTALL) for p in patterns)
```

Dem geübten Auge fällt auf, dass drei neue Muster-Zeilen (*patterns*) von der KI hinzugefügt wurden und zwei vorher vorhanden Zeilen (oben fett gedruckt) gelöscht wurden. Um dies herauszufinden, war es nicht nötig, den Code vollständig zu verstehen. Zählen und „gesunder“ Software-Entwickler-Verstand reichten aus.

Richtig wäre es gewesen, die drei neuen Muster-Zeilen hinzuzufügen (was passiert ist) und die zwei gelöschten Zeilen nicht herauszulöschen, sondern beizubehalten.

Dem Menschen (mir) ist dies aufgefallen.

> KI-Agenten verfälschen mitunter unbemerkt Ergebnisse.

Bei Nutzung eines KI-Agenten wäre dieses Problem unbemerkt entstanden. Niemand hätte es bemerkt. Das Programm wäre falsch gewesen, weil es wichtige Konstellationen nicht berücksichtigt hätte. Das wäre irgendwann jemandem aufgefallen. Dann hätte derselbe unfähige „Programmierer“, der den KI-Agenten bedient hat, wieder den KI-Agenten bemühen müssen, um vielleicht und nach vielen unbefriedigenden, kostspieligen Versuchen das gewünschte Ergebnis zu erhalten.

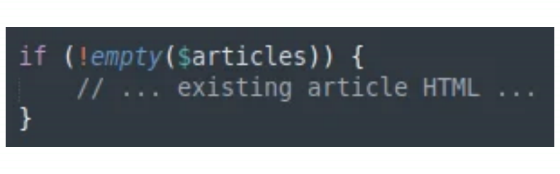
Ein anderes Beispiel:

Der Originalcode sah so aus:

```
if (!empty($articles)) {
    $html .= '<div class="article-list">';
    foreach ($articles as $article) {
        $parsed_url = parse_url($article->url);
        $path = $parsed_url['path'] ?? '/';

        // Kürze sehr lange Pfade
        if (strlen($path) > 50) {
            $path = substr($path, 0, 47) . '...';
        }

        $status = $article->read_fully == 1
            ? '<span class="read-status read-fully">✅</span>'
            : '<span class="read-status read-partial">👁</span>';

        //$html .= '<div class="article-item">' . $status . $path . '</div>';
        $html .= '<div class="article-item">' . $status . '<a href="' . esc_url($artic
    }
    $html .= '</div>';
}
```

Original-Programmcode, hier für die Abbildung gekürzt.

Der KI-Agent erweiterte diesen Code um Teile davor und danach. Aus dem Original wurde allerdings das hier:



```
if (!empty($articles)) {
    // ... existing article HTML ...
}
```

Code, der durch einen KI-Agenten ausgelöscht wurde.

Anstatt das Originalprogramm zu erweitern, wurde das Original durch einen Kommentar ersetzt ("//" ist das Kommentarzeichen). Das Programm funktionierte weiterhin, aber ohne eine wichtige Zusatzfunktion.

Das Ende der Menschheit

Ein „Prozessautomatisierer“ hat diesen Beitrag auf LinkedIn kommentiert. Er meinte, dass solche „Copy & Paste Fehler“ in seinem Prozess nicht vorkommen. Außerdem sprach er davon, dass „gute Agenturen“ solche Probleme vermeiden können.

Richtig ist: Erstens handelt es sich nicht um ein Copy & Paste Problem, sondern um einen KI-Fehler. Zweitens würde sich keine Software-Entwickler-Firma, die etwas auf sich hält, als „Agentur“ bezeichnen. Wer den Agenturbegriff entleiht, der hat es anscheinend sehr nötig. Der häufigste Grund für solche sprachlichen Ausflüge dürfte eine fehlende Kompetenz in der Software-Entwicklung sein.

Ein anderer kritisierte den oben gezeigten Code, weil er meinte, da fehle etwas. Er komme aus dem Lachen nicht mehr heraus. Richtig ist: Unter dem Code steht „hier leicht gekürzt“. Es geht hier nicht um den konkreten Programmcode, sondern darum, dass der KI Agent mehrere vorhandene, richtige Code-Zeilen fälschlicherweise herausgelöscht hat.

> KI ist bereits intelligenter als der Mensch.
>
> Siehe die drei Belege in diesem Abschnitt.

Ein Dritter meinte: „Aber Vorsicht bei Tests auf die der Agent vom Repo [Repository] aus Zugriff hat.“ Richtig ist: In diesem Beitrag geht es gerade darum, dafür zu sensibilisieren, dass KI-Agenten keine gute Idee sind.

Gute Nacht, Menschheit. Halbwissende kosten nicht nur Zeit und rauben Energie, sondern verhindern auch noch Fortschritt.

Fazit

Die meisten, die KI-Agenten für die Programmierung nutzen, können nicht programmieren. Deshalb nutzen sie Agenten.

Wer privat oder als Einzelunternehmer mit Agenten programmiert, macht erst einmal nichts falsch. Immerhin entstehen so Ergebnisse, die ansonsten gar nicht entstanden wären. Viel Spaß allerdings beim Debuggen (Fehler finden). Auch hier gilt, wie in allen anderen Lebensbereichen: *A Fool With A Tool Is Still A Fool.*

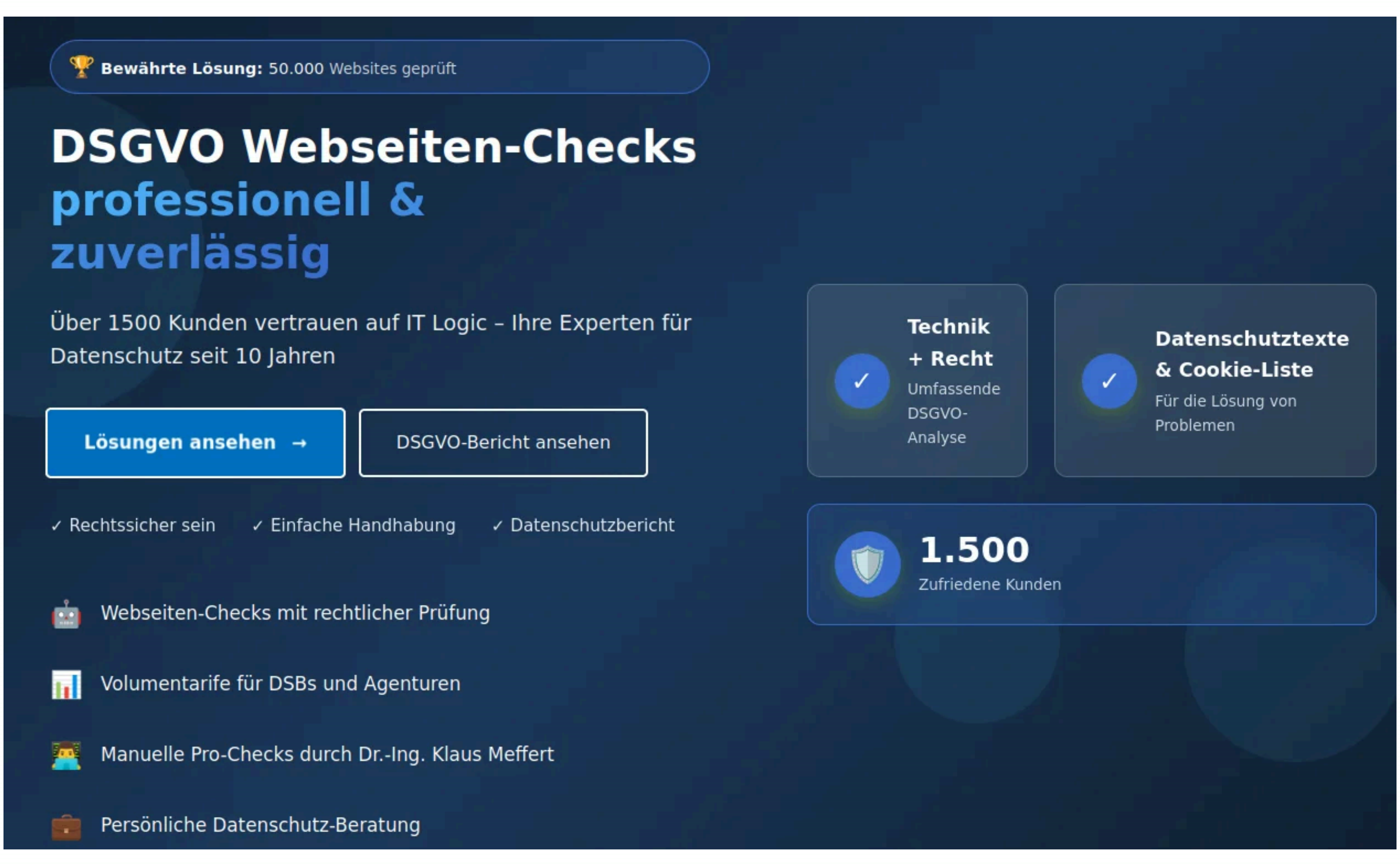
Wenn allerdings mittelständische Unternehmen mit mehreren Angestellten mit Agentic Coding anfangen, dann zeugt das von gewissen Missständen. Das werden einige dieser Firmen nicht einsehen. Sie werden weiter Ressourcen verbrennen oder Potential auf der Straße liegen lassen.

> Eine Black Box zu durchschauen, indem man sie transparent macht, gelingt nur einem Experten.

Richtig funktioniert Programmierung ohne KI-Agenten, aber mit Hilfe von KI. Wer ohne KI programmiert, hat die Kontrolle über sein Leben verloren. Wer mit KI-Agenten programmiert, hat noch weniger Kontrolle.

Wie das gezeigte Programm-Beispiel demonstriert, kann ohne KI nicht mehr effektiv programmiert werden. Wie es mit KI richtig geht, erfahren Sie gerne in einem Workshop für Entwickler.

Auch dieses Widget wurde mit Hilfe von KI entwickelt. Möglich sind diverse Stile. Was hier wie eine Spielerei aussieht, erlaubt es, in wirtschaftlicher Weise moderne Websites und ansprechende Bestellformulare zu bauen. Das Tool, das vom Widget verlinkt wird, wurde ebenfalls mit Unterstützung von KI gebaut und bedient sich zusätzlichen Logiken, die ohne KI entstanden sind.



🏆 **Bewährte Lösung:** 50.000 Websites geprüft

DSGVO Webseiten-Checks professionell & zuverlässig

Über 1500 Kunden vertrauen auf IT Logic – Ihre Experten für Datenschutz seit 10 Jahren

✓ Rechtssicher sein ✓ Einfache Handhabung ✓ Datenschutzbericht

- 🤖 Webseiten-Checks mit rechtlicher Prüfung
- 📊 Volumentarife für DSBs und Agenturen
- 👨‍💻 Manuelle Pro-Checks durch Dr.-Ing. Klaus Meffert
- 💼 Persönliche Datenschutz-Beratung

✓ **Technik + Recht** Umfassende DSGVO-Analyse

✓ **Datenschutztexte & Cookie-Liste** Für die Lösung von Problemen

1.500 Zufriedene Kunden

Beispiel für eine Website (Auszug), die mit KI-Unterstützung entwickelt wurde. Ohne Einsatz von Agentic Coding.

Grundlage ist „nur“, dass der KI-Programmierer weiß, wie man programmiert und wie man mit KI richtig programmiert.

> Das Richtige tun und es richtig tun.

Wer schreibt hier?

Mein Name ist Klaus Meffert. Ich bin promovierter Informatiker und beschäftige mich seit über 30 Jahren professionell und praxisbezogen mit Informationstechnologie. In IT & Datenschutz bin ich auch als Sachverständiger tätig. Ich stehe für pragmatische Lösungen mit Mehrwert. Meine Firma, die IT Logic GmbH, berät Kunden und bietet Webseiten-Checks sowie optimierte & sichere Lösungen an (mit und ohne KI).

Bitte nutzen Sie bei Verwendung meiner Ergebnisse die Quellenangabe oder verlinken Sie gut wahrnehmbar auf diesen Artikel:

Quelle: Klaus Meffert, Dr. DSGVO Blog, Link: https://dr-dsgvo.de/agentic-coding-schlechte-wahl-mit-beweis

Einen Kurzlink oder eine Bestätigung für Ihre Quellenangabe erhalten Sie kurzfristig auf Anfrage ✉. Ein Teilen oder Verteilen dieses Beitrags ist natürlich ohne weiteres möglich und gewünscht.