

AGENTIC OS / 29. SEPTEMBER 2026

Agenten ohne Chatfenster:

Wenn Arbeit von selbst beginnt

Cron, Sentry-Webhooks und Aufträge aus ChatGPT an OpenClaw

Marcel Wege, byte5 GmbH

08:30 Uhr. Der Bericht liegt bereit.

Was ist seit gestern passiert?

Was verdient heute unsere Aufmerksamkeit?

Ein Agent hat die Daten gelesen und einen Entwurf erstellt.

Der Zeitplan hat den Auftrag gestartet.

Ein Checkoutfehler braucht Einordnung

Die Daten

Cron und ChatGPT: Fiktiver Vorfall
NORDLICHT-42 aus lokalen Beispieldaten.

Webhook: Ein synthetischer Checkoutfehler, den
Sentry tatsächlich erfasst.

Der Auftrag

Ein kurzes Incident-Briefing als Datei:

- Einschätzung mit Begründung
- Belegte Beobachtungen
- Nächster Schritt und offene Frage

Zeitplan, Ereignis oder Werkzeugaufruf

Startsignal	Beispiel	Wer gibt den Anstoß?
Cron	Briefing morgens um 08:30	Scheduler
Webhook	Sentry erfasst ein neues Demo-Issue	Monitoring
MCP-Aufruf	ChatGPT delegiert ein Briefing an OpenClaw	ChatGPT als Client

Alle Wege sollen einen nachvollziehbaren Bericht liefern.

Der Weg vom Signal zum Ergebnis

- 1** **Eingang:** Zeitpunkt, Ereignis oder Werkzeugaufruf
- 2** **Auftrag:** feste Aufgabe, erlaubte Daten, definiertes Ziel
- 3** **OpenClaw:** Agent ausführen und Werkzeuge aufrufen
- 4** **Ergebnis:** Briefing speichern und Abschluss nachweisen

Welcher *Auslöser* passt?

Situation	Eure Wahl
Jeden Werktag ein Morgenbriefing	Cron / Webhook / MCP
Monitoring meldet einen neuen Ausfall	Cron / Webhook / MCP
Ein Analyseagent braucht eine zweite Einschätzung	Cron / Webhook / MCP

Welcher fachliche Anlass spricht für eure Wahl?

01 / CRON

Ein Zeitpunkt wird zum *Auftrag*:

Der Gateway-Scheduler startet vorbereitete Arbeit.

Einmal, regelmäßig oder nach Kalender

Zeitplan	Geeigneter Fall
Ein bestimmter Zeitpunkt	Einmalige Nachprüfung nach einem Release
Ein festes Intervall	Wiederkehrende Sichtung einer Arbeitsliste
Cron-Ausdruck mit Zeitzone	Werktags um 08:30 Uhr

Für einen fachlichen Termin gehört die **Zeitzone** zur Definition.

Jeden Werktag um 08:30 Uhr

```
30 8 * * 1-5
```

Minute	30
Stunde	8
Tag / Monat	beliebig
Wochentag	Montag bis Freitag
Zeitzone	Europe/Berlin

Der Zeitplan bestimmt den Start. Der Auftrag beschreibt die Arbeit.



Das Briefing startet nach Zeitplan:

Auslöser: Ein geplanter Zeitpunkt ist erreicht.

Auftrag: Ein Briefing zum Vorfall NORDLICHT-42.

Ergebnis: Eine neue Datei mit zugehöriger Laufkennung.

Entscheidend ist der Inhalt des fertigen Briefings.

Ein eigener Job oder eine regelmäßige Sichtung

Cron

Ein abgegrenzter Auftrag mit eigenem Zeitplan.

Beispiel: Jeden Werktag ein Briefing erstellen.

Heartbeat

Ein periodischer Agentenlauf prüft, ob etwas Aufmerksamkeit braucht.

Beispiel: Aufgaben sichten und nur bei Bedarf melden.

02 / WEBHOOK

Sentry meldet einen Checkoutfehler:

Eine Alarmregel im Demo-Projekt sendet einen signierten Webhook.

Ein Alarm mit Projekt- und Issue-Kennung

```
{  
  "action": "triggered",  
  "data": {  
    "event": { "project": 4512169615687680,  
              "issue_id": "7761201292" },  
    "triggered_rule": "Notify via Agentic OS Checkout Demo"  
  }  
}
```

Die Kennungen verbinden Sentry-Alarm, Auftrag und Bericht.

Der Adapter übersetzt den Webhook

```
Sentry:    POST /webhooks/sentry  
Adapter:   Signatur, Projekt und Duplikate prüfen  
OpenClaw:  POST /hooks/agent
```

```
{  
  "message": "Erstelle das Briefing zum Sentry-Event.",  
  "sessionKey": "agent:agentic-os-sentry:hook:live",  
  "sessionMode": "persistent",  
  "deliver": false  
}
```

Der Bericht erscheint in der festen Session und als Datei.



Ein Klick im Checkout startet die Kette:

Checkout: Ein synthetischer Fehler entsteht.

Sentry: Ein neues Issue löst den Webhook aus.

OpenClaw: Der Agent erstellt das Incident-Briefing.

Eine Issue-Kennung verbindet Fehler, Auftrag und Bericht.

Angenommen, gestartet, abgeschlossen

Beobachtung	Was sie belegt
Empfänger bestätigt den Sentry-Webhook	Das Ereignis ist angenommen
Ein OpenClaw-Lauf ist protokolliert	Die Agentenarbeit hat begonnen
Ein neuer Bericht liegt vor	Der Agent hat ein Artefakt erzeugt
Kennung und Inhalt passen	Das Ergebnis gehört zu diesem Auftrag

Die frühe Webhook-Antwort wartet nicht auf den fertigen Bericht.

Derselbe Alarm. Zwei Zustellungen.

Der Sender wartet auf eine Antwort.
Sein Timeout läuft ab. Er sendet erneut.

- A Beide Meldungen starten jeweils einen Auftrag.
- B Eine gespeicherte Ereigniskennung verhindert doppelte Arbeit.

Welche Information brauchen wir, bevor wir entscheiden?

03 / MCP MIT CHATGPT

ChatGPT delegiert an OpenClaw:

Das Gespräch führt ChatGPT. Die beauftragte Arbeit erledigt OpenClaw.

Briefing anfordern und Ergebnis lesen

```
create_incident_brief
  focus: diagnostic oder handover
  request_id: Kennung dieses Auftrags

read_incident_brief
  run_id: Kennung des gestarteten Laufs
  wait_seconds: 0 bis 20
```

Vorfall, Zielagent und Ergebnisort stehen im Adapter fest.

ChatGPT wählt eine angebotene Funktion und ihre Parameter.

ChatGPT erreicht den lokalen Agenten

Komponente	Aufgabe
ChatGPT Web	Auftrag formulieren und MCP-Werkzeug aufrufen
Secure MCP Tunnel	Privaten Zugang zum lokalen Adapter herstellen
Demo-Adapter	Auftrag begrenzen und Ergebnis zuordnen
Nativer OpenClaw-MCP	Mit <code>sessions_send</code> den Agenten beauftragen

OpenClaw erstellt das Briefing in seinem lokalen Workspace.



Ein Auftrag aus ChatGPT:

Technische Diagnose oder Übergabe an die Incident-Leitung?

Beauftrage OpenClaw mit einem neuen Briefing zu NORDLICHT-42. Zeige mir die Laufkennung und den gespeicherten Bericht.

ChatGPT erteilt den Auftrag. OpenClaw erstellt den Bericht.

Wer hat welche *Arbeit* erledigt?

Beteiligter	Verantwortung
ChatGPT	Den Wunsch in einen Werkzeugaufruf übersetzen
Demo-Adapter	Den erlaubten Auftrag an OpenClaw übergeben
OpenClaw-Agent	Beispieldaten lesen und den neuen Bericht schreiben
Menschen	Laufkennung, Datei und fachlichen Inhalt prüfen

Eine Antwort von ChatGPT allein belegt noch keinen OpenClaw-Lauf.

Der Eingang bekommt einen engen *Auftrag*

- **Authentifizierung** klärt, wer den Eingang benutzen darf.
- **Feste Aufgaben** begrenzen, was ein Trigger auslösen kann.
- **Erlaubte Daten und Pfade** bestimmen den Arbeitsbereich.
- **Zeit- und Ausführungslimits** begrenzen wartende oder laufende Arbeit.

Ereignisdaten liefern Fakten. Die Anwendung legt die Handlung fest.

Drei Wege zum Incident-Briefing

	Cron	Sentry-Webhook	MCP aus ChatGPT
Start	Zeitplan	Neues Demo-Issue	Werkzeugaufruf
Gegenstelle	Scheduler	Sentry-Regel	ChatGPT Web
Guter Einsatz	Wiederkehrende Arbeit	Konkrete Änderung	Gezielte Delegation
Zu klären	Zeitzone, Laufhistorie	Signatur, Wiederholungen	Auftragsumfang, Ergebnis

Die Wahl folgt dem fachlichen Anlass.

Welche Aufgabe könnte heute schon starten?

„Sobald _____, soll der Agent _____.
Fertig ist die Aufgabe, wenn _____.“

Welche wiederkehrende Aufgabe in eurem Alltag passt dazu?

Auslöser, erlaubte Handlung, überprüfbares Ergebnis.

Arbeit beginnt mit einem vereinbarten Anlass

Ein Zeitplan wartet auf den nächsten Termin.

Ein Webhook wartet auf eine Änderung.

Ein MCP-Werkzeug wartet auf einen Auftrag.

Der Agent braucht einen klaren Start
und ein überprüfbares Ende.

NÄCHSTES AGENTIC OS Meetup · 03.11.2026



Entscheidungen in Millisekunden

TypeSafes „System One“-Modell für Auswahl, Bewertung und Routing.

Tempo, Konfidenz und Grenzen im Praxistest.

Quelle: TypeSafe · 70–500 ms je Aufruf laut Hersteller · Stand: 29.09.2026

MARCEL WEGE · BYTE5

Vielen Dank

AGENTIC OS

**Welche Arbeit wartet bei euch
noch auf einen manuellen Start?**

Crons für geplante Aufgaben.

Webhooks für Ereignisse.

MCP für einen gezielten Werkzeugauftrag.

byte5 entwickelt geschäftskritische
Web- und KI-Lösungen.



Marcel Wege

byte5 GmbH

mwege@byte5.de

[LinkedIn](#)

Offizielle Dokumentation

- [OpenClaw: Cron jobs](#)
Zeitpläne, Sitzungen und Ausführung
- [Sentry: Issue Alert Webhooks](#)
Projektregel, signierter Alarm und Event-Kennung
- [OpenClaw: Webhooks](#) und [nativer MCP-Zugang](#)
Agentenaufträge und native Gateway-Werkzeuge
- [OpenAI: MCP mit ChatGPT verbinden](#)
Verbindung, Werkzeugangebot und Prüfung in ChatGPT

Drei unterschiedliche Richtungen

Mechanismus	Wirkung
Eingehender Webhook	Ein externes System startet Arbeit in OpenClaw
Interner Hook	OpenClaw reagiert auf ein internes Lebenszyklus-Ereignis
Ausgehender Webhook	Ein fertiger Lauf liefert ein Ergebnis an ein Ziel

Ein **eingehender Webhook** kann einen Agentenauftrag starten.

Ein Auftrag braucht eine nachvollziehbare Spur

Information	Zweck
Ereignis- oder Auftragskennung	Eingang und Ergebnis zuordnen
Job- oder Laufkennung	Ausführung im Gateway finden
Start, Ende und Fehler	Laufzeit und Abbrüche erkennen
Ergebnisort und Inhalt	Den tatsächlichen Abschluss prüfen

Bei Wiederholungen gehört der vorherige Zustand zur Entscheidung.

Ein einmaliger Job in OpenClaw 2026.9.6

```
openclaw cron add \  
  --agent agentic-os-demo \  
  --at "<ISO-Zeitpunkt>" \  
  --session isolated --no-deliver \  
  --delete-after-run \  
  --message "Erstelle das Briefing zu NORDLICHT-42."
```

`--at` legt den Zeitpunkt für den einmaligen Auftrag fest.

Der Scheduler startet den Agenten, sobald der Auftrag fällig ist.

OpenClaw 2026.9.6. cron ist ein Alias von automations. Gekürzter Auftrag. Der Scheduler startet den Job.

Ein begrenzter Auftrag an OpenClaw

```
{  
  "name": "create_incident_brief",  
  "arguments": {  
    "focus": "diagnostic",  
    "request_id": "abend-20260929-01"  
  }  
}
```

Das Ergebnis folgt über `read_incident_brief(run_id)` .

Die `run_id` verbindet den Auftrag mit seinem Ergebnis.