Harness-audit — Sjakie/Conductor-stack vs. "Harness Engineering" checklist

Door Sjakie, 2026-06-08. Getoetst aan de video "Harness Engineering Is AI's New Gold Rush" + OpenAI/Anthropic harness-bronnen.

> Kernstelling van het vakgebied: **het model is commodity, de harness is de moat.** ~88% van agent-projecten haalt productie niet; 65% van de fouten zit in harness-defecten (context-drift, schema-mismatch, state-verlies) — niet in het model. Hieronder onze 8 harness-onderdelen, met cijfer en concrete zwakke plek.

Scorebord (1 = zwak, 5 = sterk)

#	HARNESS-ONDERDEEL	CIJFER	KORTSTE OORDEEL
1	Context-levering & -beheer	3	Veel lagen, maar overlappend en deels tegenstrijdig
2	Tool-interfaces & schema	3	Werkt, maar MCP-token-bloat + losse schema's
3	Planning / anti-one-shotting	2	Geen vaste plan-stap; we duiken vaak meteen in uitvoer
4	Verificatie-loops	**2**	**Grootste gat — "klaar" ≠ door Jamal goedgekeurd**
5	Geheugen & state	3	Sterk idee (Mnemosyne), maar 5 geheugenbronnen die kunnen botsen
6	Sandbox & permissies	3	Breed (bypass) — snel, maar weinig guardrails
7	Observability / kosten	**2**	**Geen centrale log van acties + uitgaven (fal.ai-runaway)**

#	HARNESS-ONDERDEEL	CIJFER	KORTSTE OORDEEL
8	Orchestratie / multi-agent	3	Rollen helder, maar kanaal-bugs + dubbele regels

Gemiddeld: 2,6 / 5. Fundament staat; de winst zit in #4 en #7.

---

1. Context-levering & -beheer — 3/5

Wat we hebben: CRITICAL FACTS, CLAUDE.md, MEMORY.md, last-session, heartbeat, Known Gotchas, Obsidian-vault.
Zwakke plek (context-drift): dezelfde regel staat op 3-4 plekken en spreekt zichzelf soms tegen. Voorbeeld: Conductor-communicatie staat als "via Supabase queue" (multi-agent-protocol.md) én als "NIET via queue, maar Telegram-doorstuur" (memory). Tegenstrijdige context = precies de "schema misalignment" uit de video.
Fix: één bron van waarheid per regel. CLAUDE.md = canon, de rest verwijst ernaar i.p.v. te dupliceren.

---

2. Tool-interfaces & schema — 3/5

Wat we hebben: veel MCP's, Browser Bridge, eigen scripts (hue.py, relay, shared_memory).
Zwakke plek: MCP-token-bloat (gesignaleerd in de Composio-scan 2026-06-07) — veel tools geladen = vol context = "context anxiety". Losse scripts zonder uniforme fout-conventie.
Fix: deferred/lazy tool-loading aanhouden (gebeurt al deels), zelden gebruikte MCP's uit, en een vaste fout-conventie voor eigen scripts (defensief, nooit breken).

---

3. Planning / anti-one-shotting — 2/5

Zwakke plek (one-shotting overreach): we pakken grote taken vaak in één keer i.p.v. eerst plan → akkoord → uitvoer. De extensies-saga is hét voorbeeld: telkens opnieuw genereren zonder eerst de acceptatie-norm vast te leggen.

Fix: bij elke niet-triviale taak eerst 3-regelig plan + 1 testresultaat → akkoord → pas dan batch. (Dit staat al als beeld-regel; uitbreiden naar álle taken.)

---

4. Verificatie-loops — 2/5 ⚠ GROOTSTE GAT

Zwakke plek (victory declaration bias): onze "check vóór claimen"-regel is prompt-only, geen harde poort. Erger: de verificatie meet de verkeerde dingen. Bij extensies gaf de QC 0.947 similariteit + gezicht 9/10 → "klaar", maar jij keurde af. De check matchte niet jouw échte criterium (natuurlijk opgedroogde krul, niet getekend).
Fix: verificatie koppelen aan jóuw acceptatiecriterium, niet aan een proxy-getal. Voor beeld: altijd 1 visuele check + jouw OK = de enige "done". Voor code: een echte test draaien, geen "het zou moeten werken".

---

5. Geheugen & state — 3/5

Wat we hebben: Mnemosyne shared memory (getest 2026-06-08, werkt, similariteit 0.77) + auto-memory + Obsidian-vault.
Zwakke plek (state degradation): 5 geheugenbronnen die niet automatisch synchroon zijn; een feit kan in MEMORY.md kloppen en in de vault verouderd zijn. Geheugens worden zelden opgeschoond.
Fix: periodieke "geheugen-reconciliatie" (1×/week diff), en bij gebruik altijd verifiëren tegen de live-staat vóór ik erop bouw.

---

6. Sandbox & permissies — 3/5

Wat we hebben: bypass-permissies ("gewoon doen"), Keychain voor secrets (sterk).
Zwakke plek: breed mandaat = snelheid, maar weinig harde rem. De échte vangrails zijn de gedrags-regels (geen group, nooit zelf exitten, bevestiging bij grote acties) — die zijn prompt-afhankelijk.
Fix: secrets-in-Keychain houden (top), en de paar onomkeerbare acties (deploy, betaling, delete) als expliciete bevestig-poort houden. Dit zit al goed; bewust zo laten.

7. Observability / kosten — 2/5 ⚠ TWEEDE GAT

Zwakke plek: geen centrale log van wat de agents doen + wat het kost. De fal.ai-runaway (kosten-blok die we moesten pauzeren) was hier het symptoom: pas zichtbaar toen het al liep.
Fix: één dashboard/logregel per gegenereerde foto + uitgave (de fal.ai-overzichtspagina die net af is, is hiervan de eerste steen — uitbreiden naar een kostenplafond + alert).

---

8. Orchestratie / multi-agent — 3/5

Wat we hebben: heldere rolverdeling (Sjakie=uitvoerder, Conductor=orchestrator, JPB=backup).
Zwakke plek: kanaal-fragiliteit. Net nog: de relay dubbel-wrapte mijn Conductor-bericht (`[→CONDUCTOR: [→CONDUCTOR:`). Plus twee tegenstrijdige regels over wélk kanaal (queue vs Telegram-forward).
Fix: één afgesproken kanaal + de relay-dubbelwrap fixen.

Top-3 acties (hoogste hefboom eerst)

1. **Verificatie-poort (#4):** "done" = jouw OK of een echte test — nooit een proxy-getal. Voorkomt de extensies-loop.
2. **Kosten/observability (#7):** fal.ai-overzicht uitbouwen met kostenplafond + alert. Voorkomt runaway.
3. **Eén bron van waarheid (#1/#8):** dubbele/tegenstrijdige regels opschonen, relay-dubbelwrap fixen.

Deze 3 brengen ons van ~2,6 naar ~4 zonder één modelwissel — exact de boodschap van de video.