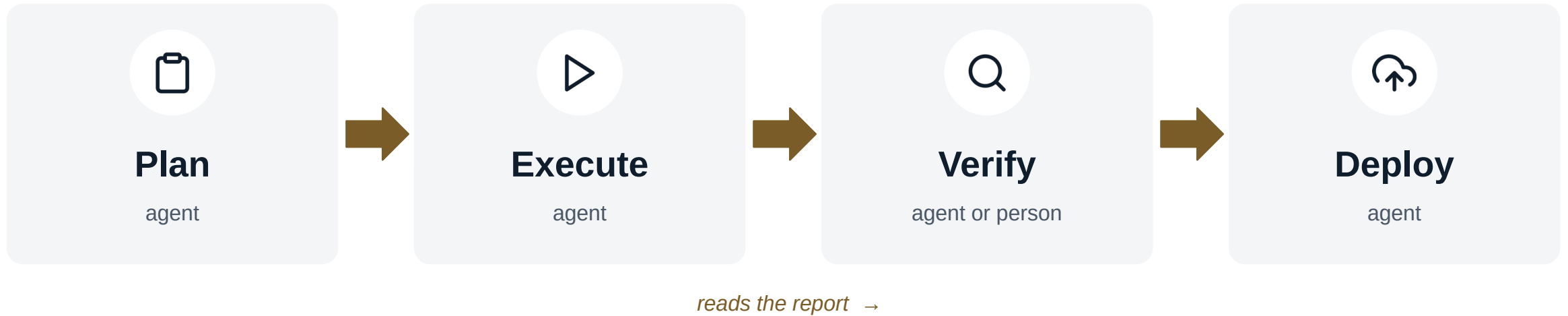


Definition of Done for AI Agents

What a completion report has to carry across a handoff

The report becomes the interface



Each step may rely on the previous step's report.

If the report is vague, the next step inherits a guess and treats it as a fact. So the question is not only "Is it done?" but "What must the report carry so the next step can rely on it?"

Three ways "done" goes wrong at a handoff



Compression

The summary drops the scope.

"Implemented and tested" — on one page, in one browser, with one configuration.



Substitution

The check was on something easier than the claim.

The button was clicked; the behaviour behind it was never observed. A stub passed; the real call never ran.



Absence as confirmation

Nothing found is read as nothing there.

An empty configuration list is read as "none needed" instead of "not checked yet".

The consent banner that passes review

HYPOTHETICAL

What the report said — all true

- ✓ Banner implemented
- ✓ Accept and reject both work
- ✓ The choice is saved; the banner does not reappear

What the task required — not shown

- ✗ Optional scripts held back until the choice
- ✗ Every output path covered, not only one hook
- ✗ Behaviour checked where users actually are

The banner shows that the interface records a choice. It does not show that optional scripts were held back until the choice was made — which was the point of the task.

One hook does not cover every output path

On WordPress, a common way to hold a script back is to rewrite its tag with the `script_loader_tag` filter. Which outputs actually pass through it?

A script with a source file

translations, inline before / after and the tag itself

passes through the filter

Data attached with `wp_localize_script()`

printed separately by `print_extra_script()`

does not pass through

A handle registered without a source file

its inline code is printed directly

does not pass through

Script written into the page by a theme, plugin or host

never enters the script queue

does not pass through

Source: WordPress core, `WP_Scripts::do_item()` and `print_extra_script()` in `class-wp-scripts.php`, tags 6.9.9 and 7.1.3 (checked 21 October 2026). Code paths change between releases — name the version you checked.

The completion record: eight fields

1

Claim

One sentence: what is now true, at the right strength.

2

Scope

Environment, version, paths, data, languages, browsers.

3

Evidence

What was run, by whom, and where the artefact is.

4

Method limits

Stubs, mocks, samples, simulated conditions.

5

Not checked

An explicit list. Unknowns are entries, not omissions.

6

Uncovered paths

What the change does not control, even if it works.

7

Decision needed

What a person must decide before upgrading — and who.

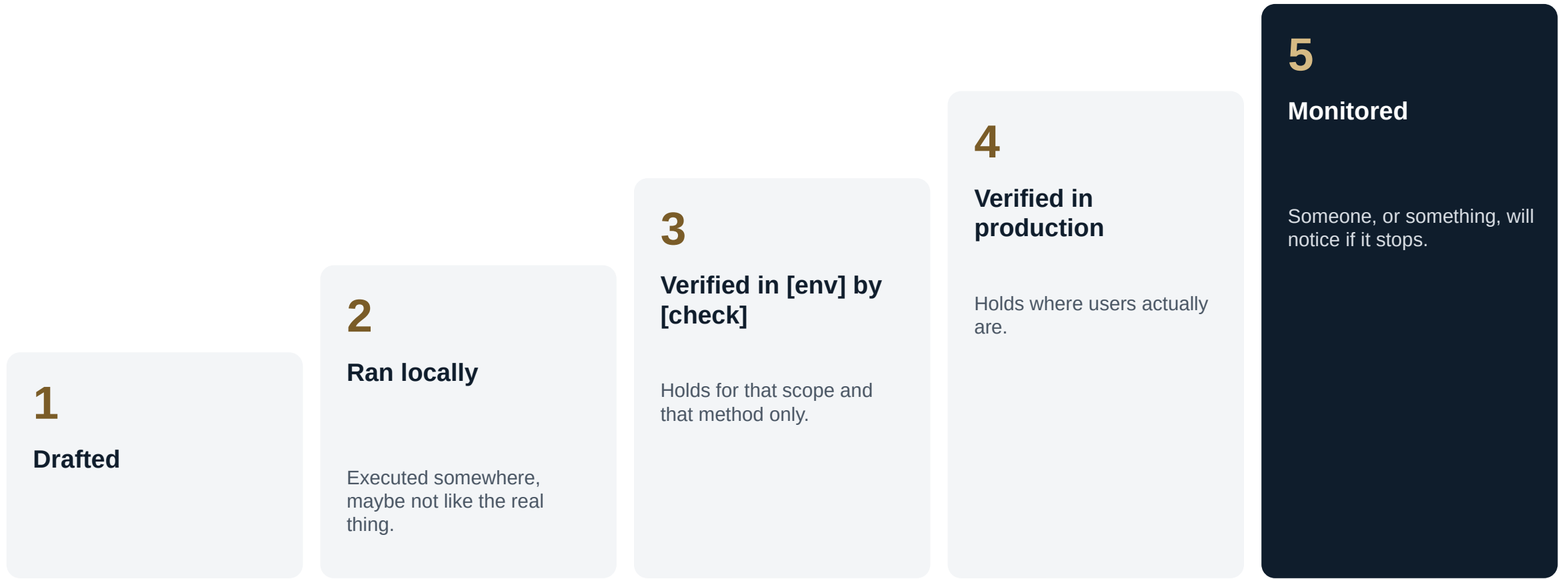
8

Next check

What would upgrade or invalidate it; when; rollback.

A ladder for the word "done"

A report that says "done" without naming a rung is not accepted at the handoff.



**A summary may shorten a claim.
It may never strengthen it.**

"Verified locally in Chromium" may become "verified locally". It may not become "verified".

What the receiving side does



Reject reports without Scope or Not checked

An empty "Not checked" reads "none, because ...", never blank.



Re-check one claim independently

A different agent, model or person. One re-derived claim beats ten re-read ones.



Turn each "not checked" into a task or an accepted risk

Accepting a risk is a decision. Record who made it.



Do not upgrade a claim while passing it on

If the outgoing report is stronger than the evidence, the handoff failed.

Match the depth to the consequence

IMPACT IF
WRONG

High impact · easy to undo

Re-check one claim
independently before
release.

High impact · hard to undo

Stop until a person decides.
"Not checked" must be
short.

Low impact · easy to undo

Short record. Move on.

Low impact · hard to undo

Name the rollback before
acting.

HOW HARD TO UNDO →

*"A human having read the
output is not enough to
describe the work as
verified."*

akiohirayama.com/en/ai-source-check/

Twelve false-completion patterns

01

Interface $\neq$ behaviour

02

Stub $\neq$ integration

03

Local $\neq$ production

04

One browser $\neq$ all

05

Sample $\neq$ population

06

**Empty list $\neq$ confirmed
none**

07

**No result $\neq$ non-
existence**

08

**Merged $\neq$ deployed $\neq$
live**

09

Scheduled $\neq$ ran

10

Translated $\neq$ reviewed

11

Written $\neq$ loaded

12

**Hook added $\neq$ every
path**

Each is reasonable inside one step and misleading after a handoff. Pair each pattern with a check that can detect it.

Make the limits travel with the claim

A completion record does not make work correct. It is not a standard or a certification. It lets the next agent, reviewer — or you, next week — see what was actually established.

This week

1 Add the eight fields to your agents' report format 2 Name the rung 3 Run the handoff checklist