MUTAGENT

FIELD GUIDE

How to diagnose agent traces.

Thousands of traces in, a short list of named root causes out. **The method, the cheap checks, and a checklist you can run on your own agents today**, by hand or with a tool.

START HERE

3 traces

A model reads one transcript, spots the bad call, tells a plausible story. That works.

3,000 traces

Now you need the pattern across all of them: a class of failures with one shared cause. It only shows up across the whole set.

Diagnosis is a dataset problem. Debugging is a single-trace problem.

THE METHOD

Eight steps, whole-set in, root causes out

STEP 01
Scan cheaply
A pattern pass over every trace. No model, no tokens.

STEP 02
Census, rank by impact
Rule out the loud-but-benign. Rank by harm, not count.

STEP 03
Group by failure kind
Errors, bad feedback, slow runs, the rest.

STEP 04
Read in parallel, capped
One reader per group. Cheap checks first, model last.

STEP 05
Read deeper as needed
Sample more only where the signal is. Say how deep.

STEP 06
Trace back to the cause
Walk each failure to its origin. Cite a line.

STEP 07
Name it
What broke, why, and where.

STEP 08
Fix the source, add a test
Change the prompt, tool, or config. Re-run to confirm.

STEP 1 IN PRACTICE: WHAT TO SCAN FOR

Cheap checks, no model, run across the whole set

SCAN FOR	HOW TO DETECT IT (CHEAP)	MEANS
Errors	error marker or a tool result flagged isError; flag if > 20%	broke outright
Slow runs	latency above the set's own fence (Q3 + 1.5×IQR), or > 10s; flag if > 10%	latency spike
Low score	thumbs-down, or normalized score ≤ 0.4; flag if > 5%	rated bad
Cost spikes	tokens near the model limit, or cost outliers vs the median run	cost overshoot
Stuck retrying	retry attempts hit the max on a trace	provider limit
Tool loops	same tool 3+ times with near-identical args, or call depth > 10	a loop
Tool errors	a tool result marked as an error	tool misuse
Dropped calls	a tool call with no matching result	lost handoff
Complaints	the user's own text contains a complaint phrase ("that's wrong", "try again")	user-complaint

It finds where to look, never the answer. The scan is blind to clean-looking wrong output, so a model read still has to follow.

STEP 7 IN PRACTICE: NAME IT ON THREE AXES

Three axes turn a symptom into an address

WHAT
the symptom

- wrong output
- missing output
- a loop
- latency spike
- cost overshoot
- format violation
- hallucination
- user complaint
- low score
- missing context

WHY
the mechanism

- underspecified prompt
- overspecified prompt
- tool misused
- tool missing
- context overflow
- provider limit
- stale data
- lost handoff
- dependency failure

WHERE
the location

- system prompt
- tool definition
- agent config
- routing config
- upstream data
- provider side
- the harness
- user input

"Wrong output" is not fixable. **"Wrong output, tool misused, in the tool definition" is.** A name you can test for, and catch next time.

ONE FAILURE, WALKED END TO END

SYMPTOM
The agent loops: the same search tool fires four times in a row, no new result.

SCAN FLAGS IT
The cheap pass catches it: same tool, near-identical args, repeated. No model needed.

WALK IT BACK
Why loop? the tool errors. Why error? the query arg is malformed. Why malformed? the prompt never states the format.

NAME + FIX
`WHAT loop`
`WHY prompt-underspec`
`WHERE system-prompt`
Add the format rule to the prompt. Add a regression test.

RUN THIS ON YOUR OWN TRACES

- [ ] **Pull the whole set** for one agent and window, not a handful.
- [ ] **Scan cheaply first** (the table above). No model yet.
- [ ] **Rule out the loud-but-benign.** Rank what survives by impact, not by count.
- [ ] **Group the survivors** by kind. Diagnose one kind at a time.
- [ ] **Read deep only where the signal is.** Start small, go deeper only if the evidence is thin.
- [ ] **Walk each failure to its origin,** the thing nothing upstream explains. Cite a trace line or a code line.
- [ ] **Name it: what, why, where.** If you cannot name it, you cannot test for it.
- [ ] **Refuse to call it a diagnosis** if you read zero traces deeply or cannot cite the cause. State how much you read.
- [ ] **Fix the source, add a regression test,** and re-run to confirm it held.

WHAT MAKES IT A DIAGNOSIS, NOT A GUESS

Cite a line
Every claim points to a trace message or a line of code. No evidence, no claim.

Refuse to guess
Zero deep reads and no prior knowledge is not a diagnosis. Say so, do not ship it.

Show coverage
Say how many traces you read and how sure that sample makes you.

Reproduce it
Pin the model and its settings. Run it twice, get the same answer.