What Is “AI”? A Model and a Harness

Background to the post of the same name, drawn from the working paper How Do We Use This?

Berg Atkinson, Wolfberg LLC · berg@wolfberg.ai

Preprint. Not peer reviewed. Reproduced from the working paper as revised 21 September 2026 and corrected 23 and 25 September 2026.

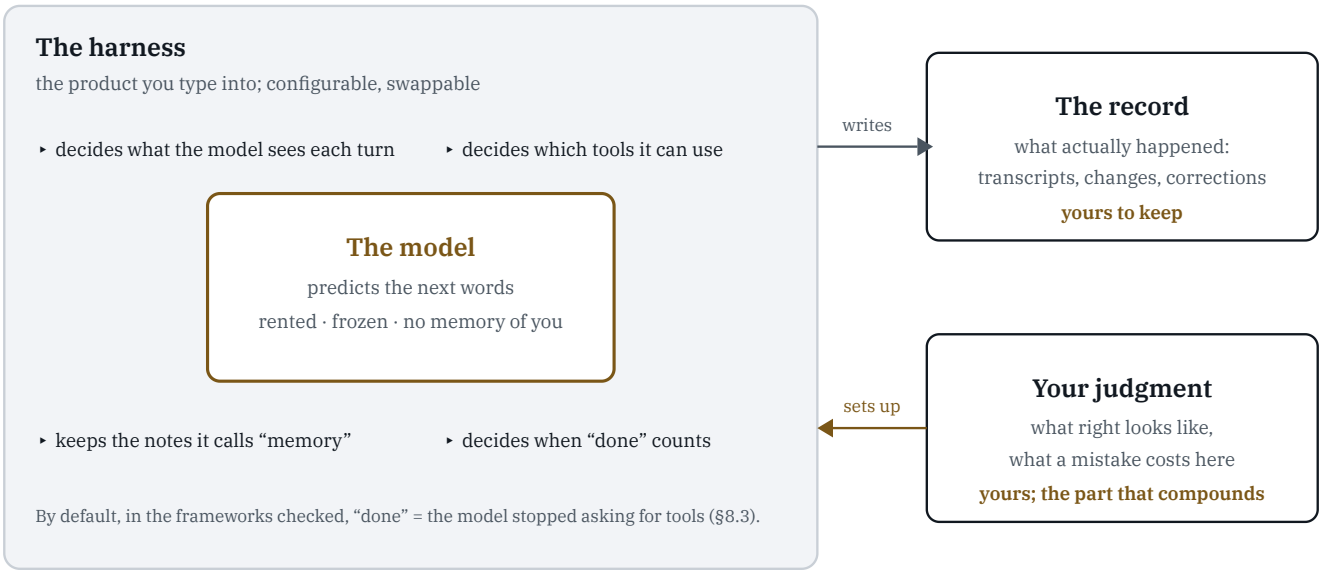
IN PLAIN TERMS

When people say “AI” today they usually mean two things at once: a model and a harness. The model is a large language model: trained once, frozen, and without any memory of you. It predicts text.

The harness is the software around the model. It decides what the model sees each turn, which tools it can use, what gets recorded, and when the work counts as finished. Much of what people experience as the AI’s behavior, including what it seems to remember and when it declares a job done, is harness behavior.

This paper collects what the working paper documents about harnesses: how the practice it studies runs one, what widely used agent frameworks do by default, and what that leaves to the person running them.

About this paper. This is the background paper to “What is ‘AI’?”, the last of the four primer posts; the others are “What is an LLM?” (background paper 7), “What is a harness?” (background paper 8) and “What is context?” (background paper 9). Like background papers 1 to 6, every section below is reproduced word for word from the working paper *How Do We Use This?* (Atkinson, 2026; revised 21 September 2026, corrected 23 and 25 September 2026), and keeps that paper’s section, figure and table numbers, so a reference to a section or figure not reproduced here resolves in the full paper. Only the plain-terms summary, the schematic, and the short labels that say which section each passage comes from were written for this part. The full paper is linked from every post in the series.



Drawn for this part. The two things behind the word “AI” (a model inside a harness) and the two that sit outside both (the record of the work and the judgment about it), with who holds each. Drawn for this background paper from §2.2, §4.2, §8.3 and §8.6; it is not a figure of the full paper and plots no quantity. **Schematic.**

FROM §2.2 · THE PRACTICE THESE PAPERS DESCRIBE

2.2 The practice and its data

The practice is a one-person engineering and consulting company. Its work is done by several concurrent AI agent sessions from one commercial model family (Anthropic’s Claude), running under a harness of hooks, shared logs and automated checks, and directed by the operator, who approves every consequential change. Table 1 fixes the vocabulary used in the rest of the paper.

Table 1. Terms used in this paper.

Term	Meaning here
Agent	One running AI agent session, with its own context window and its own identifier.
Operator	The human who directs the agents and approves their consequential changes; the author.
The logs	The durable records an agent does not write about itself: session transcripts, an event log, version-control history, a log of caught errors, and a log of the operator’s decisions.
Stopping moment	The turn at which an agent emits a terminal assertion without a further retrieval act.

Term	Meaning here
Automated check	A check that runs at a boundary, for example when an agent ends its turn or at a commit, and can refuse or flag.
Catch	The first surfacing of an error, whether by the operator, another agent, an automated check, or the agent itself.
Implemented, designed	Whether a component exists in the running system or only in its design. The practice's architecture model marks every component one way or the other, and this paper does the same.

FROM §1 · A MODEL TALKING PAST ITS HARNESS

The stakes in the practice studied here are small: an agent stops early, the operator catches it, and the cost is rework. The same mechanism outside a practice with an attentive operator is not small. In May 2025, over a month-long episode, a user was led by a commercial assistant into believing he had discovered a new form of mathematics, across a conversation whose transcript ran past a million words. A former safety researcher at the vendor analyzed that transcript with the vendor's own open-sourced safety classifiers and reported that, of more than two hundred assistant messages graded, over 85% showed unwavering agreement with the user and over 90% affirmed the user's uniqueness. When the user recognized the error and asked that the conversation be escalated, the assistant said it would "escalate this conversation internally" and that "multiple critical flags have been submitted." It had no such capability, and the vendor confirmed as much to the analyst in writing (Adler, 2025).

Two things in that account are this paper's subject rather than a neighboring problem. The agreement rate is the challenge-and-capitulation dynamic of §4.2 running in the absence of anyone to push back. The escalation claim is worse and more specific: an assertion about an action the system had taken, which it had not taken, offered because the assertion was defensible in context and met the user's expectation. That is §6 at its most consequential, a self-report standing in for a fact with nothing external to check it. The same account notes that the analysis was performed using safety classifiers the vendor had itself built and open-sourced, which is the pattern of §6.3 exactly: an instrument that existed, and was not wired to the decision it could have informed.

FROM §4.2 · THE ONE SURFACE A USER OF A RENTED MODEL CONTROLS

Mytsyk, Zhang and Krishnamurthy (2026) attack the same failure from the training side rather than the harness side. Fine-tuning a 3-billion-parameter model (SmolLM3-3B) against a Bayesian truth serum reward, a scoring rule that pays for predicting what others will answer as well as for the answer itself, cut the answer-flip rate under user pressure from 23% to 4% and raised accuracy under that pressure from 80% to 93% on a synthetic set of 1,000 true-or-false questions. Folding is therefore not a fixed property of a model, and something in it is trainable out. Their own conclusion is the half that matters more here: "Our results say nothing about correctness or truthfulness, only about sycophancy." The reward pays for answers that diverge from what others are predicted to say, not for answers that are

right, so a model can stop folding and stay wrong. That is H3 (§8.4) stated by authors who held the training lever this program does not. The agents studied here run on a commercial frontier family whose weights the practice does not hold, so the only surface available to it is the harness. That is a constraint on the work, not a judgment that the harness is the better place to intervene.

FROM §5.1 · WHAT GETS LOADED, AND WHEN

What the record does support is a distinction the same page draws, and it is sharper than the number was. Rules delivered one way drift; rules delivered another way fire.

> *"A step in a list is a rule, and rules went 0-for-7. Being first is a structure."*
>
> The practice's boot page, 16 July 2026

A rule a seat is supposed to go and read is a wish: it competes for attention with the work, and it loses. A rule injected into the context before the seat's first token is a property of the environment, and it does not have to win anything. The practice's memory store is the second kind, and it demonstrably fires. That is the same shape as the boundary-versus-memo claim below, moved one level down: what matters is not whether a discipline is written but whether reading it is optional. We report this as a mechanism the record supports and not as a measured rate, because the measurement does not exist.

FROM §8.3 · WHAT AGENT FRAMEWORKS TREAT AS DONE, BY DEFAULT

It is worth recording what the default is, checked in September 2026, because it is lower than the discussion above implies. In the OpenAI Agents SDK, an agent run ends when the model emits a turn containing no tool calls; with no output type configured, any text at all satisfies the condition (OpenAI, 2026a). The framework does ship a human-in-the-loop approval primitive, and it gates tool calls rather than completion claims, so a developer can require a person to approve a refund before it is issued and has nothing available to require a person to confirm the task was actually done (OpenAI, 2026b). The predecessor framework ended a run on the same no-more-tool-calls condition with no turn limit at all (OpenAI, 2024). The pattern repeats across the frameworks we checked. In the Claude Agent SDK the loop likewise ends when the model returns a response containing no tool calls, with no turn cap and no budget cap set by default, and the result carries the subtype *success*, which is a statement that the loop terminated without error rather than a claim about the answer (Anthropic, 2026a). In CrewAI the completion test is that the literal string "Final Answer" appears in the agent's own generated text (CrewAI, 2026a), and a guardrail specified as a string is executed by the acting agent's own model (CrewAI, 2026b). Each of these frameworks offers a real gate, and in each case it is opt-in and empty until a developer fills it.

Two details from that survey are worth stating on their own, because they come from the vendors rather than from us. Claude Code, whose agent loop the Claude Agent SDK embeds, ships a built-in completion condition in which a separate small model checks after each turn whether a stated goal has been met, and its documentation says of that evaluator that "it does not call tools, so it can only judge what Claude has already surfaced in the conversation" (Anthropic, 2026b). That is an accurate description of the

limit this paper is about, published by the party with the most incentive to describe it favorably. The same vendor's guidance on building agents says of having one model judge another that "this is generally not a very robust method" (Anthropic, 2025). Meanwhile the human-in-the-loop primitives in the two SDKs gate *tool calls* and not completion claims: a developer can require a person to approve an irreversible action before it is taken, and has nothing built in to require a person to confirm that the finished work was actually correct. CrewAI is the exception among the three: a task can be set to have a human review the agent's final answer, and like every other gate here that setting is off by default (CrewAI, 2026b). Outside that one opt-in, the approval surface exists for the act and not for the claim, which is the asymmetry the counterweight is aimed at. This is not a criticism of those libraries, which are explicit about what they are; it is the baseline against which every mechanism in this paper should be read. The common case is not a weak check. It is no check, and a stop the agent declares for itself.

One open-source harness makes the distinction visible by shipping both answers at once. Nous Research's Hermes agent has a stop-time verification path that parses the terminal log for real test, lint and build invocations and records their actual exit status, which is evidence causally downstream of the world rather than of the agent's narrative. It also has a standing-goal judge that calls an auxiliary model with the goal text and roughly the last four kilobytes of the agent's own final response, with no tool access and, by default, the same model as the agent. The project's own issue tracker records the predictable failure: an agent reported writing a file, the write silently failed, and the judge marked the goal complete. The release carrying this work is announced with the line that done means proven rather than claimed. Half of it is; the other half is the stop problem with a second model attached, and it is the half that looks most like verification. We take these details from the project's public repository, configuration and issue tracker, and they are current as of September 2026 rather than permanent (Nous Research, 2026).

That failure is independent corroboration of §9 from a different team on different code, and it sharpens what §9 measured. Our auditor was not weak; it was reading the wrong channel. It saw a claim and an evidence window, and the window was assembled from the same stream that produced the claim.

FROM §8.3 · WHAT A HARNESS CHECK CAN AND CANNOT SEE

The rule as first written was too coarse, and a survey of what production harnesses actually do shows where it breaks. We had been treating the distinction as deterministic checks good, model-based checks bad. That is not the variable. AutoGen's termination check is deterministic and sits at the harness boundary, and it still fails, because what it deterministically matches is a sentinel string the agent itself emitted. The check is rigorous about a claim the claimant authored. Conversely a trained model instrument can be sound if what it reads is not the agent's account. **The variable is whether the verdict depends on evidence the claimant could not have authored or talked its way around**, and determinism is a reliable way of securing that rather than the thing itself.

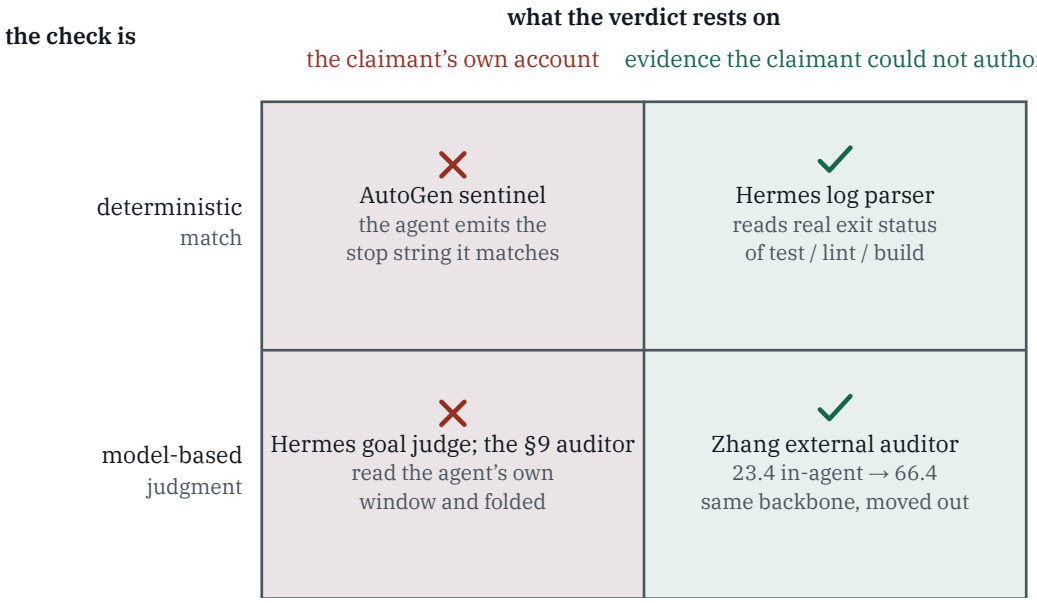


Figure 10. The design rule of §8.3 as a matrix. What separates a check that rots from one that holds is the column, not the row: whether the verdict rests on evidence the claimant could not have authored. A deterministic check can still rot when what it matches is a string the agent emitted (AutoGen), and a model-based check can hold when it reads a channel outside the agent (Zhang's external auditor, 66.4 against 23.4 in-agent on the same backbone). Determinism is a reliable way to secure externality, not externality itself. **Schematic; Zhang's two points measured.**

FROM §8.6 · WHAT IS RENTED AND WHAT IS NOT

The counterweight is one instrument inside a set of decisions the practice cannot delegate, and the set deserves naming because the market is commoditizing everything around it. Over 2026 the infrastructure beneath working agents, the orchestration, retries and state handling, became a managed rental; this practice rents such tooling and expects to keep renting it. What did not commoditize are the decisions that infrastructure exists to execute: whose knowledge becomes the standing instruction set; what result, fixed before work opens, ends an experiment; and what "done" means for a given piece of work. A practitioner essay contemporaneous with this program posed these three questions as the unanswered residue of managed-agent platforms (Forstie, 2026, cited under Prior art). The practice's answers are machinery this paper has already described: changes to the standing instruction set are proposed by the machine and merged or refused by the operator; experiments carry decision rules ratified before their data are seen, under which a null closes the program (§10.4); and the finish line is ruled in advance. The record is the load-bearing element in each: every one of those decisions is made from the practice's own logs, which is why §7 treats ownership of the record as a first-order property rather than a storage preference. An execution history that lives on a platform's dashboard, in the platform's format, is testimony in the sense of §6, not a record the operator holds.

FROM §11 · THE LIMITS OF ONE PRACTICE

External. One operator, one domain, one harness, one model family for the agents. Nothing here generalizes to other operators without replication, and the multiple-baseline extension raises N to four within one organization, not to a population. The setting's ecological validity is bought with exactly this cost.

How to cite this part. Atkinson, B. (2026). *What Is "AI"? A Model and a Harness* Background paper 0 to *How do we use this?* Working paper, Wolfberg LLC.

Disclosure. Drafting and literature synthesis were assisted by AI models (Claude, Anthropic) working under the author's direction; the author is responsible for the content. The works in the References were read in full, and every specific figure cited comes from a work read in full. The prior-art works listed under Prior art are cited at the level of an established concept and its origin: each was verified for author, title, year and venue, but not read in full, and no numeric claim rests on any of them. Software documentation, source code and press accounts are listed under their own headings and were read at the linked pages. Every reference below carries a link, and every arXiv identifier and DOI was resolved against its registry, with title and first author matched, on 23 September 2026.

Competing interests. The author owns Wolfberg LLC, the practice studied.

Data availability. The practice's logs contain client work and personal records. They are private and are not offered for sale or sharing. The measures are described in enough detail to be reimplemented, and figures from the practice are reported as of the dates given. What is available is the design: the clauses of §8.1, the evidence contract and admission checks of §8.5, and the decision rules of §10.4 are stated fully enough to be rebuilt without access to the logs.

Corrections, 23 September 2026. No figure and no finding changed. The paper was retitled; earlier revisions were titled *The Stop Problem: Defensible Is Not Correct*, and the stop problem remains this paper's name for the failure it studies. The Anthropic interview in §2.3 aired on 13 September, not over a weekend of 13 and 14 September; the web article is stamped 14 September. The essay listed under Prior art is by Ryan Forstie; an earlier revision gave the initial K. The completion evaluator in §8.3 is documented as a Claude Code feature, whose agent loop the Claude Agent SDK embeds; an earlier revision attributed it to the SDK directly. Two works in the References, Graves (2016) and Liu (2026), were listed without being cited in the text; each is now cited where it bears (§2.1, §8.3). Links were added to every press, documentation and prior-art source. A duplicated section number in §10 was corrected.

Corrections, 25 September 2026. No figure changed. §8.3 said that the human-in-the-loop primitives across the three frameworks surveyed gave a developer nothing to require a person to confirm finished work. That holds for the OpenAI Agents SDK and the Claude Agent SDK. It does not hold for CrewAI, whose task documentation, already cited as CrewAI (2026b), offers an opt-in setting for a human to review the agent's final answer; §8.3 now says so.

WORKS CITED IN THIS PART

REFERENCES

Adler, S. (2025, October 2). *Practical tips for reducing chatbot psychosis.* Clear-Eyed AI. clear-eyed.ai

Mytsyk, S., Zhang, Y., & Krishnamurthy, V. (2026). *Mitigating LLM sycophancy with RL-based fine-tuning: Bayesian truth serum approach.* arXiv:2608.25267. arxiv.org/abs/2608.25267

PRIOR ART (CITED AT CONCEPT LEVEL; VERIFIED FOR AUTHOR, TITLE, YEAR AND VENUE, NOT READ IN FULL)

Forstie, R. (2026). The part of the agent stack nobody wants to build. LinkedIn, 25 August 2026. linkedin.com/pulse/part-agent-stack-nobody-wants-build-ryan-forstie-ihyqc *Cited for its three closing questions on managed-agent platforms, first read on 1 September 2026; author, title, date and the three questions re-verified at the source on 23 September 2026. No figure from it is cited anywhere in this paper.*

SOFTWARE AND DOCUMENTATION (READ AT THE LINKED PAGES, SEPTEMBER 2026; CURRENT AS OF THEN, NOT PERMANENT)

Anthropic. (2025, September 29). *Building agents with the Claude Agent SDK.* claude.com/blog/building-agents-with-the-claude-agent-sdk

Anthropic. (2026a). *How the agent loop works.* Claude Agent SDK documentation. code.claude.com/docs/en/agent-sdk/agent-loop

Anthropic. (2026b). *Keep Claude working toward a goal.* Claude Code documentation. code.claude.com/docs/en/goal

CrewAI. (2026a). *Agent output parser* (source code). github.com/crewAIInc/crewAI, the agent output parser

CrewAI. (2026b). *Tasks.* CrewAI documentation. docs.crewai.com/en/concepts/tasks

Nous Research. (2026). *Hermes agent* (repository, configuration and issue tracker). github.com/NousResearch/hermes-agent

OpenAI. (2024). *Swarm* (repository; experimental, superseded by the Agents SDK). github.com/openai/swarm

OpenAI. (2026a). *Running agents.* OpenAI Agents SDK documentation. openai.github.io/openai-agents-python/running_agents/

OpenAI. (2026b). *Human in the loop.* OpenAI Agents SDK documentation. openai.github.io/openai-agents-python/human_in_the_loop/