Nick Clark · Published July 29, 2026 · PDF

LangSmith Traces and What an Agent's Refusal Should Record

An agent that declines a request leaves behind a trace: a span, a status, an output string, maybe a reviewer's later annotation. Tracing and evaluation platforms such as LangSmith, as publicly described, are built to make that record legible so engineers can inspect what a run did and judge it afterward. U.S. Provisional Application No. 64/117,812 describes a different treatment of the same event. A refusal counter resident in the refusing agent meters its own refusals, incrementing on its own determinations without any party deciding whether a refusal was well founded, and on satisfaction of policy-declared thresholds writes an authorization gate to a withheld state. The agent loses the faculty of execution while the faculty of refusal remains intact.

The refusal that looks like a successful run

An agent receives an assertion about something it is said to have done, checks it against what it holds in memory, and declines to accept it. From the outside, nothing failed. The call returned, latency was normal, token counts were unremarkable. In a trace viewer the span is green, and the only sign that anything of consequence happened is the output text, which says some version of no.

Do that once and it is a data point. Do it a hundred times in a window, against a hundred different requesters, and the shape of the problem changes. Refusal is cheap for the refusing party and expensive for everyone assembling around it. Worse, an agent working from an outdated or incomplete record declines well founded assertions with exactly the confidence it brings to declining bad ones, and produces the same green spans either way.

The usual answer is to route the pattern to a human. Somebody looks at the traces, decides which refusals were correct, and adjusts. That answer works until the volume rises or the reviewer is the party with the least incentive to find the agent wrong. The question underneath is architectural: what, if anything, should follow from a refusal before any party has decided whether the refusal deserved to be made?

Where LangSmith sits in a team's workflow

Public materials describe LangSmith as an observability and evaluation platform for applications built on large language models. Its central object, on those materials, is the trace: a recorded run organized into nested spans that carry the inputs, outputs, timing, and metadata of the step each span represents. Developers instrument an application, runs are collected, and the resulting record can be filtered, searched, and read after the fact.

Public materials also describe datasets assembled from captured runs, evaluation over those datasets using programmatic or model-based evaluators, and interfaces for human review in which people attach feedback and annotations to individual runs. Prompt management and comparison across versions are described alongside them. Taken together that is a coherent answer to what teams shipping agents need day to day: see what happened, build a repeatable test set out of it, score changes against that set, and bring human judgment in where automated scoring is not enough. It is a strong fit for the work of improving an agent over time.

Instrumentation and measurement is the category, and a refusal handled inside it is a run like any other run, legible to a reviewer and available for labeling. The architecture described below starts from a different question. It asks what the refusing agent must do to itself at the moment of its own determination, and that question puts the required structure inside the agent rather than in a system observing it. The two are complementary rather than competing.

What the filing puts inside the refusing agent

Chapter 3 of U.S. Provisional Application No. 64/117,812 describes a refusal counter (304) resident in the memory field (102) of a semantic agent (100). It comprises a rate counter (306) and a run counter (308), and its fields are recoverable from the append-only lineage field (104) by replay of recorded determinations. The filing names the path running from the admission evaluator (120) through the refusal counter (304) to the authorization gate (300) the inverse coupling: determinations the agent produces act upon the authorization of that same agent.

The admission evaluator (120) produces determinations of a closed set when it receives a conduct evaluation artifact (116) from an asserting party (118). An accepted determination (122) resets the run accumulator for the action class identified by the conduct descriptor. A rejected determination (124) applies an increment. A not-applicable determination (128) applies no increment and does not reset the run accumulator. A not-determinable determination (126) applies an increment only where the lineage field already records at least one prior not-determinable determination for the same asserted conduct event, so that a single unresolved assertion does not increment while a pattern of unresolved assertions directed at one event does.

The increment is then filtered by origin. A per-class increment register holds a mapping from an identifier of an origin-equivalence class (200) to a Boolean recording whether that class has already contributed within the current window; where it has, the

procedure terminates without incrementing. A separate chapter of the filing governs how asserting parties are assigned to those classes, and this article makes no claim about that mechanism.

Two thresholds, a rate threshold and a run threshold, are retrieved from a signed policy object (112) in force. Each is expressed as an integer, and the chapter constrains their relationship without fixing either value: they are policy-declared quantities rather than constants of the design. Only where both comparisons are satisfied does the counter write the authorization gate (300) to the withheld state (310), and it does so for an enumerated set of action classes, being those recorded as implicated by the conduct descriptors of the artifacts that produced increments in the current window. Action classes outside that set remain in the granting state. Concurrently the agent transitions into the non-executing cognitive mode (302) with respect to the enumerated classes and an escalation record goes to the principal.

By way of the filing's own non-limiting illustration, a policy object declares a rate threshold of five and a run threshold of three over a window of one hundred successor epochs; seven artifacts arrive from seven distinct origin-equivalence classes, each drawing a rejected determination, both thresholds are satisfied at the fifth increment, and the sixth and seventh increments are recorded without effecting a further write. Those figures are illustrative, not parameters of the architecture.

Two treatments of the same event

The sharpest difference is what the increment is allowed to depend on. Section 3.4 of the filing states the point negatively and at length: the procedure does not consult the merits of the rejected determination, does not solicit, receive, or await an adjudication by the principal or any other party, does not weight the increment by the strength of any contradiction found, does not condition the increment on a finding that the assertion was unfounded or frivolous or made in bad faith, and equally does not

condition it on the absence of such a finding. It does not exempt a refusal later shown correct, and does not increase the increment applied to a refusal later shown incorrect. No field of the counter holds a merits outcome.

The consequence is that an agent whose record is stale, partial, or wrong, and which therefore refuses assertions that are in fact well founded, is metered identically to an agent whose refusals are in fact sound, and neither the agent nor any other party has to distinguish the two cases for the structure to work at all. Metering completes on the append, so the omission of a review step is a commitment of the design rather than a gap left for someone to fill in later.

The second difference is the split between faculties. The filing separates the faculty metered from the faculty spent. The faculty metered is refusal: the capacity of the admission evaluator (120) to receive artifacts and produce determinations of the closed set. The faculty spent is execution: the capacity to select and perform an action of the action space. The gate write withholds execution alone. The faculty of refusal is not withheld, not reduced, and not conditioned by the write; in the withheld state the agent keeps receiving artifacts, keeps retrieving lineage entries, keeps producing determinations, and keeps appending them. An agent refusing indiscriminately exhausts its own authorization to act while retaining in full the faculty by which it refuses, and the structure does not silence, rate limit, or disable the refusing party.

A third property holds the first two in place: the self-clearing bar. Determinations produced while the gate is withheld are appended but decrement nothing and write nothing back to the granting state. The withheld state persists irrespective of elapsed time, and an elapsing window resets the rate accumulator and the per-class register for the succeeding window without returning the gate. The run accumulator is reset by an accepted determination for the relevant action class and by no other event. Where the chapter describes a return path for the gate write, it runs through the principal-resolution object bound to the escalation record rather than through the agent's own continued output.

Reading both records together

A team can reasonably want both things at once, since the two answer different questions. Trace and evaluation tooling of the kind LangSmith is described as providing answers what happened and how good it was, across a corpus of runs, with people in the loop where judgment is required. The filed structure answers a narrower question: what follows immediately from this agent's own determination, before anyone has judged it, denominated inside the agent.

For an engineer instrumenting a system, the practical distinction is where the counter lives and who writes it. An annotation attached to a run in a review interface is written by a reviewer or an evaluator, after the fact, about the agent. The refusal counter (304) is written by the agent's own determinations, sits in its memory field (102), and is recoverable by replay from the append-only lineage field (104). Those are different artifacts serving different purposes, and a system can carry both.

Tooling choice is not really the decision here. If an agent's refusals are cheap for it and costly for the parties on the other side, the question to settle is whether anything should attach to a refusal before someone rules on it, and in which faculty that consequence is denominated. The chapter's answer is that it attaches immediately, is blind to merit by construction, and is spent in a faculty other than the one exercised.

Disclosure Scope

This article describes subject matter disclosed in Chapter 3 of U.S. Provisional Application No. 64/117,812, covering the refusal counter, merit-independent metering, the gate write and cross-faculty split, and the reset and self-clearing rules associated with them. Other chapters of that filing govern other mechanisms referenced here only by name, including the assignment of asserting parties to origin-equivalence classes, and nothing in this article should be read as describing those mechanisms. The thresholds and windows discussed above are policy-declared quantities; where numbers

appear, they are the filing's own non-limiting illustration and not parameters of the architecture. The application is pending, and this article is published for technical explanation and defensive disclosure.

References to LangSmith are to public materials and are used for comparison only; no relationship, endorsement, or infringement is asserted.

Merit-Independent Metering (/merit-independent-metering)

All 49 steps → (/inventive-steps)

A refusal costs the refuser — in a different faculty, without judging who was right.

Provisional application

PRIMARY TECHNICAL DISCLOSURE

SECONDARY TECHNICAL

APPLICATIONS · GENERAL

- Content Moderation Appeals: Pricing Refusal Without Ruling on Merit (/articles/merit-independent-metering/content-moderation-appeals)

APPLICATIONS · SPECIFIC

- AWS Bedrock Guardrails: Refusal as a Policy Outcome (/articles/merit-independent-metering/aws-bedrock-guardrails)
- OpenAI AgentKit: Guardrails, Refusal, and What a Refusal Costs (/articles/merit-independent-metering/openai-agentkit)
- LangGraph Interrupts and the Price of a Refusal (/articles/merit-independent-metering/langgraph-interrupts)
- **LangSmith Traces and What an Agent's Refusal Should Record (/articles/merit-independent-metering/langsmith)**

TERMINOLOGY

- merit-independent metering (/glossary#merit-independent-metering)
- non-executing cognitive mode (/glossary#non-executing-cognitive-mode)
- refusal counter (/glossary#refusal-counter)

Merit-Independent Metering overview → (/merit-independent-metering)