Nick Clark · Published July 29, 2026 · PDF

Inngest: Durable Waits and Withholding as an Outcome

A durable workflow that waits for an approval and never receives one has to do something when the wait ends. In most systems that something is settled by application code, and the easiest branch to write is the one that treats silence as a decision. Chapter 5 of Adaptive Query's U.S. Provisional Application No. 64/117,812 describes a different arrangement, in which a governed withholding is recorded as an outcome entry of a recorded abstention class adverse to no party, held in a type disjoint from magnitude, with a conversion bar foreclosing its conversion into a scalar value, a default value, an operand of a threshold comparison, or a consequence adverse to any party. This article describes Inngest in its own terms, then sets out where the disclosed architecture diverges.

The branch nobody wants to write

A workflow pauses for a human approval. The approval does not arrive. The wait expires, control returns to the next line, and a developer has to decide what the absence meant. The branch that gets written is usually the convenient one: treat the empty result as a rejection, decrement something, close the request. It is convenient because the rest of the program expects a value it can compare, sum, or store in a non-nullable column.

The damage shows up downstream. Several stages later a report says the request was denied, and nothing distinguishes a denial a person issued from one a timer manufactured. Where the silent party was itself a machine deliberately holding back, the silence becomes evidence against it: counters advance, health scores fall, and correct behavior lands as a fault on somebody's ledger.

Inngest and the Adaptive Query filings meet this seam from adjacent layers. The public materials address waiting reliably; Chapter 5 of the filed provisional addresses what the resulting outcome may become in a governance record.

Inngest as it publicly presents itself

Inngest describes itself in public materials as a durable execution and workflow platform: application code authored and hosted by the developer, functions triggered by events or on a schedule, and orchestration provided around them.

The organizing idea is the step. Work inside a function is expressed as discrete steps whose completed results the platform is documented as recording, so progress survives crashes, restarts, and redeploys. On resume, completed steps are reported as not re-executed; their recorded results are supplied and execution continues, which is what allows a function to span far more wall-clock time than a single process would survive.

Around that core, the publicly documented feature set is described as including:

- **Durable sleeps and waits.** A step can sleep for a duration or until a timestamp, or pause until a matching event arrives. A wait can carry a timeout, and the documentation describes the step resolving without a matching event when that timeout elapses, control returning to the function.
- **Automatic retries.** Failed steps are retried under configured policy, the step boundary defining what gets retried rather than the whole function.

- **Flow control.** Concurrency limits, throttling, rate limiting, debouncing, and cancellation are described as declarative configuration rather than hand-written queue plumbing.
- **Observability and replay.** Runs, step timelines, inputs, outputs, and errors are described as inspectable, and runs as replayable.

On its own terms, that is coherent general-purpose infrastructure with a precise published contract: a truthful report that a wait concluded without a match, handed back to the developer's function. Assigning business meaning to that report sits in application code, the conventional boundary for an orchestrator. What follows concerns a governance layer above the one those materials describe.

Recording a withholding as a first-class outcome

Chapter 5 begins from an invariant. In accordance with an embodiment, where an input required by a determination is unavailable, incomplete, or unresolvable, the determination emits an outcome entry of a recorded abstention class identifying that input, and emits no outcome adverse to the semantic agent (100), to an asserting party (118), or to a counterparty. The invariant applies at each stage at which an input is required, irrespective of the cause of unavailability.

A degradation map (500) enumerates, for each determination stage, the input required, the abstention outcome produced upon unavailability, and the consequence foreclosed, path by path. Where the append-only lineage field (104) is unavailable, incomplete, or silent as to an asserted conduct, the admission evaluator (120) produces the not-determinable determination (126), foreclosing the rejected determination (124). Where the signed policy object (112) in force at the recorded assertion time cannot be resolved, no determination is produced, the conduct evaluation artifact (116) is appended as pending, and a structured inquiry requesting that object is generated. Where no acceptance determination is received within the window declared in the signed policy

object (112), the outcome is recorded as not-determinable, the authorization gate (300) remains in the provisional granting state, and resolution of the matter against the non-responding party is foreclosed.

One path is expressly not an abstention: an unresolvable reason-type bounds the magnitude of modification to the non-zero minimum declared in the signed policy object (112), foreclosing a magnitude of zero. That bound is policy-declared and the filing states no value for it. The result is the same as elsewhere, an outcome adverse to no party.

Load-bearing here is the conversion bar (502), which forecloses a consuming determination from converting an abstention outcome into a scalar value, a default value, an operand of a threshold comparison (510), or a consequence adverse to any party. The filing is explicit that this foreclosure is affirmative, the conversion barred by the structure of the values themselves and not by an absence of defined behavior. The outcome entry and a magnitude are values of disjoint types, with no total function mapping the former to the latter. A threshold comparison of such an entry emits an outcome of the recorded abstention class and not a Boolean; an accumulation over a set containing it emits an abstention outcome and not a sum over the remaining members. In accordance with an embodiment, the type discipline is enforced statically before the agent's instructions execute, whereby the offending determination is not constructible rather than foreclosed at the attempt.

Propagation carries the discipline forward. The outcome is appended to the append-only lineage field (104) as a first-class entry comprising the abstention class, identifiers of the producing stage and the unavailable input, and a recorded time. A determination consuming it emits an abstention outcome of its own recorded class, never replaced, resolved, or defaulted, so propagation terminates in an abstention at the final consuming determination and the unavailable input stays identifiable along the chain.

Two constructions matter for anything distributed. Where a determination requires an admissible counterparty class and the counterparty identity records (114) for the scope partition hold no admissible member, the counterparty-absence path produces a third outcome, neither execution of the action nor denial of it: the agent enters the non-executing cognitive mode (302), emits a structured inquiry naming the absent class and the delegation alternatives, appends both to the per-partition lineage record, and contracts that partition's capability envelope through coupled fields until the absence is cured. Separately, an agent that has written its authorization gate (300) to the withheld state (310) for an enumerated set of action classes and entered that same mode constructs a non-execution attestation (504) carrying the enumeration, an evidentiary basis disclosing no conduct descriptor content, an attested epoch field, an abstention-class type marker whose declared type is the abstention type (506), and an express designation that it is neither a determination nor a denial of any dispatch.

Upon a satisfied verification, a receiving agent writes that attestation as a carried abstention entry, adopting the type designation as received rather than re-deriving it, and the conversion bar (502) reaches across the boundary: no counter of the disclosing agent is incremented, nothing recording fault or diminished trust is appended to its counterparty identity record (114), and subsequent non-response in a withheld class is recorded as not-determinable rather than as a rejected determination (124) or an accepted determination (122). A failed verification is expressly not a denial.

Convergence, and where the designs diverge

Both designs converge on one commitment: an expired wait should be reported honestly, never silently forged into an answer. Inngest's public materials describe delivering that report across crashes and long horizons, and Chapter 5 depends on the same honesty when a required input proves unavailable.

Divergence begins at the layer, and at what the value may become afterward. A report handed back into a host language is an ordinary value of that language, and what happens next is settled by the code receiving it. Chapter 5 places the constraint inside the value: the outcome of an unavailable input is an entry of a recorded abstention class, held in a type disjoint from magnitude, propagating through every consuming stage, which a threshold comparison or an accumulation cannot reduce to a Boolean or a sum. What application code would otherwise settle by convention, the disclosed architecture requires be settled by construction, in an embodiment before execution begins.

A second separation concerns the counterparty. Durable orchestration addresses the progress of a workflow; Chapter 5 additionally addresses what one agent's withholding may do to another agent's books. The non-execution attestation (504) goes to each counterparty recorded as having dispatched to the disclosing agent within an emission window declared in the signed policy object (112), and to no other party, and admission advances no counter of the receiving agent denominated in any unit. That agent's dispatch-authority predicate thereafter fails where the requested action is of an enumerated class and the intended counterparty is the disclosing agent, and is unaffected for other classes and counterparties. The failure is recorded as a positive abstention, with no determination that the dispatch was impermissible and no fault of either agent. Release follows only a verified unmarked execution record in that class, a superseding attestation omitting it with a successor attested epoch field, or elapse of the time-to-live field, which elapse is expressly no evidence that the class has been restored.

Reading this from a durable workflow codebase

For a team already running durable functions, the filed chapter reduces to requirements for the governance layer above the orchestrator.

- Keep the result of an expired wait distinguishable from an answer for the life of the record, not only at the call site.
- Give that result a representation arithmetic, comparison, and accumulation cannot accept, so no later stage substitutes a default unnoticed.
- Carry the unavailable input's identity through every consuming stage instead of collapsing it into a status string.
- Settle in policy, not in scattered application code, what a counterparty's deliberate withholding may do to any score or counter kept about it.

None of this calls for replacing an orchestration platform. Chapter 5 discloses one arrangement for the governance layer that sits above one.

Disclosure Scope

This article describes subject matter of Chapter 5 of U.S. Provisional Application No. 64/117,812 and is published for defensive-publication and informational purposes. Architectural statements above are drawn from that chapter alone; other chapters of the application govern other subjects and are not described here. Quantities described as declared in a signed policy object are policy-declared, and the filing states no values for them. Class designations appearing in the chapter's worked trace are the filing's own illustration.

The application is pending. Nothing here is a legal opinion or an assertion of claim scope. References to Inngest are to public materials and are used for comparison only; no relationship, endorsement, or infringement is asserted.

Positive Abstention and the Conversion Bar (/positive-abstention)

All 49 steps → (/inventive-steps)

Refusing to act is a first-class, recorded outcome — and can't be laundered into acting.

Provisional application

PRIMARY TECHNICAL DISCLOSURE

- Positive Abstention: Withholding as a First-Class, Non-Convertible Outcome (/articles/positive-abstention/withholding-as-a-first-class-outcome)

SECONDARY TECHNICAL

- Relay Non-Execution Attestation With Depth and Cycle Bounds (/articles/positive-abstention/relay-non-execution-attestation)
- The Anti-Amplification Bound on Propagated Determinations (/articles/positive-abstention/anti-amplification-magnitude-bound)
- The Attestation Revocation Object (/articles/positive-abstention/attestation-revocation-object)
- Empty-Intersection Mutual Positive Abstention (/articles/positive-abstention/empty-intersection-mutual-abstention)
- Curing Counterparty Absence by Introduction Protocol (/articles/positive-abstention/introduction-protocol-cure)
- Recorded Counterfactual Coupling Test: Detecting When a Counterparty's Silence Moves an Agent's Own Authorization Gate (/articles/positive-abstention/recorded-counterfactual-coupling-test)
- Self-Execution Intersection Conjunct: Gating Cross-Agent Determination Propagation on the Admitting Agent's Own Per-Partition Execution Record (/articles/positive-abstention/propagation-admissibility-gate-self-execution-intersection)
- Attack-Conditional Cross-Boundary Suspension of Determination Propagation Between Persistent Semantic Agents (/articles/positive-abstention/propagation-admissibility-gate-attack-conditional-cross)
- Propagation Depth Ceilings for Cross-Agent Determination Records: A Monotonic Anti-Restore-Depth Gate Against Hop-Count Replenishment (/articles/positive-abstention/propagation-depth-ceiling-anti-restore-depth)
- Attestation Staleness Outcome and the Attestation Window Parameter: Bounding the Age of a Carried Withholding Without Recording a Rejection (/articles/positive-abstention/attestation-staleness-outcome-attestation-window-parameter)
- Evidentiary-Basis Omission for the Receiving Counterparty: Attesting a Withholding Without Naming the Recipient as Its Occasion (/articles/positive-abstention/evidentiary-basis-omission-receiving-counterparty)
- Persistence-Tier Emission Targeting for Non-Execution Attestations: Bounding the Audience of an Agent's Recorded Withholding (/articles/positive-abstention/non-execution-attestation-emission-targeting-persistence)

- Restoration Settlement Reference: Releasing a Carried Abstention Entry on a Matched-Pair Settlement Record of Gate Restoration (/articles/positive-abstention/restoration-settlement-reference-distinct-release-condition)

APPLICATIONS · GENERAL

- Safety-Critical Autonomy: Recording a Refusal to Act as a First-Class Outcome (/articles/positive-abstention/safety-critical-autonomy-abstention)
- Clinical Decision Support: When the Correct Output Is No Output (/articles/positive-abstention/clinical-decision-support-abstention)

APPLICATIONS · SPECIFIC

- NIST AI Risk Management Framework: Where Abstention Fits in Agent-to-Agent Conduct (/articles/positive-abstention/nist-ai-rmf-abstention)
- ServiceNow AI Agents: When the Correct Outcome Is No Action (/articles/positive-abstention/servicenow-ai-agents)
- Copilot Studio: Recording a Withholding as a Real Outcome (/articles/positive-abstention/copilot-studio-abstention)
- Bedrock AgentCore: Session Isolation and Governed Abstention (/articles/positive-abstention/bedrock-agentcore-abstention)
- **Inngest: Durable Waits and Withholding as an Outcome (/articles/positive-abstention/inngest)**

TERMINOLOGY

- abstention outcome (/glossary#abstention-outcome)
- conversion bar (/glossary#conversion-bar)
- non-execution attestation (/glossary#non-execution-attestation)
- positive abstention (/glossary#positive-abstention)

Positive Abstention and the Conversion Bar overview → (/positive-abstention)