Nick Clark · Published July 29, 2026 · PDF

Bedrock AgentCore: Session Isolation and Governed Abstention

An agent that declines to act produces something a downstream system has to store. In most operational stacks the storage is a failure code, a timeout, a null, or a health metric that quietly drifts downward, and the distinction between an agent that could not act and an agent that was found at fault disappears at the first arithmetic. AWS Bedrock AgentCore, as publicly described, is a managed foundation for running agents in production with isolated sessions, memory, tool access, identity, and observability. A filed architecture for positive abstention answers a different question at a different layer: a degradation map, a conversion bar, and a non-execution attestation that make a withholding a first-class recorded outcome adverse to no party, held in a type from which no total function reaches a magnitude.

What a Withholding Costs When Nobody Records It

A payments agent stops executing settlement instructions for one counterparty. What does the rest of the system now know? The orchestrator sees requests time out. A retry layer concludes the endpoint is degraded. A health score falls, a circuit breaker opens, and a dashboard reports a reliability number computed by counting silences, read by a human as evidence the agent is broken.

None of that records what happened. The agent declined a specific set of action classes and remained capable of everything else. A governed non-execution became a magnitude, and once it was a magnitude it was indistinguishable from a fault.

The same collapse happens inside one reasoning chain. A determination needs an input; the input is missing. The pipeline has to emit something, so it emits a default: zero, false, null, or the conservative denial. A later stage consumes that default without knowing it was manufactured. Three stages on, a decision stands against a party, and nothing in the trace says it rested on an input that was never there.

Two things are missing. A withholding needs a name, recorded when it occurs, identifying the unavailable input. That name then has to be structurally incapable of becoming a number, because the moment it can be added, compared, or scored, it stops recording an abstention and becomes a cost charged to somebody.

Where AgentCore Sits in the Stack

AWS Bedrock AgentCore is described in public AWS materials as a set of services for running AI agents in production rather than in a notebook. Its stated purpose is to carry the operational burden for teams building agents: hosting the agent process, providing durable memory, brokering tool access, handling identity, and giving observability into what the agent did. Public documentation also describes it as framework-agnostic, so a team can bring an agent written against the library it already uses.

Session isolation is among the properties those materials emphasize. The runtime is described as giving each session its own isolated execution environment, so that state and side effects from one session do not reach another. Anyone who has debugged a shared-process agent leaking context between tenants knows why that matters.

That is a real category, competently served. AWS also publicly documents Bedrock Guardrails as a configurable policy layer that can filter or block model inputs and outputs against content and topic policies. Between a runtime that isolates sessions and a layer that can block a response, an operator has a credible answer to where an agent runs and what stops it from saying something it should not.

Those are questions of hosting and content policy. The architecture below answers a question about the representation of an outcome, at a different layer, and the two are complementary.

The Filed Mechanism: A Type That Cannot Become a Number

Chapter 5 of U.S. Provisional Application No. 64/117,812 discloses, in accordance with an embodiment, an architecture in which no determination stage converts unavailability of a required input into an adverse consequence for any party. Each mechanism below is recited in that embodiment form.

A **degradation map (500)** enumerates, for each determination stage, the input required, the abstention outcome produced upon unavailability, and the consequence foreclosed. 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), and the consequence foreclosed is 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 arrives within the window declared in that object, the outcome is recorded as not-determinable, the authorization gate (300) stays in the provisional granting state, and what is foreclosed is resolution of the matter against the non-responding party. Further paths cover an origin-equivalence

class (200) that cannot be computed, a replenishment register (402) that cannot be read, and the absence of an admissible counterparty class, that last path producing a third outcome that is neither execution of the action nor a denial of it.

One exception is recited. Where the reason-type of an edge cannot be resolved, the magnitude of modification is bounded to the non-zero minimum declared in the signed policy object (112), for which the filing declares no value. The consequence foreclosed there is a magnitude of zero rather than an adverse magnitude, and the outcome remains adverse to no party.

Such an outcome is appended to the append-only lineage field (104) as a first-class outcome entry comprising the abstention class, an identifier of the stage that produced it, an identifier of the unavailable input, and a recorded time. A downstream determination consuming it emits an abstention outcome of its own recorded class, and no stage replaces, resolves, or defaults it. Propagation terminates in an abstention outcome at the final consuming determination, and the identity of the unavailable input is recoverable by following the chain.

The **conversion bar (502)** is what makes this hold. An abstention outcome is represented in a form disjoint from the domain of values a consuming determination can take as a magnitude: the outcome entry and a magnitude are values of disjoint types, and no total function maps the former to the latter. A threshold comparison (510) of the outcome entry emits an outcome of the recorded abstention class and does not emit a Boolean. An accumulation over a set containing the entry emits an abstention outcome, not a sum over the remaining members. The foreclosure is affirmative, the conversion being barred by the structure of the values themselves and not by an absence of defined behavior. In an embodiment, the type discipline is enforced statically before the agent's instructions execute, so a consuming determination attempting the conversion is not constructible.

Across the agent boundary, a **non-execution attestation (504)** carries the same property. A disclosing agent that has written its authorization gate (300) to the withheld state (310) for an enumerated set of action classes, and entered the non-executing cognitive mode (302), emits an attestation whose payload includes that enumeration, an enumerated evidentiary basis disclosing no conduct descriptor content, an attested epoch field drawn from its dynamic agent hash chain rather than a host clock, an abstention-class type marker whose declared type is the abstention type (506), and an express non-determination designation. Emission is complete upon emission: no acknowledgment is required, and the disclosing agent's state does not turn on whether any receiving agent admits it.

Verification on the receiving side tests the authority credential and continuity hash, the successor-continuity of the attested epoch, and resolution of the type marker against the closed enumeration in the signed policy object (112) in force. Where satisfied, the receiving agent writes a carried abstention entry as a value of the abstention type (506), adopting the designation as received rather than re-deriving it. Where a conjunct fails, the attestation is appended with the failing element, admitted to no consuming determination, and not treated as a denial of the disclosing agent.

The receiving agent's dispatch-authority predicate thereafter fails for a requested action of an enumerated class to that disclosing agent. That failure is recorded as a positive abstention and not a denial: a dispatch-authority abstention record is appended, and no determination that the dispatch was impermissible, and no fault of either agent, is recorded. Admission increments no refusal counter (304), no assertion-cost counter, and no refusal meter, and appends no attribute recording fault, breach, unreliability, degraded standing, or diminished trust to the disclosing agent's counterparty identity record (114).

Different Layers, Not Competing Ones

The divergence is one of layer, not of quality. AgentCore, as publicly described, addresses where an agent runs and how one session is isolated from another. Isolation is a containment property: it bounds what an execution can touch. The filed architecture addresses what an outcome *is* once produced, which is a representation property: it bounds what a value can become at every stage that consumes it.

Two different properties, and they compose. Containment holds across execution boundaries; representation holds across determination boundaries and across the agent boundary. A team with one still has to decide, separately, what its determinations emit when a required input is missing. The filed architecture answers that by requiring a value of the abstention type (506), disjoint from magnitude and reached by no total function.

Guardrail layers sit at a third position. A guardrail that blocks a response produces a blocked outcome, and the question the filing puts is what type that outcome carries at the next stage that reads it. The filed answer is exact: a type disjoint from magnitude, enforced in an embodiment before execution.

Coexistence, and What This Does Not Do

A plausible deployment runs the agent on managed infrastructure, uses the platform's isolation and observability for the operational envelope, and implements the abstention type inside the agent's own determination logic and in the payloads it exchanges with peers. The non-execution attestation (504) is a governed observation between agents carried in an application-level protocol, and nothing in the filing requires a particular host.

The disclosed architecture does not host agents, isolate sessions, broker tool access, or filter model output for harmful content, and offers no scheduler, memory service, or identity broker. A team adopting it still needs each of those, and the platform layer is

where they come from.

Nor does it decide whether an agent was right to withhold. In the worked trace, the write of the authorization gate (300) to the withheld state (310) occurs under the refusal-counter threshold of Chapter 3, which Chapter 5 takes as given. Chapter 5 governs the record: the withholding is emitted, verified, carried, and consumed as an outcome adverse to no party. The narrower question answered is whether the fact of it survives the trip downstream intact.

One more limit. Static enforcement of the conversion bar reaches the instructions of the semantic agent (100). Where an abstention outcome leaves that boundary, into a log sink or a metrics pipeline, the type discipline no longer travels with it. The guarantee is about determinations, not about every consumer that later reads the lineage record.

Disclosure Scope

The architecture described here is disclosed in U.S. Provisional Application No. 64/117,812, a pending application. Mechanism names, reference numerals, and outcome classes above are those of the filed specification, recited in embodiment form. Where the filing declares a quantity without a value, this article supplies none. Nothing here is a claim construction or a statement of claim scope.

References to AWS Bedrock AgentCore 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)**

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)