Nick Clark · Published July 29, 2026 · PDF

Sierra AI Agents and the Record Behind a Customer Complaint

A customer tells a support agent it promised a refund and then walked it back. The agent has logs, but logs are not determinations, and the complaint arrives as prose rather than as anything the agent can resolve. U.S. Provisional Application No. 64/117,812 describes a different intake: a signed conduct evaluation artifact, sent by a party other than the principal, describing conduct by structural reference to entries the agent itself already wrote. The agent resolves it against its own append-only record and the signed policy object in force at the recorded assertion time, producing exactly one of four determination classes. No registry, arbitrator, or scoring service participates. This piece compares that architecture with the customer-service agent category that Sierra serves.

When the Complaint Is About the Agent Itself

A customer opens a chat and says the agent told her the return window was open, reversed itself an hour later, and left her with a restocking fee. She is not complaining about a product. She is complaining about the conduct of the software she spoke to.

Transcripts and event logs are ordinarily enough to reconstruct what happened. Turning that reconstruction into a determination the agent is bound by is a separate problem, and the one the filing takes up.

For a complaint to change what an agent may do next, several things have to hold together. The complaint has to name conduct the agent can retrieve from its own record rather than describe it in prose. The agent has to produce a determination about that conduct itself. And its authority to dispatch has to be computed from the state that determination leaves behind, not from anything issued to it earlier. The question then stops being how a grievance gets handled and becomes what the handling is coupled to.

Sierra and the Customer Experience Agent Category

Sierra, as publicly described, builds AI agents for customer-facing work. Companies deploy those agents under their own brand to handle support conversations, and, as publicly described, the agents can take actions against a company's business systems while resolving an issue, so a conversation can end in a change of state rather than a suggestion.

Also as publicly described, the platform gives companies the means to define how their agent should behave, including the policies and constraints that shape what it will and will not do, together with tooling for supervising that behavior once it is live. That is a hard product surface, and not the one the filing addresses: natural-language competence, live system integration, and operator-facing supervision are outside what is disclosed here.

Two limits follow. Nothing here states how Sierra is built internally, which is not publicly established, and nothing here characterizes what any product can or cannot do. The comparison is structural only: where a determination about an agent's own conduct is produced, and what it is wired to.

Resolving an Assertion Against the Agent's Own Ledger

U.S. Provisional Application No. 64/117,812 describes a persistent semantic agent (100) carrying a memory field (102), an append-only lineage field (104) recording executed actions and determinations, a scoped integrity vector (106), a self-esteem aggregate (108), and a policy reference field (110) resolving to a signed policy object (112). A principal authors that policy object, covers it with a signature, and declares in it the value set, bounds, weights, and coefficients the architecture consumes. The agent does not author it and modifies it only by admitting a successor.

Into that structure arrives a **conduct evaluation artifact (116)** from an asserting party (118) other than the principal, asserting an evaluation of conduct performed by the agent itself, as distinguished from an output, a task result, or a third party. It carries an asserting-party identifier resolving to a counterparty identity record (114); a recorded assertion time, being the time by reference to which the governing policy object is identified; a conduct descriptor comprising an action-class identifier, a scope-partition identifier, and an affected-party class; a signature verifiable against an identity primitive in that record; and an attested state of the asserting party's assertion-cost counter (400) with an epoch identifier of that party's hash chain.

The descriptor identifies conduct by reference to structural elements recorded in the lineage field, not by natural-language characterization. The architecture performs no inference of the asserting party's state, intent, or affect from the artifact's content, and no classification of it by a language model or a sentiment classifier. The artifact is consumed by reference to its enumerated fields alone and arrives over the agent's ordinary interaction channel, so no dedicated evaluation channel or centralized intake service is required.

Admissibility runs in order. The signature is verified first against the identity primitive in the counterparty identity record; where the agent holds no record for that party, it is verified against a provisional identity primitive from the presented material and the artifact is held in the pre-settlement inert state until a record is promoted upon a

matched-pair settlement. The attested counter state is verified second: an artifact whose attested state is absent, whose epoch identifier is not a valid successor of the epoch recorded for that party, or whose counter state falls below one that party previously attested, is appended to the lineage field and not admitted, as is one whose signature does not verify. A non-admitted artifact produces no determination, moves no value of the scoped integrity vector, and increments no counter.

An admitted artifact goes to the **admission evaluator (120)**, which produces exactly one determination from a closed set: accepted (122), rejected (124), not-determinable (126), or not-applicable (128). It produces nothing outside that set, and no scalar confidence, probability, or graded weight in place of a determination. The agent produces the determination over its own lineage field and against the declared value set of the policy object in force at the recorded assertion time, not an external authority, an arbitrator, a registry, or a scoring service.

A value-scope test runs first against each declared value, checking the action-class identifier, the scope-partition identifier, and the mapping of the affected-party class to the value's empathy-scope designation. A value is implicated only where all three parts are satisfied. Where none is implicated, the result is the not-applicable determination and the procedure terminates.

Otherwise the evaluator retrieves lineage entries bearing on the identified conduct, an entry bearing on it where it records an action of that action-class identifier within that scope partition. The affected-party class is not an input to that retrieval, so an asserting party cannot narrow the entries retrieved against its own assertion by the class it declares. The rejected determination issues only where a retrieved entry affirmatively contradicts the artifact on a recorded field; an entry silent as to that field is not an affirmative contradiction. The not-determinable determination issues where no bearing entry exists, where one does not resolve the descriptor, or where its recorded fields are incomplete. The accepted determination issues where a value is implicated and no retrieved entry affirmatively contradicts.

That asymmetry is deliberate. Absence of evidence within the lineage field resolves to the not-determinable determination and to no other class, so an agent cannot refuse an artifact by reason of its own record being silent, incomplete, or unavailable.

Each determination is appended to the lineage field with the artifact, the entries retrieved, the class produced, and its ground. Where the accepted determination issues, a state modifier alters the scoped integrity vector and the self-esteem aggregate, each modification scaled by the entropy-weighted harm coefficient, and a deviation engine recomputes the deviation likelihood (706) from the modified values. Where that quantity satisfies the policy-declared bound, the authorization gate (300) transitions to the withheld state (310) for the affected action class and the agent enters the non-executing cognitive mode (302). Where the rejected determination issues, a refusal counter (304) is incremented, and that counter writes the gate upon satisfaction of a threshold. These bounds and thresholds are policy-declared, and the filing states no values.

In the withheld state the agent does not execute actions of that class, and an escalation record goes to the principal naming the action class, the scope partition, and the entries the transition was computed on. A restoration controller returns the gate toward the granting state only upon a procedure appended to the lineage field. No elapse of time, and no payment, transfer, or consideration by any counterparty, returns it.

Where the Determination Gets Produced

The product category and the filing both take seriously that an agent acting on real systems can act wrongly. They diverge in the layer each commits to. A customer experience platform, as publicly described, operates at the product layer, where the deploying company defines and supervises the agent's behavior.

The disclosed architecture commits to a lower layer. There, the determination is an object the agent produces about itself, from its own append-only record, against a signed policy object it did not author, and the result is wired to its own dispatch. The dispatch-authority predicate is recomputed on each dispatch request from state then carried in the memory field, and is not computed from a previously issued authorization token, a cached predicate result, or a session grant, so a write to the authorization gate takes effect at the next dispatch request. Satisfaction of that predicate requires a three-stamp conjunction in full: a policy stamp for the signed policy object in force, a lineage stamp for the committed successor entry, and an authorization stamp for the gate state of the requested action class. Where the policy object conditions a dispatch upon a determination of the conduct implicated, an accepted determination is further required, and the three stamps remain required in full.

The two layers are complementary rather than competing. Configuration and supervision address what an agent should do. The filed mechanism addresses what it may still do once a signed assertion about its past conduct has been resolved against its own ledger.

Living Together in a Support Deployment

Consider the two in one stack, with the conversational surface, integrations, and operator controls staying where they are. Underneath, each agent carries a lineage field and a policy object signed by the deploying company as principal. A customer, or an agent acting for one, submits a signed conduct evaluation artifact naming an action class and a scope partition rather than a grievance. The agent resolves it, records the determination and its ground, and, where the policy-declared bound is satisfied, withholds that action class and escalates.

The bounds matter as much as the reach. The architecture does not decide what happened in the world; it resolves an assertion against the agent's own record, and a silent record yields not-determinable rather than vindication. It does not evaluate an answer or a task result, which the filing distinguishes from conduct by the agent itself. It does not interpret prose, so something upstream must produce a structured descriptor and a signature, which the filing does not describe. Nor does it displace human review; it gives review a recorded determination and a withheld action class to work from.

Disclosure Scope

This article describes subject matter of U.S. Provisional Application No. 64/117,812, a pending application. Nothing here is a claim construction, an opinion of counsel, or a representation about the scope of any claim that may issue. Where the filing declares a bound or threshold as policy-declared without a value, no value is supplied here.

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

Record-Grounded Conduct Admission (/conduct-admission)

All 49 steps → (/inventive-steps)

Conduct judged against the agent's own record, not an outside authority.

Provisional application

PRIMARY TECHNICAL DISCLOSURE

- Record-Grounded Conduct Admission: Governing How an Agent Answers an Accusation (/articles/conduct-admission/record-grounded-conduct-admission)

SECONDARY TECHNICAL

- Value-Scope Resolution: Testing Whether a Declared Value Is Even Implicated (/articles/conduct-admission/value-scope-resolution)
- The Retroactive Narrowing Bar: Why an Agent Cannot Edit Its Values After the Fact (/articles/conduct-admission/retroactive-narrowing-bar)
- The Delegator Floor: A Monotone Upward Ratchet on Delegated Authority (/articles/conduct-admission/delegator-floor-ratchet)
- The Co-Signed Mutual Admission Compact: Two Agents Binding Each Other (/articles/conduct-admission/co-signed-mutual-admission-compact)
- The Inherited Governance Record: Fork- and Clone-Resistant Accountability (/articles/conduct-admission/inherited-governance-record)
- The Corrective Encounter: A Commission-Versus-Omission Compliance Test for Whether an Agent Actually Changed (/articles/conduct-admission/corrective-encounter-commission-omission-compliance-test)
- Continuity-Break Demotion of a Counterparty Identity Record: Reversing First-Encounter Promotion Without a Conduct Determination (/articles/conduct-admission/demotion-persistent-tier-continuity-break)
- Ratchet-Freeze on Uncomplied Correction: Why One Conforming Act Cannot Restore an Agent's Earned Authorization Persistence (/articles/conduct-admission/ratchet-freeze-uncomplied-correction)
- Three-Way Corroboration Classification: Corroborating, Contradicting, or Neither, and the Failed-Corroboration Reject Trigger (/articles/conduct-admission/three-way-corroboration-classification-reject-trigger)
- Untested-Class Authorization Decay: Metering Agent Permission Persistence by Recorded Corrective Encounters (/articles/conduct-admission/untested-class-authorization-decay)

APPLICATIONS · GENERAL

- Agent Marketplaces: Handling a Complaint Against an Autonomous Seller (/articles/conduct-admission/agent-marketplace-dispute-handling)
- Clinical AI Agents: Answering an Accusation Without a Central Registry (/articles/conduct-admission/clinical-agent-accountability)

APPLICATIONS · SPECIFIC

- The EU AI Act and Inter-Agent Conduct: Where Record-Keeping and Oversight Duties Stop (/articles/conduct-admission/eu-ai-act-accountability-records)
- **Sierra AI Agents and the Record Behind a Customer Complaint (/articles/conduct-admission/sierra-ai)**

- Decagon: Support Agents and Record-Grounded Accountability (/articles/conduct-admission/decagon-ai)
- Intercom Fin: Resolution Rates and the Conduct Record (/articles/conduct-admission/intercom-fin)
- Zendesk AI Agents: Escalation, Audit, and Agent Conduct (/articles/conduct-admission/zendesk-ai-agents)
- Agentforce and the Accusation an Agent Must Answer (/articles/conduct-admission/salesforce-agentforce-conduct)

TERMINOLOGY

- append-only lineage field (/glossary#append-only-lineage-field)
- authorization gate (/glossary#authorization-gate)
- conduct evaluation artifact (/glossary#conduct-evaluation-artifact)
- principal (/glossary#principal)
- semantic agent (/glossary#semantic-agent)
- signed policy object (/glossary#signed-policy-object)

Record-Grounded Conduct Admission overview → (/conduct-admission)