Nick Clark · Published April 25, 2026 · PDF

Enterprise Agent Fleets: Stable Agent Identity and Governed Tool Access Across Model Upgrades and Infrastructure Migration

Enterprises deploying fleets of AI agents face a structural problem: every model upgrade, vendor swap, or datacenter migration resets the agent's identity, audit trail, and access governance, forcing re-onboarding and breaking accountability. This application addresses that problem with an architecture built on the Agent-Resident Execution Substrate, disclosed in U.S. Provisional Application No. 64/070,239, in which each agent carries a hardware-bound persistent identity, an append-only lineage, and a cryptographically signed governance policy that survive replacement of every underlying model and update of the runtime itself.

What This Application Specifies

An enterprise that operates many AI agents (one per knowledge worker, one per business process, one per service desk queue) needs each agent to remain the same accountable entity over its working life. In practice it does not. The model behind the agent is upgraded on a vendor's cadence. Tool credentials rotate. The whole deployment migrates from one server, region, or cloud to the next. Under conventional tool-using

agent frameworks, agents are ephemeral sessions or stateless dispatch functions, so each of these events effectively spawns a new agent: the prior outcome history, the access scope, and the identity that auditors and counterparties relied on are gone.

This application specifies an enterprise agent fleet in which each agent is instead a persistent execution substrate, as disclosed in U.S. Provisional Application No. 64/070,239. In that architecture, the semantic agent is the persistent runtime authority of its computing system, and the inference models it uses are managed assets subordinate to it, held in a managed inference tool registry. The agent comprises four persistent fields: a persistent identity field, a cognitive state field, an append-only lineage field, and a governance policy field of cryptographically signed, machine-evaluable policy objects. The model artifacts loaded at any moment are merely the agent's currently available capabilities; the spec states plainly that the agent's identity is not dependent on any specific model artifact and is preserved across replacement, retraining, or removal of any subordinate model.

The governing mechanism is the substrate's continuity guarantee. Under it, the agent's identity, cognitive state, and lineage fields are not modified, reset, or interrupted by any lifecycle operation performed on a managed inference endpoint, by addition or removal of an endpoint from the tool registry, by an update to the governance policy field, or by an update to the substrate runtime itself, except by appending a record to the lineage field under a continuity proof. The continuity guarantee, the spec notes, permits arbitrary replacement of subordinate components, including replacement of every model currently registered, without any modification to the agent.

Why It Matters

For an enterprise fleet, identity that survives the model is the difference between an auditable workforce and a churn of anonymous processes. The substrate makes the agent, not the model, the unit that holds access rights and accountability. Three properties of the disclosed architecture carry the enterprise weight.

First, identity is hardware-anchored. The spec binds the persistent identity field to a hardware security element of the device (a secure enclave, trusted platform module, hardware security module, or embedded secure element) through a key-derivation operation, with the binding verifiable at any later event by re-deriving and comparing it. The identity is not transferable to a device lacking that bound hardware element except under a governed migration operation, and the substrate quarantines agent execution (suspending dispatch and refusing lifecycle operations) if the hardware element fails attestation. For a fleet, this means an agent's authority is rooted in attested hardware rather than a revocable account credential.

Second, governance is enforced per operation, not assumed. Every dispatch, every tool lifecycle operation, every ingestion, every counterparty encounter, and every runtime update is evaluated against the applicable signed policy objects, and the policy version under which it was admitted is recorded. The spec provides an enterprise server configuration explicitly: organizational policy objects govern admissibility, model size budgets, retraining schedules, and audit requirements, and may impose stricter constraints on ingestion sources, training corpus composition, and inference logging than a personal deployment would.

Third, the history is tamper-evident. The lineage field is append-only under continuity proofs, with each record carrying a cryptographic reference to its predecessor, so no prior record can be altered or deleted without producing a detectable continuity break. This is the substrate property that turns "the agent did X" into a verifiable claim an internal auditor or external regulator can check.

How It Composes With the Domain

Mapping the substrate primitives onto an enterprise fleet is direct, and each mapping is a faithful instance of what the spec already describes.

Governed tool access becomes the managed inference tool registry plus the agent-to-tool dispatcher. Each model an agent can call is a managed inference endpoint carrying a model artifact, an interface specification, and a governance scope, plus a signed, publisher-verified capability declaration. The dispatcher's routing decision is conditioned on the agent's cognitive state field, the request's input characteristics, and the applicable policy objects: candidate endpoints whose declared confidence, normative compliance, or capability fall below the policy thresholds are excluded before any call is made. Tool access in the enterprise sense (which model, for which task, under which constraints) is therefore a property the substrate evaluates at dispatch time, not a static credential.

Model upgrades become governed tool lifecycle operations. The tool lifecycle controller moves an endpoint through installed, active, retraining, updated-active, archived, and removed states, each transition gated by policy and recorded in lineage. The spec requires atomic substitution: retraining and substitution happen in a staging area distinct from the active registry, the updated artifact is promoted only on successful policy validation, and a failed validation rolls back to the prior artifact with the cause recorded. An enterprise can thus swap a model vendor or version under the agent without the agent itself changing, and a bad upgrade reverts without disturbing identity or history.

Runtime and infrastructure migration are likewise first-class. The substrate runtime (agent runtime, dispatcher, lifecycle controller, governance and resource subsystems) is updated through a staged, governed substrate-runtime update operation, with the agent's identity preserved across runtime versions by a continuity attestation that cryptographically chains the identity over the sequence of runtime versions. Moving the agent to new hardware uses the agent lifecycle controller's snapshot, transfer, and restore sequence under a migration policy object, with attestations from both originating and destination devices recorded in lineage, and may be denied if the destination fails declared hardware, governance, or trust criteria. The provisioning sequence supports a transfer-provisioning mode that initializes a new device from a

prior device's identity, lineage, cognitive state, policy objects, and personal corpus model artifacts under a governed migration. Datacenter, region, or cloud moves become governed migrations rather than re-instantiations.

Counterparties (clients, vendors, employees, contractual partners) are handled by counterparty identity records, which the spec lists for exactly this enterprise case. Each record carries a counterparty identifier, a counterparty scope object specifying admissible interactions and admissible lineage disclosure, an encounter history, and a persistence designation that can be promoted from ephemeral to persistent under organizational policy. An agent that survives its model upgrades also retains its working relationships and the access scope governing each one.

What This Enables

Concrete fleet capabilities follow from the disclosed mechanisms, all as enabling implementations rather than new technology.

A stable workforce roster. Because identity is hardware-anchored and model-independent, an enterprise can maintain a registry of agents whose identities persist across the entire upgrade cadence of their underlying models. Onboarding (granting a process, a counterparty relationship, a data scope to an agent) is done once against the agent identity, not re-done at every model release.

Vendor and model independence without re-accreditation. Multiple endpoints of distinct types and sizes can be co-resident in one agent's registry, including general-purpose models, task-specific fine-tuned models, and adapter-based variants over a shared base. The fleet can route across heterogeneous inference endpoints and substitute any of them under governance, so a model vendor change is a tool lifecycle event, not a fleet rebuild. Where a local endpoint lacks capability or capacity, the cloud-

burst forwarding subsystem can forward selected requests to a remote endpoint under a cloud-burst policy that enforces admissible destinations, disclosure scopes, encryption, and cost limits, with each forward recorded as an off-device disclosure event.

Audit-grade accountability across change. The append-only, continuity-proofed lineage gives every agent a reconstructible operational history spanning model upgrades, policy revisions, and migrations, each annotated with the policy version in force at the time. The privacy invariant adds an egress dimension auditors care about: lineage records, model artifacts, training corpora, personal corpus model parameters, and counterparty records are not transmitted off the device except under an explicit disclosure policy object, and every off-device disclosure is itself a verifiable lineage record. The spec states this permits a user or a regulatory authority to audit the complete record of off-device disclosures.

Segregated operating contexts within one agent. Scope records let a single agent identity hold independent professional, project, household, or regulatory-domain contexts, each with its own corpus policy, tool subset, and lineage partition, with cross-scope access permitted only under explicit inter-scope policy. A compliance-sensitive line of business can run as a governed scope inside the same accountable identity rather than as a separate, separately-audited agent.

Org-wide coordination without weight pooling. Federation coordinates agents across devices by exchanging lineage records, and optionally propagating governed model updates, under a federation policy declared at the user, household, or organizational scope, without a centralized federation authority and without pooling model weights. A federated agent identity record verifies through cross-device attestations that agents on multiple systems correspond to a single identity, preserved across device additions, retirements, and hardware refresh.

Boundary Conditions

This is an application framing of a provisional disclosure, U.S. Provisional Application No. 64/070,239, and an early-stage one: a provisional establishes priority and enabling disclosure but is not an issued claim set, and scope is settled only through subsequent prosecution. The enterprise scenarios here (workforce rosters, vendor swaps, datacenter migration, regulatory audit) are faithful deployments of the disclosed primitives, not independent inventions, and nothing here should be read as claiming the business practices of fleet operations themselves.

The substrate's guarantees are architectural, and their real-world strength depends on the implementation choices the spec leaves open. Hardware-anchored identity is only as strong as the device's hardware security element and its attestation; the continuity and privacy invariants depend on correct enforcement by the substrate runtime and its egress filter. The underlying building blocks the substrate composes (parameter-efficient fine-tuning methods, secure enclaves and trusted platform modules, append-only cryptographic chaining, network transport) are established prior art, and the disclosure's contribution is the agent-resident arrangement of them, not those components in isolation. This application also does not assert any specific performance, latency, throughput, or cost figures; resource budgets, retraining schedules, and migration criteria are policy-declared parameters set per deployment, not benchmarked results. Regulatory and contractual obligations that an enterprise must meet are external facts the governance and disclosure policy objects can be configured to enforce; they are not themselves part of the disclosed technology.

Disclosure Scope

The technology described in this article (the persistent hardware-bound agent identity, the managed inference tool registry and governed tool lifecycle, the cognitive-state-conditioned dispatcher across heterogeneous inference endpoints, the append-only continuity-proofed lineage, the continuity guarantee across model replacement and

substrate-runtime update, governed agent migration and transfer-provisioning, counterparty identity records, the privacy invariant and off-device disclosure framework, scope partitioning, and federation with a federated agent identity record) is disclosed in U.S. Provisional Application No. 64/070,239, the Agent-Resident Execution Substrate, on which this application is built. The enterprise framing in this article (agent fleets, knowledge-worker and business-process agents, vendor and model independence, datacenter and cloud migration, internal and regulatory audit) is provided as external domain context describing a faithful enabling implementation; it is not a representation of the patent claims, which are defined solely by the application as filed and any patent issuing from it, and references to regulatory or audit obligations describe configurable deployment context rather than legal advice.

Agent-Resident Execution Substrate (/agent-resident-execution-substrate)

All 40 steps → (/inventive-steps)

Persistent execution environment carried by the agent, not the host — identity, state, and lineage across power cycles, devices, and upgrades.

Provisional application

PRIMARY TECHNICAL DISCLOSURE

- Agent-Resident Execution Substrate, Articles (/articles/agent-resident-execution-substrate)

SECONDARY TECHNICAL

- Persistent Semantic Agent (/articles/agent-resident-execution-substrate/persistent-semantic-agent)
- Managed Inference Tool Registry (/articles/agent-resident-execution-substrate/managed-inference-tool-registry)
- Agent-to-Tool Dispatcher (/articles/agent-resident-execution-substrate/agent-to-tool-dispatcher)
- Lineage-Derived Training Signal (/articles/agent-resident-execution-substrate/lineage-derived-training-signal)

- Identity Preservation Across Upgrades (/articles/agent-resident-execution-substrate/identity-preservation-across-upgrades)
- Cognitive State-Conditioned Dispatch (/articles/agent-resident-execution-substrate/cognitive-state-conditioned-dispatch)
- Governed Tool Lifecycle (/articles/agent-resident-execution-substrate/governed-tool-lifecycle)
- Continuity-Proof Lineage (/articles/agent-resident-execution-substrate/continuity-proof-lineage)
- Substrate Runtime Continuity (/articles/agent-resident-execution-substrate/substrate-runtime-continuity)
- Personal Corpus Model Training (/articles/agent-resident-execution-substrate/personal-corpus-model-training)
- Heterogeneous Inference Endpoints (/articles/agent-resident-execution-substrate/heterogeneous-inference-endpoints)
- Atomic Lifecycle Substitution (/articles/agent-resident-execution-substrate/atomic-lifecycle-substitution)
- Integrity Signal Feedback (/articles/agent-resident-execution-substrate/integrity-signal-feedback)
- Hardware-Bound Identity (/articles/agent-resident-execution-substrate/hardware-bound-identity)
- Cognitive State Append-Only Invariant (/articles/agent-resident-execution-substrate/cognitive-state-append-only-invariant)
- Counterparty Identity Records (/articles/agent-resident-execution-substrate/counterparty-identity-records)
- Privacy Egress-Controlled Disclosure (/articles/agent-resident-execution-substrate/privacy-egress-controlled-disclosure)
- Federated Cross-Device Agent Identity (/articles/agent-resident-execution-substrate/federated-cross-device-agent-identity)

APPLICATIONS · GENERAL

- Personal AI Agents That Survive Device Loss: One Continuous Identity and a Private Corpus Across Every Device (/articles/agent-resident-execution-substrate/personal-cross-device-agents)
- **Enterprise Agent Fleets: Stable Agent Identity and Governed Tool Access Across Model Upgrades and Infrastructure Migration (/articles/agent-resident-execution-substrate/enterprise-agent-fleets)**
- Audit-Grade Agent Identity for Regulated Finance and Healthcare: Continuity-Proof Lineage Across the Agent Lifecycle (/articles/agent-resident-execution-substrate/regulated-industry-agents)
- Edge and On-Device Agents: Hardware-Bound Identity Across Heterogeneous Inference Endpoints (/articles/agent-resident-execution-substrate/edge-and-on-device-agents)
- Agent-to-Agent Commerce With Counterparty Identity Records and Egress-Controlled Disclosure (/articles/agent-resident-execution-substrate/agent-to-agent-commerce)

- Governed Tool Lifecycles for Managed Inference-Provider Ecosystems: A Substrate Approach to Owning, Routing, and Retiring AI Tools (/articles/agent-resident-execution-substrate/managed-tool-ecosystems)
- Proving Unbroken Continuity in Long-Lived Autonomous Systems Across Substrate Migration and Atomic Model Substitution (/articles/agent-resident-execution-substrate/long-lived-autonomous-systems)

Agent-Resident Execution Substrate overview → (/agent-resident-execution-substrate)