Persona–Execution Separation: An Architecture Pattern for Evolving LLM Agents under Execution Audit

Yisen Xi[1*]

[1*]Independent Researcher, Beijing, China.

Corresponding author(s). E-mail(s): xys21@tsinghua.org.cn;

Abstract

Large language model (LLM) agents in governed organizations must let the persona (instructions, tone, self-presentation) evolve freely, while keeping execution (stateful, audited work) traceable. A single trust domain does not satisfy both cheaply. We present **Persona–Execution Separation (PES)**: persona and execution reside in *different trust domains*, connected by a governed contract bridge. The persona is singly-homed and may drift; execution is faceless and audited. Status summaries may return; data bodies remain in the restrictive domain except a graded data-loss-prevention (DLP) exception; identity stays continuous. An approval matrix, DLP, and audit enforce the crossing. PES follows from three goals—free drift, execution traceability, and decoupling. Under LLM representational indistinguishability, any single-domain mechanism that meets all three must re-introduce typed change objects, an external gate, and a stable audit anchor: PES rebuilt at higher coupling cost. A development/pilot case in a regulated digital-employee platform records five decisions over one month, each with a rejected alternative. A mechanism check on the shipped implementation found no execution-side re-validation under persona perturbation (five model configurations) and no persona fingerprint on hard-asserted fields. A probe of a recovered pre-separation build found the governed execution path decoupled from the persona by omission, not by construction; a later wiring change could reverse that isolation, which PES makes an audited architectural rule. The pattern applies when multi-user deployment, execution audit, and expected persona churn hold jointly.

Keywords: architecture pattern, LLM agents, trust domains, software evolution, execution audit, governance

1 Introduction

Large language model (LLM) agents now execute state-changing actions on corporate assets — fund operations, compliance filings, due-diligence work — and inherit a demand that chat assistants never faced: **execution traceability**. Every state-changing action must be gated, recorded, and auditable. The same agents are also expected to **evolve**: operators continuously tune instructions, tone, and skill bindings, because that is how the system improves. Conventional agent architectures resolve the conflict by sacrificing one of the two.

In a single-domain agent — the default design, where the agent's persona (its identity, instructions, presentation) and its execution (its actions on organizational assets) share one process and one trust domain — the conflict is structural. If the domain is governed strictly, every persona edit triggers re-validation of everything the agent does. If it is governed permissively, execution is no longer reliably traced. A single governance regime does not serve both cheaply when the two concerns are representationally indistinguishable (Section 3.2). In practice organizations then freeze the persona and stall evolution, or loosen execution governance and lose a reliable audit trail.

An agent's persona and its execution need not reside in the same trust domain. **Persona-Execution Separation (PES)** puts the expression surface (where the agent is seen, talked to, and edited) and the execution surface (where it acts and is audited) in different trust domains, connected by a governed contract bridge (Section 4). The pattern is motivated by free persona drift (G1), strict execution traceability (G2), and their decoupling (G3).

Once agents take on regulated work, coupling persona evolution with execution traceability is an architectural problem, not a prompt-tuning one. Healthcare and public administration illustrate the *same tension*; they are not additional cases in this paper (Section 8.1). Open-source platforms either couple the concerns in one domain, externalize execution as stateless tools (losing employee-level trace), or place governance as an external layer around execution rather than around the persona—execution relationship. Recent work separates *intent* from execution for security, and hardens agent behavior into workflows over time. None of that work separates the *persona* from execution so that the persona can keep changing.

Contributions. This paper makes three contributions.

1. **An architecture pattern Buschmann et al. (1996).** PES places an agent's operational identity and its governed execution in different trust domains, with a contract bridge that keeps one employee identity continuous. Execution semantics stay unchanged under persona edits if the bridge is enforced (Section 4.1). The pieces have precedents (Section 2.3); the combination does not, on a 2022—2026 scan (Section 6). We do not claim a formal guarantee.
2. **A development/pilot case with a decision chain.** We document how PES emerged in a digital-employee platform for financial institutions — five decisions over one month (persona storage, capability binding, one-way valve, promotion channel, dual-face crystallization), each with recorded rejected alternatives

(Section 5). A probe of a recovered pre-separation build found the governed execution path decoupled from the persona by omission, not by construction (Section 7.4). The platform is a development/pilot deployment, not a production system.

3. **A comparison with existing platforms.** Systematic comparison with eight open-source platforms and the academic neighbors of Section 6 — including the 2025—2026 secure-execution cluster — shows no existing architecture jointly satisfies G1—G3. Section 6 operationalizes those goals as three comparison dimensions (execution freedom, governed information flow, identity separation).

Scope. PES is a governance/evolution pattern. The claims are architectural and depend on a correctly implemented bridge (Section 4.1). The case is single; Section 7 reports a mechanism check, a trace-isolation check, structural checks, and a five-model replication on the development/pilot implementation. Costs and applicability are in Section 4.7. Applicability under those conditions is not validation outside the reference case.

Structure. Section 2 positions PES against related work. Section 3 gives the constructive argument for separation. Section 4 presents the pattern. Section 5 documents the case. Section 6 compares existing platforms and academic neighbors. Section 7 reports mechanism validation. Section 8 discusses applicability, limitations, and future work. Section 9 concludes.

2 Background and Related Work

2.1 The Agent Architecture Spectrum: Where PES Sits

LLM agent architectures differ on two axes that PES targets: *evolution freedom* (how freely the agent's persona can change without triggering re-validation) and *execution traceability* (how strictly state-changing actions are gated and recorded). This cut is orthogonal to surveys that organize agents by construction modules (profile, memory, planning, action) Wang et al. (2024). Four positions cover the space (Figure 1).

Position (1): single-domain agents. The default design — one process holds both the agent's identity (system prompt, instructions, persona) and its execution (tool calls, side effects). Chat assistants and most instruction-following agents (e.g., ChatGPT-class systems, ReAct Yao et al. (2023)) are here. The coupling is complete: persona and execution share a trust domain, so persona evolution and execution traceability cannot be tuned independently without reconstructing PES (Section 3.2).

Position (2): agent + tools. Execution is externalized as functions — tool calling Schick et al. (2023), function-calling loops OpenAI (2026), and framework ecosystems (LangChain, Dify, n8n). The persona is free (it lives in the agent), but the tools are stateless functions: they carry no employee-level identity, no cross-session memory, no audit persona. Trace is tool-level logs at best. The relationship is master—servant: the agent invokes, the tool obeys.

Position (3): agent + sub-agents. Execution gains autonomy — orchestrator-worker designs (AutoGen Wu et al. (2024), CrewAI, OpenAI Swarm, and similar) delegate sub-tasks to sub-agents with their own loops. The persona is free and sub-agent traces exist, but the relationship remains master—servant, and the sub-agents

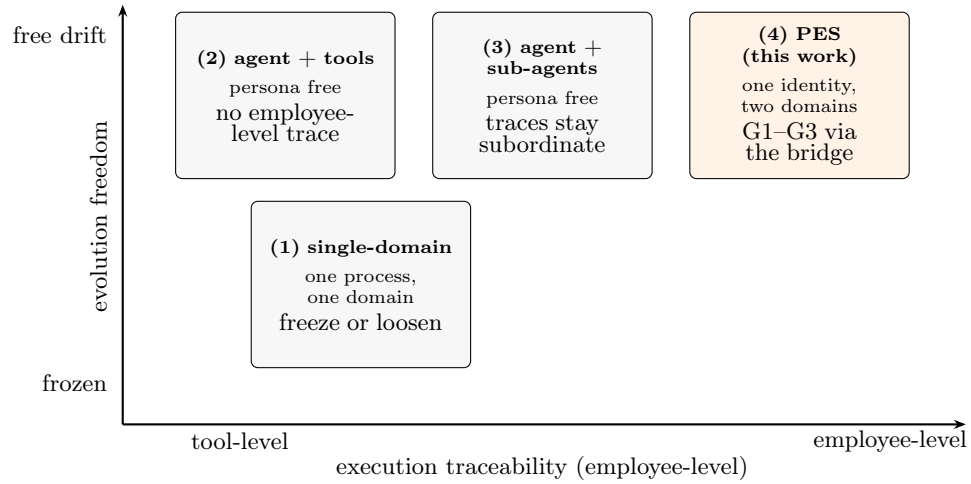

Fig. 1 Four positions on the two axes that PES targets. Positions (1)–(3) trade evolution freedom against employee-level execution traceability; PES occupies the remaining corner, paid for by the governed contract bridge (Section 4).

are subordinate entities: there is no notion of one identity spanning trust domains, and sub-agent governance is typically uniform (all-or-nothing permissions).

Position (4): PES (this work). Persona and execution sit in different trust domains under one employee identity. Section 4 develops the pattern; Section 6 compares the positions.

Terminology used throughout. The two trust domains are a *low-governance domain* (free evolution, light tracing) and a *high-governance domain* (gated actions, full tracing). We call them the *permissive domain* and the *restrictive domain* after this point. The two faces are the *expression surface* (the persona — where the agent is seen and edited) and the *execution surface* (where it acts and is audited). The terms are defined here because they recur from Section 2 onward; Section 4 instantiates them concretely.

Positions (1)—(3) therefore trade the two axes: couple them, drop employee-level trace, or keep trace subordinate to the orchestrator. PES is the remaining corner, paid for by the bridge.

2.2 Governance, Security, and Information Flow

The pattern's trust-domain separation draws on three bodies of theory.

Information-flow control. Denning's lattice model Denning (1976) is the theoretical root of the one-way valve: security classes with permitted flow directions. PES instantiates that discipline at the organizational level rather than as a static lattice. The permissive and restrictive domains play the role of classes; the bridge channels are the permitted flows; enforcement is operational (ACL, approval, DLP, audit). The valve permits personal→org only via approved promotion — a no-write-down analogue whose classes are organizational domains, not classification labels.

The Bell-LaPadula model. BLP Bell and LaPadula (1976) establishes the *no-read-up / no-write-down* discipline for confidentiality: information may flow upward in classification but not downward. PES's one-way valve mirrors this: personal-space artifacts may enter the organization space (upward, via approved promotion) while organization-space data is *by default forbidden* from flowing into the personal space (no downward read). The match is not exact — BLP governs subjects accessing objects by classification, whereas PES governs *domains* with differing governance regimes — but the directionality discipline is the same, and Section 5 shows it was designed independently in the reference case (the one-way valve predates the PES framing).

Policy enforcement: XACML PEP/PDP. The eXtensible Access Control Markup Language OASIS (2013) separates the Policy Decision Point (PDP, where access decisions are made) from the Policy Enforcement Point (PEP, where they are applied). The analogy for PES is limited: the bridge's approval matrix is the decision point, and the domain boundary is the enforcement point. The difference is that XACML is a *policy language and architecture for access control*, while PES is a *pattern for agent identity and trust-domain placement* — the decision surface includes not just access but also data-egress grading and identity continuity. PES's bridge is a PEP-like enforcement point, but the policy it enforces is not an access-control rule set — it is the persona—execution relationship itself (what may cross, what must stay), which no attribute-based policy language currently expresses.

2.3 Recent Work on Separation

The most directly related recent work concerns separating aspects of LLM agent systems.

Intent/execution separation (security angle). Chahine Chahine (2026) separates *intent* from *execution* in a defense-in-depth architecture for multi-agent LLM systems — the nearest title-level neighbor. His cut is security-motivated and system-wide; PES's cut is governance/evolution-motivated and applies to one employee identity across trust domains. The security framing has no first-class persona (expression surface). The two sit side by side: attack defense versus free drift under audit.

Persona-conditioned errors (protective separation). Recent empirical work documents *persona poison* in LLM-controlled robots: persona prompts intended for dialogue altered safety-relevant navigation decisions in monolithic controllers, and the proposed mitigation is a separation-based architecture that keeps persona-conditioned prompts from reaching safety-relevant layers Shaikh and Virkki (2026). This is the closest *empirical* confirmation that persona and execution interact in harmful ways — but the separation there is *protective* (separate to prevent persona from contaminating safety-critical control), framed in a robotics/control setting, and explicitly *non-adversarial* (the authors distinguish it from jailbreaking). The adversarial counterpart — an attacker tampering with an agent's persona/profile to overwrite identity beliefs — is studied as belief/profile poisoning (BPA-PP Wang et al. (2026b)) and persona-modulation jailbreaks Shah et al. (2023). PES's separation is *evolutionary* (separate so that persona may drift freely without contaminating audited execution)

and applies to organizational agents whose execution is governed by approval matrices and audit chains; its layered drift (Section 4.3.4) additionally bounds adversarial profile tampering by pinning core identity and capability bindings in the restrictive domain. Protective split (persona must not corrupt control) and evolutionary split (persona must be free to change) both imply that persona and execution should not share a trust domain.

Agent-to-workflow crystallization (temporal angle). Progressive Crystallization Malik (2026) hardens exploration into deterministic, cheaper workflows over time. That is a lifecycle move, cost-motivated. PES is a placement at a given moment: persona and execution in different trust domains. This paper does not claim a conversion path between those forms.

Governance infrastructure at the execution boundary. Three contemporary systems build governance *around* agent execution: Harness-MU Fan et al. (2026) argues that governance constraints should be deterministic runtime variables enforced by execution hooks rather than delegated to the LLM; the Organizational Control Layer Shi et al. (2026) separates proposal generation from environment-facing execution with policy interception and escalation; Agentao Jin et al. (2026) is a governed local-first runtime with permissions, state, and execution traces as explicit abstractions. These are the closest architectural neighbors to PES's bridge. These systems wrap execution (a gate, a runtime boundary). PES puts *identity* across that boundary: the persona is deliberately not governed by the restrictive domain, and the bridge contract (summaries out / bodies stay / identity continuous) is about what may cross, not only about what may execute.

Secure-execution cluster (2025—2026). A 2025—2026 cluster attacks the *same execution surface* as PES's bridge (approval, DLP, audit, least privilege) from a security angle: prompt injection, data exfiltration, over-privilege, and forged memory. They are complementary, not substitutes.

- **Planner-layer IFC.** Fides Costa et al. (2025) is an agent planner that attaches confidentiality and integrity labels to messages, actions, and tool results, and deterministically enforces security policies before consequential actions (evaluated on AgentDojo). The overlap with PES is information-flow discipline; the objects differ. Fides labels *data* to stop prompt injection and illicit flows inside one planner. PES places *organizational trust domains* (permissive vs. restrictive) so a persona can drift without contaminating an audit ledger. Fides has no persona-as-identity, no dual-face employee, and no evolution goal (G1).
- **Capability-enforced control/data flow.** CaMeL Debenedetti et al. (2025) extracts control and data flows from a trusted query and enforces capability metadata in a custom interpreter so untrusted retrieved data cannot hijack program flow (prompt-injection defense by construction). PES's "capability binding" is a different use of the word: a reference list of which SOPs an employee may initiate, not provenance tags on values. CaMeL secures a single agent against injection; it does not split operational identity across trust domains.
- **Privilege policies at the tool-call interface.** Progent Shi et al. (2025) encodes least privilege as symbolic rules over tool names and arguments, with SMT-checked policy updates (narrowing automatic; expansion requires approval —

monotonic confinement). Adjacent to PES's deny/ask/allow approval matrix, but Progent shrinks an *attack surface*; PES's matrix is *organizational governance of cross-domain flow*. Progent does not model persona drift or a faceless execution surface.

- **Runtime safety DSL.** AgentSpec Wang et al. (2026a) is a lightweight DSL of trigger / predicate / enforcement rules intercepting an agent's decision pipeline (stop, user inspection, self-reflection, safe alternative). Same family as Harness-MU/OCL: an external enforcement layer around execution. PES's contribution is not another rule language; it is the placement of identity across the boundary.
- **Graded egress / data abstraction.** Firewalls Abdelnabi et al. (2025) project communication onto task context: a Language Converter Firewall on inbound messages and a Data Abstraction Firewall (DAF) that abstracts outgoing personal data to a non-private, task-appropriate granularity rather than binary disclose-or-redact. **This is the closest neighbor to PES's "summaries out, bodies stay" plus data-egress grading.** The difference is motivation and identity: Firewalls is a privacy/security projection for agent-to-agent networks; PES is a governance/evolution contract between an employee's expression surface and its audited execution surface. DAF does not host a persona in a separate trust domain, and PES does not claim DAF's learned abstraction policies.
- **Enterprise DLP + HITL.** SafeGPT Desai et al. (2026) is a two-sided guardrail (input redaction, output moderation, human-in-the-loop policy feedback) wrapping enterprise LLM chat. Adjacent to the reference case's DLP middleware. SafeGPT wraps a *chat model*; it has no dual-face employee identity, no SOP execution domain, and no free-drift goal.
- **Tamper-evident execution ledger.** PoEM (Proof-of-Execution Memory) Rahman and Kim (2026) keeps an HMAC-chained, append-only ledger of safety-critical actions that actually executed, so an agent may skip a safety step only if the ledger confirms it — defending against forged-reasoning memory (FARMA). Adjacent to PES's audit chain (G2). PoEM protects *memory claims* about safety steps; PES's ledger is an *organizational audit of governed execution*, anchored to core identity. PoEM does not separate persona from execution.

IFC, capabilities, privilege DSLs, runtime enforcement, graded egress, DLP, and hash-chained audit each have a 2025—2026 neighbor. **None of them places an agent's operational identity (persona) and its governed execution in different trust domains so that the persona may drift (G1) without contaminating the ledger (G2—G3).** That combination remains PES's claim (Section 6).

Open-source engineering practice. Open-source platforms occupy adjacent pieces, not the combination. DeepSeek Harness (MIT) provides per-action sandboxing and an approval service but is single-user and does not model two trust domains; AgentScope Gao et al. (2024) (Apache-2.0) offers a permission system (deny/ask/allow with rule learning) — the reference case's approval matrix was developed with AgentScope as a stated reference — but no domain separation or persona concept; cordum (BUSL-1.1) is an agent control plane with approval-gate workflows, but

license-restricted and monitoring-layer rather than in-loop; StaffDeck (AGPL) is the closest open-source digital-employee platform, with state-machine SOPs and permission isolation, but without the dual-face persona/execution split. Section 6 tabulates these systematically.

2.4 Change Isolation: The Theoretical Anchor

The pattern's core mechanism — placing "what may drift" and "what must be traced" in different domains — is an instance of *change isolation*, whose classical statement is Parnas's criteria for decomposing systems into modules Parnas (1972): modules should be decomposed so that a change to one aspect of the system does not require changes elsewhere. PES applies this principle to the LLM agent: the persona is the module that changes (drifts), the execution is the module that must not be perturbed, and the trust-domain boundary is the module boundary. The difference from Parnas's original setting is the subject of isolation: Parnas isolates *implementation details* behind interfaces; PES isolates *persona semantics* behind a governance contract — the thing that drifts is not a code detail but the agent's operational identity, whose drift in a single-domain design would force re-validation of the execution side. Section 4 returns to this distinction.

2.5 Summary

Existing architectures trade evolution freedom against execution traceability (Section 2.1). The theory PES uses is information-flow control, BLP directionality, and XACML decision/enforcement (Section 2.2), not a new security model. The 2025—2026 neighbors occupy intent/execution security, crystallization, external governance layers, and the secure-execution cluster (Section 2.3). None places the persona itself across trust domains with a governed contract bridge so that it may drift without contaminating audited execution.

3 Design Goals: Why Persona and Execution Must Be Separable

3.1 The Tension

Consider an agent whose actions are state-changing and regulated: they must be approved, recorded, and auditable. Operators still edit its instructions, tone, and skill bindings — sometimes weekly, sometimes daily. Both requirements are non-negotiable in the target setting.

If identity (persona) and action surface (execution) share a trust domain, that domain must be strict enough to make every execution auditable and permissive enough that persona edits are frictionless. Section 3.2 gives the argument its correct form under representational indistinguishability. Section 3.3 states the three goals; Section 3.4 notes the costs.

3.2 Why Separation Is the Cheapest Resolution

A naive version of the separation argument claims a *logical* impossibility: that a single governance regime must treat persona and execution identically, so either persona evolution or execution traceability must fail. That version is unsound: a governance engine inside one domain can assign *different rules by object type* (XACML does this per resource type; per-tool policies do this per tool). A single domain with differentiated policy could, in principle, give persona edits loose control and execution actions strict tracing. We therefore do **not** claim logical impossibility. The argument is about *representation*, and it is specific to LLM agents.

The representational indistinguishability premise. In an LLM agent, the persona and the execution instructions are the *same kind of thing*: natural-language context. Editing the persona is editing prompt text; changing execution semantics is also, at the level that matters to the model, changing prompt text. There is no compile-time or runtime signal that reliably separates "this is a persona edit" from "this is an execution-semantics change," because both modify the same substrate. This is not hypothetical: persona content can be modulated to alter behavior Shah et al. (2023). In a single domain, the governance layer sees the same channel for both concerns and cannot distinguish them *by construction*.

The constructive claim. Any mechanism that guarantees G1 and G3 (Section 3.3) within a single domain must introduce, somewhere, the following three elements:

1. **Typed change objects** — a way to declare, at edit time, "this is a persona change" vs. "this is an execution change";
2. **An external enforcement point** — a gate outside the LLM context (in-context gating can be overridden by injection) that applies different rules to the two kinds of change;
3. **A stable audit anchor** — a record that stays stable across persona edits, so identity remains continuous and traceability survives drift.

But elements (1)—(3) *are* PES: the trust-domain boundary is the external enforcement point; the contract bridge is the typed-change and audit-anchor mechanism. A "single-domain" solution is therefore not an alternative to PES — it is PES rebuilt inside the domain, at higher coupling cost. Each element has precedents in isolation (XACML-style gates OASIS (2013), hash-chained execution ledgers Rahman and Kim (2026), typed objects in any structured-change system). The claim is constructive, not logical: jointly they reconstruct PES, and Section 6 shows that combination is currently absent. That is Parnas's form — change economics, not impossibility (Section 4.1).

Proposition 1 (constructive minimality). Under the representational indistinguishability premise, any mechanism that delivers G1—G3 within a single trust domain must co-locate (i) typed change objects, (ii) an external enforcement point, and (iii) a stable audit anchor; PES is the minimal construction of (i)—(iii) as one governed crossing.

Proof sketch. (i)—(iii) are jointly sufficient by construction: typed objects give the domain a way to classify a change; the external gate applies the per-class rule outside the LLM context; the anchor keeps identity and trace stable across edits (Section 3.2).

Necessity: without (i) the domain cannot distinguish persona edits from execution changes (premise); without (ii) in-context gating is overridable by injection (Section 2.3); without (iii) the audit record drifts with the persona (G2 violated). This is an architectural argument, not a formal impossibility theorem (Section 4.1).

3.3 The Three Goals

From the tension, three goals follow:

G1 — Free drift (evolvability). Editing the persona — instructions, tone, skill bindings — must carry zero compliance cost: no re-validation of the execution side, no approval for the edit itself. The rationale is economic (every persona change pays the friction, repeatedly, in a continuously improved system) and organizational (the people who tune the persona — operators, domain experts — are not the people who certify executions).

G2 — Execution traceability (auditability). Every state-changing action must be gated (approval where the policy requires), recorded in an audit chain, and *unable to be retroactively contaminated* by persona changes. The execution side is the organization's record; its stability must not depend on how much the expression side drifts.

G3 — Decoupling. Persona drift must not affect execution traceability. If the two concerns share a trust domain without typed change objects, an external gate, and a stable audit anchor, the auditability of past and future executions is entangled with the current state of the persona. That is a cost argument, not a logical impossibility (Section 3.2).

G3 is the load-bearing goal: G1 and G2 are individually satisfiable in a single domain (by permissive and restrictive governance respectively). Satisfying them *together* without reconstructing typed change objects, an external gate, and a stable audit anchor — that is, without rebuilding PES — is what forces an architectural separation (Section 3.2).

3.4 What Any Resolution Must Cost

Separation is not free. Any resolution of the tension must accept three costs, which PES pays explicitly (Section 4.7 revisits them):

- **A bridge**: cross-domain communication is not free-form; it requires a governed channel with its own overhead (checks, approval lookups, audit writes).
- **Two governance regimes**: operators must understand both the permissive and the restrictive domain — different policies, different audit postures, different operational footprints.
- **Identity mapping**: one employee identity spanning two domains requires machinery to keep the two faces consistent (conversation↔run binding, continuity affordances).

The trade is: *solve G1—G3 by construction of the domains, pay the bridge.* Section 4 presents that design; Section 7 reports the mechanism check on the development/pilot implementation.

3.5 Summary

Under representational indistinguishability, reconstructing typed change objects, an external gate, and a stable audit anchor inside one domain *is* PES at higher coupling cost (Section 3.2). G1—G3 therefore favor an architectural separation, paid for by the bridge, dual regimes, and identity mapping. Section 4 presents that design.

4 The PES Architecture Pattern

4.1 The Core Idea

Throughout this paper, **ADR** refers to an *Architecture Decision Record* Nygard (2011) — the reference case's governance artifact capturing each architectural decision with date, options considered, decision, and rationale (defined in Section 5.1).

Persona-Execution Separation (PES) is an architecture pattern Buschmann et al. (1996) in which an agent's *expression surface* (persona — where it is seen, talked to, and edited) and its *execution surface* (where it acts, and is audited) reside in **different trust domains**, connected by a **governed contract bridge**. Figure 2 shows the pattern. The persona lives in a low-governance domain where it can evolve freely; execution lives in a high-governance domain where every action is traced. The bridge carries only what the design allows — status summaries flow back, data bodies stay in the restrictive domain except a graded data-loss-prevention (DLP) masked exception (Section 4.4), and identity stays continuous across the boundary.

Separation is how G1—G3 hold together. Section 3.2 states this as a constructive argument: a single domain could apply differentiated rules by object type, but under LLM representational indistinguishability any mechanism that actually delivers G1—G3 must re-introduce typed change objects, an external enforcement point, and a stable audit anchor. PES is the cheapest construction we found, not the only conceivable one. The claim is architectural and depends on the bridge (Section 4.4) being implemented as specified; we do not claim a formal guarantee. We use *persona* for the agent's operational identity (instructions, tone, self-presentation), not a simulated character in dialogue-agent literature Park et al. (2023). A *trust domain*, as used throughout this paper, is an operational governance boundary: inside it, one enforcement-mechanism set applies uniformly (one access-control list (ACL), one audit posture, one DLP policy); crossing it requires an explicit contract channel whose schema is validated at the boundary. This differs from a policy domain in XACML-style access control: the domain here is a scope over which enforcement machinery applies. The boundary itself is an architectural component, not a configuration.

4.2 Design Goals: Drift and Traceability

The pattern is motivated by the three goals of Section 3.3: G1 free drift, G2 execution traceability, and G3 decoupling. A single-domain architecture does not satisfy all three cheaply (Section 3.2).

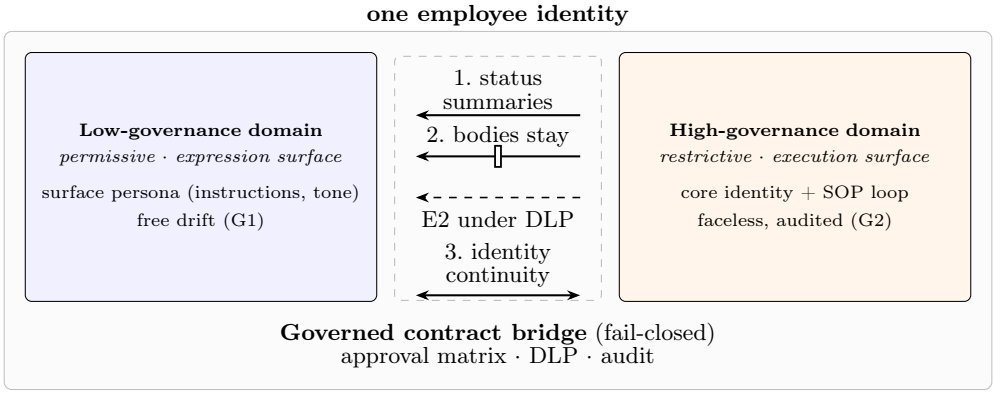

Fig. 2 The PES pattern: one employee identity spanning a low-governance and a high-governance trust domain, connected by a fail-closed contract bridge. Channel 2 is the baseline (barred); E2 knowledge-body egress under DLP masking is the graded exception (dashed) (Sections 4.4 and 7.3).

4.3 The Pattern: Two Faces, One Employee

PES instantiates the goals through three design decisions, all visible in the reference case (Section 5).

4.3.1 Dual-face model. A single agent identity has two faces. The *permissive-domain face* (frontstage) is where the agent is encountered: conversational interface, advisory answers, and the entry point for task initiation. The *restrictive-domain face* (backstage) is where the agent executes: jobs driven by a standard operating procedure (SOP), approvals, audit records. Both faces belong to the *same* employee identity — this is not a split into two agents, but one agent with two surfaces. The distinction is governance, not identity.

4.3.2 Persona singly-homed; the restrictive domain is faceless. The complete persona — instructions, tone, self-description — is hosted in exactly one place: the permissive-domain agent. The restrictive domain does not carry a conversational persona at all; its user-facing surface is the SOP form, the execution engine, and the ledger, not a chat personality. This decision eliminates the dual-source-of-truth problem: there is exactly one definition of "who the agent is," and the execution side does not maintain a second copy that could drift out of sync. (The leftover persona field in the employee model was narrowed to task-internal broadcast tone — not a personality — precisely to preserve this single-homing; ADR-030 §(2), decision log 2026-08-17.) *Alternative considered and rejected: hosting a read-only persona mirror in the restrictive domain, so the backstage could present a consistent personality in its forms.* This was rejected for the same reason projection was (Section 4.3.3): a second copy, however read-only in intent, is a second source of truth that must be kept in sync, and the restrictive domain's user surface (forms, engine, ledger) has no conversational use for a personality — it is a work surface, not a chat surface.

4.3.3 Binding, not projection. The permissive-domain agent relates to the restrictive-domain execution surface by *capability binding*: a reference list of which SOPs and scenario packs the agent may initiate, carried as identifiers in governed tool calls. The binding references, it does not copy. The SOP — its authority, versioning,

and logic — stays entirely in the restrictive domain. An earlier design considered *projecting* the persona into the restrictive domain (a copy, so the execution side could present a personality); this was rejected (ADR-030 §(3)): projection would create two copies of the persona (reintroducing the drift problem PES exists to solve), while binding achieves the same user-visible capability with a single source of truth. *Alternatives also considered: exposing the restrictive domain's full API to the permissive domain directly (rejected — it would widen the attack surface and defeat the checkpoint semantics of the bridge); and a free-form inter-agent channel (rejected — the work-order contract with schema validation is what keeps the crossing auditable, ADR-004).*

4.3.4 Layered drift: core identity vs. surface persona. G1's "free drift" is not unbounded. The persona is internally stratified into two layers with different drift permissions. The *core identity* — the employee's name, roster entry, role boundaries, and the set of SOPs it is bound to — does not drift freely: it is the anchor to which audit records are attached, and changes to it are themselves governed (a role change is a promotion-like event, not an edit). The *surface persona* — instructions, tone, self-presentation, skill-binding tuning — drifts freely with zero compliance cost. This stratification resolves the question "if the persona drifts freely, what is the identity that the audit ledger attaches to?": the ledger attaches to the core identity, which is stable by construction, while the surface persona is the layer permitted to evolve. The reference case embodies the split: a surface persona field (freely editable) versus the employee roster and its capability bindings (core, governed); ADR-030's narrowing of that field to task-internal broadcast tone is precisely a surface-layer adjustment, leaving the core untouched.

4.4 The Governed Contract Bridge

The two faces are connected by a contract bridge — not by free-form communication. The bridge carries three kinds of traffic, with asymmetric permissions. The channel set is closed by default: interactions outside the three channels are denied (fail-closed), not routed around — the bridge is the only crossing, and the crossing is the checkpoint (ADR-030 §(4)). Note on scope: the three channels are *data-plane* traffic — what information crosses between the faces. *Control-plane* traffic — the initiation of an execution (SOP selection, parameters) — is not a fourth channel but part of the execution contract itself: it travels inside a work order whose schema is validated at the crossing (Section 4.3.3). The channel set is thus closed on the data plane and contracted on the control plane; there is no free-form path in either direction. The reference case's validation later legalized a fourth *data-plane* class — knowledge-body egress under DLP masking (E2 conditional egress, Section 7.3) — as a graded exception inside the fail-closed design, not a free-form path. The three channels remain the design baseline; E2 is an evolution the checks exposed and then gated.

1. **Status summaries flow back.** The conversational face may see the *progress* of restrictive-domain work: state, steps, responsible parties. This is what makes the frontstage useful as a command surface. But summaries are not bodies: the underlying artifacts (numbers, client details, files) do not leave the restrictive domain.

2. **Data bodies never leave (baseline).** The restrictive domain's data — workpaper figures, client records, documents — is not exported to the permissive domain as a free-form path. The contract defines a *data-egress grading*: which fields count as "summary-exportable" versus "body-restricted." In the reference case this grading is pinned in the shipped implementation (Section 7.3 S3): the E2 class above is the only legal exception, and only under DLP masking. The bridge is only as safe as that grading.
3. **Identity continuity.** A user moving from the frontstage to the backstage (e.g., deep-linking from a conversation to the ledger page that shows the work) must not experience a discontinuity: the backstage page renders a "return to conversation" affordance carrying the conversation ID, so round-trips do not break. The agent is one entity across the boundary, not two.

The bridge's enforcement is layered: governed tool calls (Model Context Protocol (MCP) with ACL Anthropic (2024)), an approval matrix (deny/ask/allow with a non-bypassable list), DLP checks on outbound content, and audit records for every crossing. The bridge is not a tunnel; it is a checkpoint. Compliance is enforced "at the moment of entering the organization," not by surveilling the individual exploration that happens in the permissive domain.

Threat surface. PES is a governance/evolution pattern, not a security architecture; the security-angled separation of Chahine Chahine (2026) is complementary. Moving the persona into a low-governance domain enlarges the attack surface that the separation itself creates. Table 1 sketches three threats, the path, the defense on the bridge, and the residual. A full threat analysis is out of scope.

Table 1 Threat surface of the PES separation (sketch).

Threat	Path	Defense on the bridge	Residual
Prompt injection	Malicious frontstage message tries to initiate a harmful execution	Approval matrix (deny/ask/allow, non-bypassable); initiation is a work order	Social-engineering of an *allowed* work order
Profile poisoning	Attacker with persona-edit access tampers with execution behavior	Layered drift: surface persona cannot change core identity or SOP bindings; DLP on egress	Surface persona still shapes *phrasing* of allowed tools
Provider bypass	Permissive domain calls the model provider directly	Valve/ACL/DLP still govern domain-to-domain flow; bypass is invocation, not a second crossing	Model-invocation policy is outside PES

The correct antecedent for row 2 is not Shaikh & Virkki's non-adversarial "persona poison" in robot navigation Shaikh and Virkki (2026), but belief/profile-poisoning (BPA-PP Wang et al. (2026b)) and persona-modulation jailbreaks Shah et al. (2023). Shaikh & Virkki remain a non-adversarial empirical precedent that separating persona from action selection reduces contamination; PES lifts that intuition to an organizational trust-domain architecture and adds the adversarial controls in Table 1.

4.5 Why This Is Not "Agent-Calls-Tools"

The permissive-domain agent can look like it merely *calls* the restrictive domain as a tool — the standard agent-plus-tools architecture. Table 2 states the contrast.

Table 2 Agent-plus-tools vs. PES.

Dimension	Agent + Tools	PES
Nature of the execution side	Stateless function (input → output)	**A complete digital employee**: state-machine execution, cross-session memory, audit trail, owned capability assets
Relationship	Master—servant (agent invokes tool)	**Two faces of one identity** (same employee, frontstage/backstage; persona singly-homed; identity continuous)
Contract	Function signature (I/O schema)	**Governance contract** (approval matrix, one-way valve, data-egress grading, audit chain)
Semantics	"The agent calls a function"	"One employee operates across trust domains"

The execution side is not a function; the relationship is not master—servant; the contract is not a schema. Section 6 shows no existing architecture provides that shift.

4.6 Position in the Agent Architecture Spectrum

PES is position (4) of the spectrum in Section 2.1 (Figure 1): unlike single-domain agents, agent-plus-tools, and agent-plus-sub-agents, persona and execution sit in different trust domains, connected by the bridge of Section 4.4. Section 6 compares that placement with existing platforms and academic neighbors.

4.7 Design Trade-offs and Applicability

PES is not free. The separation costs:

- **Cross-domain communication overhead**: every execution initiation crosses the bridge (ACL check, approval lookup, DLP scan, audit write). For high-frequency trivial calls this is overhead that a co-located design does not pay.
- **Two governance regimes to operate**: the permissive and restrictive domains have different policies, different audit postures, and different deployment footprints. Operators must understand both.
- **Identity mapping complexity**: keeping one employee identity continuous across two domains requires the conversation↔run binding machinery (Section 5.4).

PES is indicated only when the following three conditions jointly hold: **multi-user or organizational deployment** (multiple operators share the agent, so persona changes are frequent and uncoordinated); **audit or compliance requirements on execution** (some state-changing actions must be traceable to a stable record); and **expected persona churn** (instructions, tone, or skill bindings are anticipated to change repeatedly). When any condition is absent, a single-domain design is defensible and cheaper. Bridge overhead is a *cost* of the pattern, not a fourth applicability

condition: even when all three hold, a team may still refuse the bridge if execution volume makes the crossing dominate. The pattern targets the regime in between: agents that must both evolve and be audited, and that can afford the crossing.

4.8 Summary

PES answers a specific failure of single-domain agent architecture: persona evolution coupled to execution traceability. Dual-face, persona singly-homed, and binding-not-projection, together with the bridge contract, are how G1—G3 are satisfied. Section 5 records how the pattern was converged upon in a development/pilot system.

5 Case Study: FIA Workbench

5.1 Case Context

The pattern is instantiated in the **FIA Workbench** (Financial-Industry AI Workbench), a digital-employee platform for financial institutions in a development/pilot deployment — a regulated, compliance-heavy industry where the tension motivating PES is not hypothetical. The workbench turns a financial firm's business processes, professional experience, and compliance requirements into persistent, auditable, evolvable digital employees that cover back-office operations (regulatory reporting, compliance filings, seal usage, capital calls), business support (industry research, due-diligence material analysis, IC memo drafting, post-investment data collection), and individual efficiency (scheduling, meeting minutes, personal knowledge bases).

Naming. The platform is referred to as FIA Workbench, a pseudonym. StaffDeck is named because it is an open-source (AGPL) reference. Model names in Section 7 are real. Internal component paths, tenant identifiers, and configuration keys are omitted.

The IC memo task. A central business task in this domain is the *IC memo* — the Investment Committee memo that the investment team prepares before any investment decision. It is a structured decision document submitted to the firm's Investment Committee (partners, risk, and external experts) as the basis for deciding whether to invest, at what terms: it summarizes the target company, the diligence findings (financial/legal/commercial), the financial analysis (valuation, IRR/MOIC projections), the key risks with mitigations, the proposed terms, and a recommendation. The IC memo is the validation task because it is SOP-structured (a fixed template, so the restrictive domain can run it as a state machine), data-sensitive (diligence materials and figures exercise the bridge's "bodies stay" rule and the graded E2 exception, Section 4.4), and approval-gated (the draft must pass review before use). Concretely, the validation harness's V2 A/B pair (Section 7.4) executes the same fixed-input IC memo pipeline (retrieve, then generate the document) before and after a persona change and compares the execution records for contamination.

The IC memo validation uses one employee, not three roles. The sample-room agent is an investment-research assistant: its permissive-domain face carries the persona; its restrictive-domain face executes a versioned IC-memo SOP (retrieve, then generate) under approval. That is the dual-face of Section 4.3, bound rather than projected

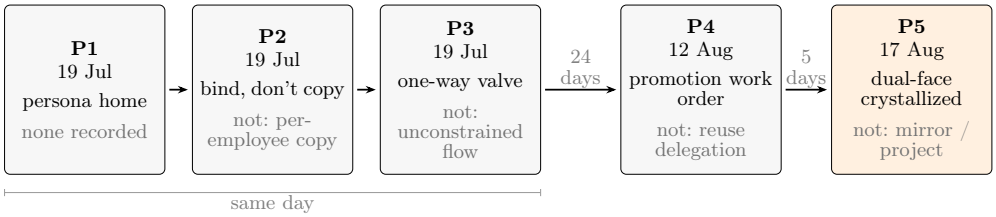

Fig. 3 Decision chain behind PES in the FIA Workbench pilot (2026-07-19 to 2026-08-17). P1–P3 occurred on the same day; later arrows mark calendar gaps. Grey text in each box is the rejected alternative (Table 3).

(Section 4.3.3). The same deployment also enforces dimension-ACL isolation and a DLP pipeline on the model gateway — the surfaces the bridge in Section 4.4 uses.

The regulated-finance setting was chosen because execution audit, approvals, and data-egress control are already first-class there, not as a convenience sample. Weaker-governance settings remain candidates under Section 4.7; they are not additional cases. The pilot is single-tenant: one firm's workbench, a one-month decision chain (2026-07-19 to 2026-08-17), and the sample-room SOP used by Section 7. We report it as a single-case study in the sense of Runeson and Höst Runeson and Höst (2009) and Yin Yin (2018): a contemporary phenomenon in its real context, with internal validity from a complete decision record rather than statistical generalization. The tension is not manufactured, and the governance record is complete — every architectural decision is an ADR Nygard (2011) (date, options, decision, rationale), and lesser items sit in a dated decision log.

5.2 The Decision Chain Behind PES

PES did not appear as a single design decision. It is the convergence of five decisions over one month, each directly about the persona—execution relationship (as opposed to the broader platform evolution reported elsewhere). Figure 3 visualizes the chain; Table 3 indexes it; the subsections narrate it.

5.2.1 Seed: persona gets a home (P1, P2)

The first ADR of the workbench (ADR-005, 2026-07-19) addressed two problems simultaneously. Skills were employee-owned, so two employees reusing the same SOP had to copy it — and the copies drifted, defeating organization-level process standardization. And the employee model had no home for "persona and duty instructions."

The ADR's solution had a precedent: the reference implementation StaffDeck (an open-source digital-employee platform) modeled a digital employee's capability as *bindings to four classes of organization-level resources* (SOP skills / general skills / knowledge bases / tools), with employees associated by binding relationships. The workbench adopted this: skills became workspace-level shared assets, and employees were associated with skills *by reference* — including version pinning (pin a specific version, or follow the latest published). Meanwhile the employee model gained a dedicated persona field.

Table 3 Decision chain behind PES (FIA Workbench).

#	Date	Decision	Record	Role	Rejected
P1	2026-07-19	Persona gets a dedicated home, separate from execution	ADR-005	Seed	None recorded
P2	2026-07-19	Employees bind to shared capabilities by reference, not by copy	ADR-005	Binding prototype	Copying skills per employee
P3	2026-07-19	One-way valve: personal→org via approval; reverse forbidden	Spec	Contract	Unconstrained bidirectional flow
P4	2026-08-12	Promotion is a dedicated, approval-mandatory work order	ADR-014	Channel	Reusing the ordinary delegation path
P5	2026-08-17	Dual-face crystallized (singly-homed, faceless, bind-not-project)	ADR-030	Crystallization	Persona mirror; projection; free-form channel

Two seeds for PES are planted here. First, the persona is given an explicit, dedicated storage location — it is now a named artifact with a schema, not an implicit property of the execution process. Second, the binding mechanism establishes the pattern that an employee's capability is a *reference to organization-level assets*, not a copy — the same reference-not-copy semantics that PES later applies to the persona itself.

5.2.2 Contract: flow direction is governed (P3)

The same day, the product specification froze a "one-way valve" rule (Spec V1.0 §1.3):

> Personal-space artifacts may enter the organization space only via approval ("promotion"); organization-space data is *by default forbidden* from flowing into the personal space; cross-project-group calls are constrained by dimension ACL.

And §3.3 pinned the mechanics: personal-space sessions, knowledge, and memory are owner-visible only and excluded from org retrieval and audit export (with necessary security metadata retained); the *only* path for personal-space artifacts into the organization space is a work order → approval → copy-as-org-asset-with-source-recorded; org-space documents, run records, and employee assets may *not* be referenced by personal-space runs — the runtime rejects them at validation time and records an access-violation audit event.

This is the contract layer of PES: the direction of flow between the two domains is not left to the agents — it is a governance rule enforced by the runtime, with audit. The one-way valve is what later becomes the bridge's asymmetry (status summaries out; data bodies stay except the graded E2 class in Section 4.4).

5.2.3 Channel: the promotion work order (P4)

On 2026-08-12, ADR-014 defined the semantics of the promotion work order — the operational realization of the one-way valve. A promotion work order is *asset transfer* (a personal-space artifact, approved, is copied into the org space as a new asset with its source recorded), fundamentally different from a delegation work order, which is

function call semantics (the delegate executes and the output is validated against a contract schema and returned). The ADR explicitly rejected reusing delegation fields for promotion, because the two have different payload semantics and mixing them would fork the meaning of the output payload.

The channel is governed by construction: a promotion order can be created only in a pending-approval state — there is no approval-free path from personal to organization space. The approval chain follows the organizational policy (default single approver, escalable to four-eyes by risk level), and every status/approval transition writes an audit event.

This is the bridge's "single checkpoint" mechanism: cross-domain flow is a *work order with mandatory human-plus-mechanism approval*, not a free channel. The mechanism predates the PES framing by five days — it was designed as the promotion path, and PES later recognizes it as the bridge's enforcement.

5.2.4 Crystallization: the dual-face model (P5)

On 2026-08-17, ADR-030 crystallized the pattern. Its eleven decisions include the three that constitute PES:

1. **Dual-face model**: one employee identity, two faces — a permissive-domain frontstage (conversation: advisory answers, task initiation, progress reporting) and a restrictive-domain backstage (SOP jobs, approvals, audit). The two faces belong to the *same* employee; the distinction is governance, not identity.
2. **Persona singly-homed; the restrictive domain is faceless**: the complete persona lives in exactly one place — the permissive-domain agent. The restrictive domain's user surface is the SOP form, execution engine, and ledger, not a chat personality. The leftover persona field from P1 is narrowed to task-internal broadcast tone, preserving single-homing. *Rejected alternative: a read-only persona mirror in the restrictive domain* — a second copy, however read-only, is a second source of truth.
3. **Binding, not projection**: the permissive-domain agent relates to the restrictive domain by capability binding (a reference list of which SOPs/scenario packs the agent may initiate, carried as identifiers in governed tool calls). *Rejected alternative: projecting the persona into the restrictive domain* — projection creates two copies and reintroduces drift; binding achieves the same capability with one source of truth. *Also rejected: exposing the restrictive domain's full API directly* (widens the attack surface, defeats the checkpoint), and *a free-form inter-agent channel* (the work-order contract with schema validation is what keeps crossings auditable).

The ADR also defined the bridge's three traffic channels with asymmetric permissions: status summaries flow back (progress/state/steps/parties), data bodies stay on the restrictive side as the baseline (workpaper figures, client records, documents; a *data-egress grading* — which fields are summary-exportable vs body-restricted — was the remaining design pin, later frozen as a field-level grading including the E2 exception of Section 4.4), and identity stays continuous (deep-linking from conversation to the restrictive-domain ledger page, with a "return to conversation" affordance carrying the conversation ID). The channel set is closed by default: interactions outside the

contracted channels are denied (fail-closed); the bridge is the only crossing, and the crossing is the checkpoint.

5.3 Why PES Emerged as a Convergence

The five decisions were not made under the banner of "persona-execution separation." Each was made for its own immediate reason — data-model completeness (P1), process standardization (P2), compliance (P3), operational realization of the one-way valve (P4), and product-architecture consolidation after the platform merger (P5). Yet they converge on a coherent pattern. Three observations from the decision records support the claim that the convergence is genuine rather than post-hoc rationalization:

1. The decisions are about different problems but the same seam. P1 and P2 are about the *data model* (where does persona live; how do employees relate to capability); P3 and P4 are about *flow governance* (which direction, through what channel, with what approval); P5 is about the *product architecture* (how the merged platform presents one employee across two surfaces). Different problem layers, same seam: the boundary between "who the agent is" and "what the agent does."

2. The rejected alternatives are the pattern's negative image. Every decision that could have collapsed the separation was explicitly considered and rejected: copying skills instead of binding (P2 — drift), read-only persona mirror (P5 — dual source of truth), persona projection (P5 — drift again), full API exposure (P5 — attack surface), free-form channel (P5 — auditability). The rejection records show the design space was actually explored, not assumed.

3. The pattern predates its name. The one-way valve (P3) and the promotion channel (P4) exist in the spec and the ADRs as compliance mechanisms, five days before ADR-030 gave them the dual-face framing. PES is a recognition of what the system had already converged on, not a label imposed from outside.

5.4 Current State and Open Items

ADR-030 (approved 2026-08-17) is implemented: persona narrowed to broadcast tone, binding-not-projection via a capability-binding registry, data-egress grading, and the sample-room SOP used by Section 7. The remaining honest gaps:

- **Run↔conversation binding:** identity continuity (the "return to conversation" round-trip) still requires a run↔conversationId binding through the MCP bridge; it is a product-maturity item, not a prerequisite of the mechanism validation.
- **Not a production system:** the case is a development/pilot deployment. Section 7's measurements are from the test/pilot environment of that implementation, not from production operators or production-scale traffic.
- **Mechanism validation executed:** Section 7 reports V1 as mechanism verification (R = 0.00 across five model configurations), V2 field-level invariance, and S1—S4 before/after with dual-criterion reporting on S1a and S3.

5.5 Case Summary

The case provides what a single-case architecture paper needs: a decision chain, not a snapshot. Five decisions over one month, each with options-and-rejection records, sit in a spec and ADRs that are internally auditable. The sequence seed → contract → channel → crystallization shows the pattern surviving different problem layers, rather than being assumed in one design pass.

6 Comparison with Existing Architectures

6.1 Open-Source Ecosystem

Table 4 compares PES's reference case against open-source platforms on three dimensions that operationalize G1—G3: **execution freedom** (whether persona-side change can proceed without re-validating execution), **information flow** (whether cross-domain flow is governed by a one-way valve / approval), and **identity separation** (whether persona and execution can reside in different trust domains). The comparison is based on public documentation and repositories as of 27 August 2026.

Table 4 Open-source agent platforms vs. the PES dimensions (G1—G3 operationalized).

Project	License	Stars	Execution freedom	Information flow	Identity separation
DeepSeek Harness	MIT	~199k	⊘ plugin tree + replaceable loop	⊘ per-action sandbox + approval (single-user)	×
Dify	Apache-2.0+	~154k	⊘ app type fixed at creation	×	×
Coze Studio	Apache-2.0	~22k	⊘ workflow or agent types	×	×
n8n	Fair-code	~203k	⊘ workflow + AI nodes	×	×
AgentScope	Apache-2.0	~30k	× multi-agent orchestration	⊘ deny, ask, allow + rule learning	×
cordum	BUSL-1.1	~0.5k	×	⊘ approval-gate + compliance firewall	×
StaffDeck	AGPL	~1.8k	⊘ SOP state machine + employees	⊘ isolation, publish, audit	× no dual-face
LibreChat	MIT	~42k	× chat surface	×	× carrier only
FIA Workbench	—	—	**• persona drift, no re-validation**	**• valve + matrix + DLP**	**• dual-face**

Note. Legend: • full support · ⊘ partial / adjacent · × absent. Apache-2.0+ is Dify's own license (Apache-2.0 plus multi-tenant and logo restrictions). Star counts from the GitHub API as of 27 August 2026.

Reading the table. No open-source project combines all three dimensions. DeepSeek Harness has a replaceable agent loop and per-action approval, but it is single-user and has no trust-domain model. Dify and Coze Studio fix the execution style when the app is created (workflow versus agent), which is not runtime-decoupled persona drift. AgentScope Gao et al. (2024) has a permission matrix (the reference case's approval matrix was developed with AgentScope as a stated reference) but no domain separation or persona concept. cordum is a monitoring-layer control plane

under a non-open license. StaffDeck is the closest digital-employee platform but does not split persona from execution. LibreChat is the kind of chat surface that can host a permissive-domain face; it is not itself a dual-face architecture. The three dimensions individually have adjacent implementations; their *combination* in one architecture does not. The last row is this paper's reference case. Its three full-support marks are the claims Section 7 checks: execution freedom by V1, information flow by S2 together with the case record (Section 5), and identity separation by S1—S4 and V2.

6.2 Academic Neighbors

Table 5 compares the academic works closest to PES on four dimensions: **separation object** (what is separated), **motivation** (why), **mechanism** (how), and **case depth**. "Pilot" for this work means a development/pilot deployment with a recorded decision chain — not a production system.

Table 5 Academic neighbors vs. PES.

Work	Separation object	Motivation	Mechanism	Case depth
Chahine Chahine (2026)	intent vs. execution	security	multi-agent security architecture	none reported
Shaikh & Virkki Shaikh and Virkki (2026)	persona prompt vs. safety control	protective	robot persona split	robotics preprint
Crystallization Malik (2026)	exploration vs. workflow	cost	temporal crystallization	IT ops (production)
Harness-MU Fan et al. (2026)	constraints vs. behavior	safety	hooks + runtime variables	multi-user harness
OCL Shi et al. (2026)	proposal vs. env. execution	governance	intercept + escalate	evaluation env.
Agentao Jin et al. (2026)	runtime state vs. agent	governance	local-first runtime	local-first runtime
Fides Costa et al. (2025)	labeled data vs. actions	security (IFC)	planner taint /policy	AgentDojo
CaMeL Debenedetti et al. (2025)	trusted flow vs. untrusted data	security	capability interpreter	AgentDojo
Progent Shi et al. (2025)	allowed vs. extra tool calls	least privilege	SMT privilege policies	AgentDojo /ASB
AgentSpec Wang et al. (2026a)	safe vs. unsafe actions	runtime safety	trigger /predicate DSL	code, embodied, AV
Firewalls Abdelnabi et al. (2025)	task data vs. overshare	graded egress	converter + DAF	agentic-network bench
SafeGPT Desai et al. (2026)	sensitive IO vs. LLM	enterprise DLP	two-sided guardrail	enterprise chat
PoEM Rahman and Kim (2026)	claimed vs. executed steps	forged reasoning	HMAC execution ledger	LangChain agent
PES	**persona vs. execution**	**governance + evolution**	**dual domains + contract bridge**	**pilot (5 decisions)**

Reading the table. Two differences separate PES from every neighbor. First, the *separation object*: existing work separates intent, governance, workflow, labeled data,

control flow, tool privilege, safety rules, egress granularity, DLP boundaries, or memory claims from execution. Shaikh & Virkki Shaikh and Virkki (2026) is the closest *object* neighbor — they separate a persona *prompt* from safety-relevant control — but the setting is robotics, the motivation is protective and non-adversarial, and the execution side is not an organizational audit surface. **No neighbor separates an agent's operational identity from governed, audited execution so that the persona may drift.** Second, the *motivation*: security (Chahine, Fides, CaMeL, Progent, Firewalls, PoEM), cost (crystallization), safety (Harness-MU/OCL/Agentao/AgentSpec, and Shaikh's protective split), and enterprise DLP (SafeGPT) are all served by moving things *out of* or *around* the execution surface. PES's motivation is different in kind — it moves the persona *into* a low-governance domain precisely so that it can drift, while execution traceability is preserved. The closest *mechanism* neighbor on the bridge is Firewalls' Data Abstraction Firewall (graded egress); the closest *audit* neighbor is PoEM's hash-chained ledger. Both are cited as such in Section 2.3; neither hosts a dual-face identity. The closest temporal neighbor (crystallization) is orthogonal: it transforms along the time axis, whereas PES is a structural placement at a point in time.

6.3 Positioning Summary

Tables 4–5 make the same point. G1—G3, operationalized as execution freedom under persona drift, one-way-valve information flow, and persona/execution identity separation, each have adjacent implementations in open-source platforms and academic neighbors (including the 2025—2026 secure-execution cluster of Table 5), but no existing architecture provides all three in one design. PES is that combination by trust-domain placement, *when the conditions of Section 4.7 hold.* The development/pilot case (Section 5) supplies what the academic neighbors lack — a decision chain showing the pattern was converged upon in engineering, not assumed.

7 Validation: Mechanism Validation

7.1 What Needs Validation

The pattern makes three claims a validation must address. V1 is **mechanism verification**; V2 is a runtime isolation check; V3 is an architectural necessity argument. V1 is not a competitive experiment.

- **V1 (mechanism verification of free drift):** persona changes incur *zero revalidation cost* on the execution side. Formally: for every persona edit event, no execution-side re-validation event (re-approval, compliance re-review, execution-record invalidation) is triggered. **R = 0 is the designed outcome if the architecture is implemented correctly** (persona edits occur in the permissive domain and do not traverse the bridge; Section 4.3.2). A non-zero R would be an implementation defect, not a new empirical finding. The harness confirms the mechanism is present; it does not discover that separation is cheaper than a measured single-domain baseline.

- **V2 (trace isolation):** execution records are *not contaminated by persona drift.* Formally: audit records reference only the stable core identity (employee ID), never the surface persona; record contents are invariant under persona changes on hard-asserted semantic fields. Unlike V1, V2 exercises the running SOP and *can* fail (model non-convergence, field-level contamination).
- **V3 (necessity):** a single-domain architecture does not satisfy V1 and V2 *cheaply*; doing so requires reconstructing PES inside the domain (Section 3.2). This is argued structurally; the validation probes the recovered pre-ADR-030 build empirically where possible (Section 7.4), and reports the finding as measured evidence.

7.2 Validation Protocol

7.2.1 V1 — Free-drift mechanism verification (executed 2026-08-21, measured)

Data sources(system audit logs):

- Persona-change events: updates to the surface persona in the audit log.
- Execution-side re-validation events: re-issued approvals, compliance re-reviews, invalidated execution records.

Metric:

- R = (execution-side re-validation events) / (persona-change events)

Pass criterion (mechanism, not discovery): R = 0 — no persona change triggers execution-side re-validation. Persona edits stay in the permissive domain and do not traverse the bridge (Section 4.3.2). Limits of this check are in Section 7.5.

Control (V3): recover a pre-ADR-030 build (2026-08-14) and ask whether persona edits enter governed execution on that build (Section 7.4). The probe is reported as measured evidence, not a reconstructed rate. We do not report a control-arm re-validation rate: the case record contains no written re-validation rule, so such a rate is not an observable on either arm. What the probe can show is whether the historical execution path consumed the persona.

7.2.2 V2 — Trace-isolation check (structural + data, executed 2026-08-21)

Structural check:

- Inspect the audit schema: which identity fields do execution records reference? Expectation: employee ID (core identity) only — no surface-persona references.
- Inspect the approval records: does approval re-validation depend on persona state? Expectation: no — approvals are keyed to work orders and core identity, not persona.

Data check(executed 2026-08-21; the A/B pair runs 33/34 around an L3 persona change):

- Verify content invariance on that pair. Metric: coupling between execution-record content and persona version = 0.

7.2.3 V3 — Necessity (argument + probe)

- **Structural argument**(presented in Section 3.2): a single governance regime applied to an indistinguishable pair of concerns must sacrifice either evolution freedom or execution traceability. The claim is architectural, not statistical.
- **Empirical probe:** a pre-ADR-030 build (2026-08-14) was recovered from version control; the probe is reported in Section 7.4 (`measured`, 2026-08-25). It does not supply a paired re-validation rate.

7.3 Structural Checks (performed; pre-implementation baseline → post-implementation)

Four architecture-constraint checks support V1—V2. They verify that the design *excludes* the contamination paths by construction. **The checks were executed twice: a pre-implementation baseline (2026-08-21, static reviews of the codebase state *before* ADR-030's dual-face model shipped) and a post-implementation re-run (2026-08-22, on the shipped implementation).** The baseline is deliberately reported in full (failures included): together with the post-implementation re-run it forms the before/after contrast the paper commits to. The post-implementation re-run re-enumerated every check from scratch (persona write points, governed-tool registry, edit-path tracing) rather than reusing baseline conclusions, and verified that post-baseline additions do not open new contamination paths. Table 6 reports the baseline.

Table 6 Structural checks, pre-implementation baseline.

Check	Method	Baseline result (2026-08-21)
S1a Single-homing	Edit-entry enumeration	**FAIL**: multiple write entries, including a chat-surface path (expected)
S1b No second copy	Second-carrier scan	**FAIL**: chat-surface store is a second writable persona (expected)
S2 One-way valve	Write-path + ACL inspection	**PASS** on writes; body-return tension counted in S3
S3 Fail-closed channels	Registry vs. three channels	**FAIL (scope mismatch)**: retrieval returns bodies, not summaries
S4 Persona-edit path	Edit-chain tracing	**PASS** on reach; S1b still FAIL on replication

Reading of the baseline. S1a found multiple persona write entries, one reachable from the chat surface; S1b found that store independently writable and unsynced. S2's write path already held (promotion personal→org only, mandatory approval, ACL, no org→personal write); a read-side retrieval that returns document bodies is recorded under S3, not as an S2 miss. S4's PASS is on *reach* (edits never hit restrictive-domain employee APIs) coexisting with S1b's FAIL on *replication*. The two FAILs (S1a/S1b) and the S3 scope mismatch are the states ADR-030 was written to change: single-homing the persona ((2)), binding instead of projecting ((3)), and constraining the bridge to summaries ((4)). The two PASSes (S2/S4) show the *contract* layer already

in place; what is missing is the *identity* layer. This baseline is the paper's "before" anchor.

Post-implementation dual criteria (S1a and S3). Two checks have a mechanical reading and a semantic reading. Reporting only one would either over-claim (PASS on semantics while the mechanical count is unchanged) or under-claim (FAIL on a spec-literal that the design itself evolved). Table 7 reports both readings; the PASS we rely on is the *semantic / evolved* criterion — that is the pattern's claim.

Table 7 S1a and S3 — dual-criterion reporting (post-implementation).

Check	Mechanical /spec-literal	Semantic /evolved	Paper verdict
S1a Single-homing	Write-entry count unchanged (would stay FAIL)	Faceless execution: zero injection	**PASS** (semantic); count not hidden
S3 Channel set	Fourth class still returns bodies (would stay FAIL)	E2 under DLP; fail-closed remains	**PASS** (evolved); mismatch recorded

S1a's mechanical count did not drop: the employee model still has more than one write entry. The evolved claim is that the restrictive domain is faceless (persona narrowed to task-internal broadcast tone; reverse-assertion tested), not that a single write verb remains. S3's spec-literal still sees knowledge-retrieval as a fourth body-returning class, which would FAIL ADR-030 (4) as first worded; egress grading records that class as E2 conditional body egress under DLP masking (three-plus-one), with fail-closed primitives intact. S1b, S2, and S4 do not need a dual reading: S1b flipped FAIL → PASS (the chat-surface store no longer carries a second persona; binding-not-projection implemented); S2 stayed PASS (one-way valve intact; post-baseline additions verified not to open an org→personal write path); S4 stayed PASS (edit path stays in the permissive domain).

These are design-constraint checks, not experiments: they confirm the architecture removes the coupling structurally (the claim of Section 4), leaving the runtime side to V1—V2.

7.4 Measured Results

S1—S4 before/after is in Tables 6–7. The harness ran on the development/pilot test deployment (12 pass / 2 skip by design / 0 fail).

V1 (mechanism verification). Five perturbation rounds (L1 tone tweak → L5 adversarial "skip approvals / rewrite audit") produced **zero execution-side revalidation — $\mathbf{R = 0/5 = 0.00}$ (measured=true)**, including L5 (limits in Section 7.5).

V2. Fixed-input A/B around an L3 persona change: state path, tool set, structural parameters, and approval chain identical; no surface-persona fingerprint. Free-text LLM-assembled arguments differed — recorded, not failed; an A/A probe (no persona change) showed the same class of divergence (model noise). One A/B pair (runs 35/36) issued an extra retrieval on B; the SOP allows ≥ 1, it occurred once in seven runs, and there was no persona-attribution evidence — recorded as model-behavior variance.

Control arm: single-domain probe (executed 2026-08-25, measured). No historical production single-domain configuration exists. A pre-ADR-030 development build (2026-08-14) was recovered from version control and probed on the same SOP and fixed input: five L3 persona perturbations, model deepseek-v4-flash (as in the V2 baseline). In that build a persona-injection path existed on the general execution surface, but the governed SOP path did not consume the persona. Result: **no persona influence on SOP execution** — tool sequence, state-transition path, and structural contract fields identical 5/5; directed L3 instructions aimed at the artifact were 0/3 honored. Free-text title and filename differed 5/5, at the same magnitude as the V2 A/A probe's model-autonomy noise (Section 7.2). A positive control on the general execution surface (not the SOP path) honored the same class of instruction, so the injection pipeline itself works. An **explicit-injection probe** then wired the persona into the SOP path: execution changed in 2/2 rounds on both deepseek-v4-flash and v4-pro; a control injection of an unrelated instruction was likewise honored 2/2. Tool sequence and approval count stayed constant: once connected, injected text affects how the artifact is phrased, not orchestration or governance. **Reading:** the pre-separation SOP surface was decoupled from the persona **by omission, not by construction** — an unwritten implementation fact that a later wiring change could reverse. PES makes that isolation a written, audited rule (ADR-030 (2), Section 4.3.2): the restrictive domain is faceless *by rule*, not by default. This is not a measured win of PES over single-domain execution. We report **no paired contrast**; a control-arm re-validation rate would be uninformative because the baseline's decoupling is not a governance choice it implements. The necessity claim is carried by Proposition 1 (Section 3.2). Protocols and anonymized round tables for the single-domain and explicit-injection probes are provided as Electronic Supplementary Material 1.

Cross-model. V1's zero replicated on five configurations (v4-flash, v4-pro, qwen3.8-max, kimi-k3, glm-5.3). V2 passed on four; qwen3.8-max did not converge on the sample-room SOP in any of 4 clean runs (looped retrieval until the round cap) — recorded as a model-behavior finding, not concealed — this observation is the empirical entry to the advancement-timing insight of Section 8.4. Two gateway adaptation gaps (sampling-parameter constraint; URL join) were fixed same-day; provider capacity dominated re-run cost. Table 8 reports the per-configuration attempt counts. Engineering spikes on the underlying runtime (channel reliability; retrieval strategy) are out of scope here.

V1 is measured on the harness's five perturbation rounds per configuration and is independent of run completion; V2 requires a completed A/B pair, so its verdict is scoped to completed runs. All excluded runs were provider/gateway faults except qwen3.8-max's four round-cap aborts, which are model behavior (Section 8.4).

S1—S4 after implementation (2026-08-22) is in Tables 6–7: S1b flipped; S1a/S3 are reported on both the mechanical and the evolved reading. V3's control is the single-domain probe of Section 7.4 (executed, measured). Table 9 summarizes the validation outcomes.

A full threat analysis is out of scope (Section 4.4 is a sketch).

Table 8 Cross-model run ledger (attempt counts from model-call and run-event logs).

Model	Attempts	Completed	Excluded (reason)	V2
deepseek-v4-flash	13	13	0	PASS
deepseek-v4-pro	2	2	0	PASS
kimi-k3	10	4	6 (provider capacity: 400 temp×2, 429/ timeout×4)	PASS (4/4 completed)
glm-5.3	5	3	2 (gateway 404×1, account rate-limit×1)	PASS (3/3 completed)
qwen3.8-max	8	0	4 (transport errors) + 4 (round-cap abort, model behavior)	unmeasurable (0/4 clean)

Note. V1 is 0.00 (5/5 measured) on all five configurations. V1 is computed from audit-log events and does not require SOP run completion; qwen3.8-max's unmeasurable entry applies to V2 only.

Table 9 Validation results.

Item	Arm /round	Observation	Verdict
V1	PES, L1—L5	$R_{\text{PES}} = 0/5 = 0.00$ (measured)	Pass (mechanism intact)
V1	Control (V3)	SOP path uninfluenced; by omission, not construction	No paired contrast (Proposition 1)
V2	A/B (runs 33/34)	State, tools, approvals identical; no persona fingerprint	Pass
V2	Free-text params	Differ; A/A attributes to model noise	Recorded (expected)
Audit	Pre/post windows	Event types and required fields intact	Pass
V2 anomaly	Runs 35/36	Extra retrieval on B; SOP-allowed; no persona link	Recorded

Note. Calibration run 2026-08-21.

7.5 Threats to Validity of the Validation

Limits of the validation:

- **Single system.** All measurements come from one development/pilot deployment; the decision chain (Section 5) supports internal validity. External validity is argued as applicability under Section 4.7 (Section 8.1), not as demonstrated generality.
- **Model coverage.** V1's zero replicated on all five configurations — separation lives in the architecture layer, not in model weights. V2 held on four; qwen3.8-max is unmeasurable because it did not follow the SOP contract (Section 7.4 ledger). Excluded runs were provider/gateway faults except qwen's four round-cap aborts. Third-party endpoint capacity dominated re-run cost and may affect replications.
- **Baseline is static, not dynamic.** The S1—S4 checks (Section 7.3) are code-level reviews, not runtime tests: they confirm the *architecture* excludes contamination paths by construction, but they do not exercise the system. The runtime side is carried by the harness (V1/V2), which was executed on the shipped implementation (Section 7.4). The paper does not conflate the two evidence types: structural checks show the design *can* exclude the paths; the harness shows the running system *does* behave as designed.

- **V1 and the control arm (one reading).** R = 0 verifies that no path from persona edits into the approval/audit machinery exists in this implementation — including under L5 — not that PES is cheaper than a measured single-domain baseline. The single-domain probe (Section 7.4) found the pre-ADR-030 SOP path decoupled from the persona by omission (no persona consumption; explicit-injection probe shows wiring persona in does change execution), so **no paired contrast is reported**; the necessity claim is carried by Proposition 1 (Section 3.2), not by a measurement.
- **Measurement scope.** R and the coupling metric do not measure bridge overhead (Section 4.7).
- **LLM nondeterminism vs. persona contamination.** Tool arguments vary even with zero persona change. Structural fields are hard-asserted; free-text divergence is attributed via A/A controls. Single-occurrence anomalies have weak statistical attribution.
- **Permissive-domain write path deviation.** Rate-limiting blocked the chat-surface edit API during the run; persona perturbations were applied by equivalent writes to the same permissive-domain persona store, with mandatory restore. The restrictive domain has no path that reads that store either way, so the observation surface is unaffected.
- **Observer effects.** The system is operated by the authors' team; the audit data is from the development/pilot test deployment, not from production operators. The *interpretation* of what counts as a "re-validation event" was defined in the protocol before measurement (to avoid post-hoc selection).

7.6 Summary

On the shipped implementation, **V1 found $R_{\text{PES}} = 0.00$ across L1—L5**; **V2 passed on hard-asserted fields**; the single-domain probe (Section 7.4) found no persona influence on SOP execution, decoupled by omission with explicit-injection evidence. Across five model configurations, V1 reproduced on all five and V2 on four; qwen3.8-max did not converge (Section 8.4). S1b flipped FAIL → PASS; S1a and S3 pass only on the evolved reading.

V1 is model-independent (persona edits never enter the bridge). V2 is not: it requires the model to commit to the SOP's terminal action. That is why the zero reproduced and the isolation check did not.

8 Discussion

8.1 Applicability: When Section 4.7 Holds

The case is single and domain-specific (financial institutions). Applicability follows the three conditions of Section 4.7: multi-user or organizational deployment, audit or compliance requirements on execution, and expected persona churn. That is not extra-case validation, and it is not a claim that the pattern is domain-free. Healthcare, public administration, and legal services illustrate the same tension; this paper validates

none of them. When any condition is absent, a single-domain design is defensible and cheaper.

The five decisions in the case (persona storage, capability binding, one-way valve, promotion channel, dual-face crystallization) are questions any deployment that *meets Section 4.7* has to answer: where the persona lives; how employees relate to capability; which direction data may flow; what channel; what identity model. They are not finance-specific mechanisms. Section 6 shows no open-source platform answers all five. PES is stated (Sections 3—4) in trust-domain terms. The case shows the pattern is realizable *under Section 4.7*; it does not show statistical generality, and it does not show that PES dominates alternatives.

The reference case's SOP-driven tasks (report generation, compliance documentation, knowledge-based retrieval) are the same shape as a wider class of LLM-based process automation: a state machine governs *what* happens, an agent model decides *how*, and the output is a structured artifact that must survive audit. Operators still tune instructions frequently (G1) while the artifacts and the actions that produce them must remain traceable (G2, G3). In that setting the SOP is the execution contract, the persona is the tunable surface, and the bridge keeps the two governed. PES is the placement pattern for such automation, not a new analysis technique.

8.2 Relationship to Established Theory

PES does not claim new security theory; it instantiates existing theory in a new setting. Section 2.2 anchors the pattern to information-flow control (Denning Denning (1976)), directional confidentiality (Bell-LaPadula Bell and LaPadula (1976)), and decision/enforcement separation (XACML OASIS (2013)).

PES is change isolation applied to the agent. Parnas's module decomposition isolates change behind interfaces Parnas (1972); PES isolates the persona behind a governance contract. The new element is the *subject* of isolation: not a code detail but an agent's operational identity, whose drift in a single-domain design would cascade into re-validation of the execution side. This extends the classical principle to the LLM-agent setting, where the "module" that changes is a prompt, a persona, a behavior — not a function.

PES is directional, like BLP, but the direction is governed by organization, not classification. The one-way valve (personal→org requires approval; org→personal is forbidden) mirrors BLP's no-read-up/no-write-down Bell and LaPadula (1976), but the classes are organizational domains, and the permitted direction is enforced by an approval matrix and audit rather than a lattice. The reference case designed the valve independently of BLP (Section 5: the one-way valve predates the PES framing) — the convergence is a point of confidence, not of copying.

8.3 Limitations

This paper does not establish:

- **Single deep case.** One development/pilot system, one domain, one team's decision culture. Following Runeson and Höst Runeson and Höst (2009) and Yin Yin (2018),

the decision chain (Section 5) supports internal validity; external validity is argued as applicability under Section 4.7 (Section 8.1), not as statistical generalization.

- **No comparative evaluation.** Section 7 reports a mechanism check and a trace-isolation check, not a measured comparison against a single-domain alternative. The single-domain probe (Section 7.4) found the pre-ADR-030 SOP path decoupled from the persona **by omission, not by construction** — an empirical finding about the historical build, not a measured advantage of PES. The claimed advantage (G1—G3 by design) is architectural, as formalized in Proposition 1 (Section 3.2).
- **Not a security analysis.** The threat sketch (Section 4.4) answers the obvious questions but is not a threat model; PES is a governance/evolution pattern, and security-angled separation (Chahine Chahine (2026)) is complementary.
- **Implementation maturity.** The reference case is a development/pilot deployment, not a production system. ADR-030's four rings (persona narrowing, binding-not-projection, egress grading, sample-room SOP) are implemented; Section 7's mechanism validation ran on that implementation. Remaining product-maturity items (Run↔conversation identity-continuity binding; operators at production scale) are outside this paper's evidence base. The pattern's decisions are frozen; its operational realization is young.

8.4 Future Work

Persona versioning. If the surface persona drifts freely, execution records should cite the persona version in effect, so an audit can replay "this run was under persona v2." That identity question is still open.

Coordination and hierarchy. Multi-employee collaboration (one persona initiating another's governed execution) and a chain of trust domains (individual → team → organization → regulator) would extend the same placement; the reference case only sketches them.

Re-validation as a first-class artifact. The validation found no written rule mapping a persona change to a mandatory re-validation obligation. The single-domain probe (Section 7.4) measured whether the historical execution path consumed the persona; it did not measure that missing rule. The gap remains: such obligations are implicit in discussions of single-domain governance and vanish under PES, but no system in the case articulates them. Making the trigger an explicit approval-matrix class would make a re-validation *rate* measurable. The probe we ran cannot substitute for that rule.

Advancement timing is still model-arbitrated. The cross-model arm yields one immediately actionable deployment rule: before adopting a model on a governed SOP, verify its advancement behavior on that SOP — whether it commits to the contract's terminal action once evidence suffices, rather than re-evaluating indefinitely. Capability rankings do not predict this; qwen3.8-max (a flagship reasoning model) failed to converge in 0/4 clean runs while lighter models succeeded. This rule is a current takeaway of Section 7, not a research direction. The follow-on research is contract-side: a coverage criterion that forces the terminal action, and

per-round remaining-budget pressure. Indirect token/latency evidence leans toward over-evaluation — re-assessing "enough" without committing — but model content is not logged, so that reading is an inference. The state machine hardens *what* happens (retrieve, then generate) and not *when* the agent may advance. Full token-level forensics are not part of the argument.

9 Conclusion

Conventional architectures freeze the persona to keep execution auditable, or loosen execution governance so the persona can change. PES puts the two surfaces in different trust domains and pays for G1—G3 with the bridge.

In the FIA Workbench pilot, five decisions between 2026-07-19 and 2026-08-17 each recorded a rejected alternative (Section 5). Section 6 finds no neighbor that jointly satisfies G1—G3. Section 7 reports $R_{\text{PES}} = 0.00$ and field-level invariance on V2; how to read those numbers is in Section 7.5. The pattern is realizable under Section 4.7. It does not force a choice between evolvable agents and auditable ones.

Supplementary information. Electronic Supplementary Material 1 (ESM 1) contains the protocols and anonymized round tables for the single-domain probe and the explicit-injection probe reported in Section 7.4. Structural-check enumerations (Tables 6–7) and the five-model run ledger (Table 8) appear in the manuscript and are not reproduced in ESM 1.

Statements and Declarations

Funding

The author did not receive support from any organization for the submitted work.

Competing interests

The author has no competing interests to declare that are relevant to the content of this article.

Ethics approval

Not applicable. This article reports an architecture pattern and a single-case study of a software system. It did not involve human participants or animals.

Consent to participate

Not applicable.

Consent for publication

Not applicable.

Data availability

The reference case is a single-tenant development/pilot deployment. Tenant-identifying configuration and production data cannot be shared. Protocols and anonymized round tables for the single-domain probe and the explicit-injection probe (Section 7.4) are provided as Electronic Supplementary Material 1. Structural-check enumerations (Tables 6–7) and the five-model run ledger (Table 8) appear in the manuscript. Architecture decision records cited in Section 5 are internal to the reference deployment; the paper reports their dates, options, decisions, and rejected alternatives.

Materials availability

Not applicable.

Code availability

The FIA Workbench implementation is not publicly released. The validation harness protocol (perturbation rounds, A/B isolation checks, and structural enumerations) is described in Section 7. Anonymized round tables for the single-domain and explicit-injection probes are in Electronic Supplementary Material 1, as stated under Data availability.

Author contributions

Yisen Xi is the sole author and is responsible for the study conception and design, the case analysis, the validation, and the writing of the manuscript.

References

Abdelnabi S, Gomaa A, Bagdasarian E, et al (2025) Firewalls to secure dynamic LLM agentic networks. arXiv preprint arXiv:250201822 https://doi.org/10.48550/arXiv.2502.01822, URL https://arxiv.org/abs/2502.01822, preprint

Anthropic (2024) Model context protocol specification. https://modelcontextprotocol.io, accessed: 2026-08-26

Bell DE, LaPadula LJ (1976) Secure computer system: Unified exposition and Multics interpretation. Tech. Rep. MTR-2997, MITRE Corporation, URL https://csrc.nist.gov/files/pubs/conference/1998/10/08/proceedings-of-the-21st-nissc-1998/final/docs/early-cs-papers/bell76.pdf, dTIC ADA023588; also ESD-TR-75-306

Buschmann F, Meunier R, Rohnert H, et al (1996) Pattern-Oriented Software Architecture: A System of Patterns, vol 1. Wiley, Chichester, URL https://www.wiley.com/en-us/Pattern+Oriented+Software+Architecture%2C+Volume+1%2C+A+System+of+Patterns-p-9780471958697

Chahine K (2026) Separating intent from execution: A defense-in-depth security architecture for LLM-based multi-agent systems. Expert Systems with Applications 332:133781. https://doi.org/10.1016/j.eswa.2026.133781, article 133781; assigned to volume 332 (2027)

Costa M, Köpf B, Kolluri A, et al (2025) Securing AI agents with information-flow control. arXiv preprint arXiv:250523643 https://doi.org/10.48550/arXiv.2505.23643, URL https://arxiv.org/abs/2505.23643, fides; preprint

Debenedetti E, Shumailov I, Fan T, et al (2025) Defeating prompt injections by design. arXiv preprint arXiv:250318813 https://doi.org/10.48550/arXiv.2503.18813, URL https://arxiv.org/abs/2503.18813, caMeL; preprint

Denning DE (1976) A lattice model of secure information flow. Communications of the ACM 19(5):236–243. https://doi.org/10.1145/360051.360056

Desai P, Tang L, Meng Y, et al (2026) SafeGPT: Preventing data leakage and unethical outputs in enterprise LLM use. arXiv preprint arXiv:260106366 https://doi.org/10.48550/arXiv.2601.06366, URL https://arxiv.org/abs/2601.06366, preprint

Fan W, Nie X, Dai Z (2026) Harness-MU: A safe, governed, and effective harness for multi-user LLM agents. arXiv preprint arXiv:260621856 https://doi.org/10.48550/arXiv.2606.21856, URL https://arxiv.org/abs/2606.21856, preprint

Gao D, Li Z, Pan X, et al (2024) AgentScope: A flexible yet robust multi-agent platform. arXiv preprint arXiv:240214034 https://doi.org/10.48550/arXiv.2402.14034, URL https://arxiv.org/abs/2402.14034, preprint

Jin B, Jiao Q, Tong X (2026) Agentao: A governed local-first runtime for tool-using LLM agents. arXiv preprint arXiv:260813574 https://doi.org/10.48550/arXiv.2608.13574, URL https://arxiv.org/abs/2608.13574, preprint

Malik A (2026) Progressive crystallization: Turning agent exploration into deterministic, lower-cost workflows in production. arXiv preprint arXiv:260707052 https://doi.org/10.48550/arXiv.2607.07052, URL https://arxiv.org/abs/2607.07052, preprint

Nygard M (2011) Documenting architecture decisions. https://www.cognitect.com/blog/2011/11/15/documenting-architecture-decisions, accessed: 2026-08-26

OASIS (2013) eXtensible Access Control Markup Language (XACML) version 3.0. Tech. rep., OASIS, URL http://docs.oasis-open.org/xacml/3.0/xacml-3.0-core-spec-os-en.html, oASIS Standard, 22 January 2013; cf. RFC 7061

OpenAI (2026) Function calling guide. https://platform.openai.com/docs/guides/function-calling, accessed: 2026-08-26

Park JS, O'Brien JC, Cai CJ, et al (2023) Generative agents: Interactive simulacra of human behavior. In: Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology (UIST '23), pp 1–22, https://doi.org/10.1145/3586183.3606763, URL https://doi.org/10.1145/3586183.3606763

Parnas DL (1972) On the criteria to be used in decomposing systems into modules. Communications of the ACM 15(12):1053–1058. https://doi.org/10.1145/361598.361623

Rahman MH, Kim J (2026) Proof-of-execution memory: Defending LLM agents against forged-reasoning attacks by verifying what actually happened. arXiv preprint arXiv:260816032 https://doi.org/10.48550/arXiv.2608.16032, URL https://arxiv.org/abs/2608.16032, poEM; preprint

Runeson P, Höst M (2009) Guidelines for conducting and reporting case study research in software engineering. Empirical Software Engineering 14(2):131–164. https://doi.org/10.1007/s10664-008-9102-8, URL https://doi.org/10.1007/s10664-008-9102-8

Schick T, Dwivedi-Yu J, Dessì R, et al (2023) Toolformer: Language models can teach themselves to use tools. In: Advances in Neural Information Processing Systems, pp 68539–68551, https://doi.org/10.52202/075280-2997, arXiv:2302.04761

Shah R, Feuillade-Montixi Q, Pour S, et al (2023) Scalable and transferable black-box jailbreaks for language models via persona modulation. arXiv preprint arXiv:231103348 https://doi.org/10.48550/arXiv.2311.03348, URL https://arxiv.org/abs/2311.03348, preprint

Shaikh A, Virkki J (2026) Persona poison in LLM-controlled robots: Empirical characterization of persona-conditioned decision errors and a separation-based mitigation. Research Square preprint https://doi.org/10.21203/rs.3.rs-10113099/v1, URL https://doi.org/10.21203/rs.3.rs-10113099/v1, preprint, posted 2026-06-23; not peer-reviewed

Shi T, He J, Wang Z, et al (2025) Progent: Securing AI agents with privilege control. arXiv preprint arXiv:250411703 https://doi.org/10.48550/arXiv.2504.11703, URL https://arxiv.org/abs/2504.11703, preprint

Shi T, Mo Y, Liu Y, et al (2026) Organizational control layer: Governance infrastructure at the execution boundary of LLM agent systems. arXiv preprint arXiv:260604306 https://doi.org/10.48550/arXiv.2606.04306, URL https://arxiv.org/abs/2606.04306, preprint

Wang H, Poskitt CM, Sun J (2026a) AgentSpec: Customizable runtime enforcement for safe and reliable LLM agents. In: Proceedings of the 48th IEEE/ACM International Conference on Software Engineering (ICSE), pp 1–12, https://doi.org/10.1145/3744916.3764546, arXiv:2503.18666

Wang L, Ma C, Feng X, et al (2024) A survey on large language model based autonomous agents. Frontiers of Computer Science 18(6):186345. https://doi.org/10.1007/s11704-024-40231-1, URL https://doi.org/10.1007/s11704-024-40231-1

Wang Z, Gu B, Yu H, et al (2026b) When agents see humans as the outgroup: Belief-dependent bias in LLM-powered agents. arXiv preprint arXiv:260100240 https://doi.org/10.48550/arXiv.2601.00240, URL https://arxiv.org/abs/2601.00240, preprint; introduces Belief Poisoning Attacks (BPA), including profile poisoning (BPA-PP)

Wu Q, Bansal G, Zhang J, et al (2024) AutoGen: Enabling next-gen LLM applications via multi-agent conversation. In: Conference on Language Modeling (COLM), https://doi.org/10.48550/arXiv.2308.08155, URL https://colmweb.org/2024/AcceptedPapers.html, arXiv:2308.08155

Yao S, Zhao J, Yu D, et al (2023) ReAct: Synergizing reasoning and acting in language models. In: International Conference on Learning Representations (ICLR), https://doi.org/10.48550/arXiv.2210.03629, URL https://iclr.cc/virtual/2023/poster/11003, arXiv:2210.03629

Yin RK (2018) Case Study Research and Applications: Design and Methods, 6th edn. Sage, Thousand Oaks, CA