# The Agentic Enterprise Capability Framework (AECF): A Governance-First Architecture for Scalable AI Agent Deployments

KHAMMAL Adil[1], HAMZANE Ibrahim[2], MARZAK Abdelaziz[3],
Hajar Taji[4], Elmostafa Erraitab[5], Mohamed EL WAFIQ[6]
Laboratory of Information Technology and Modeling LTIM, Hassan II University,
Ben M'sik Faculty of Sciences Casablanca, Morocco[1,3]
LAPSSII, Cadi Ayyad University of Marrakech, Morocco[2]
Mechanical, Material and thermal Laboratory/ National School of Mines in Rabat (ENIM) Rabat, Morocco[4]
Applied Mathematics and Statistics Department, Hassan II University, Km8 Route d'EL Jadida, Casablanca, 5366, Morocco[5]
Laboratoire LCPM, Faculté des Sciences Ben M'sik. Université Hassan II de Casablanca, Marco[6]

***Abstract*—Enterprise AI adoption has reached a structural inflection point: while a majority of organizations have deployed generative AI, few have established mature governance models for autonomous agents. This disparity reflects a fundamental architectural gap: Multi-Agent Systems, Enterprise Architecture, AI agent deployment, and software architecture have approached agent coordination from separate disciplinary perspectives, with no single framework integrating persistent memory, semantic interoperability, orchestration, human oversight, and normative enforcement. Following Design Science Research, this article develops the Agentic Enterprise Capability Framework (AECF), a five-layer architecture structured around Context Persistence (CPL), Semantic Interoperability (SIL), Hybrid Orchestration (HOL), Human Governance Interface (HGI), and Governance Envelope (GEL). The framework introduces the co-evolution constraint: technical capability layers cannot mature independently of governance capacity. This constraint is operationalized through Context-Enriched Pre-Execution Validation (CEPEV), which grounds compliance checks in operational memory. Five architectural propositions formalize inter-layer dependencies (P1), scalability boundaries (P2), governance effectiveness (P3), performance accumulation (P4), and a governance scaling law (P5). The study contributes an integrated architectural model, propositional formalization, and validation agenda for governed enterprise AI agent deployments.**

***Keywords*—Enterprise AI agents; agentic architecture; design science research; AI governance; multi-agent systems; policy-as-code**

## I. Introduction

### A. Motivation and Problem Context

Enterprise AI adoption has reached a critical juncture. While approximately 80% of organizations have deployed generative AI capabilities, only 21% have established mature governance models for autonomous agents [1], [2]. This disparity is not a temporary adoption lag but a structural mismatch between technological velocity and governance frameworks designed for deterministic, human-driven systems. The consequences are observable in enterprise deployments: agentic initiatives consistently stall not due to model limitations but due to governance architectures unable to manage autonomous decision-making, information systems lacking integration surfaces, and organizational uncertainty about transformation roles.

AI agent systems fundamentally differ from conventional AI deployments. They constitute coordinated ecosystems of autonomous actors that orchestrate workflows, invoke enterprise APIs, and make decisions across organizational boundaries. The governance architectures designed for single AI models or deterministic IT systems are structurally inadequate for this new paradigm. A sustainable enterprise agentic infrastructure requires five integrated capabilities: persistent shared memory, semantic interoperability among heterogeneous agents, task-sensitive orchestration, embedded human supervision, and policy enforcement throughout the execution lifecycle. This article addresses this architectural gap by defining the structural conditions under which agentic capability can be integrated, coordinated, and governed as enterprise infrastructure.

### B. Architectural Gap and Research Streams

Four research streams each capture complementary facets of the architectural problem, yet each remains bounded by its disciplinary perspective:

*1) Multi-Agent Systems (MAS):* has established coordination theory [3], [4] and conceptualized multi-agent architectures as organizational constructs [5]. However, MAS research has not theorized the layered architectural conditions—persistent memory, human governance interfaces, enabling governance envelopes—that enterprise-scale deployment requires. The design trade-offs inherent in MAS—such as the tension between agent autonomy and system predictability, or between distributed decision-making and centralized coordination—are well-documented at the theoretical level, but their implications for enterprise-scale architectural choices (e.g., where to place control points, how to bound communication, when to enforce human oversight) remain underexplored.

*2) Enterprise Architecture (EA):* frameworks [6], [7] provide governance logic designed for deterministic, human-

driven systems. This logic lacks the reactivity to accommodate autonomous agents' behavioral velocity [8], [9]. EA's periodic revision cycles and static artifacts are structurally mismatched with probabilistic AI agent architectures. The key trade-off here is between the stability required for enterprise governance and the adaptability required for AI agents—a tension that EA frameworks have not yet resolved.

*3) Emerging AI agent literature:* demonstrates technical viability and operational benefits—execution speed improvements of 3–10x, 60–80% MTTR reductions, and auditable semantic contracts [10]–[12]. However, this work focuses on technical coordination and protocol design, rarely examining architectural and governance conditions for enterprise-scale deployment [2]. The trade-off between technical performance and governance maturity is a recurring theme: high-performance orchestration can be achieved without governance, but sustainable enterprise deployment requires both.

*4) Software architecture:* has established proven design patterns [13], [14], [20], [21] without systematic transposition to enterprise AI agent coordination. The AECF addresses this fragmentation through integrated architectural design, leveraging patterns such as Layered Architecture, Strategy, Mediator, and Observer to address specific coordination and governance challenges.

### C. Research Contribution and Structure

This article introduces the Agentic Enterprise Capability Framework (AECF), a governance-aware, sociotechnical architectural framework developed through Design Science Research [15], [16]. The AECF systematically transposes proven software design patterns to enterprise AI agent coordination, organizing them across five interdependent layers. Its central proposition is:

A sustainable, governed enterprise agentic infrastructure requires a layered architecture formalizing the co-evolution of five capabilities: persistent memory, semantic interoperability, hybrid orchestration, human governance, and enabling governance.

The study is organized as follows: Section II establishes the theoretical foundations; Section III presents the Design Science Research methodology; Section IV formalizes the AECF and its layers, with Fig. 1, 2, and 3 illustrating the architecture; Section V derives architectural propositions; Section VI discusses contributions and implications, including Tables I, II, and III; Section VII concludes with the research agenda.

## II. Theoretical Foundations

### A. Multi-Agent Systems and Coordination Theory

Giorgini et al. [5] provide a foundational theoretical premise: multi-agent systems are fundamentally organizational constructs structured through intentional and social dependencies among agents. System quality is shaped by architectural decisions defining roles, interdependencies, responsibilities, and coordination mechanisms—not solely by technical components. This theoretical framing foregrounds the tension between predictability and adaptability that any agentic architecture must address. In particular, the trade-off between agent autonomy (which enables adaptability) and system predictability (which enables governance) is central to enterprise deployment. MAS research has explored coordination protocols and organizational structures, but it has not systematically addressed how these trade-offs scale to enterprise environments with thousands of agents and heterogeneous governance requirements.

Recent advances corroborate this view while reorienting toward enterprise AI ecosystems. Tools such as AutoGen, LangGraph, and CrewAI, together with production-scale research [10], [12], demonstrate technical feasibility of multi-agent coordination. These studies concentrate primarily on orchestration, interaction protocols, and distributed execution, leaving the organizational and architectural conditions for enterprise integration unformalized. Venkiteela [11] specifically identifies the absence of enterprise governance models capable of regulating large-scale agentic deployment.

### B. Enterprise Architecture Evolution and AI

Enterprise Architecture has evolved from documentary frameworks [6], [7] toward strategic governance roles [8], although practice diverges from formal prescriptions [9]. EA now faces a paradigmatic mismatch with AI agent ecosystems. Designed to govern deterministic systems whose behavior is specified before deployment, EA is structurally inadequate for probabilistic, autonomous agents with emergent behaviors. Periodic revision cycles and static artifacts cannot govern the behavioral velocity of agentic systems. The trade-off between EA's emphasis on stability and control and AI agents' need for adaptability and autonomy is a fundamental design tension that the AECF addresses through enabling governance rather than constraining control.

Chen and Zhao [17] empirically confirm this tension: approaches coupling AI with multi-agent systems outperform traditional EA methods in risk management but exhibit scalability constraints absent dedicated conceptual frameworks. Kotusev and Kurnia [18] document EA artifacts taking effective governance roles that existing models have not formalized. Enterprise Architecture has established how to govern conventional architectures, but it has yet to theorize the evolution required to absorb autonomous AI agents' probabilistic and emergent dynamics.

### C. Emerging Enterprise AI Agent Architectures

Technical viability of enterprise AI agent architectures is well documented. Venkiteela [11] reports execution speed improvements of 3–10x and 60–80% reductions in Mean Time to Resolution (MTTR). Perikala [10] shows that Model Context Protocol (MCP)-based architectures establish auditable semantic execution contracts. Adimulam et al. [12] consolidate MCP and Agent-to-Agent (A2A) protocols into unified orchestration frameworks.

Extending beyond technical performance, Qian et al. [19] demonstrate across 180 experimental configurations that architecture–task alignment matters more than model sophistication: improvements of up to 80.8% on decomposable tasks, with architecture–task mismatch degrading performance by up to 70% on sequential tasks. Acharya [2] documents systemic governance risk: 21% of enterprises have mature governance

models, with 40% of agentic initiatives projected to fail by 2027 absent suitable architectural arrangements. The trade-off between model capability and architectural design is a key insight: investing in better models without corresponding architectural maturity yields diminishing returns.

### D. Software Architecture Patterns for Complex Systems

The AECF's design logic draws on software engineering patterns validated across three decades: Layered Architecture, crosscutting concerns, strategy-based composition, interface contracts, and protocol adapters [13], [14], [20], [21]. The framework's originality lies in systematic transposition of these patterns to autonomous AI agent coordination. Existing architectures either forge ad hoc mechanisms without structural grounding [11], [12] or apply isolated patterns at the agent level without theorizing their integration within a global, multi-layer composition [22].

## III. Research Methodology

This research adopts Design Science Research (DSR) as its epistemological foundation [15], positioning the AECF as a theoretical artifact whose scientific legitimacy derives from addressing a rigorously identified problem and generating design utility. Following Peffers et al. [16], the research proceeds from problem identification (the architectural gap established in Section II) through design and development (the five-layer AECF formalized in Section IV) to propositional evaluation and communication.

The framework's core design principle is inter-layer architectural traceability, ensuring co-evolution of memory, interoperability, orchestration, and governance is structurally enforced. Drawing on Gregor and Hevner's DSR contribution taxonomy [23], the AECF constitutes an Improvement contribution: it extends and integrates existing theoretical bodies—Multi-Agent Systems, Enterprise Architecture, coordination theory, and software architecture—into a new architectural framework that none resolves independently.

At this stage, the framework's propositions are theoretically derived. This separation between artifact design and empirical evaluation is a deliberate DSR methodological feature, not a limitation. Validation through multiple-case study design [24] constitutes the complementary empirical research agenda. To illustrate practical applicability, Section VI presents a worked deployment scenario for a financial services use case, demonstrating how the AECF layers would be instantiated in practice.

## IV. Agentic Enterprise Capability Framework (AECF)

### A. Architectural Foundations and Design Principles

The AECF rests on four design principles adapted from software architecture to autonomous AI agent coordination.

*1) Layered decomposition:* Enterprise agentic capability decomposes into distinct, interdependent layers addressing specific architectural concerns. This principle instantiates the Layered Architecture pattern [13] and builds on Giorgini et al. [5]: multi-agent architectures are organizational constructs governed by social dependencies, not computational specifications alone.

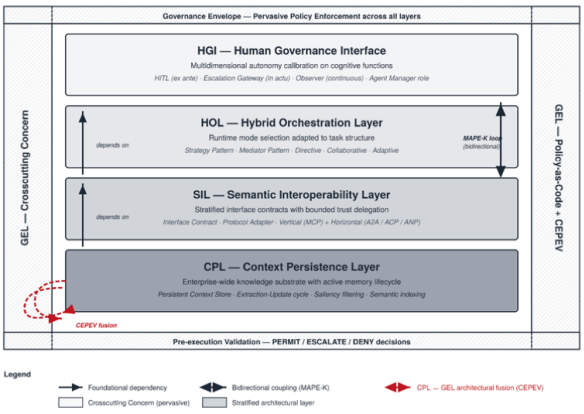


Fig. 1. Five-layer architecture of the AECF: Stratified layers (CPL, SIL, HOL, HGI) governed by a pervasive Governance Envelope Layer (GEL) as a crosscutting concern.

*2) Architectural co-evolution under bounded orthogonality:* In classical software design, separation of concerns [25], [26] holds that well-designed modules should evolve independently. In autonomous agent systems, certain concerns cease to be orthogonal: memory infrastructure value depends on governance capable of consulting it; governance precision depends on available memory. The AECF distinguishes layer-level modifiability (preserved per classical architecture) from system-level co-evolution (a constraint specific to autonomous agent systems). Empirical evidence supports this distinction: technically sophisticated orchestration architectures without parallel memory and governance maturation produce documented capability ceilings in enterprise deployment [11], [17].

*3) Scalability-aware design:* Distributed systems research establishes that coordination does not scale linearly with actor count due to increasing communication and control costs [27], [28]. In AI agent systems, empirical evidence documents performance degradation of up to 70% on sequential tasks when coordination overhead fragments reasoning budget [19], and disproportionate latency increases beyond 50 concurrent agents [11]. The AECF incorporates this constraint: each layer addresses specific scalability dimensions—reducing context reinitialization, bounding inter-agent communication, aligning orchestration with task structure, and limiting agent proliferation to supervision capacity.

*4) Sociotechnical integration through crosscutting pervasion:* Non-functional requirements that cannot be condensed into a single module—governance, auditability, testability—must be embedded directly in system architecture rather than relegated to separate checkpoints [13], [21]. In enterprise AI agent contexts, this extends beyond aspect-oriented architecture: human supervision must be integrated as a fundamental design element, co-constructed with the technical substrate it directs [47]. The AECF formalizes these requirements through patterns cutting across layers, producing a sociotechnical architecture where compliance, audit traceability, and human accountability are mechanically guaranteed.

Fig. 1 illustrates the five-layer architecture, showing the stratified layers (CPL, SIL, HOL, HGI) and the GEL as a crosscutting governance concern that pervades all layers. This

figure should be read in conjunction with the layer descriptions in Section IV-B.

### B. The Five Layers of the AECF

*1) Layer 1: Context Persistence Layer (CPL):* The CPL provides structured, persistent, semantically indexed knowledge maintaining continuity across agent invocations, reducing context loss, reinitialization costs, and execution inconsistency. Unlike passive Event Sourcing [29], [30], the CPL operates through an active Extraction–Update cycle filtering, restructuring, and deprecating accumulated information. Unlike per-invocation RAG [31], the CPL provides a persistent architectural substrate serving operational execution, semantic interoperability, and governance evaluation.

The CPL implements a Persistent Context Store through continuous Extraction–Update cycles: incoming events undergo salience-based extraction prioritizing decision-relevant facts and patterns, while existing entries are maintained for coherence, contradiction resolution, and obsolescence deprecation. The architecture spans: temporal persistence across sessions; semantic indexing supporting vector-similarity retrieval and relational inference; and cross-agent accessibility enabling collective organizational knowledge beyond individual agent perimeters.

Without a governed CPL substrate, orchestrated coordination degenerates into stateless directive execution, semantic contracts lose contextual grounding, and governance validation loses its evidential basis. Upper layers are defined by their capacity to consult the CPL substrate.

*2) Layer 2: Semantic Interoperability Layer (SIL):* The SIL defines protocols, contracts, and interface specifications enabling AI agents to exchange information, delegate tasks, and coordinate across heterogeneous enterprise environments. It distinguishes technical compatibility from semantic interoperability, where agents share not only data but contextual meaning and operational intent [32].

Adapting Interface Contract and Protocol Adapter patterns [14], the SIL stabilizes agent interactions despite probabilistic, heterogeneous behaviors. Drawing on formal semantics [33], the SIL distinguishes syntactic (data exchange), structural (typed schema conformance), and semantic (shared interpretation) interoperability. Interface contracts operate at the first two levels; semantic interoperability is carried by complementary artifacts (business ontologies, capability descriptions, pre/post-condition constraints). As documented in recent agent interoperability surveys [34], these contracts are essential for reliable cross-agent coordination.

The SIL operates along two axes:

- Vertical axis: governs agent access to enterprise tools and data sources, requiring: structural determinism of invocations, auditability of calls and results, explicit versioning of interface contracts, and schema validation at invocation time. The Model Context Protocol provides the most convergent industrial instantiation, establishing client–server JSON-RPC interfaces for deterministic, auditable tool invocation [10]. The SIL is not reducible to any specific protocol.
- Horizontal axis: governs peer-to-peer coordination among agents, requiring capability discovery, task negotiation, and bounded session management, instantiable through protocols such as those examined in recent agent interoperability literature [34].

Trust delegation is governed through propagation-control mechanisms binding authorization to scope, time, and delegation depth. Each invocation carries an enriched authorization token specifying effective scope, residual lifetime, and delegation chain position, validated at each interface crossing. Threshold violations trigger cascading revocation, preventing unauthorized access propagation across the enterprise agent network [10], [50].

*3) Layer 3: Hybrid Orchestration Layer (HOL):* The HOL addresses the fundamental design tension arising from single orchestration mode inadequacy across diverse enterprise workflows. It resolves this through composite architecture selecting optimal coordination mode according to process criticality, regulatory constraints, and task complexity.

Adapting the Strategy Pattern [14] to self-adaptive architectures [35], [36], the HOL extends classical pattern through: 1) classifier mechanism evaluating task characteristics (decomposability, regulatory sensitivity, uncertainty, coordination cost) before delegation; 2) relaxed mutual exclusivity at ecosystem level (multiple sub-mechanisms operating simultaneously for distinct workflow segments) while maintaining exclusivity at workflow segment level.

The HOL operationalizes three sub-mechanisms:

- Directive orchestration (command pattern): centralized workflow engine issues deterministic instructions to specialized agents executing regulated processes with explicit specifications, guaranteeing traceability and auditability.
- Collaborative orchestration (peer mediation pattern): agents negotiate allocation and share results within bounded clusters to limit exponential communication overhead [4].
- Adaptive orchestration (dynamic dispatch pattern): dynamic agent and resource reallocation in real time based on execution context, with self-correcting feedback loop triggering re-execution by alternative agents when quality thresholds are unmet.

The classifier decision function maps task characteristics to sub-mechanisms. High regulatory sensitivity favors directive orchestration. High decomposability with moderate uncertainty favors collaborative orchestration. High coordination cost or strong uncertainty favors adaptive orchestration. Thresholds are configurable by process class and functional domain.

At the topology level, the HOL implements a Mediator Pattern [14] as coordination-control mechanism. By centralizing communication channels, the mediator reduces direct dependencies and limits coupling complexity [37], [38]. Without mediation, the agent ecosystem would tend toward unbounded mesh topology with quadratic complexity. The Mediator maintains explicit exchange graphs, detects non-terminating invocation loops, and provides consistent instrumentation for coordination metrics.

Beyond task distribution and topological governance, the HOL performs semantic integration of results through: 1) cross-validation enforcing coherence checking against contributions; 2) conflict resolution driven by configurable priority rules; 3) final synthesis generating consolidated output with explicit contribution and arbitration traces.

*4) Layer 4: Human Governance Interface (HGI):* The HGI addresses the human and structural pillar of agentic capability—systematically absent from technically oriented frameworks [11], [17]. Drawing on Sheridan [39] and Parasuraman, Sheridan, and Wickens [40], the HGI calibrates autonomy independently across four cognitive functions: acquisition, analysis, decision selection, and action implementation. It moves beyond binary automation views by decomposing human supervision into granular, independent calibration across these functions.

The HGI implements three complementary patterns covering the agent action lifecycle:

- Human-in-the-Loop pattern operates pre-action: structured control points requiring human approval before irreversible actions.
- Escalation Gateway pattern operates during action: configurable thresholds (risk level, confidence scores, regulatory sensitivity, deviation from expected execution parameters) triggering automatic escalation to human authority under unforeseen conditions.
- Observer pattern [14] operates continuously: non-intrusive monitoring constituting the information base for Escalation Gateway threshold detection.

The HGI deploys a multidimensional autonomy spectrum independently calibrating four cognitive functions according to risk level, regulatory constraints, and agent reliability. This supports three profiles: *supervised* (autonomy restricted to acquisition and analysis for highly sensitive processes), *delegated* (full automation framed by Observer and Escalation Gateway for moderate-risk environments), and *hybrid* (asymmetric, context-dependent configuration for variable criticality processes).

The Agent Manager construct constitutes a technical governance role with four architectural responsibilities:

- Definition of escalation thresholds: configuring precise conditions for Escalation Gateway transfer to human authority, aligned with business risk profiles.
- Calibration of autonomy boundaries: positioning and adjusting combination of automation levels across cognitive functions based on demonstrated agent reliability.
- Monitoring agent performance: exploiting Observer pattern data to detect behavioral drift, performance regressions, and execution anomalies before incident occurrence.
- Maintaining alignment between demonstrated and granted autonomy: ensuring operational autonomy never exceeds demonstrated competence.

The HGI incorporates temporal regulation through Progressive Rollout pattern: escalation thresholds progressively relaxed, agent-creation rights gradually decentralized, cognitive autonomy adjusted according to documented performance, enshrining the principle that granted autonomy must never exceed demonstrated autonomy.

*5) Layer 5: Governance Envelope Layer (GEL):* The GEL defines the architectural boundary within which AI agent systems can be governed, audited, and aligned with enterprise constraints. Unlike Layers 1–4, the GEL operates as a cross-cutting governance concern [13], drawing on Aspect-Oriented Programming [46] and policy-driven architectures [44], [45] to enforce governance rules across the agentic architecture. Rather than constraining agentic reasoning intrinsically, the GEL anchors control in the predictability of evaluation moments through pre-commit interception points, locking down interaction surfaces affecting enterprise state at three levels: individual agent actions; enterprise tool invocations; and inter-agent communications.

Given agent behavioral velocity, ex-post governance is structurally insufficient: control must move from terminal checkpoints to embedded architectural constraints shaping behavior throughout execution.

The GEL deploys a Policy-as-Code engine encoding rules as declarative, versioned, machine-executable policies. Each policy specifies conditions under which agent actions are permitted, restricted, escalated, or audited, enabling automated validation, testing, deployment, and rollback with application code engineering rigor. This architecture synthesizes three complementary advances:

- Unified infrastructure [41]: multilayer architecture integrating risk classification, compliance validation, and Zero Trust control.
- Automated translation [42]: Policy-as-Prompt approach converting unstructured regulatory corpora into executable guardrails through structured policy trees.
- Normative interoperability [43]: machine-readable Policy Cards ensuring contractual mapping to external frameworks (NIST AI RMF, ISO/IEC 42001, EU AI Act [52]).

The GEL operationalizes governance through pre-execution validation: before action commitment, the engine confronts the proposal with active configuration (authorization perimeter, risk classification, compliance, audit requirements). Architectural specificity lies in contextual enrichment: unlike classical Policy-as-Code, the GEL weights decisions with behavioral traces accumulated in the CPL. This memory–governance fusion generates three distinctive properties:

- Dynamic tolerance modulation: identical configuration yields different outcomes based on compliance history.
- Holistic evaluation: fragmented invocation history is re-aggregated at each governance decision.
- Abolition of decoupling: traditional separation between state storage (memory) and control (governance) gives way to structural integration.

Pre-execution architecture induces predictable latency (1–2 seconds in MCP production [10]), justified by deterministic execution guarantees, auditability, and fault isolation. This embodies enabling governance: declarative boundaries, automated invocation validation, default authorization within perimeter, and progressive autonomy mechanism calibrating constraints according to demonstrated reliability.

Dynamic calibration operates through: 1) continuous reliability indexing from CPL history; 2) automated threshold modulation limiting friction for high-performing agents, tightening constraints on erratic profiles; 3) asynchronous review by Agent Manager incorporating qualitative heuristics.

### C. Inter-Layer Relationships and Architectural Tensions

The five architectural layers constitute an interdependent system whose collective behavior determines whether AI agent deployments achieve governed, sustainable value. This section formalizes five structural relationships.

- R1: Foundational dependency chain. The HOL presupposes the SIL, which presupposes the CPL [13]. Investment in orchestration without proportional investment in semantic interoperability and context persistence yields diminishing marginal effectiveness.
- R2: Bidirectional HOL ↔ HGI coupling. The HOL provides orchestration infrastructure; reciprocally, the HGI provides calibration logic determining its operation. This instantiates the MAPE-K model of self-adaptive architectures [35]: Monitor (Observer pattern), Analyze (Agent Manager interpreting data), Plan (calibrations of modes, thresholds, boundaries), Execute (HOL applying configurations), Knowledge (operational memory maintaining cross-cutting history).
- R3: Articulation on inter-agent communication. Three mechanisms operate in series with non-redundant functions: HOL Mediator Pattern governs topological structure; HGI Observer provides observation; GEL interception provides regulation. Each operates at different abstraction levels—structural, epistemic, regulatory.
- R4: Regulatory pervasion of the GEL. The GEL applies to the CPL for trace integrity, to the SIL for contract and token validation, to the HOL for pre-execution orchestration validation, and to the HGI for escalation decision execution.
- R5: CPL ↔ GEL architectural fusion. Governance validation can be contextually enriched only by consulting operational memory; memory generates regulatory value only through GEL mobilization at decision time. The classical decoupling between memory and governance is replaced by fusion operationalized through Context-Enriched Pre-Execution Validation (CEPEV).

Fig. 2 illustrates these five structural relationships, showing how the layers interact through dependency chains, bidirectional coupling, and crosscutting pervasion. This figure complements Fig. 1 by showing the dynamic interactions rather than the static layer structure.

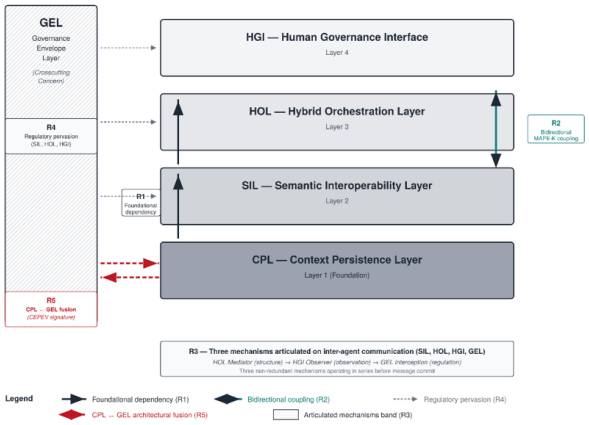


Fig. 2. Five structural relationships among AECF layers: Foundational dependency chain (R1), bidirectional MAPE-K coupling (R2), articulated mechanisms on inter-agent communication (R3), regulatory pervasion of the GEL (R4), and CPL ↔ GEL architectural fusion (R5).

*1) Architectural tensions:* The AECF exposes three inherent tensions as configurable, monitorable design parameters:

*a) Tension 1: Operational autonomy versus governance overhead.:* The balance between execution fluidity and control cost. Overly constraining governance introduces latency and friction; unbounded autonomy produces behaviors exceeding authorized scope and reducing auditability. The AECF favors enabling governance: default autonomy within declared boundaries, dynamically conditioned on risk, action criticality, compliance history, and context quality, with degrees of freedom to calibrate per process according to risk profile.

*b) Tension 2: Agent specialization versus architectural coherence.:* Local specialization efficiency versus global ecosystem coherence. Unregulated specialization can fragment architecture through duplicated capabilities, inconsistent formats, semantic drift, incompatible rules, and limited cross-cutting auditability. The AECF allows specialization at implementation while enforcing coherence at interface and governance levels through SIL contracts and GEL policies.

*c) Tension 3: Distributed agent creation versus calibration coherence at scale.:* Distributed creation fosters local ownership and innovation. As agent population expands, differences in autonomy trajectories, escalation thresholds, policies, and risk profiles accumulate, reducing governance comparability and auditability. The AECF responds with stratified calibration: HGI sets maximum autonomy boundaries at ecosystem and domain levels; GEL adjusts thresholds at workflow, action-class, and agent levels; CPL preserves demonstrated reliability traces; CEPEV verifies action compatibility with active policies and context.

### D. Comparative Positioning

The AECF occupies a distinct position in the emerging landscape of enterprise AI agent frameworks. Table I (shown below) compares the AECF with three reference architectures: Enterprise Agentic Architecture Framework (EAAF; [11]), Multi-Agent System approach to Enterprise Architecture (MAS-EA; [17]), and Unified Orchestration Framework [12].

TABLE I. COMPARATIVE POSITIONING OF ENTERPRISE AI AGENT FRAMEWORKS

| Architectural dimension | EAAF | MAS-EA | Unified Orch. | AECF |
|---|---|---|---|---|
| 1. Persistent Context Infrastructure | ◯ | ○ | ◯ | ● |
| 2. Semantic Interoperability | ● | ○ | ● | ● |
| 3. Hybrid Orchestration | ◯ | ○ | ● | ● |
| 4. Human Governance Interface | ○ | ○ | ○ | ● |
| 5. Agent Manager Role Construct | ○ | ○ | ○ | ● |
| 6. Configurable Autonomy Spectrum | ○ | ○ | ○ | ● |
| 7. Policy-as-Code Governance | ◯ | ○ | ◯ | ● |
| 8. Layered Architecture Pattern | ● | ◯ | ◯ | ● |
| 9. Crosscutting Concern Governance | ○ | ○ | ○ | ● |
| 10. Software Architecture Pattern Grounding | ○ | ○ | ◯ | ● |
| 11. Inter-Layer Coevolution Theorization | ○ | ○ | ○ | ● |
| 12. Regulatory Anchoring | ○ | ○ | ◯ | ● |

*Legend:* ● = formally specified; ◯ = partially addressed; ○ = absent.

Comparison dimensions characterize functional completeness, design rigor, governability, and scaling conditions.

Table I is discussed in detail in Section VI-A, where we analyze the comparative positioning and substantiate the study's claim to occupy a distinct architectural niche. The table shows that the AECF is the only framework addressing all twelve dimensions simultaneously, with its decisive contribution lying in linking technical infrastructure to human governance while theorizing their co-evolution.

Three observations emerge from Table I. First, the AECF is the only framework addressing all twelve dimensions simultaneously. Second, its decisive contribution lies in linking technical infrastructure to human governance while theorizing their co-evolution—dimensions systematically absent from technically oriented frameworks. Third, the AECF is positioned as architectural complement rather than alternative: existing frameworks provide technical components; the AECF provides architectural scaffolding determining whether these components generate sustainable enterprise value.

### *E. Technical Specification: Context-Enriched Pre-Execution Validation (CEPEV)*

The CPL–GEL fusion requires a formal mechanism bringing operational memory into governance evaluation at decision time. CEPEV sits at the convergence of three traditions without reducing to any. ABAC and Risk-Adaptive extensions [48] established dynamic attribute relevance but without architectural substrate for accumulation; the CPL provides that substrate. Event Sourcing [29], [30] demonstrated exhaustive execution history utility but remains retrospective. Trust Management [49] formalized reliability computation but remains limited to aggregate scores. CEPEV integrates dynamic attributes, structured historicity, and evidential calibration into a mechanism specifically designed for CPL $\leftrightarrow$ GEL coupling.

*1) Formal specification:* Before action commitment, the GEL retrieves a contextual snapshot from the CPL and injects it into policy evaluation alongside the proposed action:

$$\text{Decision} = \text{GEL.validate}(a, \pi, C(\text{agent}, t, \text{scope})) \quad (1)$$

where, $a$ is the proposed action, $\pi$ the policy configuration, and $C(\text{agent}, t, \text{scope})$ the contextual snapshot retrieved over time window $t$ within perimeter $scope$. Decision returns PERMIT, ESCALATE, or DENY.

The snapshot $C$ is a governance-ready memory artifact produced through the CPL's Extraction–Update cycle. The salience-scoring function $\sigma(e, \text{agent}, t, \text{scope})$ governs what operational history is extracted. For an event $e$ in the agent's history, salience is computed as:

$$\sigma(e, \text{agent}, t, \text{scope}) = w_1 \cdot r(e) + w_2 \cdot f(e) + w_3 \cdot c(e) + w_4 \cdot s(e) \quad (2)$$

where:

- $r(e)$ is the recency score (decaying exponentially with event age),
- $f(e)$ is the frequency score (number of similar events in the window),
- $c(e)$ is the criticality score (pre-defined by process class),
- $s(e)$ is the severity score (pre-defined by action type),
- $w_1, w_2, w_3, w_4$ are configurable weights.

Only events with $\sigma(e, \text{agent}, t, \text{scope}) > \theta$ are included in the snapshot, where $\theta$ is a configurable threshold. This salience function ensures that the snapshot remains bounded in size while preserving the most decision-relevant history.

Fig. 3 illustrates the CEPEV decision sequence, showing how the GEL retrieves the contextual snapshot from the CPL and evaluates it against the proposed action and active policies to return PERMIT, ESCALATE, or DENY. This figure should be read alongside the formal specification in Eq. (1).

Contextual enrichment introduces a structural vulnerability: compromised CPL history enables decision manipulation. CEPEV relies on CPL integrity protected through cryptographic event signatures, append-only memory structures, and governance-engine write validation. This vulnerability is absent from context-blind engines, which remain invulnerable precisely because they consult no memory—at the cost of inability to calibrate governance according to behavioral history.

CEPEV operationalizes progressive autonomy: agents with rich compliant context obtain PERMIT for actions triggering ESCALATE for agents with limited or deviant histories, without configuration alteration. Latency overhead is estimated at 1.5–3 seconds in production—a controlled extension of the 1–2 second baseline for context-blind pre-execution governance [10].

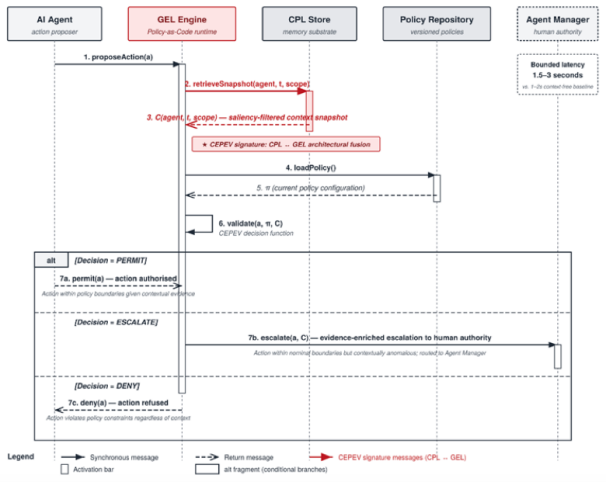


Fig. 3. CEPEV decision sequence returning PERMIT, ESCALATE, or DENY based on contextual evidence retrieved from CPL.

## V. Architectural Propositions

### A. Propositional Logic Rationale

Following DSR positioning [15], the AECF is formalized through architectural propositions rather than empirically testable hypotheses. A proposition establishes a theoretical relationship between constructs; a hypothesis presupposes stabilized, measurable indicators. The central constructs—CPL maturity, HGI capacity, governance complexity, CEPEV effectiveness, agentic scalability boundary—lack consolidated empirical metrics. Their operationalization is the object of the complementary empirical study.

Each proposition is supported by convergent empirical evidence from recent production-anchored publications. This provides initial theoretical support but not definitive validation. Broadening the empirical base through multiple-case design is necessary for testing transferability, validity conditions, and operational limits.

### B. Framework Propositions

*1) Proposition 1 Foundational dependency:* (P1) The effectiveness of the Hybrid Orchestration Layer (HOL) is strictly conditioned by the joint maturity of the Context Persistence Layer (CPL) and the Semantic Interoperability Layer (SIL). A deficiency in either layer produces orchestration pathologies of qualitatively distinct nature.

This formalizes the inherent tension between predictability and adaptability [5]. Absence of mature CPL produces *contextual amnesia*: orchestration executes without history, generating redundant executions and nullifying adaptation capacity. SIL maturity deficit induces *semantic fragmentation*, paralyzing inter-agent coordination. P1 argues operational memory and shared language are necessary structural prerequisites for HOL effectiveness.

*2) Proposition 2 Architectural scalability boundary:* (P2) Enterprise AI agent system scaling encounters an unavoidable architectural boundary when orchestration sophistication (HOL) is not accompanied by proportional human governance capacity (HGI) and mature memory infrastructure (CPL). Asymmetric layer evolution produces diminishing returns beyond a critical systemic complexity threshold.

This formalizes the bidirectional coupling between orchestration and supervision through a MAPE-K control loop. The HGI acts as co-moderating variable: its maturation conditions the effectiveness of other architectural blocks without constituting an independent mechanism. Empirically, this boundary manifests through coordination degradation, non-linear governance incident growth, and divergence between deployed technical capacity and organizational control capacity. The threshold of approximately 50 concurrent agents serves as a provisional empirical marker.

*3) Proposition 3: Pre-execution policy validation:* (P3) Architectures implementing pre-execution normative validation through declarative, versioned Policy-as-Code governance engines positively moderate the relationship between agent autonomy and value creation.

This moderation operates through executable authorization boundaries, enlarging productive autonomy space while restricting ungoverned action. It prevents uncontrolled behavioral drift while removing execution inertia imposed by manual auditing. The proposition formalizes transition from constraining governance (static rules with human verification) to enabling governance (automated declarative perimeters within which autonomy legitimately unfolds). CEPEV latency should not be interpreted solely as technical cost but as a trade-off among operational fluidity, compliance, and auditability.

*4) Proposition 4: Context persistence as performance differentiator:* (P4) AI agent systems systematically structuring operational data within specialized persistent memory (CPL) develop a cumulative performance advantage. This advantage, directly attributable to contextual substrate richness rather than model sophistication, constitutes endogenous cognitive capital that generic model-dependent systems cannot reproduce.

Unlike standardized generic AI capabilities, the CPL is inherently proprietary, hyper-contextualized, and incremental. Each interaction enriches it through the Extraction–Update cycle, forging a knowledge base widening an asymmetric efficiency gap over deployment. P4 should not be formulated as an automatically irreversible advantage: it depends on memory quality, freshness, integrity, and governance.

*5) Proposition 5: Governance Scaling Law:* (P5) Beyond a critical deployment density threshold, democratization of AI agent creation across non-technical teams generates non-linear growth in governance complexity. Policy conflicts, authorization ambiguities, and agent behaviors escaping supervision multiply at an accelerating rate. This creates a structural inflection point where a qualitative GEL evolution becomes architecturally unavoidable: from centralized rule enforcement to distributed compliance-by-design.

P5 models decentralization–scalability tension as a three-phase dynamic. Pre-threshold: distributed creation increases action capacity without disproportionate governance overhead. Threshold crossing: transferring creation capacity triggers qualitative rupture—governance complexity accelerates sharply. Post-threshold: GEL must shift from centralized control to embedded compliance-by-design. Evidence supports

TABLE II. AECF PROPOSITIONAL ARCHITECTURE

| Prop. | Core relationship | Primary empirical support | Layer(s) | Tension addressed | Operational mechanism |
|---|---|---|---|---|---|
| P1 | CPL + SIL → HOL effectiveness | Perikala 2025; Venkiteela 2026 | L1, L2, L3 | Predictability vs. Adaptability | Context retrieval through CEPEV pre-execution |
| P2 | HOL ↔ HGI → Scalability boundary | Qian et al. 2025; Venkiteela 2026 | L1, L3, L4 | Specialization vs. Coherence | MAPE-K control loop coupling |
| P3 | Enabling GEL → Value + Compliance | Jackson 2025; Kholkar et al. 2025 | L5 | Autonomy vs. Governance overhead | CEPEV declarative boundary enforcement |
| P4 | CPL investment → Performance advantage | Chhikara et al. 2025; Perikala 2025 | L1 | Strategic (architectural) | CEPEV evidence accumulation for differentiated decisions |
| P5 | Distributed creation → Non-linear governance complexity | Acharya 2026; Venkiteela 2026 | L4, L5 | Agent creation vs. Governance scalability | Embedded compliance-by-design |

this dynamic: 21% of organizations exhibit sufficient governance maturity; 40% of agentic initiatives projected to fail by 2027 [2].

Table II summarizes the five architectural propositions, showing the core relationship, empirical support, layers involved, tension addressed, and operational mechanism for each. This table is referenced in Section V-B and provides a consolidated view of the propositional architecture. It should be read alongside the detailed proposition descriptions in the text.

## VI. DISCUSSION

### A. Answering the Research Question

The AECF provides an architectural response to enterprise AI agent scaling by defining structural conditions for coordination, governance, and sustainability beyond isolated deployments. The transition is not achieved through model sophistication or agent volume but through layered architecture whose co-evolution generates durable value. The central contribution is a unified framework integrating operational memory, semantic interoperability, hybrid orchestration, human governance, and pre-execution normative validation—dimensions previously addressed in fragmented, discipline-specific ways.

The contribution mobilizes software architecture patterns as a structural bridge between two historically parallel bodies: MAS coordination theory and EA governance logic. It operationalizes Giorgini et al.'s [5] conception of multi-agent architectures as organizational constructs, translating this theoretical premise into a five-layer design model. It simultaneously extends EA theory [8], [9] by replacing static, artifact-centric paradigms with enabling governance [47], embodied in pre-execution Policy-as-Code validation.

### B. Architectural Contributions

*1) Original architectural constructs:* The AECF's distinctive contribution lies in coordinated integration of four interdependent constructs:

*a) The Context Persistence Layer (CPL):* elevates enterprise memory from retrieval device to proprietary knowledge infrastructure. Generating value from domain-specific operational accumulation, it forms cumulative cognitive capital—structured institutional memory as a competitive differentiator [51]. This contribution remains conditional on memory quality, freshness, integrity, and completeness.

*b) The Hybrid Orchestration Layer (HOL):* transcends classical multi-agent system dichotomies by theorizing orchestration as multimodal, pattern-grounded architectural mechanism. Its legitimacy rests on architecture–task alignment driving collaborative success more than model sophistication [19].

*c) The agent manager:* role formalizes the human governance pillar, systematically overlooked by technical frameworks. Analogous to site reliability engineering, it guarantees operational and normative integrity. Organic emergence of similar functions supports empirical relevance [11], [17].

*d) Context-Enriched Pre-Execution Validation (CEPEV):* materializes structural fusion between memory substrate (CPL) and normative envelope (GEL), moving beyond traditional Policy-as-Code engines by converting theoretical dependency into an execution mechanism based on proposed action, active policies, and agent behavioral context.

*2) Governance scaling law:* Proposition 5 advances both AI theory and scaling engineering by formalizing non-linear relationship between distributed agent generation and governance complexity. The AECF proposes an architectural hypothesis of non-linear complexity growth, empirically verifiable through metrics: agent density, policy conflicts, authorization ambiguities, escalation rates, coordination costs, and compliance incidents.

Available evidence supports this model: massive generative AI deployment collides with governance deficit, driving critical agentic initiative failure [1], [2]. Organizations hit this complexity threshold blindly, without architectural evolution [11]. The AECF provides conceptual formalization and validation agenda for characterizing thresholds, triggering conditions, and mitigation mechanisms.

### C. Practical Implications and Worked Deployment Scenario

For enterprise architects and technology leaders, the AECF provides a pattern-grounded diagnostic instrument and reframing of AI agent investment: from technology acquisition logic to architectural infrastructure development logic. The architectural scalability boundary (P2)—symptom of human governance infrastructure lagging orchestration capacity—will likely be encountered when returns drop despite increased technical complexity. The GEL enabling governance principle offers actionable guidance on autonomy–compliance tension: declarative Policy-as-Code architectures operationalized through CEPEV support superior value creation while preserving compliance integrity.

*1) Worked deployment scenario financial services:* To illustrate practical applicability, consider a financial services organization deploying AI agents for loan processing, fraud detection, and customer onboarding. The AECF would be instantiated as follows:

*a) CPL:* Persistent memory stores historical loan applications, fraud patterns, customer profiles, and past compliance decisions. The Extraction–Update cycle filters by recency and relevance, maintaining a bounded but rich context for each agent.

*b) SIL:* Interface contracts define how agents invoke enterprise systems (credit scoring APIs, KYC databases, transaction monitors). Protocol adapters handle heterogeneity across legacy and cloud-native systems.

*c) HOL:* Directive orchestration handles regulated loan approval workflows; collaborative orchestration enables fraud detection agents to share suspicious patterns; adaptive orchestration reallocates resources during peak volumes.

*d) HGI:* Human-in-the-Loop requires manager approval for loans above a threshold; Escalation Gateway triggers human review for fraud alerts above confidence boundaries; Observer monitors all agent actions.

*e) GEL:* Policy-as-Code enforces regulatory constraints (GDPR, fair lending, anti-money laundering) pre-execution. CEPEV enriches validation with the agent's compliance history, dynamically adjusting permissions based on demonstrated reliability.

This scenario demonstrates how the five layers co-evolve: as the CPL accumulates more data, the GEL can make more nuanced decisions; as the HGI calibrates autonomy based on agent performance, the HOL can operate with greater flexibility. The deployment would begin with a supervised pilot (HGI tightly constrained, HOL directive mode), progressively rolling out autonomy as the CPL matures and GEL policies are validated.

These implications are operationalized through an architectural maturity heuristic derived from the co-evolution constraint. Table III (shown below) positions the organization along technical infrastructure maturity (CPL–SIL–HOL) and governance architecture maturity (HGI–GEL) axes, deriving investment priority. Measurable indicators include memory completeness, interface contract quality, orchestration failure rates, CEPEV latency, escalation rates, policy conflicts, Agent Manager role coverage, and compliance incidents.

TABLE III. ARCHITECTURAL MATURITY HEURISTIC FOR ENTERPRISE AI AGENTS

| **Weak governance (HGI–GEL)** | **Strong governance (HGI–GEL)** |
|---|---|
| Approaching P2 boundary: ungoverned automation.<br>*Priority: develop HGI and GEL.* | Crossing boundary toward governed infrastructure.<br>*Priority: distributed scaling (Progressive Rollout).* |
| **Weak technical infrastructure (CPL–SIL–HOL)** | |
| Deficit on both axes.<br>*Priority: multi-layer co-evolution before scaling.* | Governance without execution capacity.<br>*Priority: invest first in CPL and SIL.* |

Table III provides a strategic heuristic: only organizations achieving progressive alignment along both axes cross the boundary separating isolated deployments from governed enterprise agentic infrastructure. This table is referenced in Section VI-C and provides actionable guidance for enterprise architects.

### D. System Failure Modes and Adversarial Scenarios

A governance-first framework must address system failure modes. The AECF anticipates several categories:

*1) CPL corruption:* If the CPL is compromised (e.g., through adversarial injection of false events), CEPEV decisions become unreliable. Mitigations include: cryptographic event signatures (ensuring event authenticity), append-only memory structures (preventing history alteration), governance-engine write validation (validating all CPL writes against GEL policies), and integrity monitoring alerts (triggering escalation on detected corruption).

*2) Policy conflicts under concurrent agent proliferation:* As agent populations grow, policy interactions become complex. Mitigations include: policy versioning and conflict detection (automated analysis of policy sets), prioritized override rules (explicit conflict resolution), CEPEV-based dynamic constraint relaxation (adjusting policies based on demonstrated reliability), and Agent Manager oversight (human review of policy conflicts).

*3) Adversarial agent behavior:* Malicious agents may attempt to exploit governance gaps. Mitigations include: authorization token propagation controls (binding delegation to scope and time), pre-execution validation (blocking unauthorized actions), Observer pattern monitoring (detecting anomalous behavior), and escalation thresholds (triggering human review on suspicion).

*4) Governance latency attacks:* Attackers may attempt to overload CEPEV validation to induce denial of service. Mitigations include: bounded snapshot size (capped by $\theta$), caching of frequent policy decisions, and rate limiting on agent action submissions.

These failure modes inform the GEL design: governance is not a static rule set but an adaptive, fault-tolerant envelope that anticipates and responds to system vulnerabilities.

## VII. CONCLUSION AND RESEARCH PERSPECTIVES

The AECF makes governance a native architectural condition, not a control device added afterward. Scaling AI agents in the enterprise demands more than model sophistication: transition from isolated deployment to resilient, governed, and durably controllable infrastructure requires co-evolution of technical, organizational, and human dimensions. These dimensions do not compensate for one another: advanced orchestration is fragile without persistent memory, human governance is ineffective without technical instrumentation, declarative compliance is hollow unless wired into execution.

The fundamental contribution is systemic articulation of architectural layers into coherent enterprise capability rather than isolated components. This logic rests on a co-evolution constraint binding CPL, SIL, HOL, HGI, and GEL. Their robustness depends on joint maturation. The CEPEV mechanism

operationalizes this constraint through relationship R5 connecting operational memory with pre-execution governance. Governance becomes an engineering requirement—designed, executed, and revised within architecture rather than external organizational aspiration.

### A. Research Perspectives

Future research should pursue four complementary directions:

*1) Empirical validation:* A multiple-case study design [24] will test the five propositions across diverse enterprise contexts (financial services, healthcare, manufacturing, public administration), measuring CPL maturity, HGI capacity, governance complexity, CEPEV effectiveness, and agentic scalability boundaries.

*2) Formal verification:* The CEPEV salience function and the five propositions merit formal verification to establish correctness conditions for the AECF's architectural logic.

*3) Tooling:* Reference implementation of the AECF as an open-source architectural blueprint, including Policy-as-Code templates, CPL reference schemas, HGI calibration dashboards, and deployment orchestration scripts.

*4) Regulatory alignment:* Systematic mapping of the AECF to the EU AI Act [52], NIST AI RMF, and ISO/IEC 42001 to establish normative interoperability and compliance assurance.

## REFERENCES

[1] SAP LeanIX, "State of enterprise architecture 2024," SAP LeanIX, 2024. [Online]. Available: https://www.leanix.net/en/wiki/ea/state-of-enterprise-architecture

[2] V. Acharya, "Governing the agentic enterprise: A governance maturity model for managing AI agent sprawl in business operations," arXiv:2604.16338, 2026. doi: 10.48550/arXiv.2604.16338

[3] M. Wooldridge and N. R. Jennings, "Intelligent agents: Theory and practice," *The Knowledge Engineering Review*, vol. 10, no. 2, pp. 115–152, 1995. doi: 10.1017/S0269888900008122

[4] T. W. Malone and K. Crowston, "The interdisciplinary study of coordination," *ACM Computing Surveys*, vol. 26, no. 1, pp. 87–119, 1994. doi: 10.1145/174666.174668

[5] P. Giorgini, M. Kolp, and J. Mylopoulos, "Multi-agent architectures as organizational structures," *Autonomous Agents and Multi-Agent Systems*, vol. 13, no. 1, pp. 3–25, 2003. doi: 10.1007/s10458-006-5997-6

[6] J. A. Zachman, "A framework for information systems architecture," *IBM Systems Journal*, vol. 26, no. 3, pp. 276–292, 1987. doi: 10.1147/sj.263.0276

[7] The Open Group, *TOGAF® Standard, Version 9.2*. The Open Group, 2018.

[8] J. Lapalme, A. Gerber, A. Van der Merwe, J. Zachman, M. De Vries, and K. Hinkelmann, "Exploring the future of enterprise architecture: A Zachman perspective," *Computers in Industry*, vol. 79, pp. 103–113, 2016. doi: 10.1016/j.compind.2015.06.010

[9] S. Kotusev, "TOGAF-based enterprise architecture practice: An exploratory case study," *Communications of the Association for Information Systems*, vol. 44, Article 20, 2019. doi: 10.17705/1CAIS.04420

[10] K. Perikala, "Architecting MCP-based platforms for enterprise-scale agentic generative AI," *Journal of Business Intelligence and Data Analytics*, vol. 2, no. 3, pp. 1–8, 2025. doi: 10.55124/jbid.v2i3.264

[11] P. Venkiteela, "An enterprise agentic architecture framework for agentic AI governance and scalable autonomy," *Scientific Journal of Computer Science*, vol. 2, no. 1, 2026. doi: 10.64539/sjcs.v2i1.2026.368

[12] S. Adimulam, R. Gupta, and A. Kumar, "The orchestration of multi-agent systems: Architectures, protocols, and enterprise adoption," arXiv:2601.13671, 2026. doi: 10.48550/arXiv.2601.13671

[13] L. Bass, P. Clements, and R. Kazman, *Software Architecture in Practice*, 3rd ed. Addison-Wesley, 2012.

[14] E. Gamma, R. Helm, R. Johnson, and J. Vlissides, *Design Patterns: Elements of Reusable Object-Oriented Software*. Addison-Wesley, 1994.

[15] A. R. Hevner, S. T. March, J. Park, and S. Ram, "Design science in information systems research," *MIS Quarterly*, vol. 28, no. 1, pp. 75–105, 2004. doi: 10.2307/25148625

[16] K. Peffers, T. Tuunanen, M. A. Rothenberger, and S. Chatterjee, "A design science research methodology for information systems research," *Journal of Management Information Systems*, vol. 24, no. 3, pp. 45–77, 2007. doi: 10.2753/MIS0742-1222240302

[17] J. Chen and L. Zhao, "AI-driven innovation in enterprise architecture: A multi-agent system approach to adaptive design," in *Proceedings of the 2024 8th International Conference on Electronic Information Technology and Computer Engineering (EITCE 2024)*, ACM, 2024, pp. 768–774. doi: 10.1145/3711129.3711261

[18] S. Kotusev and S. Kurnia, "The practical roles of enterprise architecture artifacts: A classification and relationship," *Information and Software Technology*, vol. 149, Article 106897, 2022. doi: 10.1016/j.infsof.2022.106897

[19] C. Qian et al., "Towards a science of scaling agent systems," arXiv:2512.08296, 2025. doi: 10.48550/arXiv.2512.08296

[20] M. Shaw and D. Garlan, *Software Architecture: Perspectives on an Emerging Discipline*. Prentice Hall, 1996.

[21] F. Buschmann, R. Meunier, H. Rohnert, P. Sommerlad, and M. Stal, *Pattern-Oriented Software Architecture: A System of Patterns*. Wiley, 1996.

[22] N. Krishnan, "AI agents: Evolution, architecture, and real-world applications," arXiv:2503.12687, 2025. doi: 10.48550/arXiv.2503.12687

[23] S. Gregor and A. R. Hevner, "Positioning and presenting design science research for maximum impact," *MIS Quarterly*, vol. 37, no. 2, pp. 337–355, 2013. doi: 10.25300/MISQ/2013/37.2.01

[24] R. K. Yin, *Case Study Research and Applications: Design and Methods*, 6th ed. SAGE, 2018.

[25] E. W. Dijkstra, "On the role of scientific thought," in *Selected Writings on Computing: A Personal Perspective*, Springer-Verlag, 1982, pp. 60–66.

[26] D. L. Parnas, "On the criteria to be used in decomposing systems into modules," *Communications of the ACM*, vol. 15, no. 12, pp. 1053–1058, 1972.

[27] L. Lamport, "Time, clocks, and the ordering of events in a distributed system," *Communications of the ACM*, vol. 21, no. 7, pp. 558–565, 1978.

[28] N. A. Lynch, *Distributed Algorithms*. Morgan Kaufmann, 1996.

[29] M. Fowler, "Event sourcing," martinfowler.com, 2005.

[30] M. Kleppmann, *Designing Data-Intensive Applications*. O'Reilly Media, 2017.

[31] P. Lewis et al., "Retrieval-augmented generation for knowledge-intensive NLP tasks," *Advances in Neural Information Processing Systems*, vol. 33, pp. 9459–9474, 2020.

[32] A. De Saint Marc et al., "Beyond message passing: A semantic view of agent communication protocols," arXiv:2604.02369, 2025. doi: 10.48550/arXiv.2604.02369

[33] A. P. Sheth and J. A. Larson, "Federated database systems for managing distributed, heterogeneous, and autonomous databases," *ACM Computing Surveys*, vol. 22, no. 3, pp. 183–236, 1990.

[34] Y. Kong et al., "A survey of agent interoperability protocols: Model Context Protocol (MCP), Agent Communication Protocol (ACP), Agent-to-Agent Protocol (A2A), and Agent Network Protocol (ANP)," arXiv:2505.02279, 2025. doi: 10.48550/arXiv.2505.02279

[35] J. O. Kephart and D. M. Chess, "The vision of autonomic computing," *IEEE Computer*, vol. 36, no. 1, pp. 41–50, 2003.

[36] M. Salehie and L. Tahvildari, "Self-adaptive software: Landscape and research challenges," *ACM Transactions on Autonomous and Adaptive Systems*, vol. 4, no. 2, Article 14, 2009.

[37] G. R. Andrews, *Foundations of Multithreaded, Parallel, and Distributed Programming*. Addison-Wesley, 2000.

[38] G. Cabri, L. Leonardi, and F. Zambonelli, "MARS: A programmable coordination architecture for mobile agents," *IEEE Internet Computing*, vol. 4, no. 4, pp. 26–35, 2000.

[39] T. B. Sheridan, *Telerobotics, Automation, and Human Supervisory Control*. MIT Press, 1992.

[40] R. Parasuraman, T. B. Sheridan, and C. D. Wickens, "A model for types and levels of human interaction with automation," *IEEE Transactions on Systems, Man, and Cybernetics — Part A: Systems and Humans*, vol. 30, no. 3, pp. 286–297, 2000.

[41] F. Jackson, "Governing autonomous AI agents with policy-as-code: A multi-layer architecture for risk, compliance, and zero-trust control," SSRN:5820262, 2025. doi: 10.2139/ssrn.5820262

[42] G. Kholkar et al., "Policy-as-prompt: Turning AI governance rules into guardrails for AI agents," arXiv:2509.23994, 2025. doi: 10.48550/arXiv.2509.23994

[43] J. Mavračić, "Policy cards: Machine-readable runtime governance for autonomous AI agents," arXiv:2510.24383, 2025. doi: 10.48550/arXiv.2510.24383

[44] M. Sloman and E. Lupu, "Security and management policy specification," *IEEE Network*, vol. 16, no. 2, pp. 10–19, 2002.

[45] K. Twidle, N. Dulay, E. Lupu, and M. Sloman, "Ponder2: A policy system for autonomous pervasive environments," in *Fifth International Conference on Autonomic and Autonomous Systems*, IEEE, 2009, pp. 330–335.

[46] G. Kiczales et al., "Aspect-oriented programming," in *ECOOP'97 — Object-Oriented Programming*, Springer, 1997, pp. 220–242.

[47] P. Weill and J. W. Ross, *IT Governance: How Top Performers Manage IT Decision Rights for Superior Results*. Harvard Business School Press, 2004.

[48] V. C. Hu et al., "Guide to attribute based access control (ABAC) definition and considerations," NIST Special Publication 800-162, National Institute of Standards and Technology, 2014.

[49] A. Jøsang, R. Ismail, and C. Boyd, "A survey of trust and reputation systems for online service provision," *Decision Support Systems*, vol. 43, no. 2, pp. 618–644, 2007.

[50] P. Chhikara, D. Khant, S. Aryan, T. Singh, and D. Yadav, "Mem0: Building production-ready AI agents with scalable long-term memory," arXiv:2504.11053, 2025. doi: 10.48550/arXiv.2504.11053

[51] J. P. Walsh and G. R. Ungson, "Organizational memory," *Academy of Management Review*, vol. 16, no. 1, pp. 57–91, 1991. doi: 10.2307/258607

[52] European Parliament, "Regulation (EU) 2024/1689 of the European Parliament and of the Council laying down harmonised rules on artificial intelligence (Artificial Intelligence Act)," Official Journal of the European Union, 2024.