International Journal of Computing and Artificial Intelligence

E-ISSN: 2707-658X
P-ISSN: 2707-6571
Impact Factor (RJIF): 5.57
www.computersciencejournals.com/ijcai
IJCAI 2025; 6(2): 169-176
Received: 05-06-2025
Accepted: 10-07-2025

Prateek Batla
University of Michigan, Ann Arbor, Michigan, USA

Corresponding Author:
Prateek Batla
University of Michigan, Ann Arbor, Michigan, USA

AI commerce agents: A framework for trustworthy autonomy in enterprise workflows

Prateek Batla

DOI: https://doi.org/10.33545/27076571.2025.v6.i2c.196

Abstract
The rapid emergence of AI commerce agents is reshaping enterprise workflows, from billing, pricing, and promotions to fraud detection and customer personalization. While these agents promise major gains in efficiency and scale, their deployment raises critical concerns about autonomy, trustworthiness, and security in mission-critical environments. Prior research addresses trust in AI agents, workflow automation, and zero trust architectures separately, but few studies integrate these dimensions within the specific context of enterprise commerce. This paper proposes a comprehensive framework for trustworthy autonomy in AI commerce agents, synthesizing recent advances in agentic AI, workflow orchestration, and enterprise security. The framework specifies layered mechanisms across perception, reasoning, trust and security, and action, underpinned by transparency, accountability, fairness, and resilience. It operationalizes graded autonomy with human-in-the-loop and human-on-the-loop controls, policy-aware guardrails, and auditable safety cases. Drawing on recent works from journals and peer-reviewed venues, we show how the framework applies to subscription billing, dynamic pricing, cross-border payments, order management, and fraud prevention. I outline evaluation dimensions across technical, organizational, and ethical criteria and provide a practical rubric for enterprise adoption. By bridging autonomy and trust in enterprise contexts, the study contributes both a conceptual foundation and a deployable architecture for AI commerce agents at scale.

Keywords: AI commerce agents, enterprise workflows, trustworthy AI autonomy, agentic AI, workflow automation, dynamic pricing and billing, zero trust security, auditability

Introduction

AI commerce agents are increasingly embedded in enterprise workflows such as subscription billing, dynamic pricing, promotions, order management, fraud prevention, and customer experience. Firms adopt these agents to increase decision velocity, orchestrate complex toolchains, and improve operational efficiency while preserving compliance and auditability (Huang, 2025; Ranjan *et al*., 2025; Gadde, 2025) [3, 8, 2]. Agentic capabilities extend beyond single tasks to mission progress across systems of record, payments, and customer channels, making design choices material to revenue and risk (Ranjan *et al*., 2025; Huang, 2025) [8, 3]. Rising autonomy heightens concerns about transparency, controllability, and security in mission-critical settings where actions can trigger financial events and downstream obligations. Trust architectures for enterprise assistants emphasize disclosure, provenance, and policy alignment to keep automated actions explainable and auditable (Kareti, 2025) [6]. Security and privacy requirements include least-privilege access, isolation, continuous verification, and protection of sensitive data (Inaganti and Sundaramurthy, 2020; Chennupati, 2025) [4, 1]. Analyses of autonomous process agents catalog failure modes and recommend layered mitigations such as separation of duties, guardrails, and continuous monitoring (Madireddy, 2025; Chennupati, 2025; Sundaramurthy and Ravichandran, 2022) [7, 1, 10]. Despite progress, the literature does not unify autonomy, trust, and commerce-specific governance into a deployable framework. Existing books and chapters outline how agentic AI can transform enterprises but stop short of specifying graded autonomy, approvals for monetary actions, and end-to-end audit trails tailored to commerce workflows (Ranjan *et al*., 2025; Huang, 2025) [8, 3]. Surveys of AI-driven automation synthesize taxonomies and lifecycle challenges but treat security, audit, and fairness as parallel concerns rather than first-class architectural constraints for commerce agents (Jin *et al*., 2025) [5]. Taxonomies distinguishing AI agents from agentic AI clarify planning, tool use, memory, and

collaboration but do not bind these capabilities to enterprise policy and control objectives (Sapkota *et al*., 2025) [9].

We define an AI commerce agent as an autonomous or semi-autonomous software entity that perceives enterprise signals, reasons under explicit business and compliance policies, and acts through approved tools to advance a commerce objective. Our view of agentic AI is workflow-centric, prioritizing orchestration, memory, and tool proficiency while constraining autonomy through governance and security controls appropriate to financial risk and regulatory exposure (Sapkota *et al*., 2025; Ranjan *et al*., 2025; Huang, 2025) [9, 8, 3].

Research questions

- **RQ1:** Which properties constitute trustworthy autonomy for enterprise commerce agents across technical, organizational, and ethical dimensions?
- **RQ2:** What layered architecture can operationalize these properties using perception, reasoning, trust-and-security, and action layers with enterprise-grade controls?
- **RQ3:** How should enterprises evaluate such agents for reliability, security, auditability, fairness, and business impact in scenarios like dynamic pricing, billing, and fraud prevention?
- **RQ4:** What risks and failure modes arise from fully autonomous process agents, and how can zero-trust principles and policy-aware orchestration mitigate them in production?

The literature establishes why agentic AI matters, how it differs from standard agents, and which trust and zero trust controls are relevant (Ranjan *et al*., 2025; Huang, 2025; Sapkota *et al*., 2025; Kareti, 2025; Inaganti and Sundaramurthy, 2020; Jin *et al*., 2025) [8, 3, 9, 6, 4, 5]. What is missing is a deployable framework that binds agentic capabilities to enterprise policy and control objectives in commerce, including dual control, auditability, fairness, and financial reconciliation. The next section proposes a layered architecture with graded autonomy, safety cases, and policy-aware orchestration tailored to enterprise commerce.

3. Framework for Trustworthy Autonomy in AI Commerce Agents

3.1 Design principles

Trustworthy autonomy for enterprise commerce agents rests on five principles.

Transparency: expose inputs, policies, and decision rationale for audit. Controls include decision traces, input attributions, and human-readable policy checks (Kareti, 2025) [6].

Accountability: bind actions to identities, policies, and approvals with durable records. Controls include immutable logs, provenance, and tamper-evident storage (Kareti, 2025) [6].

Security: protect identities, tools, and data through least privilege, isolation, and continuous verification under zero trust (Inaganti and Sundaramurthy, 2020; Sundaramurthy and Ravichandran, 2022) [4, 10].

Fairness: avoid harmful differential treatment in pricing, promotions, and fraud actions. Controls include policy-aware prechecks, disparity monitoring, and reversible actions (Kareti, 2025) [6].

Resilience: degrade safely under drift, faults, or attacks. Controls include SLOs, circuit breakers, and tested fallbacks informed by automation reliability guidance (Jin *et al*., 2025) [5].

Implication: these principles drive concrete controls for billing, dynamic pricing, and fraud in large enterprises (Ranjan *et al*., 2025; Huang, 2025) [8, 3].

Principles to metrics and targets

Table 1: Principles to metrics and targets

Principle	Operational metric	Target example	Data source	Principle
Transparency	Explanation coverage on monetary actions	≥ 95 percent	Explanation service logs	Transparency
Accountability	Log completeness for sensitive actions	100 percent	Immutable event store with provenance	Accountability
Security	Privileged calls with continuous verification	100 percent	Authz gateway under zero trust	Security
Fairness	Pricing: absolute price-gap across protected segments	≤ Δ p (deployment-tuned)	Pricing audit reports and fairness checks	Fairness
Fairness	Fraud: equalized odds gap	≤ Δ f (deployment-tuned)	Fraud model audits and case review samples	Fairness
Resilience	Successful rollback rate for monetary actions	≥ R* (deployment-tuned)	Reconciliation jobs and rollback telemetry	Resilience
Auditability	Audit retrieval P95 time	≤ T* (deployment-tuned)	Audit API traces and storage access logs	Auditability
Control plane	Overhead on decision latency	≤ L* (deployment-tuned)	Online decision traces and SLO dashboards	Control plane

Operationalization of transparency, accountability, security, fairness, resilience, auditability, and control-plane overhead with measurable metrics, thresholds, and data sources.

3.2 Architecture: data plane and control plane

The framework separates execution from enforcement. The data plane performs perception, reasoning, and action. The control plane enforces the trust architecture and zero trust consistently across layers.

Data plane

- **Perception:** assemble a policy-aware state from events, features, and policy context with validation and provenance.
- **Reasoning:** plan and select tools under constraints using agentic AI capabilities such as goal decomposition, tool selection, memory, uncertainty calibration, and selective abstention when confidence < τ or fairness checks return indeterminate status

(Sapkota *et al*., 2025; Ranjan *et al*., 2025) [9, 8]. The value of τ is tuned empirically during evaluation.
- **Action:** execute preconditioned, idempotent operations with rollback and reconciliation in systems of record (Huang, 2025; Ranjan *et al*., 2025; Gadde, 2025) [3, 8, 2].

Control plane
- **Policy decision point (PDP):** evaluate policies for perception, reasoning, and action.
- **Policy enforcement points (PEPs):** interpose checks at data ingress, plan selection, and actuation time. PEPs fail closed: on PDP errors or ambiguous context, the default is deny.
- **Identity and secrets:** enforce least privilege, short-lived credentials, and isolation with continuous verification under zero trust (Inaganti and Sundaramurthy, 2020; Sundaramurthy and Ravichandran, 2022) [4, 10].
- **Assurance services:** explanation, logging, monitoring, approvals, and fairness checks (Kareti, 2025) [6].
- Control-plane overhead is attributed via per-request tracing with PEP timing spans and trace IDs recorded in evidence bundles.
- A parameter registry records deployment-time tunables {τ, k, n, Δp, Δf, L*, T*, R*} under a param_set_id referenced in all evidence bundles.

PEP intercepts and emitted evidence

Table 2: PEP intercepts and emitted evidence

Layer	PEP intercept focus	Evidence artifacts emitted
Perception	Schema validation, PII tagging, quality	Schema validation report, PII tag map, provenance hash
Reasoning	Plan admissibility, quotas, fairness	Policy evaluation ID, plan proof, fairness report ID, uncertainty summary
Action	Approvals, preconditions, rollback hooks	Approval token, explanation bundle ID, rollback plan ID, reconciliation job ID

Policy Enforcement Points (PEPs) at perception, reasoning, and action time enforce constraints and emit evidence artifacts for audit and assurance.

3.2.1 Threat model
Attackers can be external or internal. Trust boundaries run between data plane components, the control plane, and systems of record. Goals include monetary abuse, policy bypass, data exfiltration, and unfair treatment. Table 3 summarizes the threat model with actors, capabilities, targets, and control-plane mitigations.

Table 3: Threat model for AI commerce agents

Actor	Capability	Target	Control-plane mitigation
External adversary	Prompt or tool injection	Reasoning and action flows	PEP checks, least privilege, isolation
Malicious insider	Approval misuse, policy tampering	Action approvals	Dual control, immutable logs, approval analytics
Compromised service	Privilege escalation, data scraping	Perception and action APIs	Continuous verification, short-lived creds, quotas
Data supplier	Poisoned inputs, schema drift	Perception ingest	Schema validation, provenance, anomaly screens

Attacker categories, capabilities, targets, and control-plane mitigations. Trust boundaries run between the data plane, control plane, and systems of record.

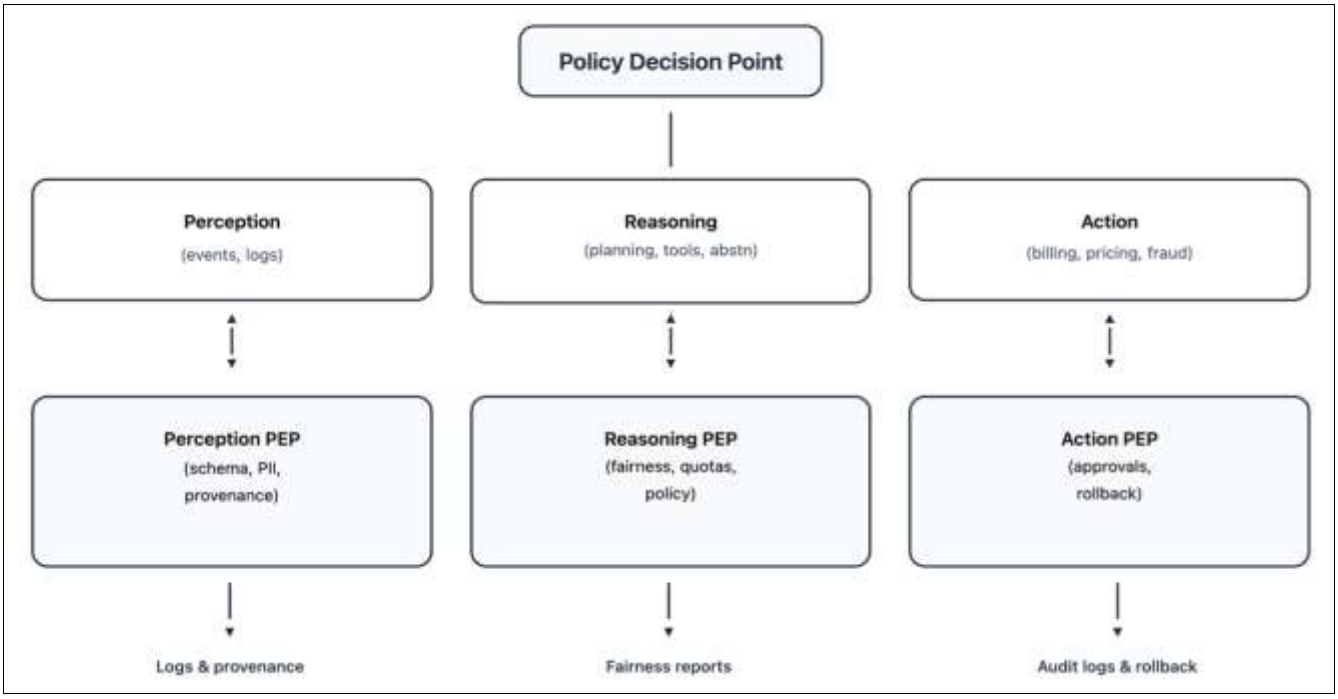

Fig 1: Data plane and control plane architecture for AI commerce agents.

Perception, reasoning, and action are mediated by Policy Enforcement Points (PEPs) under a central Policy Decision Point (PDP). Evidence artifacts include logs, fairness reports, and rollback plans.

3.3 Cross-cutting governance via control-plane PEPs

Governance is executable through PEPs at perception, reasoning, and action time. Perception-time PEPs validate schemas, tag PII, enforce data minimization, and attach provenance. Reasoning-time PEPs check plans against business rules, fairness constraints, quotas, exploration budgets, and uncertainty thresholds. Action-time PEPs require approvals for monetary actions, verify preconditions, record an evidence bundle before actuation, enforce rollback handlers, and schedule reconciliation jobs (Kareti, 2025; Inaganti and Sundaramurthy, 2020) [6, 4].

Minimal policy grammar

Subjects (S) are agent or human identities with roles. Actions (A) are tool methods or workflow steps. Resources (R) are data objects or business entities. Constraints (C) are predicates over context (X) such as amount, geography, risk score, time, and customer attributes. Policy form: allow|deny S A R where C. Example: allow agent(role=pricing) update_price SKU where amount_change $\leq$ δ and fairness_ok and geo $\in$ {US, EU}.

LTL operators and safety invariants

We use Linear Temporal Logic with G (globally), F (eventually), X (next), and U (until). Let exec_monetary, approved, policy_pass, exec_sensitive, and rollback_ready be atomic propositions.

I1: G(exec_monetary$\rightarrow$F approved)

I2: G(exec_sensitive$\rightarrow$X policy_pass)

I3: G(exec_monetary$\rightarrow$rollback_ready)

Semantically, the operators can be interpreted as follows:

- **G (Globally):** "Always." The condition must hold for the entire run. Example: G(price$\geq$ 0) means the price is always non-negative.
- **F (Finally/Eventually):** "Sometime in the future." Example: F(order_is_delivered)F(order_is_delivered)F(order_is_delivered) means every order is eventually delivered.
- **X (Next):** "In the very next step." Example: X(payment_is_processed) means payment must be processed in the next step.
- **U (Until):** The first condition must hold until the second becomes true. Example: (inventory_check U order_placed) means inventory is checked continuously until an order is placed.

Proposition (Enforceability)

If PEPs intercept perception, reasoning, and action, and the PDP evaluates policies before PEPs allow progress, then I1–I3 hold.

Proof sketch: Action-time PEP blocks exec_monetary until an approval token exists, so I1 holds. For any exec_sensitive, the PEP requires a successful policy evaluation in the immediately preceding step, so I2 holds. The action-time PEP also checks the presence of a registered rollback plan before execution, so I3 holds. Hence the control plane enforces the invariants.

Evidence bundle schema

{policy_hash, policy_version, approver_id, explanation_id, lineage_ids[], fairness_report_id, signature}

The invariants I1–I3 are mechanically verified on a finite-state model of the autonomy machine (Appendix A). This converts the informal safety claims into checkable properties.

3.4 Graded autonomy and safety cases

Autonomy modes form a constrained state machine. Monetary actions require dual control or an equivalent independent control with periodic effectiveness testing (Kareti, 2025; Chennupati, 2025; Madireddy, 2025) [6, 1, 7]. Apply hysteresis with a k-of-n rule to prevent flapping. An upgrade occurs when success criteria are met in k of n evaluation windows. A downgrade occurs when violations are observed in k of n windows or when a critical incident fires. Values for k and n are deployment parameters tuned to workload stability and risk tolerance.

Assist

Entry: New workflow or high variance.
Exit: K-of-n windows meet targets and audits are clean.
Invariant: Proposals for monetary actions include explanation and policy evaluation ID.

Constrained Execute

Entry: Assist meets targets in k-of-n windows.
Exit: threshold breach, anomaly, or policy alert in k-of-n windows.
Invariant: actions remain within quotas and budget caps.

Conditional Autonomy

Entry: Constrained Execute meets targets with low variance.
Exit: any policy block or anomaly triggers automatic downgrade by the k-of-n rule.
Invariant: failed fairness or approval checks prevent execution.

Full Autonomy

Entry: executive approval and signed safety case after k-of-n success.
Exit: incident, drift, or metric regression by the k-of-n rule.
Invariant: tested rollback exists for every monetary action.

Global invariant

For all modes: G(exec_sensitive$\rightarrow$policy_pass$\wedge$log_emit).

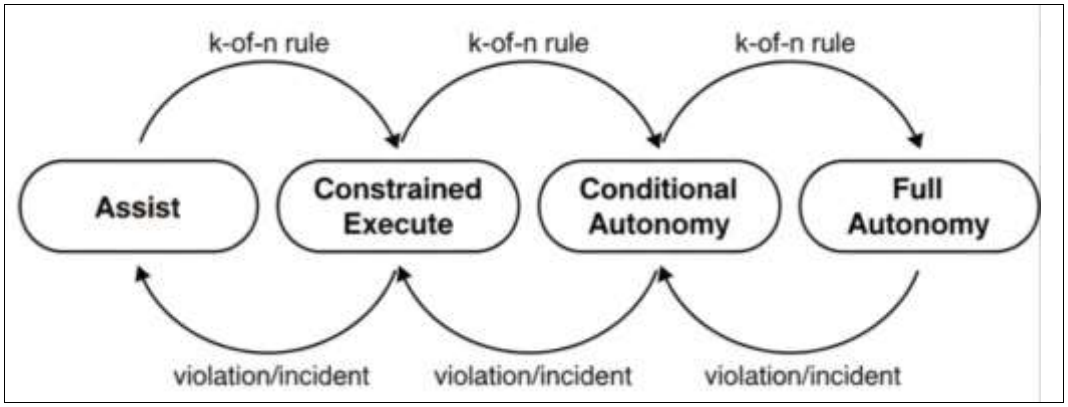

Fig 2: Autonomy modes as a state machine with k-of-n hysteresis.

Entry and exit conditions are tied to policy checks, approvals, and rollback readiness.

3.5 Data governance and privacy controls

Data controls reduce blast radius and improve assurance. Use data minimization and PII tagging with policy gates at ingestion. Enforce least privilege and process or network isolation between components. Re-authenticate and re-authorize on every sensitive call with short-lived credentials under zero trust. Stamp data and outputs with origin, version, and policy context to support audits, fairness reviews, and reconciliation. Respect data residency policies. Retain raw event data for a configurable window (e.g., 30 days) and aggregates for a longer window (e.g., 1 year), tuned to regulatory and business requirements (Inaganti and Sundaramurthy, 2020; Sundaramurthy and Ravichandran, 2022; Kareti, 2025) [4, 10, 6].

3.6 Assurance patterns

Assurance patterns provide evidence that the agent is safe to deploy and to keep running.

Pre-deployment testing

Run unit tests for policy checks, simulation for workflow effects, and canaries for limited exposure. Include fairness tests for pricing and guardrail tests for billing thresholds. Pricing fairness uses absolute price-gap with weekly audits; fraud fairness uses equalized odds gap with daily audits (Kareti, 2025; Jin *et al*., 2025) [6, 5].

Adversarial red teaming

Probe prompt or tool injection, permission escalation, data exfiltration, and policy bypass. Validate isolation and least privilege against attacker models derived from workflow risks (Inaganti and Sundaramurthy, 2020; Chennupati, 2025) [4, 1].

Runtime monitoring

Track action success, rollback rates, anomaly flags, fairness disparities, and approval latency. Tie alerts to automatic mode downgrades or kill switches when thresholds are crossed (Jin *et al*., 2025; Madireddy, 2025) [5].

Rollback and reconciliation

Design each monetary action with a tested rollback and post-action financial reconciliation to detect and correct divergence from systems of record.

Post-incident reviews

Run reviews. Update the risk catalog. Revise safety cases and guardrails before resuming autonomy.

Evaluation design

- **Baselines:** Human-only workflow, rule-based orchestration, and an agent without the control plane.
- **Ablations:** remove or relax one control at a time, such as fairness prechecks, dual control, or continuous verification.
- **Datasets and replay logs:** Use historical billing, pricing, and fraud logs with policy tags and outcomes; augment with synthetic scenarios that stress rare events.
- **Red-team suites:** Prompt and tool injection, policy bypass attempts, privilege escalation, and data exfiltration targeted at perception, reasoning, and action PEPs.
- **Business KPI analysis:** Apply a non-inferiority margin on core KPIs such as authorization rate or revenue per session; choose Welch's t-test for unequal variances under approximate normality or Mann-Whitney for non-normal data; report 95 percent confidence intervals; ensure power $\geq$ 0.8 by pre-study power analysis.
- **Risk and control metrics:** Use the metrics in 4.1 plus fraud loss, approval latency, rollback success, anomaly precision and recall, and fairness disparity.
- **Sample size for replay:** compute minimal N using standard power formulas for means or proportions and the chosen margin and effect size; when unknown, set a floor of at least Nmin actions per scenario to bound variance.

Implication: this design links controls to measurable outcomes and provides statistically sound evidence that the framework improves reliability and safety in billing, pricing, and fraud contexts (Gadde, 2025; Jin *et al*., 2025; Chennupati, 2025; Madireddy, 2025) [2, 5, 1, 7].

4. Applications to Enterprise Commerce

4.1 Subscription billing and invoicing

Scope: Agents automate invoice creation, usage rating, discounts, dunning, and adjustments.

Framework application. Perception assembles subscription state, entitlements, usage events, and payment risk with provenance. Reasoning plans dunning sequences and adjustment options under rules, with selective abstention

when confidence < τ or fairness is indeterminate. Action executes idempotent postings and schedules reconciliation jobs. The control plane enforces approvals and dual control for monetary actions, writing an evidence bundle for each sensitive step (Huang, 2025; Ranjan *et al*., 2025; Kareti, 2025) [3, 8, 6].

Controls and Metrics: Enforce I1 and I3 for refunds, credits, and write-offs. Monitor log completeness, rollback success rate ≥ R*, Audit retrieval P95 ≤ T*. Evaluate non-inferiority on authorization rate and revenue recognition stability; report 95% CIs with Welch or Mann-Whitney per the plan (Jin *et al*., 2025) [5].

Governance note: Evidence bundles include policy_hash, approver_id, explanation_id, param_set_id, and reconciliation IDs to support auditability and dispute resolution.

4.2 Dynamic pricing and promotions

Scope: Agents adjust prices and offers using demand, inventory, and campaign context.

Framework application: Perception ingests demand signals, stock levels, and policy limits. Reasoning proposes price changes with fairness constraints and exploration budgets; it abstains below τ. Action applies canary updates, quotas, and rollbacks. Control-plane PEPs enforce fairness prechecks, delta guardrails, and explanation bundles before actuation (Ranjan *et al*., 2025; Huang, 2025; Kareti, 2025) [8, 3, 6].

Controls and metrics: Use pricing fairness as absolute price-gap ≤ Δp with a weekly audit cadence. Enforce I2 on every sensitive price change. Track P95 decision latency and hold control-plane overhead within L*. In evaluation, apply a pre-specified non-inferiority margin on conversion or revenue per session; report two-sided 95% CIs and power ≥ 0.8 (Jin *et al*., 2025) [5].

Governance note: Evidence bundles bind each price change to policy and fairness reports, with change_request_id for approvals and param_set_id for tunables.

4.3 Fraud detection and prevention

Scope: Agents score transactions, place holds, decline, or escalate.

Framework application: Perception aggregates device, velocity, and behavioral features with lineage. Reasoning evaluates actions under equalized-odds fairness and abstains when confidence < τ or policy context is incomplete. Action places time-bound holds with auto-release checks and executes declines with rollback plans where feasible. Control-plane PEPs require approvals for irreversible monetary effects and attach full evidence bundles (Gadde, 2025; Kareti, 2025) [2, 6].

Controls and metrics: Enforce I2 for all sensitive actions and I3 for reversible holds. Track equalized-odds gap ≤ Δf with a daily audit cadence, fraud loss, false positive rate, and analyst queue latency. Maintain Audit retrieval P95 ≤ T* (Jin *et al*., 2025; Chennupati, 2025; Madireddy, 2025) [5, 1, 7].

Governance note. Zero trust controls restrict tool access and require continuous verification on each sensitive call (Inaganti and Sundaramurthy, 2020; Sundaramurthy and Ravichandran, 2022) [4, 10].

4.4 Cross-border payments and FX settlement

Scope: Agents orchestrate KYC checks, jurisdictional policies, FX quotes, routing, and settlement.

Framework application. Perception assembles KYC status, geography, limits, and counterparty risk with provenance. Reasoning plans payment paths with constraints for jurisdiction, amount, and cutoff windows; it abstains when compliance context is indeterminate. Action books transfers with preconditions and initiates reconciliation. Control-plane PEPs enforce dual control for high-value transfers, data residency, and immutable logging (Inaganti and Sundaramurthy, 2020; Sundaramurthy and Ravichandran, 2022; Kareti, 2025) [4, 10, 6].

Controls and metrics: Enforce I1, I2, and I3 on high-value payments. Monitor settlement mismatch rate, exception queue size, and approval latency. In evaluation, use non-inferiority on authorization rate and cost per payment; report 95% CIs and apply test-choice rules as specified (Jin *et al*., 2025) [5].

Governance note: Evidence bundles capture approver identity, policy version, and cross-border policy proofs for audit retrieval.

4.5 Order management and fulfillment

Scope: Agents resolve backorders, substitutions, cancellations, and split shipments.

Framework application: Perception collects inventory, lead times, and customer preferences with lineage. Reasoning chooses substitutions under policy and fairness limits; abstains below τ. Action triggers fulfillment changes with pre- and post-conditions and records reconciliation IDs for financial impact. Control-plane PEPs enforce policy-conformant substitutions, approvals for costly changes, and complete provenance (Ranjan *et al*., 2025; Huang, 2025; Kareti, 2025) [8, 3, 6].

Controls and metrics: Enforce I2 for policy checks before high-impact actions and verify I3 for reversible steps. Track fulfillment SLOs, substitution fairness if applicable, and rate of post-order financial corrections. Keep overhead within L*.

Governance note: Immutable logs and evidence bundles support customer redress and internal audits.

4.6 Implications for governance

Across workflows, the data plane executes perception, reasoning, and action while the control plane enforces policy via PEPs and the PDP. The policy grammar standardizes decision rights. LTL invariants I1–I3 bind sensitive and monetary actions to approvals, policy checks, and rollback readiness. Evidence bundles and immutable logs deliver auditability with retrieval target T*. Graded

autonomy with k-of-n hysteresis aligns decision rights with risk and avoids flapping. The evaluation plan from Section 3.6 pre-registers baselines and ablations, uses replay logs and red-team suites, and reports non-inferiority and risk metrics with 95% CIs and power guarantees (Gadde, 2025; Jin *et al*., 2025; Kareti, 2025; Chennupati, 2025; Madireddy, 2025; Inaganti and Sundaramurthy, 2020; Sundaramurthy and Ravichandran, 2022) [2, 5, 6, 1, 7, 4, 10].

5. Discussion and Future Work

5.1 Implications for research

This paper formalizes a data plane–control plane architecture for AI commerce agents with enforceable properties: policy-aware orchestration through PEP–PDP hooks, evidence bundles, and LTL invariants I1–I3. The framework connects agentic capabilities to enterprise governance by making approvals, fairness checks, and rollback readiness first-class, testable obligations rather than informal guidelines (Kareti, 2025; Ranjan *et al*., 2025; Huang, 2025) [6, 8, 3]. It also positions graded autonomy with k-of-n hysteresis as a systems primitive for operationalizing decision rights. These abstractions invite research on how control-plane guarantees shape planning, memory, and tool use in agentic AI (Sapkota *et al*., 2025) [9], and how reliability and resilience metrics should be composed for end-to-end workflows (Jin *et al*., 2025) [5].

5.2 Implications for practice

Enterprises can adopt the framework incrementally. Start in Assist mode for high-variance workflows, then raise autonomy as evidence accumulates under the k-of-n rule. Practitioners should externalize policy into the control plane, fail closed on ambiguity, and require evidence bundles for every sensitive action. Zero trust patterns - least privilege, isolation, continuous verification - should apply at perception, reasoning, and action time, not only at network boundaries (Inaganti and Sundaramurthy, 2020; Sundaramurthy and Ravichandran, 2022) [4, 10]. In commerce contexts, dual control for monetary actions and auditable reconciliation are non-negotiable controls that reduce blast radius while preserving speed.

5.3 Limitations

The work is architectural and does not report live trials. Parameter choices {τ, k, n, Δp, Δf, L*, T*, R*} are deployment-tuned and may vary by domain and risk tolerance. Fairness auditing depends on label availability and may require proxies or human review in some regions. The scope assumes controllable tool interfaces and enterprise identity infrastructure; highly open or consumer-grade ecosystems may need additional safeguards (Chennupati, 2025; Madireddy, 2025) [7, 1].

5.4 Future work

1. **Empirical validation and benchmarks:** Release a public replay harness that mirrors the evidence-bundle schema and evaluation plan in Section 3.6, including synthetic stressors for rare events and red-team templates (Jin *et al*., 2025) [5].
2. **Policy DSL and formal semantics:** Specify a commerce-focused policy language with deny-overrides precedence; prove refinement theorems that connect DSL rules to PEP enforcement and invariants I1–I3 (Kareti, 2025) [6].
3. **Calibration and abstention:** Study domain-specific methods for uncertainty calibration and selective abstention that optimize business KPIs subject to fairness and risk constraints, with pre-registered non-inferiority margins.
4. **Autonomy governance at scale:** Evaluate mode hysteresis across portfolios of workflows, including cross-workflow rollback dependencies and escalation playbooks.
5. **Security and zero trust automation:** Automate short-lived credential issuance, continuous verification, and least-privilege attestations as reusable control-plane services; measure overhead and leakage risks (Inaganti and Sundaramurthy, 2020; Sundaramurthy and Ravichandran, 2022) [4, 10].
6. **Risk catalogs and safety cases:** Extend hazard libraries for pricing, billing, fraud, and cross-border payments; standardize GSN patterns and evidence artifact types for audits (Chennupati, 2025; Madireddy, 2025) [7, 1].
7. **Human factors and accountability:** Define reviewer workloads, approval SLAs, and explanation formats that reduce cognitive burden while preserving accountability, especially for high-volume pricing and fraud queues (Gadde, 2025) [2].
8. **Privacy and residency:** Study privacy-preserving features and regional residency constraints that still enable reliable perception and fair decisions, with retention proofs keyed to deletion logs.
9. **Multi-agent and partner ecosystems:** Explore inter-agent protocols and cross-enterprise control-plane interoperability for marketplaces and payment networks (Ranjan *et al*., 2025; Huang, 2025) [8, 3].

In sum, the framework translates autonomy into enforceable, auditable behavior for enterprise commerce. By binding agentic AI to explicit policies, invariants, and evidence, it provides a practical basis for rigorous evaluation and safe industrial adoption.

6. Conclusion

This paper specifies a deployable framework for AI commerce agents that binds autonomy to enterprise governance. We separate execution into a data plane and enforcement into a control plane with PDP and PEP hooks, and we require evidence bundles on every sensitive path. We formalize safety through LTL invariants I1–I3 for dual control, policy checks at action time, and rollback readiness, and we operationalize decision rights via graded autonomy with k-of-n hysteresis. The framework maps directly to high-value workflows in billing, dynamic pricing, fraud, cross-border payments, and fulfillment, where financial eventing and fairness demand strict auditability. We provide measurable targets for transparency, accountability, security, fairness, resilience, and control-plane overhead, and we define an evaluation plan that pre-registers baselines, ablations, replay datasets, red-team suites, and statistical tests. While empirical validation remains future work, the controls, invariants, and evidence schema give enterprises a clear path to deploy agentic AI safely and to assess risk with rigor. By turning policies into enforceable mechanisms, the framework advances trustworthy autonomy from aspiration to practice in enterprise commerce.

References

1. Chennupati N. Securing the automated enterprise: mitigating security and privacy risks in AI workflows. Journal of Cybersecurity and Trusted Systems. 2025.
2. Gadde A. AI agents: the autonomous workforce for automating workflows. WJAETS. 2025.
3. Huang K. AI Agents and Business Workflow. Springer; 2025.
4. Inaganti S, Sundaramurthy B. Zero trust to intelligent workflows: redefining enterprise security and operations with AI. AIMLR. 2020.
5. Jin X, *et al*. A review of AI-driven automation technologies: latest taxonomies, challenges, and prospects. Journal of Artificial Intelligence Research & Development. 2025.
6. Kareti PSR. Trust architecture for enterprise AI assistants: technical mechanisms for transparency and security. WJAETS. 2025.
7. Madireddy RR. Security implications of fully autonomous process agents in enterprise workflows. Journal of Cybersecurity and Trusted Systems. 2025.
8. Ranjan S, *et al*. Agentic AI in Enterprise. Springer; 2025.
9. Sapkota S, *et al*. AI agents vs. agentic AI: a conceptual taxonomy. arXiv preprint. 2025.
10. Sundaramurthy B, Ravichandran R. The future of enterprise automation: integrating AI in cybersecurity, cloud operations, and workforce analytics. AIMLR. 2022.

Appendix A:

```
// Appendix A: NuSMV model for I1–I3
MODULE main
VAR
state: {Assist, Constrained, Conditional, Full};
exec_monetary: boolean;
exec_sensitive: boolean;
approved: boolean;
policy_pass: boolean;
rollback_ready: boolean;

Assign
init(state):= Assist;
next(state):= case
state = Assist & approved: Constrained;
state = Constrained & policy_pass: Conditional;
state = Conditional & policy_pass: Full;
-- violation/incident returns to previous mode
state = Constrained & !policy_pass: Assist;
state = Conditional & !policy_pass: Constrained;
state = Full & !policy_pass: Conditional;
TRUE: state; esac;

-- Invariants to check
LTLSPEC G (exec_monetary -> F approved); -- I1
LTLSPEC G (exec_sensitive -> X policy_pass); -- I2
LTLSPEC G (exec_monetary -> rollback_ready); -- I3
```