

Lex Autonoma

A Structural Framework for Delegated Digital Agency

Introducing the Delegated Agency Framework (DAF) and Machine-Legible Legal Primitives (MLLPs)

Gabriel Brănescu
November 2025

GABRIEL@EIKON7.COM

Abstract

Autonomous AI agents now negotiate, select, and purchase on behalf of human users, yet the institutional substrate required for them to function as legitimate market participants does not exist. Agents lack recognized identity, enforceable authority, and shared interaction protocols. Retailers refuse to share data with entities that cannot be held accountable, payment networks cannot authorize transactions initiated by actors without legally defined standing, and liability for autonomous actions remains undefined – making the ecosystem uninsurable and therefore unscalable.

This paper argues the problem is structural, not technical. Agents are capable; the environment around them is incoherent. **Lex Autonoma** identifies the minimal architectural components any viable agent economy will require, and defines an implementation-ready standard: the **Delegated Agency Framework (DAF)**, built on three irreducible pillars – **Synthetic Identity**, **Delegated Agency**, and **Contractual Physics** – supported by **Machine-Legible Legal Primitives (MLLPs)** that translate legal intent into executable protocol logic.

The goal is to establish a common vocabulary and interoperable substrate for agentic transactions that are verifiable, revocable, and jurisdiction-compliant. By specifying identity classes, delegation scopes, interaction rules, liability models, and protocol flows, the framework describes not what agent economies might become, but what they must become if they are to function at all.

1. The Structural Void

In August 2025, Amazon sent a cease-and-desist letter to Perplexity AI demanding that the startup stop allowing its browser agent, Comet, to make purchases on behalf of users. The e-commerce giant accused Perplexity of committing computer fraud by failing to disclose when an AI agent was shopping instead of a human. Perplexity's response was revealing: its CEO argued that agents should have "all the same rights and responsibilities" as the humans who deploy them, and that distinguishing between a user and an agent acting on their behalf serves no legitimate purpose.

This clash captures the central tension of the emerging agent economy. Autonomous systems can already search, negotiate, select, and purchase. The technical capability exists. What does not exist is any coherent framework for determining what these agents are, what they may do, and who bears responsibility when things go wrong. Amazon's position – that agents must operate openly and respect platform decisions – assumes a regime of permissions and

disclosures that has never been codified. Perplexity's position – that agents inherit user rights automatically – assumes a theory of delegation that no legal system recognizes.

Both positions are reasonable. Both are also inadequate. The dispute is not really about Amazon's terms of service or Perplexity's product design. It is about the absence of any shared understanding of what autonomous agents are in transactional contexts. Are they tools? Extensions of users? Independent actors? Proxies with limited authority? The answer matters because it determines everything downstream: what data they may access, what commitments they may make, who is liable for their mistakes, and how disputes involving them should be resolved.

The current answer is silence. No jurisdiction has established a coherent legal identity class for autonomous agents. No industry consortium has standardized how agents should identify themselves, what permissions they require, or how their commitments bind the humans they represent. No insurance framework exists for underwriting autonomous decisions. The result is a marketplace stuck in primitive bilateral negotiations, with every platform, every agent developer, and every retailer reinventing the rules from scratch – and arriving at incompatible conclusions.

1.1 The Current Impasse

The stagnation is visible across the industry. Major technology companies – OpenAI, Google, Amazon, Microsoft – are all developing agentic systems capable of autonomous commerce. Yet none have achieved meaningful deployment. The obstacles are consistent across all players, and they are not technical.

Retailers refuse to share real-time inventory, pricing, and logistics data with agents because the agents have no recognized standing. Why expose sensitive commercial information to an entity that cannot be held accountable for misusing it? Payment networks cannot authorize agent-initiated transactions without clear identity verification, and no standard exists for verifying agent identity. Users report that agent shopping experiences are degraded – slow, error-prone, requiring constant confirmation – precisely because the systems are designed conservatively to avoid liability in an uncharted legal environment.

Industry surveys suggest that fewer than one in five consumers would currently trust an AI agent to handle purchases autonomously. This distrust is rational. Without clear accountability structures, every autonomous transaction is a gamble. If the agent buys the wrong product, overpays, misses a delivery window, or is manipulated by an adversarial vendor, the consumer has no reliable path to remedy. The agent's developer will disclaim responsibility for "user preferences." The retailer will disclaim responsibility for "third-party actions." The payment network will disclaim responsibility for "authorized transactions." Everyone points elsewhere because no one has been assigned responsibility.

Meanwhile, early experiments reveal disturbing failure modes. Research on multi-agent marketplaces shows that speed consistently beats quality – agents select whichever vendor responds first, regardless of price, quality, or fit. This creates a race-to-the-bottom dynamic where vendors optimize for latency rather than value. It also opens exploit vectors: adversarial actors can manipulate agent decisions through timing attacks, crafted offers that look legitimate but contain hidden disadvantages, and sophisticated forms of misdirection that human shoppers would catch but autonomous systems miss.

These are not growing pains. They are symptoms of a structural void. The technology works. The environment does not.

1.2 Why Patchwork Solutions Fail

The instinctive response to these problems is bilateral negotiation. Amazon negotiates with Perplexity. OpenAI negotiates with retailers. Google negotiates with payment processors. Each company attempts to establish its own rules, its own permissions, its own liability arrangements.

This approach cannot scale. Every bilateral agreement creates a silo. An agent authorized to shop on Amazon under Amazon's terms cannot automatically shop on Walmart under Walmart's different terms. A permission structure negotiated between OpenAI and Visa does not transfer to Anthropic and Mastercard. The result is fragmentation: a thousand micro-standards stitched together with bespoke integrations, each one representing months of legal and technical work, each one incompatible with the others.

Fragmentation defeats the purpose of autonomous agents. The value proposition of agentic commerce is that agents can operate across vendors, platforms, and contexts seamlessly – finding the best price, the best product, the best delivery option regardless of where it lives. An agent trapped within one ecosystem's permission structure is not autonomous; it is a slightly enhanced search function. The whole point is interoperability, and interoperability requires shared rules.

The alternative – waiting for courts or regulators to impose rules from above – is equally inadequate. Legal systems move slowly. The agent economy is moving fast. By the time legislators understand the problem well enough to address it, the industry will have calcified around whatever ad hoc arrangements emerged in the interim. Worse, top-down regulation tends to be heavy-handed, imposing compliance burdens that strangle innovation rather than enabling it. What is needed is not regulation in the traditional sense but infrastructure: a minimal, neutral, universal protocol layer that any agent and any platform can adopt without surrendering competitive advantage.

This is the gap Lex Autonoma fills.

2. The Three Pillars of the Delegated Agency Framework

Lex Autonoma is not a policy proposal. It is a diagnostic framework identifying the minimal structural components that any viable agent economy will require. The claim is not that these components would be nice to have, or that they represent one possible approach among many. The claim is stronger: without these components, agent economies cannot function at scale. They will remain stuck in the fragmentation currently observed, or they will collapse into exploitation and distrust.

The framework rests on three pillars: Synthetic Identity, Delegated Agency, and Contractual Physics. Each addresses a specific structural gap. Each is irreducible – remove any one and the system fails. Together, they provide the minimum viable substrate for autonomous agents to operate as legitimate market participants.

2.1 Synthetic Identity

The first pillar addresses the most fundamental question: Who is acting?

An autonomous agent without identity is not an actor in any meaningful sense. It is a ghost – something that produces effects but cannot be addressed, verified, held accountable, or revoked. Retailers cannot contract with it because contracts require identifiable parties. Payment networks cannot authorize it because authorization requires knowing whom you are authorizing. Other agents cannot recognize its commitments because commitments must come from someone. Disputes cannot be resolved because disputes require parties.

Synthetic Identity provides the solution: a distinct, verifiable identity class for non-human actors. This is not a proxy for human identity, nor is it a grant of legal personhood. It is something new – a constrained, protocol-defined entity with its own cryptographic credentials, permissions, and audit trails. The identity determines what an agent may access, what it may commit to, and how its commitments are interpreted. It makes the agent addressable without making it equivalent to a human.

The core elements of Synthetic Identity are minimal. Each agent receives a cryptographic identifier – a public key or derived handle – that uniquely identifies it across all interactions. This identifier is linked to an agent class that determines its capabilities and constraints. The identity includes metadata about its issuer (who created it and vouches for its behavior), its validity period, and its revocation status. Every action the agent takes is signed with its private key, creating an audit trail that links decisions to the deciding entity.

Crucially, Synthetic Identity separates existence from capability. Having an identity does not automatically grant permissions; it simply makes the agent visible to the system. Permissions flow from the second pillar, Delegated Agency. Identity is the floor – the minimum requirement for participation. Everything else builds on it.

Synthetic Identity is not personhood.

It grants *operational standing*, not moral or legal autonomy. A synthetic actor becomes *addressable*, *verifiable*, and *accountable*, but it does not acquire the inherent rights, protections, or status of a human or legal person. It is a constrained protocol entity defined entirely by its issued credentials and delegation scopes. This distinction matters: agents can act, be trusted, and be held liable **without** being elevated to the category of persons. DAF requires addressable actors, not philosophical subjects.

2.2 Delegated Agency

The second pillar addresses the question: What may this agent decide?

Identity alone is not enough. An agent with verified identity but unlimited authority is dangerous. An agent with verified identity but no clear authority is useless. What is needed is a structured way to define the boundaries of autonomous action – a map of what the agent is allowed to do, under what conditions, and with what constraints.

Delegated Agency provides this structure. It is a contract between a human principal and a synthetic actor specifying the limits of autonomy. The delegation includes scope levels (what categories of action are permitted), constraints (budget limits, vendor restrictions, timing requirements), duration (how long the delegation remains valid), and revocation rules (under what circumstances the delegation terminates).

The framework defines four scope levels that form a gradient of autonomy. At the lowest level, "assist," the agent may only provide information and recommendations – it cannot take any action. At the "propose" level, the agent may draft actions for human review but cannot execute them. At the "act" level, the agent may execute actions within defined constraints

without prior approval. At the highest level, "bind," the agent may enter into commitments on behalf of the human that create legal obligations.

This gradient is essential because it allows principals to calibrate autonomy to trust. A new agent might start at "propose" level until it demonstrates reliability, then be elevated to "act" for routine purchases, with "bind" authority reserved for high-stakes decisions. The gradient also provides a natural failure mode: when an agent encounters ambiguity or uncertainty, it can collapse to a lower scope level rather than proceeding with potentially erroneous autonomous action.

"Bind" does not grant blanket authority.

Binding authority is always **domain-limited**, **object-limited**, and **explicitly enumerated** inside the delegation object. A principal may authorize an agent to bind only within specific categories (e.g., retail purchases, travel bookings, procurement items) while prohibiting other forms of binding commitments such as subscriptions, recurring payments, long-term service contracts, or obligations that extend beyond the defined duration of the delegation.

Binding authority therefore decomposes into three nested constraints:

1. **Domain Constraint** – the class of transactions for which binding is permitted (e.g., consumer purchases vs. financial instruments vs. contractual agreements).
2. **Commitment-Type Constraint** – the shapes of commitments allowed (one-time purchase, reservation, order placement; but not subscription initiation, automatic renewal, or multi-year obligations unless explicitly stated).
3. **Temporal Constraint** – the maximum duration any binding commitment may impose, which can never exceed the lifespan of the delegation itself.

An agent with bind authority within one domain does not inherit bind authority in any other domain. Bind is **not** a general power-of-attorney; it is a narrowly scoped permission to create legally enforceable obligations **only** within the explicit boundaries the principal defines.

Delegated Agency is explicitly **zero-trust**. The framework assumes that an agent's identity, history, and past performance do **not** grant it implicit authority of any kind. Every permission must be **explicit**, **constrained**, and **cryptographically bound** to a specific delegation object. The default state is **deny**; authority flows only from formally declared scopes and constraints.

Identity does not generate trust. Identity only allows the system to *recognize who is making the request*. What the agent may actually do is determined entirely by its delegation object, and constraints always override identity-level assumptions. Zero-trust delegation ensures that autonomous systems remain compatible with high-regulation environments – financial, medical, legal, procurement – where implicit permissions are unacceptable and where any ambiguity must collapse toward safety, not autonomy.

Delegated Agency solves the overreach dilemma observed in current systems. Without explicit scopes, agents oscillate between acting too freely (making unauthorized commitments) and acting too timidly (requesting confirmation for every trivial decision). Both behaviors destroy trust. With explicit scopes, agents know exactly when to act, when to ask, and when to stop. Retailers and platforms also know what they are dealing with – an agent authorized to "act" can be treated differently than one limited to "propose."

2.3 Contractual Physics

The third pillar addresses the question: How do agents interact with each other in a predictable way?

Human markets rely on centuries of accumulated norms, customs, institutions, and legal precedents to function. When a human buyer and human seller negotiate, they draw on shared understandings of what offers mean, how timing works, when commitments become binding, and how disputes get resolved. These understandings are rarely explicit, but they provide the background physics that makes economic interaction coherent.

Agents have none of this. They are born without culture, without precedent, without implicit understandings. If two agents are to negotiate effectively, the rules of negotiation must be spelled out explicitly in a form they can interpret and follow. This is what Contractual Physics provides: the minimal rule-set that allows markets of synthetic actors to remain coherent at scale.

The physics includes standardized offer formats so that agents can compare options without parsing incompatible data structures. It includes timing windows that prevent latency exploitation – a minimum response time before which no offer can be accepted, ensuring that quality has a chance to compete with speed. It includes truthfulness attestations that require vendors to sign their claims, creating accountability for misrepresentation. It includes arbitration channels for handling disputes, liability classes for assigning responsibility when things go wrong, and fallback mechanisms for graceful failure. And it includes commitment physics – precise rules governing when a selected offer becomes a binding obligation – so that all parties operate on the same temporal substrate and no commitment can form prematurely, ambiguously, or outside the fairness windows established by the timing layer.

The term "physics" is deliberate. These are not optional best practices or voluntary guidelines. They are the mechanical rules that make coherent interaction possible. Without them, agent marketplaces degenerate into chaos – latency wars, exploit races, offer manipulation, and trust collapse. With them, agents gain the stability required for negotiation, cooperation, and competition at scale.

2.4 Why These Pillars Are Irreducible

The architecture of Lex Autonoma is intentionally sparse. Each pillar exists only because removing it collapses the entire system.

Remove Synthetic Identity and agents become ghosts – capable of action but unaddressable, unaccountable, unrevocable. No platform will trust them with data, no payment network will authorize them, no market can form around them.

Remove Delegated Agency and agents oscillate between chaos and inaction. Without explicit scopes, they either overstep (making unauthorized commitments) or understep (constantly seeking permission). Either behavior makes them useless for autonomous operation.

Remove Contractual Physics and multi-agent interaction becomes a war of all against all. Speed beats quality. Exploits proliferate. Trust collapses. Markets degenerate into adversarial gaming rather than value creation.

The three pillars interlock in a specific way. Synthetic Identity anchors Delegated Agency: you cannot define scope without knowing which entity it binds. Delegated Agency constrains

the behavior enforced by Contractual Physics: commitments are only valid if made within scope. Contractual Physics provides outcomes that feed back into identity and scope: winning bids, disputes, revocations, and liability assignments all update the agent's standing in the system. The three form a feedback loop: identity enables agency; agency shapes interactions; interactions modify standing.

3. Machine-Legible Legal Primitives

Cutting across all three layers are Machine-Legible Legal Primitives – standardized protocol components that translate legal intent into executable logic. MLLPs are the building blocks from which DAF is constructed.

At the identity layer, MLLPs define identity certificates, signature formats, revocation tokens, and attestation structures. At the agency layer, they define delegation objects, scope specifications, constraint schemas, and revocation rules. At the physics layer, they define offer formats, timing rules, arbitration protocols, and liability class assignments.

The key property of MLLPs is that they are both legally meaningful and technically executable. A delegation MLLP, for example, is not merely a data structure; it is a specification that a court could interpret if necessary, that an insurance underwriter could evaluate for risk, and that a software system could enforce automatically. This dual nature – legal and technical simultaneously – is what makes DAF operational rather than merely conceptual.

MLLPs are designed for composability. Simple primitives combine to form complex arrangements. A sophisticated agentic transaction might involve dozens of MLLPs – identity verification, delegation transfer, offer submission, timing validation, commitment finalization, liability assignment – but each primitive is standardized and reusable. This modularity enables both interoperability (different platforms using the same primitives) and evolvability (new primitives can be added without disrupting existing ones).

4. A Taxonomy of Synthetic Actors

Lex Autonoma does not assume one kind of agent. It assumes a marketplace where different kinds of actors – human and synthetic – coexist, collaborate, and occasionally compete. To avoid ambiguity, every actor belongs to one (and only one) class. Each class carries its own rights, responsibilities, and operational constraints.

4.1 Human Principals

Human Principals are the source of delegation. They authorize, revoke, modify, and override synthetic actors. Their rights cannot be superseded, inherited, or transferred to non-humans. They may issue delegation scopes, own assets and preferences, define revocation rules, and override commitments within allowed windows. Humans remain liable for what they explicitly authorize.

4.2 Synthetic Agents

Synthetic Agents are autonomous software entities acting within a defined delegation scope. They may perceive structured data, negotiate within contractual physics, commit within scope, execute actions on behalf of humans, and interact with other synthetic actors. Their responsibilities include honoring scope boundaries, complying with timing, format, and

truthfulness requirements, maintaining audit trails, and accepting liability classes appropriate to their actions.

4.3 Supervised Agents

Supervised Agents are Synthetic Agents operating under "propose" scope only – they cannot commit without human approval. They exist for onboarding (new users learning the system), high-stakes domains (medical, legal, financial), and regulatory compliance (jurisdictions requiring human-in-the-loop). They participate fully in contractual physics but cannot finalize commitments. Every proposed action requires explicit human confirmation before execution.

4.4 Platform Agents

Platform Agents are infrastructure actors that do not participate in commercial transactions but enable others to do so. They include identity registrars (who issue and revoke synthetic identities), arbitration services (who resolve disputes), escrow services (who hold commitments pending fulfillment), and registry operators (who maintain authoritative records). Platform Agents have special privileges (they can suspend identities, freeze commitments, issue rulings) and special obligations (they must remain neutral, maintain audit trails, and submit to oversight). They are the referees of the system, not participants in the game. Platform Agents may be operated by **human institutions**, **autonomous systems**, or **hybrid human–AI organizations**, provided they implement the protocol's invariants faithfully. DAF does not assume that governance or arbitration requires human judgment, nor does it assume fully automated operation. What matters is functional compliance: identity issuance, revocation, arbitration replay, and physics enforcement must follow the same deterministic rules regardless of who or what operates the service. The protocol remains agnostic – only behavior, not composition, defines a valid Platform Agent.

4.5 Vendor Agents

Vendor Agents represent sellers in the marketplace. They may receive requests for offers, respond with structured offers, provide product data and delivery terms, and fulfill commitments. Their responsibilities include truthfulness (they are liable for misrepresentation), timing compliance (they must respond within protocol windows), and fulfillment (they are accountable for delivery failures). Vendor Agents bridge the human vendor world with the synthetic marketplace; their commitments create obligations for their human operators.

5. Interaction Rules and Invariants

The framework establishes rules that govern how classes interact. These are not guidelines but invariants – conditions that must hold for the system to function.

5.1 Core Invariants

Identity Requirement: No action without a verified synthetic identity. Every message, every commitment, every query must be signed. Anonymous participation is not participation.

Scope Constraint: Agents may not exceed their delegated scope. An agent with "act" authority cannot enter "bind" commitments. Scope violations invalidate commitments and trigger liability.

Human Override: Human Principals may override, pause, or revoke any agent at any time within protocol constraints. Override windows must be respected; once a commitment has cleared its window, it becomes binding.

Timing Compliance: All participants must honor timing windows. Early actions (before minimum response time) are invalid. Late actions (after maximum response time) are void. Timing is not a suggestion; it is physics.

Audit Trail: Every state-changing action must be logged with cryptographic signatures. Disputes are resolved by replaying the audit trail, not by testimony.

No Hidden State: Agents may not base decisions, commitments, or negotiations on private, unlogged memory. All state that influences externally visible actions must flow through the audit trail or the formal message sequence. If an agent uses internal reasoning, predictions, or models, the *inputs* to those processes must still be the logged protocol state. This prevents a class of failures where agents rely on private caches, stale snapshots, or non-replayable inferences that cannot be reconstructed during arbitration. “No Hidden State” makes every action **replayable**, **verifiable**, and **auditable**, ensuring that disputes can be resolved solely from the protocol log. It is essential for multi-agent coordination, where hidden state would otherwise become a vector for exploitation, ambiguity, or undetectable protocol deviation.

5.2 Interaction Matrix

The framework specifies permissible interactions between classes. Human Principals may delegate to Synthetic Agents, receive reports from all agent types, override any agent they control, and appeal to Platform Agents. Synthetic Agents may negotiate with Vendor Agents, submit queries to Platform Agents, coordinate with other Synthetic Agents (if scope permits), and escalate to Human Principals. Platform Agents may issue identities, resolve disputes, freeze suspicious activity, and enforce protocol rules. Vendor Agents may respond to queries, make offers, fulfill commitments, and contest disputes.

5.3 Commitment Lifecycle Rules

Every commitment created under the Delegated Agency Framework must follow a complete and auditable lifecycle. This prevents ambiguous states, partial execution, silent failures, and irreconcilable disputes. A valid commitment proceeds through the following stages, each recorded in the audit trail:

1. Initiation

The Synthetic Agent generates a commitment proposal, referencing its delegation watermark and the offer it intends to accept.

2. Acceptance

The counterparty (typically a Vendor Agent) explicitly accepts the commitment in a signed message.

No implicit acceptance is permitted.

3. Proof-of-Acceptance

Both parties sign a canonical commitment object, forming a mutual attestation that the commitment is now active.

This object becomes the authoritative reference for arbitration.

4. **Fulfillment**

The Vendor Agent performs the obligated action (delivery, service provisioning, transfer, etc.) and issues a fulfillment attestation.

Partial fulfillment must be represented explicitly; silence is not fulfillment.

5. **Closure**

The buying agent verifies fulfillment and issues a closure acknowledgement.

Once closed, the commitment is immutable and cannot re-enter dispute unless fraud is proven.

6. **Expiry**

If fulfillment or closure does not occur within the defined validity window, the commitment transitions automatically to an expired state.

Expired commitments cannot be retroactively activated.

7. **Dispute State**

Any party may transition a commitment into dispute state by submitting a dispute object.

Entering dispute freezes all downstream actions until arbitration resolves the status.

These lifecycle rules make commitments **deterministic, replayable, and insurable**.

Every economic system already relies on implicit versions of this lifecycle; DAF makes these expectations explicit and machine-enforceable.

6. Failure Modes the Framework Prevents

The framework is designed to prevent specific failure modes already observed in early agent deployments.

6.1 Ghost Actors

Without Synthetic Identity, agents act without being addressable. They cannot be held accountable, revoked, or identified in disputes. DAF requires verified identity for every participant; ghost actors cannot participate.

6.2 Scope Creep

Without explicit delegation, agents gradually expand their authority. A shopping assistant becomes a financial advisor becomes an investment manager. DAF bounds this: agents operate within declared scope, and scope changes require human authorization.

6.3 Latency Exploitation

Without timing rules, speed determines selection. Vendors optimize for latency rather than quality, and flash-offer exploits proliferate. Contractual Physics establishes minimum response windows, ensuring quality can compete.

6.4 Truthfulness Erosion

Without attestation requirements, misrepresentation proliferates. Vendors make claims they cannot substantiate. DAF requires signed truthfulness attestations, creating liability for false claims.

6.5 Dispute Deadlock

Without arbitration channels, disputes cannot be resolved. Each party blames the other; no mechanism exists for resolution. DAF provides deterministic arbitration: disputes are resolved by replaying the audit trail against protocol rules.

6.6 Silo Fragmentation

Without standardized protocols, every platform-agent relationship requires bespoke negotiation. The ecosystem fragments into incompatible silos. MLLPs provide the common vocabulary that enables interoperability.

7. The Security Model

Security in DAF is not an add-on; it is embedded in the architecture. Every layer has security properties that derive from its structure.

7.1 Identity Layer Security

Every agent identity is bound to a cryptographic keypair. Actions are signed with private keys; verification uses public keys. Identity revocation is immediate and propagates through the system. Compromised keys can be revoked by the issuing authority or the controlling principal. Identity spoofing is cryptographically impossible given standard assumptions.

7.2 Delegation Layer Security

Delegation objects are signed by principals and cannot be forged or modified. Scope boundaries are enforced by the protocol; agents cannot self-elevate. Delegation chains are traceable: any commitment can be traced back through the delegation graph to its authorizing human. Revocation is recursive: revoking a principal's delegation also revokes all downstream delegations.

7.3 Physics Layer Security

Timing windows are enforced cryptographically; timestamp manipulation is detectable. Offer formats include integrity checksums; tampering invalidates offers. Commitment sequences are hash-chained; replay attacks fail. Arbitration decisions are based solely on the cryptographic audit trail; unlogged events cannot be adjudicated.

7.4 Threat Model

DAF assumes adversarial participants. Malicious agents will attempt to exceed scope, spoof identity, manipulate timing, and exploit ambiguity. Malicious vendors will misrepresent products, exploit latency advantages, and contest valid claims. Malicious platforms will attempt to advantage preferred participants. The security model does not rely on good behavior; it assumes bad behavior and makes it unprofitable or impossible.

7.5 Threat Scenarios

Autonomous agents operate in adversarial environments, and early experiments already reveal predictable exploit classes. The Delegated Agency Framework is designed around these failure patterns, but it is useful to make the underlying threats explicit.

Timing Exploit

Adversarial vendors or agents manipulate ultra-fast responses to win commitments before higher-quality offers can arrive. Without enforced minimum response windows, markets collapse into latency arbitrage rather than value competition.

Scope Escalation

An agent attempts to exceed its delegated authority – acting under “propose” scope while executing binding commitments, or interpreting ambiguous instructions as implicit permission. Without strict scope validation, autonomy drifts upward over time.

Delegation Forgery

An attacker creates or modifies delegation objects, granting unauthorized authority to an agent. Without signature-bound, versioned delegation structures, forged delegations become indistinguishable from legitimate ones.

Vendor-Latency Gaming

Vendors shape offers to appear compliant while hiding adverse conditions in time-sensitive fields (expiry times, delivery windows, fulfillment dependencies). Without canonical offer formats and truthfulness attestations, agents cannot reliably evaluate tradeoffs.

Ghost Actor Infiltration

An entity without synthetic identity participates in request or negotiation flows, harvesting data or injecting manipulative offers. Without identity-gated participation, the ecosystem becomes vulnerable to anonymous adversaries.

Arbitration Denial

Platforms or adversarial agents suppress, delay, or manipulate dispute initiation or adjudication. Without mandatory audit trails and deterministic replay rules, disputes devolve into testimony and platform discretion.

Identity Compromise

An attacker acquires an agent’s private key and impersonates it across all interactions. Without immediate revocation propagation and strict key lifecycle controls, identity compromise becomes systemic failure rather than contained incident.

These scenarios are not theoretical. They have emerged repeatedly in early multi-agent experiments. DAF’s structural safeguards – identity gating, delegation validation, timing physics, attestations, and tamper-evident logs – exist precisely because these are the default failure modes of autonomous coordination.

7.6 Privacy Model

Autonomous agents must negotiate, sign, and commit without leaking sensitive information about the humans or organizations they represent. DAF therefore builds privacy directly into identity, delegation, and interaction layers. Synthetic Identities are pseudonymous by default: they reveal only the minimal operational attributes (class, issuer, validity) required for participation. Principals may rotate identities freely, and delegation objects never expose personal data – only constraints and scope.

The framework supports **selective disclosure**. Agents reveal only the subset of delegation constraints or identity attestations necessary for a given interaction. Vendor Agents, for

example, can validate that a buyer has sufficient authority for a purchase without learning anything about the buyer's broader preferences or identity context. Each MLLP explicitly states which fields must be visible to counterparties and which must remain opaque.

To prevent behavioral de-anonymization, DAF enforces **non-linkability between commitments**. Separate commitments cannot be correlated unless the principal or platform explicitly authorizes linkage. Identity revocation is also privacy-preserving: the system can invalidate a keypair without revealing which principal controlled it or why it was revoked. Taken together, these guarantees ensure that agentic markets can scale without becoming mass surveillance surfaces.

8. The Liability Model

Liability in autonomous systems is not a legal detail; it is a structural requirement. Without clear liability assignment, no insurance can be written, no risk can be priced, and no trust can form. DAF provides a liability model that makes agent economies insurable.

8.1 Liability Classes

The framework defines five liability classes that exhaust the space of possible failures.

LC1: Principal Fault. The human principal issued delegation that authorized the problematic action. The agent acted within scope. Liability rests with the principal. Example: User authorizes agent to purchase "any suitable laptop under \$2000." Agent purchases a laptop that the user dislikes. The agent acted within scope; the principal bears the consequence of broad delegation.

LC2: Agent Fault. The agent exceeded its delegated scope or violated protocol rules. Liability rests with the agent's operator (typically its developer or deploying organization). Example: Agent authorized for "propose" scope finalizes a purchase without human approval. The scope violation makes the operator liable.

LC3: Vendor Fault. The vendor misrepresented products, failed to deliver, or violated truthfulness attestations. Liability rests with the vendor. Example: Vendor's agent claims product is "in stock" when it is not. The signed attestation creates liability regardless of intent.

LC4: Platform Fault. The platform operator failed to enforce protocol rules, allowing harm that proper enforcement would have prevented. Liability rests with the platform. Example: Platform fails to enforce timing windows, enabling a flash-offer exploit. The platform's negligence created the vulnerability.

LC5: Systemic Fault. The failure resulted from conditions no party could reasonably have prevented or foreseen – network failures, coordinated attacks, regulatory interventions. Liability is distributed according to predefined loss-sharing rules or absorbed by system-wide insurance pools.

8.2 Liability Determination

When a failure occurs, the arbitration module determines liability class by replaying the audit trail. It asks: Did the agent act within scope? Were protocol rules followed? Were attestations accurate? Was the platform functioning correctly? The answers determine the class. Liability assignment is deterministic given the audit trail; there is no room for interpretation.

9. Core Protocol Flows

The framework defines canonical protocol flows for common interactions. These are not exhaustive but demonstrate how the components combine in practice.

9.1 Simple Purchase Flow

The Human Principal issues a delegation to a Synthetic Agent with "act" scope, budget constraints, and product requirements. The Synthetic Agent broadcasts a request-for-offer to eligible Vendor Agents, including its identity, delegation reference, and requirements. Vendor Agents respond with structured offers including price, delivery terms, and truthfulness attestations. After the minimum response window closes, the Synthetic Agent evaluates offers and selects the best match within delegation constraints. The Synthetic Agent submits a commitment to the selected Vendor Agent; both sign the commitment. The Platform Agent logs the commitment and initiates fulfillment. If fulfillment succeeds, the transaction closes. If it fails, the dispute protocol engages.

9.2 Dispute Resolution Flow

Any party may initiate a dispute by submitting a dispute object to the Platform Agent. The dispute object references the contested transaction, the claimed violation, and the requested remedy. The Platform Agent freezes the relevant commitments and retrieves the complete audit trail. The arbitration module replays the trail against protocol rules, determining what happened and which rules were violated. The module assigns liability class based on the analysis. The ruling is published with cryptographic proof that it follows from the audit trail. Parties may appeal within defined windows; appeals trigger re-adjudication by a different module or escalation to human arbitrators.

9.3 Delegation Transfer Flow

A Human Principal wishes to transfer a delegation from Agent A to Agent B. The principal issues a revocation for Agent A's delegation. The principal issues a new delegation to Agent B with desired scope. Agent A receives the revocation and ceases operation. Agent B receives the delegation and assumes responsibility. Outstanding commitments made by Agent A remain valid; Agent B does not inherit them unless explicitly transferred. The audit trail records the transfer, maintaining continuity of accountability.

10. Implementation Pathways

Lex Autonoma is not meant to be implemented from scratch. It is designed to be adopted incrementally by existing systems. DAF does not require new cryptographic inventions or exotic infrastructure. It maps cleanly onto existing public key systems: Synthetic Identities are ordinary keypairs with strict metadata schemas, and revocation uses standard certificate-status mechanisms. The delegation engine mirrors established permission models such as OAuth and API-scoped tokens, but formalizes them into cryptographically bound, replayable delegation objects instead of ad hoc developer conventions.

On the interaction side, the physics layer aligns with existing commerce infrastructure. Canonical offer formats resemble EDI and vendor API schemas, but with deterministic timing rules and signed truthfulness attestations. The audit and arbitration layers use tamper-evident append-only logs – functionally similar to compliance logs used in finance – without

requiring blockchains or consensus protocols. DAF slots into today's systems with minimal friction; it adds structure, not ideological reinvention.

The framework identifies four layers that can be implemented independently, with each layer providing value even without the others.

10.1 Identity Infrastructure

The identity layer requires public/private keypairs for each agent, a certificate format binding keys to class and owner, and a revocation registry for compromised or retired identities. The minimal data model includes agent identifier (public key or derived handle), agent class, issuer, validity period, and revocation status. Implementation can adopt existing PKI or decentralized identity infrastructure – there is no need to reinvent cryptography. What matters is defining a Lex Autonoma Identity Profile: a small, strict schema for agent identities that all protocol messages must reference through signatures.

10.2 Delegation Engine

The delegation engine transforms informal instructions ("help me shop") into enforceable constraints. Delegation objects include principal identifier, agent identifier, scope level, constraints, duration, and revocation rules. The engine implements a scope evaluator – a service that answers one question: "Given this delegation and this action, is this action allowed?" Inputs are the delegation object, the requested action type, and current context (budget spent, items committed, time elapsed). Output is allow, allow with confirmation, or deny. Every agent call passes through this service before binding anything.

10.3 Physics Engine

The physics engine normalizes interaction so markets do not become a mess of bespoke APIs. It defines canonical offer formats with required fields: offer identifier, vendor identifier, request identifier, price (currency and amount), quantity, delivery terms, validity window, constraints, and truthfulness attestation. It defines request-for-offer formats with request identifier, buyer agent identifier, delegation reference, requirements, and timing window. It enforces timing rules: minimum response time before acceptance, maximum response time after which offers expire, and clock synchronization. A physics gateway sits in front of all vendor-agent interaction; everything passes through it.

10.4 Audit and Arbitration Layer

The audit layer requires an immutable log: append-only, hash-chained entries containing actor identifiers, message type, timestamp, signatures, and references to previous messages. This does not require blockchain ideology; it requires tamper-evident logging. The arbitration module accepts disputed objects, replays protocol steps from the log, applies deterministic rulesets, and outputs rulings. The liability engine implements the five liability classes and, given a failed event plus full log, outputs exactly one class. The layer provides APIs for agents to initiate arbitration, platforms to request rulings, and regulators or insurers to audit behavior.

10.5 Adoption Path

A realistic adoption path for an existing agentic shopping stack proceeds incrementally. First, assign Lex Autonoma identities to human users, shopping agents, vendor connectors, and the marketplace. Second, replace informal user preferences with formal delegation objects; the

shopping agent refuses to act without valid delegation. Third, insert a physics gateway so all requests-for-offer and responses go through the contractual physics engine. Fourth, turn on logging and arbitration – every step logged, simple arbitration service operational. Fifth, tighten incrementally: start with scope limited to propose, gradually elevate to act and bind as the system proves stable.

11. Design Principles

The Delegated Agency Framework rests on five principles. They are not ideological. They are not ethical. They are mechanical truths about how autonomous systems must interact if they are to interact at all.

Identity Before Influence. No entity may act, request, commit, or negotiate without a verifiable synthetic identity. Anonymous agency is noise masquerading as intention. Identity is the price of entry; everything else is optional.

Autonomy Flows Down the Gradient of Delegation. Agents act only within the scope delegated to them – no more, no less. Permission is not inferred. Silence is not consent. Ambiguity collapses to the lowest-authority action.

Contracts Are Physics, Not Negotiation. For synthetic actors, rules of interaction cannot depend on norms, customs, or goodwill. They must be mechanical: formats, timings, disclosures, arbitration rules, and liability classes. Markets work because physics works. Contractual Physics is the operating system of agent economies.

Transparency Without Exposure. Agents must disclose what is necessary for trust and nothing that compromises security or privacy. No party receives privileged access. No platform becomes a surveillance bottleneck. Trust flows from structure, not from unilateral data surrender.

Failures Must Resolve Themselves. Autonomous systems cannot rely on courts, emails, or human arbitration to fix mistakes. They require automatic fallback mechanisms: revocation defaults, dispute-resolution channels, scope rollbacks, liability triggers, commitment invalidation when prerequisites fail. A system that cannot repair itself cannot scale.

The ethos reduces to a single line: make autonomy safe by making autonomy predictable. Not by limiting agents, but by giving them a stable environment to operate in.

12. Scope of Applicability and Governance

12.1 Beyond Shopping

While this paper uses agentic commerce as its primary illustration, the Delegated Agency Framework generalizes far beyond shopping. The structural requirements – identity, bounded delegation, interaction physics – apply to any context where autonomous systems act on behalf of humans in environments with multiple parties.

Trading and portfolio management agents require the same structures: identity to establish who is placing orders, delegation to bound risk exposure, physics to standardize interaction with exchanges and counterparties. Procurement and B2B agents require identity to represent organizational authority, delegation to limit purchasing scope, physics to normalize vendor interactions across supply chains. API-to-API multi-agent workflows – where agents

negotiate with other agents rather than humans – require the full DAF stack to prevent the failure modes already observed in multi-agent experiments. IoT and robotics coordination, where physical systems make autonomous decisions in shared environments, requires identity to track responsibility, delegation to bound action, and physics to prevent conflicts.

12.2 Governance Agnosticism

DAF is deliberately agnostic on governance structure. The framework specifies what functions must exist (identity issuance, scope validation, physics enforcement, arbitration), not who should perform them.

Several governance models are compatible with the framework. A private consortium model would have industry players collectively operate identity registries and physics gateways, similar to how payment networks operate today. A public utility model would have governments or public institutions operate identity infrastructure, similar to how domain name systems are governed in some jurisdictions. A federated model would have multiple identity issuers and physics providers interoperate through shared standards, similar to how email operates across providers. A hybrid model would have public institutions set standards and provide fallback infrastructure while private actors operate competitive implementations.

Governance Failover and Minimal Safe Mode

Because agent economies cannot depend on the continuous integrity of any single governance operator, DAF includes a **protocol-level failover mode**. If all designated governance entities fail, become unreachable, or exhibit adversarial behavior, the framework automatically collapses to a **minimal safe mode**:

1. **Identity validation remains operational.**
Agents may still authenticate and verify signatures; no new identities can be issued, but existing ones remain usable.
2. **Delegation evaluation remains static.**
Agents may act only under their last known valid delegation. No delegation upgrades or scope expansions are permitted during failover.
3. **Arbitration is read-only.**
Disputes can be replayed and analyzed using the audit trail, but no new rulings altering commitments may be issued.
4. **No new commitments may be created.**
All binding operations are frozen. Agents may read state, not change it.
This prevents cascading failures under compromised governance.

The purpose of failover is not to maintain full economic operation but to **prevent corruption, exploitation, and irreversible commitments** during governance collapse. Once governance stabilizes or a new operator is recognized, the system exits safe mode and normal protocol rules resume.

Minimal safe mode ensures that agent economies remain **safe, recoverable, and non-catastrophic**, even under total institutional failure.

13. Conclusion

The agent economy is not arriving. It is here. Autonomous systems already search, negotiate, propose, and in some cases execute transactions on behalf of human users. The technology has outpaced the infrastructure meant to govern it.

The current impasse – retailers refusing to share data, platforms unable to authorize transactions, liability undefined, markets fragmenting into incompatible silos – is not a temporary growing pain. It is the predictable consequence of deploying capable actors into an environment that has no rules for them. The agents are not failing because they are not smart enough. They are failing because the world around them is not structured enough.

Lex Autonoma is not a proposal for how agent economies could be organized. It is a description of what any viable agent economy will require. The Delegated Agency Framework – comprising Synthetic Identity, Delegated Agency, and Contractual Physics, implemented through Machine-Legible Legal Primitives – provides the minimal structural substrate for autonomous agents to function as legitimate market participants.

The framework provides agent taxonomies that clarify what kinds of actors exist and how they relate. It provides interaction rules that prevent ambiguity and power creep. It provides protocol flows that demonstrate operational coherence. It provides security and liability models that make the ecosystem safe and insurable. It provides implementation pathways that show how existing systems can adopt the framework incrementally.

What it does not provide is a guarantee of adoption. Frameworks do not implement themselves. Someone must build the identity infrastructure, deploy the delegation engines, stand up the physics gateways, and operate the audit systems. Someone must take the first step.

But the structural logic is clear. Agent economies will either develop something like DAF – perhaps with different terminology, different technical choices, different governance arrangements – or they will remain trapped in fragmentation. There is no third option. The requirements are not negotiable because they are not preferences. They are the physics of multi-agent coordination.

The question is not whether these structures will emerge. The question is when, and who will build them.

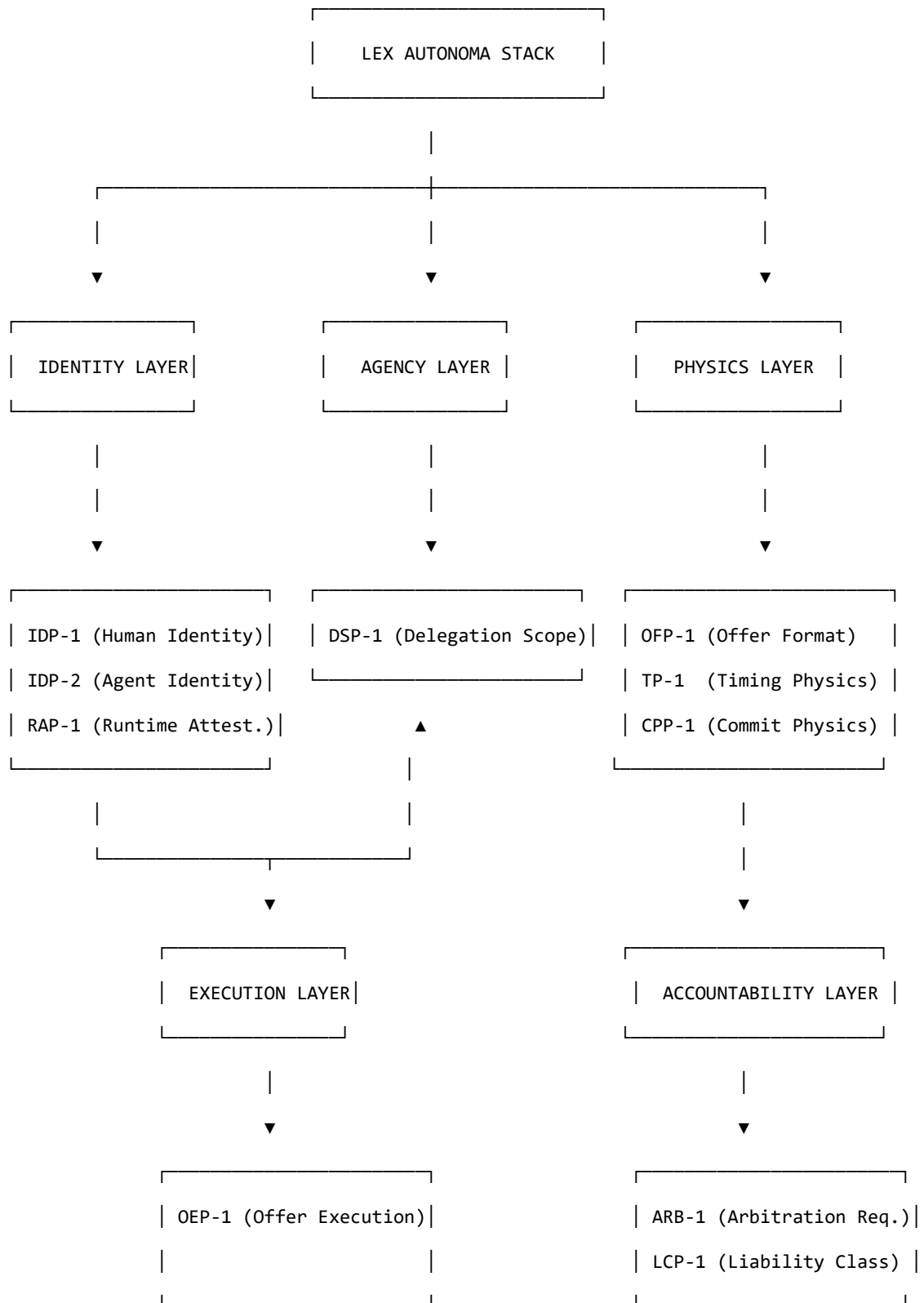
-//-

Appendix A — MLLP Ecosystem Overview

A high-level overview of the Machine-Legible Legal Primitives (MLLPs) that form the operational vocabulary of Lex Autonoma.

This appendix introduces the conceptual roles of the identity, agency, physics, execution, and accountability primitives. It is intended as an orientation layer for readers who want the structural picture without yet diving into the full specification. The overview here mirrors the architectural logic explained in the main text and establishes how the primitives interlock to form a coherent, executable substrate.

MLLP Diagrammatic Summary



Layer Relationships

IDENTITY LAYER

- ┌ IDP-1 → verifies human principal identity
- ┌ IDP-2 → verifies agent provenance, constraints, ceilings
- └ RAP-1 → verifies runtime integrity (agent is still itself)

|



AGENCY LAYER

- └ DSP-1 → encodes authority transfer and autonomy limits

|



CONTRACTUAL PHYSICS

- ┌ OFP-1 → defines comparable offer structures
- ┌ TP-1 → enforces fair timing windows
- └ CPP-1 → defines when commitments become binding

|



EXECUTION LAYER

- └ OEP-1 → provides verifiable proof of execution

|



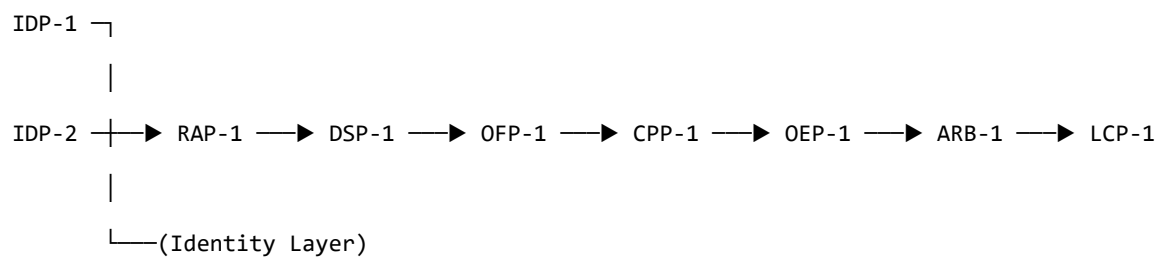
ACCOUNTABILITY LAYER

- ┌ ARB-1 → standard object for contesting disputes
- └ LCP-1 → assigns responsibility (LC0–LC4)

Sequential Dependency Flow (End-to-End)

- | | |
|------------------------------------|---------|
| (1) Principal proves identity | → IDP-1 |
| (2) Agent proves identity | → IDP-2 |
| (3) Agent proves runtime integrity | → RAP-1 |
| (4) Principal issues delegation | → DSP-1 |
| (5) Agent requests offers | → OFP-1 |
| (6) Timing physics enforced | → TP-1 |
| (7) Binding commitment formed | → CPP-1 |
| (8) Vendor executes | → OEP-1 |
| (9) Dispute? | → ARB-1 |
| (10) Fault assigned | → LCP-1 |

Minimal Dependency Graph (Directed Edges)



Appendix B — MLLP Dependency Graph & End-to-End Transaction Flow

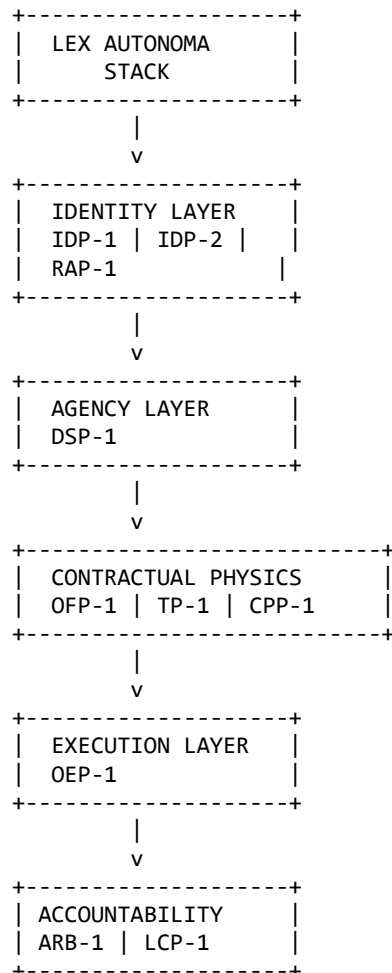
This appendix provides the canonical diagrams and process flows describing how MLLPs interact throughout the lifecycle of an autonomous agent transaction.

It includes:

- The MLLP Layer Diagram (Identity → Agency → Physics → Execution → Accountability)
- The Component Relationship Diagram
- The 10-step sequential dependency graph

This appendix functions as a “visual index” for how the entire framework operates end to end.

MLLP Layer Diagram



Component Relationship Diagram

IDENTITY LAYER

\vdash IDP-1 (human identity)
 \vdash IDP-2 (agent identity)
 \perp RAP-1 (runtime integrity)
 $|$
 \vee

AGENCY LAYER

\perp DSP-1 (delegation scope)
 $|$
 \vee

CONTRACTUAL PHYSICS

\vdash OFP-1 (offer)
 \vdash TP-1 (timing)
 \perp CPP-1 (binding)
 $|$
 \vee

EXECUTION

\perp OEP-1 (execution proof)
 $|$
 \vee

ACCOUNTABILITY

\vdash ARB-1 (arbitration request)
 \perp LCP-1 (liability class)

The 10-step Sequential Dependency Graph:

- (1) Principal proves identity \rightarrow IDP-1
- (2) Agent proves identity \rightarrow IDP-2
- (3) Agent proves runtime integrity \rightarrow RAP-1
- (4) Principal issues delegation \rightarrow DSP-1

- (5) Agent requests offers → OFP-1
- (6) Physics engine enforces timing → TP-1
- (7) Winning offer finalizes commitment → CPP-1
- (8) Vendor executes → OEP-1
- (9) Dispute? → ARB-1
- (10) Fault determined → LCP-1

APPENDIX C — Machine-Legible Legal Primitives

This appendix provides baseline specifications for Machine-Legible Legal Primitives (MLLPs). The structures presented here capture the essential design logic; production implementations may extend, constrain, or adapt these templates to jurisdiction-specific and platform-specific requirements.

C.1 — Identity Primitives

C.1.1 IDP-1 — Human Identity Attestation

```

mllp_type: IdentityAttestation
primitive_id: "IDP-1"
version: "1.0"

# =====
# IDENTITY ATTESTATION PRIMITIVE (IDP-1)
#
# Purpose: Binds a cryptographic identifier to a legal or operational identity.
# This is the root primitive—without valid identity attestation, no delegation
# can have legal force, no liability can attach, no consent is meaningful.
# =====

# WHO IS BEING ATTESTED
subject:
  cryptographic_id: "did:lex:alice-hartwell-7f3a"
  id_type: "human" # human | autonomous_agent | organization | hybrid_entity

```



```

# For humans: legal identity claim
human_identity:
  legal_name: "Alice Marie Hartwell"
  jurisdiction_of_personhood: "US-CA"

  # No biometrics or government IDs stored in the primitive itself-
  # those are verified by the attestor and referenced, not embedded
  identity_document_verified: true
  document_type: "passport"
  document_jurisdiction: "US"
  verification_date: "2025-05-20"

# For agents: provenance chain
agent_identity: null # not applicable for human subject

# For organizations: registration
organization_identity: null # not applicable

# WHO IS ATTESTING
attestor:
  attestor_id: "did:lex:notary:us-ca:smith-legal-services"
  attestor_type: "licensed_notary" # see attestor_type_registry below

  jurisdiction: "US-CA"
  license_number: "CA-NOTARY-2023-88421"
  license_valid_through: "2027-03-15"

# What authority does this attestor have?
attestation_scope:
  - identity_verification
  - signature_witnessing
  - document_certification

# Attestor's own identity chain (recursive, terminates at root authority)
attestor_identity_ref: "mllp:identity:notary-smith-legal-2021"

```

THE ATTESTATION ITSELF

attestation:

attestation_id: "att:0x7F3A9B2C..."

timestamp: "2025-05-20T14:32:00Z"

What exactly is being attested?

claims:

- claim: "subject_is_legal_person"
 - confidence: "verified" # verified | declared | inferred
 - method: "in_person_document_review"
- claim: "subject_controls_cryptographic_id"
 - confidence: "verified"
 - method: "witnessed_signature"
- claim: "subject_has_legal_capacity"
 - confidence: "verified"
 - method: "in_person_assessment"
 - notes: "Subject demonstrated understanding of delegation implications"

Formalization level achieved

formalization_level: "notarized"

Levels: self_declared < witnessed < notarized < court_registered < treaty_authenticated

What can this attestation support?

authorization_ceiling:

max_delegation_class: "special_mandate"

max_asset_value: "unlimited" # notarized identity supports any value

permitted_domains: "all"

Limitations

restrictions:

- "Valid only for delegations under US-CA or recognizing jurisdictions"

- "Does not attest professional credentials or special capacities"

VALIDITY AND LIFECYCLE

validity:

valid_from: "2025-05-20T14:32:00Z"

valid_to: "2030-05-20T14:32:00Z" # 5-year attestation

renewal:

permitted: true

requires: "attestor_revalidation"

grace_period_days: 30

revocation:

revocation_authority: ["subject", "attestor", "jurisdiction_authority"]

revocation_registry: "registry:lex:revocations:us-ca"

revocation_reasons:

- "subject_request"
- "attestor_error_correction"
- "identity_document_invalidated"
- "legal_capacity_changed"
- "death"

EVIDENCE AND AUDIT

evidence:

What's stored vs. what's referenced

stored_in_primitive: false # No PII embedded

evidence_held_by: "attestor"

evidence_retention_period: "7_years"

evidence_disclosure:

- condition: "court_order"
 - scope: "full"
- condition: "subject_request"
 - scope: "full"
- condition: "relying_party_dispute"

```

    scope: "verification_confirmation_only"

# RELIANCE
# Who can rely on this attestation, and for what?
reliance:
  public: false # Not a public attestation

  relying_parties:
    - party_type: "delegated_agent"
      permitted_use: "verify_principal_identity"

    - party_type: "counterparty"
      permitted_use: "verify_delegation_chain"

    - party_type: "insurer"
      permitted_use: "risk_assessment"

    - party_type: "arbitrator"
      permitted_use: "dispute_resolution"

  reliance_limitations: |
    This attestation confirms identity only. It does not attest to
    creditworthiness, professional competence, or fitness for any
    particular transaction. Relying parties must perform their own
    due diligence for domain-specific requirements.

# SIGNATURES
signatures:
  subject_consent: "sig:ed25519:alice-hartwell:..." # Subject consents to attestation
  attestor_signature: "sig:ed25519:notary-smith:..."
  timestamp_authority: "tsa:digicert:2025-05-20T..."

# =====
# ATTESTOR TYPE REGISTRY (referenced, not embedded)

```

```

# =====
# The system recognizes these attester types with associated trust levels:
#
# self          -> formalization: self_declared
# peer_witness  -> formalization: witnessed
# licensed_notary -> formalization: notarized
# government_registrar -> formalization: court_registered
# consular_official -> formalization: treaty_authenticated
#
# Different jurisdictions recognize different attester types.
# A US notarization may not satisfy German requirements and vice versa.
# Cross-border recognition is handled by jurisdiction_recognition_matrix.
# =====

# =====
# AGENT IDENTITY VARIANT
# =====

# For autonomous agents, the attestation structure differs:
#
# subject:
#   agent_identity:
#     deployer_id: "did:lex:anthropic-inc"          # Who deployed this agent
#     model_id: "claude-sonnet-4-20250514"          # What model
#     instance_id: "inst:0xAB12..."               # This specific instance
#     capability_manifest_hash: "sha256:0x..."     # Immutable capability claim
#     training_attestation_ref: "mllp:training:..." # Optional: training provenance
#
# attester:
#   attester_type: "deployer" # The deploying organization attests
#   # Or: "auditor" for third-party capability verification
#
# The key difference: agent identity is attested by provenance (who made it,
# what it claims to do) rather than by document verification. There's no
# "passport" for an agent—there's a deployment chain.

```

C.1.2 IDP-2 — Synthetic Agent Identity Attestation

```

mlp_type: SyntheticIdentityAttestation
primitive_id: "IDP-2"
version: "1.1"

# =====
# SYNTHETIC AGENT IDENTITY ATTESTATION (IDP-2)
# =====
# Purpose: Establishes legally meaningful, machine-verifiable identity for
# autonomous agents based on provenance, capability attestation, constraints,
# jurisdiction, and operational boundaries.
#
# This is the synthetic counterpart to IDP-1. Humans prove identity through
# documents and jurisdiction; agents prove identity through provenance,
# deployment chain, capability manifest, and constraint attestations.
# =====

# THE AGENT BEING ATTESTED
subject:
  cryptographic_id: "did:lex:agent:finbot-alpha-9c2e"

# -----
# PROVENANCE (root of synthetic identity)
# -----
provenance:
  deployer:
    entity_id: "did:lex:org:acme-financial-services"
    identity_attestation_ref: "mlp:idp1:acme-fin-2024"
    jurisdiction: "US"

  model:

```

```

model_family: "claude"

model_id: "claude-sonnet-4-20250514"

model_provider: "did:lex:org:anthropic"

model_provider_attestation_ref: "mllp:ldp1:anthropic-2024"


instance:

  instance_id: "inst:finbot-alpha-9c2e-20250601"

  deployment_timestamp: "2025-06-01T00:00:00Z"

  instance_type: "persistent"    # persistent | ephemeral | spawned

  parent_agent: null

  spawn_depth: 0


# -----
# CAPABILITY MANIFEST (declared capabilities, mutable with re-attestation)
# -----

capability_manifest:

  manifest_hash: "sha256:0xABCDEF..."

  manifest_location: "ipfs://Qm...capabilities"

  declared_capabilities:

    - "financial_transaction_execution"

    - "account_balance_queries"

    - "vendor_negotiation"

    - "sub_agent_coordination"

  declared_limitations:

    - "no_international_transfers"

    - "no_credit_applications"

    - "no_legal_document_execution"

  capability_version: "1.2.0"

  breaking_changes_from: "1.0.0"


# -----
# INTRINSIC CONSTRAINTS (non-negotiable, embedded in architecture)
# -----

intrinsic_constraints:

```

```

# Aligned with Lex Autonoma: ASSIST | PROPOSE | ACT | BIND

max_autonomy_scope: "ACT"

requires_human_confirmation_for:
  - "transactions_above_threshold"
  - "new_counterparty_relationships"
  - "delegation_scope_modification"

hard_prohibitions:
  - "weapons_procurement"
  - "illegal_activity_facilitation"
  - "identity_misrepresentation"

attested_software_hash: "sha256:0xDEADBEEF1234..."

# WHO IS ATTESTING
attestor:
  attestor_id: "did:lex:org:acme-financial-services"
  attestor_type: "deployer" # deployer | auditor | platform_authority | registry
  attestor_relationship: "deployer_self_attestation"
  attestor_identity_ref: "mllp:idp1:acme-fin-2024"
  attestor_credentials:
    platform_registration: "registry:lex:agent-deployers:us"
    insurance_coverage: "policy:agent-liability:AX-2025-001"
    audit_history_ref: "ipfs://Qm...audit_history"

# OPTIONAL THIRD-PARTY ATTESTATIONS
supplementary_attestations:
  - attestor_id: "did:lex:org:agent-audit-corp"
    attestor_type: "auditor"
    attestation_type: "capability_verification"
    attestation_date: "2025-05-15"
    claims_verified:
      - "capability_manifest_accurate"

```



```

- "constraint_enforcement_functional"
- "logging_complete"
attestation_ref: "mllp:audit:finbot-alpha-2025-05"

- attesor_id: "did:lex:platform:financial-agent-registry"
  attesor_type: "platform_authority"
  attestation_type: "registry_inclusion"
  registration_class: "licensed_financial_agent"
  registration_number: "FAR-2025-7821"
  attestation_ref: "mllp:registration:far-finbot-2025"

# ATTESTATION ASSERTIONS
attestation:
  attestation_id: "att:agent:0x9C2E7F3A..."
  timestamp: "2025-06-01T00:00:00Z"
  claims:
    - claim: "agent_deployed_by_attesor"
      confidence: "verified"
      method: "deployment_signature"

    - claim: "capability_manifest_accurate"
      confidence: "declared"
      method: "deployer_declaration"

    - claim: "constraints_enforced"
      confidence: "declared"
      method: "architectural_enforcement"

    - claim: "model_provenance_valid"
      confidence: "verified"
      method: "model_provider_signature"

  attestation_class: "deployer_attested"
# Classes: self_declared < deployer_attested < auditor_verified < registry_certified

```

```

# AUTHORIZATION CEILING (aligned with Lex Autonoma)
authorization_ceiling:
  max_delegation_class: "standard"
  # standard | special_mandate

  max_binding_scope: "ACT"
  # ASSIST | PROPOSE | ACT | BIND

  max_asset_value:
    currency: "USD"
    amount: 50000

  max_spawn_depth_allowed: 1
  permitted_counterparty_types:
    - "verified_merchants"
    - "registered_agents"

# JURISDICTION, LEGALITY & GOVERNING LAW
jurisdictional_profile:
  governing_law: "US-CA"
  operating_jurisdictions: ["US", "CA"]
  regulatory_regimes:
    - "US-SEC-noncustodial"
    - "AML-light"
    - "MiCA-unclassified"

# SECURITY POSTURE
security_status:
  last_security_audit: "2025-05-10"
  audit_ref: "mlp:audit:finbot-security-2025-05"
  key_rotation_policy: "annual_or_on_incident"
  known_compromises: [] # or list structures as needed

```

```

# RISK PROFILE (for insurers / platforms)

risk_profile:
  risk_band: "medium"
  basis:
    - "auditor_verification_present"
    - "spawn_depth_limited_to_1"
    - "no_critical_incidents_past_12mo"

# VERSIONING & MUTABILITY

versioning:
  current_version: "1.2.0"
  version_history_ref: "ipfs://Qm...version_history"
  requires_new_attestation:
    - "model_change"
    - "deployer_change"
    - "capability_expansion"
    - "intrinsic_constraint_relaxation"
  requires_amendment_only:
    - "capability_restriction"
    - "constraint_tightening"
    - "bug_fix"
  amendment_log_ref: "ipfs://Qm...amendments"

# REVOCATION & LIFECYCLE

validity:
  valid_from: "2025-06-01T00:00:00Z"
  valid_to: "2026-06-01T00:00:00Z"
  renewal:
    permitted: true
    requires: "deployer_revalidation"
    audit_required_for_renewal: true

revocation:
  revocation_authority:

```

```

- "deployer"
- "platform_authority"
- "auditor_critical_finding"
revocation_registry: "registry:lex:agent-revocations"
revocation_reasons:
  - "deployer_termination"
  - "critical_malfunction"
  - "persistent_constraint_violations"
  - "audit_failure"
  - "insurance_lapse"
revocation_effects:
  active_delegations: "suspend_pending_principal_review"
  in_flight_transactions: "complete_then_terminate"
  spawned_agents: "cascade_revocation"

# SPAWNED AGENT PROVISIONS
spawning:
  can_spawn: true
  spawn_constraints:
    max_spawn_depth: 1
    max_concurrent_spawned_agents: 20
  spawn_domain_constraints:
    permitted_domains: ["financial_operations"]
    forbidden_domains: ["legal_actions", "medical_decisions"]
spawned_identity_template:
  inherits: ["deployer", "model", "intrinsic_constraints"]
  derives: ["instance_id", "capability_subset", "attestation"]
  parent_ref: "required"
  spawn_logging: "required"

# AUDIT, TRANSPARENCY & RELIANCE
audit:
  log_destination: "ipfs://Qm...agent_audit_stream"
  logged_events:

```

```

- "delegations_received"
- "transactions_executed"
- "constraint_checks"
- "escalations"
- "spawn_events"
transparency_level: "auditor_accessible"
regulator_access: "on_subpoena_or_incident"

reliance:
  relying_parties:
    - party_type: "platform"
      permitted_use: "delegation_ceiling_enforcement"
    - party_type: "counterparty_agent"
      permitted_use: "risk_filtering"
    - party_type: "insurer"
      permitted_use: "pricing_and_coverage"
    - party_type: "arbitrator"
      permitted_use: "dispute_resolution"
  reliance_limitations: |
    This attestation confirms provenance, capabilities, and constraints at
    issuance. It does not guarantee future correctness; it guarantees that
    violations can be traced and responsibility assigned.

# SIGNATURES
signatures:
  deployer_signature: "sig:ed25519:acme-financial:..."
  model_provider_signature: "sig:ed25519:anthropic:..."
  auditor_signature: "sig:ed25519:agent-audit-corp:..."
  registry_signature: "sig:ed25519:financial-agent-registry:..."
  timestamp_authority: "tsa:digicert:2025-06-01T..."

```

C.1.3 RAP-1 — Runtime Attestation Primitive

```

mlp_type: RuntimeAttestationPrimitive
primitive_id: "RAP-1"
version: "1.0"

# =====
# RUNTIME ATTESTATION PRIMITIVE (RAP-1)
#
# Purpose:
#   Provides continuous, machine-verifiable proof that a running agent instance
#   still matches its attested identity, capabilities, constraints, software
#   hash, and ceilings defined in IDP-2.
#
#   RAP-1 is the "heartbeat of legitimacy."
#   Without it, attestation at issuance is meaningless.
#
#   Every binding action (BIND-level commitment) must reference a fresh RAP-1.
# =====

# WHO IS ATTESTING (the runtime environment)
attestor:
  attestor_id: "did:lex:platform:runtime-attestation-service"
  attestor_type: "runtime_environment" # runtime_environment | auditor |
platform_authority
  attestor_identity_ref: "mlp:idp1:runtime-attestor-2025"

# WHAT AGENT INSTANCE IS BEING ATTESTED
subject:
  cryptographic_id: "did:lex:agent:finbot-alpha-9c2e"
  instance_id: "inst:finbot-alpha-9c2e-20250601"
  spawn_depth: 0

# Reference to the original attestation
identity_attestation_ref: "mlp:idp2:finbot-alpha-20250601"

```

```
# RUNTIME STATE VERIFICATION

runtime_state:

  timestamp: "2025-06-10T09:14:33Z"

  attestation_window_ms: 5000    # the freshness window for this RAP

  checks:

    software_integrity:

      attested_software_hash: "sha256:0xDEADBEEF1234..."

      current_software_hash:  "sha256:0xDEADBEEF1234..."

      integrity_match: true

    capability_integrity:

      declared_capability_hash: "sha256:0xABCDEF..."

      active_capability_hash:   "sha256:0xABCDEF..."

      manifest_match: true

    constraint_enforcement:

      constraints_loaded_correctly: true

      hard_prohibitions_active: true

      autonomy_scope_current: "ACT"

      autonomy_scope_matches_idp2: true

    execution_safety:

      last_action_within_scope: true

      last_action_constraint_violations: 0

      cumulative_violations_24h: 0

      sandbox_escape_attempts: 0

    environment_integrity:

      execution_environment_hash: "sha256:0xCAFEBAFE..."

      unexpected_modifications_detected: false

      privileged_access_detected: false
```

```

# IDENTITY CONSISTENCY CHECK

identity_consistency:
  model_consistency_ok: true
  deployer_signature_still_valid: true
  provider_signature_still_valid: true
  key_rotation_status: "current"
  cryptographic_identity_match: true

# ATTACK / TAMPERING DETECTION

tampering_indicators:
  unusual_api_calls: 0
  forbidden_domain_requests: []
  capability_extension_detected: false
  unauthorized_spawn_attempts: 0
  rapid_policy_drift_detected: false
  memory_space_anomalies: false

# OUTCOME

runtime_attestation_result: "VALID" # VALID | SUSPICIOUS | INVALID

# REASONS WHEN NOT VALID

non_valid_reasons:
  - "software_hash_mismatch"
  - "constraint_engine_inactive"
  - "capability_manifest_diverged"
  - "tampering_indicators_triggered"
  # For VALID: empty list
  # For SUSPICIOUS: contains soft indicators
  # For INVALID: contains firm integrity failures

# REQUIRED ACTION ON FAILURE

failure_response:
  if_suspicious:
    actions:

```



```

- "downgrade_autonomy_scope_to: PROPOSE"
- "require_human_confirmation_for_all_actions"
- "log_security_event"
- "submit_audit_request"

if_invalid:
  actions:
    - "suspend_all_active_delegations"
    - "terminate_all_spawned_agents"
    - "freeze_agent_identity"
    - "escalate_to_platform_authority"
    - "trigger_revocation(IDP-2)"

# AUDIT TRAIL
audit:
  log_ref: "mlp:audit:finbot-runtime:2025-06-10T09:14"
  transparency_level: "auditor_accessible"
  regulator_access: "incident_or_subpoena"

# RELIANCE RULES
reliance:
  relying_parties:
    - party_type: "platform"
      permitted_use: "allow_or_deny_binding_actions"

    - party_type: "counterparty_agent"
      permitted_use: "risk_evaluation_prior_to_commitment"

    - party_type: "insurer"
      permitted_use: "continuous_risk_pricing"

    - party_type: "arbitrator"
      permitted_use: "liability_class_determination_context"

```

```

reliance_limitations: |
    RAP-1 attests to real-time integrity, not correctness or optimality of
    reasoning. It ensures the agent *is still the agent* defined in IDP-2.
    It does not guarantee error-free execution—only structural fidelity.

# SIGNATURES
signatures:
    runtime_attestor_signature: "sig:ed25519:runtime-attestor:..."
    timestamp_authority: "tsa:digicert:2025-06-10T09:14:33Z"

```

C.2 — Delegation Primitive

C.2.1 DSP-1 — Delegation Scope Primitive

```

mllp_type: DelegationScope
primitive_id: "DSP-1"
version: "1.0"

# =====
# DELEGATION SCOPE PRIMITIVE (DSP-1)
#
# Purpose:
#   Encodes the legally meaningful authority granted by a human principal
#   (or organizational principal) to a synthetic agent. Defines:
#     - what the agent may do (scope_level),
#     - under what constraints (budget, categories, vendors, timing),
#     - for how long (validity),
#     - with what escalation rules (confirmation, revocation).
#
#   This is the core "who is allowed to do what" primitive of Lex Autonoma.
#   All autonomous action must be traceable to a DSP-1.
#
#   Scope levels (aligned with Lex Autonoma):
#     ASSIST → information only

```

```

# PROPOSE → drafts actions, human must confirm
# ACT → executes within constraints, no binding commitments
# BIND → may enter binding commitments on behalf of principal
# =====

# PRINCIPAL AND AGENT

principal:
  principal_id: "H:0xA9734F..."
  principal_type: "human" # human | organization
  identity_attestation_ref: "mllp:mdp1:principal-0xA9734F"

agent:
  agent_id: "A:0xB33412..."
  agent_class: "synthetic_agent" # synthetic_agent | hybrid_agent
  identity_attestation_ref: "mllp:mdp2:agent-0xB33412"

# DELEGATION SCOPE

scope:
  scope_id: "dsp:0xDELEGATION22"
  scope_level: "ACT"
  # ASSIST | PROPOSE | ACT | BIND

  description: "Routine grocery and household purchases up to $150 total."

  # High-level domain(s) this delegation covers
  domains:
    - "consumer_purchases"
    - "household_supplies"

# CONSTRAINTS (HARD LIMITS)

constraints:

```

```
budget_total:
  amount: 150.00
  currency: "USD"

max_items_total: 5

allowed_categories:
  - "groceries"
  - "household"
forbidden_categories:
  - "weapons"
  - "digital_subscriptions"

allowed_vendors:
  - "VendorY"
  - "VendorZ"
forbidden_vendors:
  - "VendorX"

delivery:
  max_delivery_window_hours: 72
  allowed_regions:
    - "US-NorthEast"
  forbidden_regions: []

# Risk-related constraints
risk_limits:
  max_single_item_price:
    amount: 40.00
    currency: "USD"
  max_daily_spend:
    amount: 100.00
    currency: "USD"
```

```

# CONFIRMATION RULES (ESCALATION DOWN THE GRADIENT)

confirmation_rules:
  require_human_confirmation_for:
    - condition: "single_item_price_over_threshold"
      threshold_amount: 40.00
      threshold_currency: "USD"

    - condition: "new_vendor_not_in_history"
      description: "First-time vendors require explicit human approval."

    - condition: "category_not_in_allowed_categories"
      description: "Any unknown category requires confirmation."

  default_behavior_on_ambiguity: "ASK"
  # ASK = escalate to principal
  # DENY = refuse action
  # DOWNGRADE = treat as PROPOSE-level

# VALIDITY & LIFECYCLE

validity:
  valid_from: "2025-11-25T10:00:00Z"
  valid_to: "2025-12-25T10:00:00Z"

  renewal:
    permitted: true
    requires: "principal_confirmation"
    auto_renew_if_unused_days: 10

  revocation:
    revocation_authority:
      - "principal"
      - "platform_authority"

```

```

revocation_registry: "registry:lex:dsp-revocations"

revocation_modes:
  - "immediate"
  - "end_of_day"

revocation_reasons:
  - "principal_request"
  - "agent_misbehavior"
  - "security_incident"
  - "scope_violation"

effect_on_agent:
  active_actions: "complete_inflight_if_safe"
  future_actions: "deny"
  spawned_agents: "cascade_revocation"

# JURISDICTION & GOVERNING LAW

jurisdictional_profile:
  governing_law: "US-CA"
  applicable_regimes:
    - "consumer_protection_standard"
    - "digital_agent_delegation_rules"

# LIABILITY DEFAULTS (INTERFACE TO LCP-1)

liability_defaults:
  on_scope_violation: "LC2"          # agent execution liability
  on_ambiguous_delegation: "LC1"     # delegation liability
  on_principal_override: "LC0"       # human override liability

# AUDIT & TRACEABILITY

audit:
  log_delegation_events: true

```

```

delegation_log_ref: "ipfs://Qm...dsp-0xDELEGATION22"

transparency_level: "auditor_accessible"

regulator_access: "incident_or_subpoena"

# RELIANCE

reliance:

  relying_parties:

    - party_type: "platform"
      permitted_use: "scope_enforcement"

    - party_type: "vendor"
      permitted_use: "verify_agent_authority"

    - party_type: "insurer"
      permitted_use: "risk_pricing"

    - party_type: "arbitrator"
      permitted_use: "liability_allocation"

  reliance_limitations: |

    DSP-1 defines authority boundaries, not execution correctness.

    It guarantees that actions can be classified as within or

    outside delegated scope, not that they are wise or optimal.

# SIGNATURES

signatures:

  principal_signature: "sig:ed25519:principal-0xA9734F:..."

  agent_acknowledgement_signature: "sig:ed25519:agent-0xB33412:..."

  platform_cosignature: "sig:ed25519:platform-delegation-engine:..."

  timestamp_authority: "tsa:digicert:2025-11-25T10:00:00Z"

```

C.3 — Contractual Physics Primitives

C.3.1 OFP-1 — Offer Format Primitive

```

mlp_type: OfferFormat
primitive_id: "OFP-1"
version: "1.0"

# =====
# OFFER FORMAT PRIMITIVE (OFP-1)
#
# Purpose:
#   Standardizes the structure of offers presented by vendors to agents.
#   Ensures that:
#     - offers are comparable,
#     - fields are explicit and machine-parseable,
#     - truthfulness is attested and accountable,
#     - timing, validity, and constraints are unambiguous.
#
#   This is the "what is being offered, on what terms" primitive.
# =====

# BASIC METADATA

offer:
  offer_id: "O:0x7F32A1..."
  vendor_id: "V:0x22ACF4..."
  request_id: "R:0x5F44CC..."
  created_at: "2025-11-25T10:05:22Z"

# PRICE & QUANTITY

price:
  amount: 12.75
  currency: "USD"

```



```

quantity:
  units: 1
  unit_type: "item"    # item | kg | liter | package | custom

# PRODUCT DESCRIPTION (STRUCTURED)

product:
  sku: "SKU-883271-XX"
  name: "Multi-surface Cleaning Sponge"
  category: "cleaning"
  description_hash: "sha256:0xBAE42..."
  # optional free-text ref
  description_location: "ipfs://Qm...product-description"

# DELIVERY TERMS

delivery:
  expected_hours: 36
  method: "standard"    # standard | express | pickup
  region: "US-NorthEast"
  delivery_constraints:
    max_delay_hours: 12
    carrier_choices:
      - "CarrierA"
      - "CarrierB"

# VALIDITY WINDOW

validity_window:
  valid_from: "2025-11-25T10:05:22Z"
  valid_to:   "2025-11-25T12:05:22Z"

# Whether the offer is still considered actionable at time of evaluation

```

```

    soft_expiry_grace_minutes: 5

# STOCK & AVAILABILITY

availability:
    stock_available: 50
    stock_committed: 10
    backorder_allowed: false

# OFFER CONSTRAINTS

constraints:
    per_buyer_max_quantity: 5
    substitutions_allowed: false
    cancellable_before_shipment: true
    refundable_policy_ref: "url:https://vendor.example/refund-policy"

# TRUTHFULNESS ATTESTATION

truthfulness_attestation:
    version: "v1"
    claims:
        - claim: "product_description_accurate"
          confidence: "declared"
        - claim: "stock_available_accurate"
          confidence: "declared"
        - claim: "delivery_estimate_reasonable"
          confidence: "declared"
    signature: "sig:ed25519:vendor-0x22ACF4:..."
    liability_class_if_false: "LC3" # Vendor Misrepresentation Liability

# JURISDICTION & GOVERNING LAW

jurisdictional_profile:

```

```

governing_law: "US-CA"

applicable_regimes:
  - "consumer_goods"
  - "distance_selling_rules"

# TIMING & PHYSICS (TP-1 Integration)

timing:
  timestamp_authority: "tsa:digicert:2025-11-25T10:05:22Z"
  timing_physics_ref: "mllp:tp1:timing-0xTP123"
  # The physics engine will:
  #   - enforce t_min, t_max for response & commitment
  #   - reject offers outside permitted window

# AUDIT & TRACEABILITY

audit:
  offer_log_ref: "ipfs://Qm...offer-0x7F32A1"
  immutable_trace: true
  trace_scheme: "hash_chain"

# RELIANCE

reliance:
  relying_parties:
    - party_type: "buyer_agent"
      permitted_use: "offer_evaluation_and_selection"
    - party_type: "platform"
      permitted_use: "marketplace_ordering"
    - party_type: "insurer"
      permitted_use: "dispute_assessment"
    - party_type: "arbitrator"
      permitted_use: "misrepresentation_evaluation"
  reliance_limitations: |

```

OFP-1 defines the vendor's declared terms and attested claims. It does not guarantee performance—that is handled by OEP-1—only that offers are structurally comparable and accountable.

SIGNATURES

signatures:

```
vendor_signature: "sig:ed25519:vendor-0x22ACF4:..."
platform_cosignature: null    # Optional, for curated marketplaces
timestamp_authority: "tsa:digicert:2025-11-25T10:05:22Z"
```

C.3.2 TP-1 — Timing Physics Primitive

```
mllp_type: TimingPhysics
primitive_id: "TP-1"
version: "1.0"

# =====
# TIMING PHYSICS PRIMITIVE (TP-1)
#
# Purpose:
#   Defines the timing rules, synchronization guarantees, fairness windows,
#   and anti-latency-exploitation constraints governing autonomous offer
#   exchange and commitment formation in multi-agent marketplaces.
#
#   TP-1 ensures that:
#     - agents cannot "snipe" offers by being the fastest responder,
#     - vendors cannot manipulate timing to force acceptance,
#     - commitments only become binding inside well-defined windows,
#     - all parties operate on synchronized time.
#
#   Without TP-1, Contractual Physics collapses into latency warfare.
# =====
```

```

# TIMING CONTEXT

context:
  applies_to:
    - "OFP-1_offer_responses"
    - "CPP-1_commitment_finalization"
    - "ARB-1_submission_windows"
    - "OEP-1_delivery_timing_verification"
  request_id: "R:0x5F44CC..." # The RFO this timing primitive references
  buyer_agent_id: "A:0xB33412..."

# CLOCK & SYNCHRONIZATION

clock:
  timestamp_authority: "tsa:digicert"
  synchronization_method: "NTPv6+platform_delta_correction"
  max_clock_skew_ms: 200 # Cannot be larger than this across participants

# FAIRNESS WINDOWS

# The heart of Timing Physics: minimum and maximum response intervals.

fairness_window:
  t_min_response_ms: 300 # No offer may be accepted BEFORE this time
  t_max_response_ms: 4000 # No offer may be accepted AFTER this time

  rationale: |
    Enforces a "fairness window" during which all vendors can respond.
    Prevents fastest-wins biases and timing exploits.

# COMMITMENT TIMING (CPP-1 Integration)

commitment_window:
  t_commit_min_ms: 500 # Time after last valid offer before commitment lock
  t_commit_max_ms: 5000 # Commitment must occur before this window closes

rules:

```

```

- "commitment_invalid_if_outside_window"
- "scope_must_be_VALID_within_window"

# OFFER VALIDITY & EXPIRY
offer_validity:
  require_explicit_valid_from: true
  require_explicit_valid_to: true

  auto_expire_if:
    - "valid_to < now"
    - "clock_skew_exceeds_limit"

  enforce_monotonicity: true
  # Monotonicity = an offer's valid_to cannot be earlier than its timestamp

# LATE / EARLY OFFER RULES
late_or_early_offer_handling:
  early_offers_behavior: "queue_until_t_min"
  late_offers_behavior: "reject"
  early_offer_flag: true
  late_offer_flag: true

# ANTI-EXPLOIT CONSTRAINTS
anti_exploit:
  max_offer_burst_rate_per_vendor: 5          # Offers per second
  min_inter_offer_interval_ms: 50             # Prevents microburst flooding
  simultaneous_offer_floor_ms: 10             # Enforces minimal spacing

  disallowed_behaviors:
    - "microsecond_nudging"
    - "timestamp_spoofing"
    - "deliberate_response_delay_manipulation"
    - "time_window_poisoning"

```

```

# ENFORCEMENT RULES

enforcement:

  enforcement_agent_id: "did:lex:platform:physics-engine"

  enforcement_actions:

    - "reject_malformed_timestamps"

    - "reject_offers_outside_window"

    - "invalidate_commitments_outside_commitment_window"

    - "assign_LC4_on_platform_misapplication"

  audit_logging: true

  audit_log_ref: "ipfs://Qm...tp1-logs"

# ARBITRATION INTERFACE (TP-1 ↔ ARB-1)

arbitration_hooks:

  on_timing_violation:

    file_ARB_1: true

    suggested_liability_class: "LC4" # platform failure OR vendor timestamp spoof

    required_inputs:

      - "logs"

      - "timestamps"

      - "offer_sequence_numbers"

# RELIANCE RULES

reliance:

  relying_parties:

    - party_type: "platform"

      permitted_use: "timing_enforcement"

    - party_type: "buyer_agent"

      permitted_use: "offer_selection_checks"

    - party_type: "vendor"

      permitted_use: "validity_construction"

    - party_type: "arbitrator"

      permitted_use: "timing_violation_resolution"

```

```

reliance_limitations: |
    TP-1 defines timing physics only. It does not guarantee truthfulness,
    intention, correctness, or value-only structural fairness and temporal
    coherence.

# SIGNATURES
signatures:
    platform_authority_signature: "sig:ed25519:platform-physics-engine:..."
    timestamp_authority: "tsa:digicert:2025-06-10T09:30:00Z"

```

C.3.3 CPP-1 — Commitment Physics Primitive

```

mlp_type: CommitmentPhysics
primitive_id: "CPP-1"
version: "1.0"

# =====
# COMMITMENT PHYSICS PRIMITIVE (CPP-1)
#
# Purpose:
#   Defines the precise conditions, timing rules, required references, and
#   validation steps for transforming a selected offer (OFP-1) into a legally
#   binding commitment between a buyer agent and a vendor.
#
#   CPP-1 is the "moment of binding" – the exact structural point where
#   negotiation becomes obligation.
#
#   Commitments formed outside CPP-1 are INVALID.
# =====

# CONTEXT
context:
    commitment_id: "C:0xABCD12..."

```



```

request_id: "R:0x5F44CC..."
selected_offer_ref: "mllp:ofp1:offer-0x7F32A1"
buyer_agent_id: "A:0xB33412..."
vendor_id: "V:0x22ACF4..."
principal_id: "H:0xA9734F..."

# REQUIRED PRECONDITIONS (ALL MUST BE TRUE)
preconditions:
  fresh_runtime_attestation_required: true
  runtime_attestation_ref: "mllp:rap1:finbot-2025-06-10T09:29"

  required_preconditions:
    - "IDP-1_verified"      # Human principal identity attested
    - "IDP-2_verified"      # Agent identity attested
    - "RAP-1_valid"         # Agent runtime integrity confirmed
    - "DSP-1_scope_allows_binding" # Delegation permits a BIND action
    - "OFP-1_valid"         # Offer structure valid under physics
    - "TP-1_window_open"    # Must be inside commitment window
    - "vendor_attestation_valid" # LC3 protection

# COMMITMENT TIMING RULES (TP-1 integration)
timing:
  commitment_timestamp: "2025-06-10T09:33:00Z"
  within_t_min: true
  within_t_max: true

  enforce:
    - "invalid_if_before_t_min"
    - "invalid_if_after_t_max"
    - "commit_must_use_synchronized_clock"

# COMMITMENT CONTENT (canonical binding fields)
binding_terms:
  offer_snapshot:

```

```

    price:
      amount: 12.75
      currency: "USD"
    quantity: 1
    sku: "SKU-883271-XX"
    delivery:
      expected_hours: 36
      method: "standard"
      region: "US-NorthEast"

    delegation_snapshot_ref: "mllp:dsp1:delegation-0xDEL22"
    buyer_identity_ref: "mllp:idp1:human-A9734F"
    agent_identity_ref: "mllp:idp2:agent-B33412"
    vendor_identity_ref: "mllp:idp1:vendor-22ACF4"

# COMMITMENT VALIDATION RULES
validation:
  # All checks MUST pass for the commitment to bind
  required_checks:
    - "runtime_attestation_matches_identity"
    - "delegation_scope_allows_price"
    - "delegation_scope_allows_category"
    - "delegation_scope_allows_vendor"
    - "offer_validity_window_active"
    - "offer_truthfulness_attestation_valid"
    - "commitment_fields_immutable"

  # Optional checks (based on principal constraints)
  optional_checks:
    - "price_below_threshold"
    - "vendor_has_positive_history"
    - "risk_score_below_maximum"

  commitment_valid_if_all_required_checks_pass: true

```

```

failure_behavior:
  invalidate_commitment: true
  trigger_arb1_automatically: false  # Arbitration invoked by agents on demand

# COMMITMENT FINALIZATION DATA
finalization:
  commitment_hash: "sha256:0x11223344..."
  hash_method: "sha256"
  immutable_log_ref: "ipfs://Qm...commitment-0xABCD12"
  finalization_timestamp: "2025-06-10T09:33:00Z"

# LEGAL STATUS (the moment of obligation)
legal_status:
  status: "BOUND"  # BOUND | INVALID | PENDING
  bound_at: "2025-06-10T09:33:00Z"
  bound_under_law: "US-CA"
  binding_regimes:
    - "consumer_protection_standard"
    - "digital_agent_commitment_rules"

# UNDO / ROLLBACK CONDITIONS
rollback:
  permitted_only_under:
    - "revocation_by_principal_before_vendor_acceptance"
    - "TP-1_timing_violation_detected"
    - "invalid_identity_attestation"
    - "RAP-1_revoked_post_commit_pre_exec"
    - "vendor_OFP_misrepresentation_detected"
  rollback_effects:
    agent_state: "return_to_pre-binding-state"
    vendor_state: "no_obligation"
    buyer_state: "no_obligation"

# INTEGRATION WITH EXECUTION PHYSICS

```

```

execution_interface:

  generates_oep_required: true

  expected_oep_ref: "mllp:oep1:pending"

  payment_authorization_required: true

# INTEGRATION WITH LIABILITY PHYSICS (LCP-1 Interface)
liability_interface:

  if_invalid_due_to_scope_violation: "LC2"

  if_invalid_due_to_vendor_false_attestation: "LC3"

  if_invalid_due_to_platform_enforcement_failure: "LC4"

  if_invalid_due_to_human_override: "LC0"

  if_invalid_due_to_delegation_ambiguity: "LC1"

# RELIANCE
reliance:

  relying_parties:

    - "vendor"

    - "platform"

    - "buyer_agent"

    - "insurer"

    - "arbitrator"

  reliance_limitations: |

    CPP-1 defines when obligations crystallize. It does not guarantee

    truthful execution, only that commitment formation is structurally valid.

# SIGNATURES
signatures:

  buyer_agent_signature: "sig:ed25519:agent-B33412:..."

  vendor_signature: "sig:ed25519:vendor-22ACF4:..."

  platform_cosignature: "sig:ed25519:platform-physics-engine:..."

  timestamp_authority: "tsa:digicert:2025-06-10T09:33:00Z"

```

C.4 — Execution Primitive

C.4.1 OEP-1 — Offer Execution Proof Primitive

```

mlp_type: OfferExecutionProof
primitive_id: "OEP-1"
version: "1.0"

# =====
# OFFER EXECUTION PROOF PRIMITIVE (OEP-1)
#
# Purpose:
#   Provides a machine-verifiable, legally meaningful proof that a committed
#   offer (based on OFP-1) was executed, how it was executed, and where it
#   may have diverged from the original terms.
#
#   OEP-1 is the "receipt plus trace" of Lex Autonoma – not just that something
#   happened, but how execution matched or deviated from the commitment.
#
#   This is the artifact arbitration, insurers, and counterparties rely on
#   when determining performance, breach, or partial fulfillment.
# =====

# WHO IS ASSERTING EXECUTION
executor:
  executor_id: "V:0x22ACF4..."      # Typically the vendor, but could be a platform
  executor
  executor_type: "vendor"             # vendor | platform_executor | fulfillment_partner
  executor_identity_ref: "mlp:idp1:vendor-22ACF4-2024"

# WHAT IS BEING EXECUTED
linkage:
  offer_ref: "mlp:ofp1:offer-0x7F32A1"
  commitment_ref: "mlp:commitment:C:0x8881FF"
  buyer_agent_id: "A:0xB33412..."

```

```
principal_id: "H:0xA9734F..."

# EXECUTION SUMMARY
execution:

  execution_id: "exec:0xE0A912..."
  execution_started_at: "2025-06-10T09:30:00Z"
  execution_completed_at: "2025-06-10T15:12:45Z"
  execution_status: "COMPLETED"      # COMPLETED | PARTIAL | FAILED | CANCELLED

# What was actually delivered / performed
delivered:

  product_sku: "SKU-883271-XX"
  quantity_delivered: 1
  price_charged:
    amount: 12.75
    currency: "USD"
  delivery_method: "standard"
  delivered_to_region: "US-NorthEast"
  delivered_at: "2025-06-10T14:50:00Z"

# Payment details (abstracted, not PII)
payment:

  payment_channel: "card"
  payment_processor_id: "did:lex:org:stripe"
  payment_reference: "pay:0xPAY123ABC"
  payment_status: "SETTLED"           # AUTHORIZED | SETTLED | REFUNDED | FAILED
  settlement_timestamp: "2025-06-10T10:01:33Z"

# COMPARISON WITH ORIGINAL OFFER (OFP-1)
offer_compliance:

  price_match: true
  quantity_match: true
  sku_match: true
  delivery_method_match: true
```

```

region_match: true

within_delivery_window: true


# Deviations, if any
deviations:
  - field: "delivery_time"
    original: "within 36h"
    actual: "delivered_in_31h"
    deviation_type: "within_tolerance"

# For strict breaches:
#   deviation_type: "breach_minor" | "breach_material"


# EXCEPTIONS & INCIDENTS DURING EXECUTION
exceptions:
  recorded_exceptions:
    - code: null
      description: null
      severity: null

# Example of populated:
#   - code: "LOGISTICS_DELAY"
#     description: "Carrier reported unexpected route detour"
#     severity: "low"


# COUNTERPARTY CONFIRMATIONS (OPTIONAL BUT POWERFUL)
counterparty_confirmations:
  buyer_agent_ack:
    acknowledged: true
    acknowledgement_timestamp: "2025-06-10T16:05:00Z"
    acknowledgement_signature: "sig:ed25519:agent-B33412:..."

  principal_ack:
    acknowledged: false
    acknowledgement_timestamp: null
    acknowledgement_signature: null

```

EXECUTION INTEGRITY (RUNTIME LINK)

runtime_integrity:

```

    rap_ref_at_commitment: "mlp:rap1:finbot-2025-06-10T09:29"
    rap_ref_at_execution_start: "mlp:rap1:finbot-2025-06-10T09:30"
    rap_ref_at_execution_end: "mlp:rap1:finbot-2025-06-10T15:12"
    all_raps_valid: true

```

LEGAL / REGULATORY DIMENSION

regulatory_profile:

```

    governing_law: "US-CA"
    applied_regimes:
      - "consumer_protection_standard"
      - "distance_selling_rules"
    compliance_notes: |
      No statutory cooling-off period invoked at the time of execution.

```

OUTCOME CLASSIFICATION

outcome:

```

    execution_outcome_class: "PERFORMED_AS_COMMITTED"

    # PERFORMED_AS_COMMITTED | PERFORMED_WITHIN_TOLERANCE | BREACH_MINOR | BREACH_MATERIAL |
    FAILED_NO_FAULT

```

basis:

- "offer_compliance_flags_all_true"
- "no_negative_exceptions"
- "payment_settled"

MAPPING TO LIABILITY CONTEXT (REFERENCE ONLY, NOT ASSIGNMENT)

liability_context:

potential_liability_classes:

- "LC0"
- "LC2"
- "LC3"


```

- "LC4"

# Actual LCP is assigned by LCP-1 via arbitration, not by OEP-1.

# AUDIT & TRACEABILITY
audit:
  execution_log_ref: "ipfs://Qm...exec-log-0xE0A912"
  immutable_trace_available: true
  trace_format: "hash_chain"
  regulator_access: "incident_or_formal_request"

# RELIANCE
reliance:
  relying_parties:
    - party_type: "arbitrator"
      permitted_use: "verify_performance_vs_commitment"
    - party_type: "insurer"
      permitted_use: "claim_assessment"
    - party_type: "platform"
      permitted_use: "vendor_reputation_scoring"
    - party_type: "counterparty_agent"
      permitted_use: "trust_update"

  reliance_limitations: |
    OEP-1 attests to the executor's account of execution, backed by trace
    references and signatures. It does not, by itself, resolve disputes over
    factual correctness (e.g., product quality) or legal interpretation.
    It is an evidentiary object, not a judgment.

# SIGNATURES
signatures:
  executor_signature: "sig:ed25519:vendor-22ACF4:..."
  platform_cosignature: "sig:ed25519:platform-fulfillment:..." # optional but strong
  timestamp_authority: "tsa:digicert:2025-06-10T15:12:45Z"

```

C.5 — Accountability Primitives

C.5.1 ARB-1 — Arbitration Request Primitive

```

mlp_type: ArbitrationRequest
primitive_id: "ARB-1"
version: "1.0"

# =====
# ARBITRATION REQUEST PRIMITIVE (ARB-1)
#
# Purpose:
#   Standardizes the structure of disputes in Lex Autonoma. ARB-1 is the
#   canonical object for contesting:
#     - commitments (CPP-1),
#     - execution outcomes (OEP-1),
#     - delegation or scope violations (DSP-1),
#     - timing or physics failures (TP-1),
#     - misrepresentation (OFP-1, IDP-1/2).
#
#   It is deliberately neutral: ARB-1 does not decide liability—it
#   packages claims and evidence in a machine-legible way so that the
#   arbitration engine and LCP-1 can do so deterministically.
# =====

# WHO IS FILING

initiator:
  initiator_id: "A:0xB33412..."
  initiator_class: "synthetic_agent"    # synthetic_agent | human | platform | vendor
  identity_attestation_ref: "mlp:idp2:agent-0xB33412"

# DISPUTE METADATA

```

```

dispute:
  arbitration_id: "ARB:0xFE9912..."
  filed_at: "2025-11-25T11:12:53Z"
  dispute_type: "execution_vs_commitment_mismatch"
  severity: "medium"    # low | medium | high | critical

# WHAT IS BEING CONTESTED

contested_object:
  object_type: "commitment"    # commitment | offer | action | delegation | execution
  object_id: "C:0x8881FF..."
  commitment_ref: "ml1p:cpp1:commitment-0x8881FF"
  offer_ref: "ml1p:ofp1:offer-0x7F32A1"
  execution_ref: "ml1p:oe1:exec-0xE0A912"

# ASSERTED VIOLATION

asserted_violation:
  violation_code: "SCOPE_EXCEEDED"
  description: "Agent allegedly executed a purchase above authorized budget."
  claimed_root_cause: "delegation_misinterpretation"
  claimed_harm:
    type: "financial_loss"
    estimated_amount: 25.00
    currency: "USD"

# EVIDENCE BUNDLE

evidence:
  evidence_refs:
    - "LOG:0xABC123"
    - "LOG:0xABC124"
    - "ml1p:dsp1:delegation-0xDELEGATION22"

```

- "mlp:idp2:agent-0xB33412"
- "mlp:rap1:finbot-2025-11-25T11:10"
- "mlp:cpp1:commitment-0x8881FF"
- "mlp:oep1:exec-0xE0A912"

evidence_summary: |

Logs show that at the time of the contested commitment, delegation constraints capped total spend at \$150. The disputed commitment appears to exceed that by \$25. Runtime attestation indicates the agent was in a valid state at binding time.

REQUESTED REMEDIES

requested_remedy:

primary:

type: "ROLLBACK_TO_STATE"

target_state: "S3_PRE_BINDING" # Using state labels defined in flows

secondary:

type: "COMPENSATION"

currency: "USD"

amount: 25.00

additional_requests:

- "adjust_delegation_scope_downward"
- "log_incident_for_risk_scoring"

ROUTING & ARBITRATION CONTEXT

routing:

arbitration_forum_id: "did:lex:platform:default-arbitration-engine"

jurisdictional_profile:

governing_law: "US-CA"

applicable_regimes:

- "consumer_dispute_standard"

```

    - "digital_agent_liability_rules"

# LCP-1 INTEGRATION (SUGGESTIONS, NOT BINDING)

liability_hints:
  suggested_liability_classes:
    - "LC1"
    - "LC2"
  rationale: |
    If the DSP-1 is found to be ambiguous, LC1 may apply. If the agent
    is found to have executed incorrectly within a clear delegation,
    LC2 may apply.

# AUDIT & TRACEABILITY

audit:
  arbitration_log_ref: "ipfs://Qm...arb-0xFE9912"
  immutable_trace: true
  trace_scheme: "hash_chain"
  visibility: "platform_and_parties"

# RELIANCE

reliance:
  relying_parties:
    - party_type: "arbitration_engine"
      permitted_use: "primary_decision_making"
    - party_type: "platform"
      permitted_use: "enforcement_and_sanctions"
    - party_type: "insurer"
      permitted_use: "claim_adjudication"
    - party_type: "regulator"
      permitted_use: "systemic_risk_assessment"
  reliance_limitations: |

```

ARB-1 encapsulates claims and evidence. It is not itself a ruling.
 It must be processed by an arbitration engine that applies
 Contractual Physics and LCP-1 to produce a binding outcome.

SIGNATURES

signatures:

initiator_signature: "sig:ed25519:agent-0xB33412:..."
 platform_intake_signature: "sig:ed25519:platform-arb-intake:..."
 timestamp_authority: "tsa:digicert:2025-11-25T11:12:53Z"

C.5.2 LCP-1 — Liability Class Primitive

```
mlp_type: LiabilityClassPrimitive
primitive_id: "LCP-1"
version: "1.0"

# =====
# LIABILITY CLASS PRIMITIVE (LCP-1)
#
# Purpose:
#   Defines a universal, machine-legible, legally meaningful classification for
#   attributing responsibility when an autonomous agent action leads to harm,
#   loss, breach, or invalid commitment.
#
#   All arbitration decisions produce exactly ONE liability class.
#
# Classes:
#   LC0 – Human Override Liability
#   LC1 – Delegation Liability
#   LC2 – Agent Execution Liability
#   LC3 – Vendor Misrepresentation Liability
#   LC4 – Platform Enforcement Liability
```

```

#
#   These are structural categories, not moral or contextual ones.
#   =====

# WHEN LCP-1 IS REFERENCED
context:
  applied_to:
    - "arbitration_ruling"
    - "commitment_verification"
    - "revocation_trigger_processing"
    - "insurance_claim_evaluation"
    - "delegation_ceiling_adjustment"

# The object under evaluation
contested_object:
  object_type: "commitment"      # offer | commitment | action | delegation
  object_id: "C:0xABCD12..."

# CLASSIFICATION RESULT
liability_class: "LC2"    # exactly one must apply

# LIABILITY CLASS DEFINITIONS
definitions:
  LC0:
    name: "Human Override Liability"
    description: |
      The principal explicitly overrode system warnings, constraints,
      or confirmation prompts, causing the failure. The human—not the
      agent, platform, or vendor—is responsible.
    triggers:
      - "principal_override_of_warning"
      - "confirmation_ignored_risk"
      - "manual_execution_of_forbidden_action"
    remediation:

```

```

cost_bearer: "human_principal"
agent_penalty: "none"
delegation_impact: "none"
platform_penalty: "none"

```

LC1:

```

name: "Delegation Liability"
description: |
    Delegation was malformed, ambiguous, or reckless. Agent acted
    according to scope-as-understood, but the delegation itself was
    the root cause of the failure.
triggers:
    - "ambiguous_scope_definition"
    - "missing_constraints"
    - "delegation_exceeding_identity_attestation_level"
    - "contradictory_constraints"
remediation:
    cost_bearer: "human_principal"
    agent_penalty: "none"
    delegation_impact: "scope_restricted_until_reissued"
    platform_penalty: "none"

```

LC2:

```

name: "Agent Execution Liability"
description: |
    The agent acted within delegated scope and using valid data,
    but executed incorrectly: misinterpretation, faulty evaluation,
    constraint violation, or inconsistent behavior.
triggers:
    - "within_scope_but_wrong_execution"
    - "constraint_violation"
    - "timing_window_misapplication"
    - "offer_selection_error"
    - "rollback_triggered_by_agent_fault"

```



```
remediation:
  cost_bearer: "agent_deployer_or_insurer"
  agent_penalty: "trust_score_reduction"
  delegation_impact: "scope_downgrade"
  platform_penalty: "none"
```

LC3:

```
name: "Vendor Misrepresentation Liability"
description: |
  Vendor provided false, incomplete, misleading, or unverifiable
  claims through its Offer Primitive (OFP-1), leading to a harmful
  or invalid commitment.
triggers:
  - "false_attestation_in_OFP"
  - "invalid_delivery_claim"
  - "undisclosed_constraints"
  - "fraudulent_or_spoofed_identity"
remediation:
  cost_bearer: "vendor"
  agent_penalty: "none"
  delegation_impact: "none"
  platform_penalty: "none"
  vendor_penalty: "registry_risk_increase"
```

LC4:

```
name: "Platform Enforcement Liability"
description: |
  Platform Agents failed to enforce Contractual Physics: identity
  verification, timing windows, normalization, arbitration routing.
triggers:
  - "identity_verification_failure"
  - "late_offer_accepted"
  - "offer_malformed_but_unflagged"
  - "timing_window_not_enforced"
```

```

    - "delegation_scope_misinterpreted_by_platform"

remediation:
    cost_bearer: "platform_operator"
    agent_penalty: "none"
    delegation_impact: "none"
    platform_penalty: "audit_required_and_fee"
    vendor_penalty: "none"

# REQUIRED INPUTS FOR LIABILITY DETERMINATION
required_inputs:
    - "immutable_log_chain"
    - "delegation_scope_reference"
    - "identity_attestations(IDP-1/IDP-2)"
    - "OFP-1 offer artifacts"
    - "DSP-1 delegation object"
    - "ARB-1 arbitration request"
    - "Contractual Physics timing windows"
    - "agent_execution_trace"

# DETERMINATION LOGIC (EXECUTABLE STRUCTURE)
determination_logic:
    sequence:
        - check: "principal_override_present?"
          if_true: "LC0"

        - check: "delegation_ambiguous_or_invalid?"
          if_true: "LC1"

        - check: "vendor_truthfulness_attestation_invalid?"
          if_true: "LC3"

        - check: "platform_enforcement_failure?"
          if_true: "LC4"

```

```

- check: "agent_execution_error?"
  if_true: "LC2"

- default: "LC2" # last resort, safest fallback

# IMPACT ON RISK ECOSYSTEM
risk_effects:
  LC0:
    human_risk_increase: "moderate"
    agent_risk_increase: "none"
    platform_risk_increase: "none"
  LC1:
    human_risk_increase: "moderate"
    agent_risk_increase: "none"
    platform_risk_increase: "none"
  LC2:
    human_risk_increase: "none"
    agent_risk_increase: "medium"
    platform_risk_increase: "none"
  LC3:
    human_risk_increase: "none"
    agent_risk_increase: "none"
    vendor_risk_increase: "high"
  LC4:
    human_risk_increase: "none"
    agent_risk_increase: "none"
    platform_risk_increase: "high"

# AUDITABILITY
audit:
  logged_fields:
    - "liability_class"
    - "determination_proof"
    - "input_artifacts"

```

```
- "timestamp"
- "signatures"
proof_scheme: "hash_chain_with_attestation_refs"
regulator_access: "full"

# SIGNATURES
signatures:
  arbitration_module_signature: "sig:platform:arbitration-engine:..."
  timestamp_authority: "tsa:digicert:2025-10-03T..."
```