Shared Object Networking 2.0

A Layered Architecture for Persistent, Decentralizable Knowledge and Agentic Reasoning

Part 1: Core Framing and Theory

Bill Weber
CyberFoundry
bill@cyberfoundry.io

September 15, 2025

Abstract

Modern AI systems answer fluently but forget, aggregate data but lose provenance, and promise privacy while leaking context. The common failure is architectural: we have applications for knowledge, but not a protocol for knowledge. Shared Object Networking (SON) 2.0 introduces such a protocol, treating knowledge as a networked resource governed by thin invariants that make it portable, governable, and replayable across implementations. In SON, every unit of knowledge is a persistent object with identity, schema, evidence, and obligations; layers persist as distinct z-axis strata that separate observations from inferences; gates enforce separation of duties at every durable write; and reasoning produces Query Result Objects (QROs) that bind answers to their inputs, policies, and costs. These commitments are deliberately minimal: they fix only what is persistence-binding and replay-critical, leaving algorithms, stores, and ontologies as implementation choices. This paper specifies SON's nine-layer, three-tier stack, establishes the protocol/implementation doctrine, and introduces its contestability discipline based on confidence, reputation, and evidence. The result is a substrate where provenance and policy travel with objects, obligations are enforced at consumption, and answers remain reproducible "as of" the moment they were produced. SON is not another database or model, but the protocol that allows many of them to coexist without making knowledge fragile.

Keywords: knowledge protocol, z-axis persistence, provenance, replayability, Query Resolution Object, Confidence–Reputation–Evidence, orchestration, agentic AI, policy enforcement, federated governance

Table of Contents

Introduction

The current wave of systems can speak, search, and summarize, but they rarely think in a way we can reuse. Large language models are fluent pattern-makers; they can imitate reasoning, but they do not structure it. Their outputs are difficult to replay “as of” a moment, hard to audit across teams, and brittle when stitched together with proprietary retrieval or workflow code. The result is a ceiling on innovation: we can scale models, but we cannot systematically compose capabilities across disciplines to produce durable, novel synthesis.

Shared Object Networking (SON) 2.0 begins from a different premise. Innovation does not arrive as a single giant model; it emerges when many small, well-scoped skills—extraction, alignment, hypothesis, verification, policy enforcement—are framed inside a common protocol so they can be combined, replaced, and improved without breaking the whole. The missing piece in today’s stacks is precisely that protocol: a way to structure thought from user intent, through reasoning, to data retrieval and use, so that each step is explicit, portable, governed, and replayable.

SON provides that framework. It treats knowledge as objects—packets with stable identity, declared schema, evidence, and an obligations envelope—so units of meaning can travel across implementations without losing provenance or policy. It treats context as z-axis layers, so observations, references, and inferences persist as distinct strata instead of being collapsed into a single, fragile store. And at every point where artifacts become durable, SON uses separation-of-duties gates: producers propose; gates commit, binding each admission to what was actually inspected and to the policy snapshot in force. These are protocol-level commitments: thin, universal, and independent of vendor choices or algorithms.

Reasoning under SON is likewise standardized without being centralized. When a user states a goal, the system decomposes it into subgoals, delegates work, and assembles a Query Result Object (QRO)—a compact, durable record that binds the answer to its exact inputs (with hashes and policies), the evidence and thresholds applied, and the gas spent to achieve it. QROs make answers replayable “as of” their context and make progress measurable. A lightweight Confidence–Reputation–Evidence (CRE) discipline constrains acceptance: correlated evidence is discounted, diverse corroboration is rewarded, and future trust tracks realized—not promised—improvement. Error is not hidden; it becomes fuel for learning.

This protocol stance changes what systems can do together. Because objects, layers, gates, and QROs are common contracts, skills from different teams and disciplines interoperate by design. A musicologist’s extractor, a statistician’s aligner, and a compliance officer’s policy pack can be composed in one run without bespoke glue. Implementations remain free to choose models, indexes, or stores; SON fixes only what two independent implementations must share to work safely together: what enters a layer, what leaves, what the gate records, and what cannot be weakened.

The architecture that follows is deliberately simple: three data layers that objectify and contextualize information and curate it toward publication; three reasoning layers that delegate work, orchestrate results, and emit QROs; and three user layers where exports are standardized, applications consume inspected artifacts, and obligations are enforced at use. Throughout the paper we use a notional “Miles Davis” strand to keep the discussion concrete without tying it to any one technology: biographies become Person objects, discographies become Work and Performance objects, contexts remain distinct, consolidation aligns them under explicit thresholds, and a QRO explains exactly how a query such as “collaborations in 1959” was answered and under what obligations it may be reused.

This is not another application or a bigger model. It is a protocol for structuring thought—from intent to reasoning to retrieval—that makes innovation a matter of composing reliable pieces rather than reinventing monoliths. In the sections that follow, we set the protocol/implementation doctrine, specify each layer with a uniform contract and gate, and show how contestability and replayability keep knowledge durable as systems evolve.

A Manifesto for Human Agency in Knowledge Systems

We begin from a simple premise: information systems exist to serve people, not the other way around. In an era where models ingest, transform, and generate at super-human scale, the basic terms of that service must be re-stated as rights. These rights are not matters of interface preference; they are protocol obligations that determine what a system must disclose, what it may do, and how it must be corrected. SON is introduced here not as another implementation pattern, but as the minimal infrastructure of rights required to make modern AI accountable to the people it affects.

The right to be known only as you intend. Identity is not a default setting; it is a deliberate act. A person must be able to disclose a chosen identity, adopt a different role, or withhold identity where policy permits, at the moment of intent. SON encodes this at the boundary: Layer 8 binds microidentity and purpose to a request; Layer 9 makes that binding explicit and reversible. No layer below may infer or escalate identity on the user's behalf.

The right to specific, bounded consent. Access that carries obligations—retention, redaction, watermarking, cross-domain use—must proceed only on the basis of consent stated in human terms and bound to the scope of a single request. Consent is not a banner to be accepted once; it is a time-stamped condition attached to a contract. SON's policy envelopes, enforced at consumption (Watchtower), carry those terms forward so results arrive with the obligations that shaped them, not as unexplained absences.

The right to explanation grounded in artifacts. AI that cannot show its work asks for trust it has not earned. A person is entitled to ask why and to be shown—not a reconstructed story—but the actual chain that justified an answer: the objects consulted, the layers admitted, the evidence chains cited, the confidence and reputational signals weighed, and the degradations applied. In SON, provenance is not a log-level option; it is a structural guarantee: Query Resolution Objects, consistency envelopes, and evidence chains make the "why-path" retrievable as a matter of protocol.

The right to revise by new commitment, not mutation. People change their minds. Systems must make that safe. Edits to scope—excluding a source, tightening a window, raising a threshold, seeking deeper corroboration—must be expressed as new contracts with fresh envelopes, not as silent mutations of prior artifacts. SON's z-axis layering preserves this discipline: facts remain in declarative layers; hypotheses and narratives live beside them; revision proceeds by addition rather than erasure.

The right to revocation without amnesia. The "right to be forgotten" is often framed as deletion from a single database; modern AI makes that naive. What is required is precision withdrawal: the ability to retire layered references and prevent future computation from relying on them, while preserving a faithful record of what was done under earlier terms. SON's separation of objects from layer-level assertions permits prospective revocation with auditable history—an honest answer to forgetting in systems that cannot un-train the past.

The right to minimal, proportional disclosure. Default exposure must be metadata, not payload; visibility must follow least-privilege by policy, not convenience by implementation. SON's Watchtower design treats decryption as an outcome of policy evaluation at consumption, so graphs can be shared openly while payloads remain sealed. Redaction and watermarking are not UI features; they are obligations that travel with results.

The right to economic transparency. People should not have to speak in SLOs and "gas," but they should never be surprised by them. Outcome-level choices—quick scan vs. deeper corroboration—must be translated by the application into enforceable parameters and authorized spend, and the system must disclose what was requested versus what was achieved, and what was authorized versus what was spent. SON makes cost and degradation part of the record, not hidden scheduler heuristics.

The right to portability of terms. Agency should travel with knowledge. When results cross systems, the identity stance, consent, scope, and obligations attached to the request must travel with them—or the receiving system must disclose the mismatch and seek fresh consent. SON treats these as first-class, time-stamped assertions that can be honored across federated layers without re-interpreting the person's will.

The right to opposability and remedy. Systems will be wrong. People must be able to contest an answer with structured feedback that becomes part of the epistemic process, not a thumb-icon that vanishes into a training set. In SON, acceptance, rejection, and qualification are Feedback Records linked to concrete artifacts; they recalibrate confidence and reputation without retroactively mutating prior claims.

These rights are not idiosyncratic to a single regulation; they are the operational expression of well-understood principles: the data minimization and purpose limitation of GDPR, the individual's interest in erasure and rectification, the scientific norm that a claim's provenance and confidence must be inspectable. Where today's AI often loses the thread—answers without sources, models without memory of how they learned what they purport to know—SON supplies the missing infrastructure: persistent objects for stable reference; layered relationships to keep fact, inference, and context distinct; contracts and envelopes so operations are bounded and reproducible; policy enforcement at consumption so privacy is honored in practice, not only in policy; and a user boundary where intent becomes binding and explanation becomes a right.

We propose this manifesto not as rhetoric but as a design test. Any system that claims to be robust should demonstrate, in its protocol, how a person can assert identity or withhold it; accept or refuse obligations; demand and receive explanations backed by artifacts; revise scope by new contract; revoke prospectively without erasing history; see costs and degradations as facts, not inferences; carry terms across systems; and contest results in ways the machine is obliged to record. SON offers these guarantees by construction. They do not slow innovation; they make innovation safe to adopt. They are the conditions under which stronger, more resilient AI can be built—and under which people remain authors of their intent rather than subjects of a machine's convenience.

Manifesto to Protocol Mapping

The manifesto articulates a set of human rights over data, privacy, and knowledge systems. SON 2.0 provides the protocol infrastructure that turns these rights into enforceable commitments. The correspondence between rights and mechanisms is not accidental: each protocol element was designed to guarantee one of these assertions.

Identity as a deliberate act. The manifesto demands that a person choose when and how to disclose identity. SON implements this through microidentities bound at Layer 8 and asserted at Layer 9. Contracts and envelopes always carry the identity stance chosen, and Watchtower ensures that lower layers cannot infer or escalate identity without consent.

Consent as a condition of access. Consent must be specific, bounded, and attributable. In SON, this is achieved by embedding obligations in policy envelopes that accompany contracts. Watchtower enforces these obligations at the moment of consumption, while audit logs preserve what was accepted and under which terms.

Explanation grounded in artifacts. The right to explanation requires that answers show their grounds. SON guarantees this through Query Resolution Objects, provenance chains, confidence scores, and evidence handles that must be carried forward with every response. Distributed GraphRAG formalizes this by requiring corroboration and evidence accumulation before promotion.

Revision by new commitment. The manifesto distinguishes revision from mutation. SON implements this with envelopes and contracts: changes in scope, sources, or thresholds produce new contracts under new envelopes. Historical states remain replayable, and revisions are additive, not retroactive.

Revocation without amnesia. Precision withdrawal is achieved in SON by retiring or pruning z-axis layers, isolating dependent inferences, and marking withdrawn assertions while leaving the historical record intact. This supports GDPR-style rights to erasure without corrupting provenance.

Minimal, proportional disclosure. The manifesto calls for least-privilege visibility. SON achieves this by exposing object metadata while sealing payloads under Watchtower-enforced obligations. Redaction, watermarking, and cache controls are protocol features, not UI conveniences.

Economic transparency. While users never see gas or SLO directly, SON requires that every request carry an envelope with authorized budgets, and that achieved versus requested outcomes are disclosed in audit trails. Applications at Layer 8 present this as human-meaningful quotes; the system records spend and degradation as part of the answer.

Portability of rights. The manifesto asserts that agency must travel with knowledge. SON's federated design allows consent, identity stance, and policy envelopes to accompany results across nodes. When terms cannot be honored downstream, the receiving system must either seek fresh consent or decline execution.

Opposability and remedy. Feedback must be more than an emoji. SON formalizes contestation through Feedback Records at Layer 7, which tie user judgments (accept, reject, qualify) to specific artifacts. These become part of the epistemic cycle and adjust reputation or confidence without retroactively mutating claims.

Together, these mappings show that the rights articulated in the manifesto are not aspirational slogans but protocol-level guarantees. SON provides the structural features—layers, envelopes, contracts, provenance, and policy enforcement—that make these rights concrete, auditable, and reproducible.

Architecture Overview

The architecture of Shared Object Networking (SON) 2.0 is intentionally thin: a three-tier, nine-layer protocol whose purpose is to make knowledge persistent, governable, and replayable across heterogeneous systems. Rather than prescribing a platform, SON fixes only the boundary contracts—what enters a layer, what leaves, and what the gate records. Everything else—storage engines, indices, or reasoning strategies—remains an implementation choice. This thinness is essential: interoperability rests on stable invariants, not shifting vendor practices.

At the foundation are the data layers (L1–L3), which transform raw inputs into stable, portable objects and preserve them in distinct z-axis strata. External events, documents, or media streams do not count as SON knowledge until they are objectified: each object carries a stable identifier, declared schema, provenance, and an obligations envelope (policy metadata enforced at consumption). Layer 2 embeds these objects into per-layer graphs that capture relationships without collapsing epistemic contexts. Layer 3 consolidates within a stratum—improving quality, detecting gaps, and strengthening internal consistency—so each layer endures long enough to be reused, contested, or audited. A biography of Miles Davis may become a Person object in one layer, his albums Work objects in another, and a recording session a Performance object in a third; the strata remain distinct rather than collapsed into a single profile.

Above them sit the reasoning layers (L4–L6). Here SON specifies how goals are decomposed, work delegated, and results returned. Layer 4 reasons across strata, aligning or holding competing claims in tension under the discipline of Confidence, Reputation, and Evidence (CRE). Layer 5 orchestrates sub-tasks, allocates “gas” (an explicit computational budget), and binds obligations to the reasoning process. Layer 6 emits Query Result Objects (QROs): compact, durable records that tie an answer to its exact inputs, evidence, and policy snapshot. A query such as “collaborations in 1959” yields not only a list of musicians but also a QRO that records which objects were consulted, which thresholds applied, and what obligations govern reuse.

Finally, the user layers (L7–L9) connect SON to its human and institutional environment. Layer 7 standardizes exported model views—feature sets, embeddings, or subgraphs—ensuring that what leaves SON for downstream engines remains grounded in audited objects. Layer 8 hosts applications that consume SON artifacts without bypassing gates; their authority derives from provenance and obligations already attached. Layer 9 enforces sovereignty and governance at the point of use: a Watchtower service evaluates whether a user may decrypt a sealed payload, whether obligations apply, and under what conditions knowledge may be shared or withdrawn. In aggregate, these layers make SON not only a substrate for computation but also a system of record for why the system believed what it believed at a given moment.

This tiered architecture does not prescribe a single ontology or reasoning strategy. Instead, it guarantees that knowledge travels as objects with provenance, strata remain distinct, durable reasoning leaves behind QROs, and obligations are enforced at consumption. These minimal commitments are sufficient to make knowledge interoperable across implementations, auditable across time, and contestable across disciplines.

Motivation and Position

Modern AI stacks are powerful but brittle. Retrieval-augmented generation (RAG) improves factuality but remains ephemeral: pipelines assemble context on the fly and discard it, leaving nothing auditable or replayable. Large language models (LLMs) excel at fluency but not accountability; provenance is opaque and error correction ad hoc. Knowledge graphs provide structure but tend to be proprietary and difficult to merge.

The result is siloed systems where answers expire with sessions, provenance is missing, and trust depends on vendor implementations.

The motivation for Shared Object Networking (SON) is to break this pattern by introducing a protocol for knowledge. Protocols succeed when they are thin and durable: TCP/IP and HTTP/HTML worked not because they were rich in features, but because they defined just enough for heterogeneous systems to interoperate. SON applies the same principle. It specifies a minimal set of invariants—object identity, schema declaration, provenance and obligations, z-axis persistence, gate decisions, and replayable Query Result Objects (QROs)—that allow knowledge to remain portable, governable, and accountable across implementations.

This stance is grounded in a lineage of prior work. The original Shared Object Networking (2009–2010) introduced object packetization and z-axis layering, showing how declarative facts could be separated from contextual or inferred interpretations. Persistent Object Networks (2024) extended this model with distributed validation, confidence-weighted relationships, and pattern crystallization into stable strata. Most recently, Distributed GraphRAG via SON (2025) operationalized the approach in a modern stack: hybrid mapping to well-known objects, vector and graph alignment without a global schema, and agentic modules that discover, validate, and promote knowledge under explicit provenance and trust constraints. Each step demonstrated feasibility; SON 2.0 consolidates them into a protocol surface that can be independently implemented and widely adopted.

Positioning SON against related approaches clarifies its role. Compared with RAG, SON produces durable artifacts: every query leaves behind a QRO that records inputs, evidence, thresholds, and obligations, enabling replay and contestation. Compared with monolithic LLMs, SON externalizes knowledge into objects and layers so models reason against explicit, policy-bound artifacts rather than memorized weights. Compared with standalone knowledge graphs, SON is ontology-agnostic and protocol-minimal: it standardizes packets, persistence, and replay without mandating any particular ontology or reasoning strategy.

The result is not another database, model, or application, but a protocol: a set of agreements about how components interact so they can be interchangeable, extensible, and flexible. By fixing only what is persistence-binding, replay-critical, wire-visible, or consumption-critical, SON enables heterogeneous systems to cooperate safely without central control. Implementations remain free to choose data stores, indices, inference methods, or orchestration frameworks. What they must share are only the thin contracts that make answers accountable and knowledge durable.

Protocol vs. Implementation

Before drawing the stack, it is essential to mark the boundary: what counts as protocol and what belongs to implementation. The promise of SON 2.0 is that many small, specialized skills can be composed into durable reasoning systems without fragility. That promise only holds if there is a clear, shared understanding of what every conforming system must honor and where they are free to diverge.

A commitment qualifies as protocol only if it satisfies one of four interoperability tests. It must govern something that is persistence-binding, shaping what can be durably written into a layer and how that act is recorded; or it must be replay-critical, without which an answer cannot be reconstructed “as of” its moment; or it must be wire-visible, determining what is carried across organizational or system boundaries; or it must be consumption-critical, defining how obligations are enforced at the moment of use. Rules that fail all four tests may be useful practices, but they are not protocol: two independent systems can still interoperate safely without them.

Within that frame, this paper standardizes a thin set of invariants. Knowledge must travel as objects rather than opaque blobs: each object carries a stable identifier, a declared schema, a verifiable evidence chain, and an obligations envelope that moves with it. Knowledge must persist in ordered layers rather than amorphous heaps: SON specifies nine layers, grouped into three tiers, each with a defined input, output, and gate. Every point where artifacts become durable must pass through a gate, an inspector that records the explicit decision—admission, transformation, quarantine, or rejection—together with the policy snapshot in force. Reasoning must be replayable: answers are emitted as Query Result Objects (QROs), compact records that bind a goal to its inputs, the evidence used, the thresholds applied, and the costs incurred. Finally, acceptance must be disciplined: promotion into more trusted layers must be explicit, diversity-aware, and recorded, even if the precise mathematics of Confidence–Reputation–Evidence (CRE) are left to later papers.

Everything beyond this surface—choice of storage engine, index structure, extractor, or interface—is implementation. Teams are free to innovate and optimize, and indeed are expected to do so, without breaking interoperability.

To signal the strength of commitments, we adopt the familiar convention of requirement keywords. A rule marked MUST is required for interoperability or safety, and a system that violates it is no longer speaking the protocol. A rule marked SHOULD expresses strong guidance: deviations are permitted only with a clear, recorded reason and may invite reduced trust or reputation. A rule marked MAY is optional; the protocol neither requires nor forbids it. In all cases, we avoid undefined shorthand, introducing acronyms and terms in full—Query Result Object (QRO), Confidence–Reputation–Evidence (CRE), and separation of duties (SoD)—so that the protocol can be read without hidden context.

The remainder of this paper develops the normative surface in detail: the layer contracts, the behavior of gates, the handling of obligations, and the guarantees of replay. These are the thin commitments that allow two independent implementations to work together without central control. Later papers in the series provide concrete strategies for agent design, orchestration, access control, security posture, and applied domains such as security operations. Those are valuable guides but not requirements.

By binding only what is persistence-critical, replay-essential, wire-visible, or consumption-enforced, SON remains deliberately light. Yet these few rules have depth: they ensure that every object carries provenance and obligations, every durable change leaves a signed decision, and every answer can be reconstructed as it was believed at a moment in time.

Implementation Considerations and Preview of Subsequent Work

The boundaries drawn above deliberately restrict SON to a thin protocol surface. Yet the act of defining a protocol is never detached from implementation: it is shaped by a sense of what realizations are possible and what mistakes they must prevent. SON 2.0 is no exception. While the protocol commits only to objects, layers, gates, and replayable reasoning, the design has been informed by practical experience with contemporary graph databases, workflow engines, and agent frameworks. These considerations do not constrain the protocol, but they illustrate how it can be embodied in working systems without narrowing the space of conforming designs.

A straightforward implementation path helps to make the abstraction tangible. One can imagine objects persisted in a relational store, projected into graph layers for traversal, and guarded by inspectors that serve as the only points of durable write. Reasoning agents can be orchestrated by a workflow engine that enforces separation of duties and records every decision, while gas budgeting provides an explicit accounting of effort so that queries cannot expand unbounded. At the edge, a Watchtower service enforces obligations at the moment of consumption, ensuring that sensitive payloads remain encrypted until policy permits their release. None of these choices are required, but each shows how the thin commitments of the protocol map cleanly to present-day technologies.

The purpose of sketching such possibilities is not to legislate architecture but to show feasibility. Readers should see how the invariants described in the architecture overview—stable objects, distinct strata, inspected gates, replayable QROs—can be realized in practice without bending the protocol toward any single database or inference engine. Later papers in this series develop these non-normative dimensions in depth: how agents embody the Confidence–Reputation–Evidence discipline as goal-seeking housekeepers; how Watchtower formalizes obligation enforcement as a separation of decision and execution; how gas becomes the economic and security substrate of reasoning. Here, they appear only as foreshadowing, to suggest that the minimal surface fixed by SON 2.0 is not only theoretically sufficient but operationally robust.

CRE Agents as Goal-Seeking Housekeepers (Layers 2–6)

The architectural overview established SON as a protocol of thin but durable commitments: knowledge must persist as objects with provenance, strata must remain distinct, gates must record the decisions that make changes durable, and reasoning must leave behind replayable artifacts. Within that frame, the Confidence–Reputation–Evidence (CRE) discipline supplies a mechanism for quality control. It does so not by legislating a single algorithm but by defining the conditions under which agents operate: their work must be explicit, bounded, and replayable, and their outcomes must be recorded so that future systems can contest or repeat them.

From Layer 2 onward, CRE agents act as housekeepers. They are not designed to replace user-facing reasoning but to maintain the epistemic hygiene of the substrate. Their function is to propose refinements, detect inconsistencies, and test connections, always under constraint and always leaving a trace. In a contextualization layer, this may mean probing whether two objects are duplicates, or whether a relation suggested by schema features is robust. In a consolidation layer, it may mean testing the impact of stricter thresholds, reindexing a subgraph, or validating whether contradictions are persistent rather than transient. Each proposal carries its own evidence and a quantified confidence; acceptance or rejection is itself a recorded event.

As the system moves into the reasoning tier, the agents' role expands from local housekeeping to cross-layer coordination. At Layer 4, they arbitrate among competing claims, holding multiple perspectives in tension when evidence does not justify premature consensus. At Layer 5, their proposals are framed as explicit plans—requests for additional evidence, invitations to query a neighboring stratum, or suggested decompositions of a complex goal. These plans are accepted, degraded, or declined against declared budgets and service-level objectives, ensuring that reasoning remains proportional to available resources. At Layer 6, the result of this disciplined process is captured in a Query Result Object (QRO): not merely an answer, but a structured account of which layers were consulted, which thresholds applied, how confidence shifted, and what costs were incurred.

In this way, CRE agents operationalize the principle that error should become fuel rather than contamination. When an inconsistency is found, it is not erased but preserved as a contestable claim, its provenance intact. When an experiment fails to increase confidence, the failure is recorded, reducing the reputation of the strategy without corrupting the corpus. Over time, reputation accrues to agents, sources, and methods that prove effective, and diminishes for those that do not. The discipline is therefore cumulative: every run leaves behind not only results but also a history of how those results were obtained, so that trust becomes grounded in performance rather than assertion.

These agents are deliberately framed as housekeepers because their task is to maintain the conditions under which reasoning remains transparent, accountable, and reproducible. They do not dictate outcomes; they enforce the separation of duties that makes outcomes contestable. Later papers develop two mechanisms that complete this picture. Gas accounting ensures that agents act as economic citizens, expending measurable effort for every refinement they attempt, while Watchtower enforcement ensures that obligations attached to objects are honored at the point of use. In the present paper it is enough to recognize that CRE agents, operating across Layers 2–6, embody the protocol's minimal commitments: persistence with provenance, separation without collapse, and reasoning that can be replayed as it was believed at a moment in time.

Economic and Security Dimensions of CRE Agents

The operation of CRE agents does not occur in isolation. To improve the quality of knowledge, they must often reach beyond their local strata: consulting federated repositories, negotiating with distributed autonomous organizations (DAOs) for access to curated datasets, or invoking external services through an API or Model Context Protocol (MCP). These outward-facing calls raise two intertwined challenges: they consume resources, and they expose the system to potential poisoning or misuse. SON addresses both by binding such activity to the dual disciplines of gas accounting and z-axis isolation.

Economically, every external action is treated as a transaction. An agent cannot issue an unbounded query or pursue endless corroboration without consequence; it must expend gas against its declared budget. Gas is not an abstract metaphor but a concrete measure of computational and retrieval effort, enforced by the same gates that regulate persistence. In practice, this makes participation sustainable: agents that waste resources see their reputation decline, while those that deliver measurable confidence gains per unit of gas rise in standing. More positively, it creates a marketplace of contribution. A DAO that curates high-quality reference data can charge for access, and agents that spend wisely can demonstrate efficiency by producing more trustworthy results at lower cost. Economic citizenship thus becomes part of epistemic credibility.

Security considerations are equally central. By requiring that every external call be isolated in its own z-axis layer, SON ensures that the consequences of corruption or poisoning are confined. If an external feed introduces compromised claims, those claims remain bound to the layer in which they were ingested, together with their provenance. Pruning that layer severs their influence without rewriting the rest of the graph. Moreover, because each call is bound into a Query Result Object and subject to Watchtower review at the moment of consumption, any obligations declared by the external party—license restrictions, privacy constraints, or export controls—must be enforced before the information can shape downstream reasoning.

The result is a system in which CRE agents are not only curators of epistemic quality but also responsible citizens of a shared economy and guardians of security boundaries. Their exploratory behavior is not suppressed, but it is disciplined: they are free to federate, to test, and to experiment, yet each action carries a measurable cost and remains quarantined to the layer that produced it. In this way, SON reconciles openness with control. Error becomes fuel rather than contagion, and external contributions enrich the substrate without undermining its integrity.

Synthesis: How the Layers Cohere

Taken together, the three tiers specify a single, continuous contract for how knowledge is made durable, reasoned over, and consumed. The data tier ensures that what enters the system is objectified and situated: objects are created with stable identity, declared schema, provenance, and an obligations envelope; relations are admitted into per-layer graphs without collapsing distinct epistemic contexts; consolidation improves the internal quality of each stratum without rewriting the objects it hosts. The reasoning tier then operates on that prepared substrate under explicit constraint: cross-layer analysis preserves tension among competing claims when evidence requires it; orchestration plans work under a finite gas budget; and each run is summarized as a Query Result Object (QRO) that binds goals to inputs, thresholds, obligations, and cost, making answers replayable "as of" their moment. Finally, the user tier standardizes what leaves the system and how it is governed: exports are bounded and provenance-preserving, applications inherit protocol guarantees rather than reinvent them, and a Watchtower enforces obligations at the point of use.

This composition is intentional. By fixing only the boundary contracts—what a layer accepts, what it emits, and what the gate records—SON allows implementations to vary internally without risking interoperability. A Person object, a Work object, and a Performance object may be realized in different stores or indexed by different engines, yet the moment they cross a layer boundary the same minimal invariants apply: identity and schema must be declared; evidence must be named; obligations must travel; and any durable admission must leave a signed decision. Because those invariants persist through the reasoning tier and into the user tier, a QRO produced by one implementation is intelligible—and auditable—by another, and an exported model view can be consumed without losing lineage or policy.

The architecture is also cumulative. Each layer adds structure without erasing what came before: objectification does not foreclose alternative contexts; consolidation does not suppress competing claims; cross-layer reasoning leaves behind a narrative of its own decision path; consumption-time governance enforces obligations without rewriting the underlying objects. Error becomes a controlled input rather than a contaminant. A disputed session date, for example, remains available as a contestable claim with intact provenance until additional evidence justifies promotion or demotion; either way, the outcome and its cost are recorded in a QRO and can be revisited later under the same conditions.

What follows in this paper develops the implications of this synthesis. We make the protocol/implementation boundary explicit, then show how non-normative mechanisms—CRE agents, gas budgeting, and Watchtower enforcement—fit within that boundary to keep knowledge both reusable and accountable. The aim is not to prescribe a platform, but to demonstrate that the thin, stable commitments fixed here are sufficient to compose many different systems into a single, durable substrate for reasoning.

Tier I — Data Layers

Tier I establishes the durable footing on which the rest of SON stands. Its mandate is simple and strict: bind boundary traffic to a contract; convert traffic into objects with identity, schema, and evidence; preserve those objects in z-axis strata; and improve each stratum's internal quality without collapsing epistemic boundaries. The tier comprises four layers—boundary management (L0), objectification and persistence (L1), layer construction and sessioned graph access (L2), and intra-layer optimization (L3). Together they transform flux into replayable knowledge that Layer II can reason over without guessing what the substrate meant.

Protocol roles, not product choices. Each layer implements the same two protocol roles: a worker agent that performs functional duties (e.g., connecting, validating formats, objectifying, building graphs, running optimizations), and a security agent that gatekeeps all durable actions (authenticate, authorize, decide, and record). These agents are replayable actors: they maintain SON identities and publish capability descriptors; their decisions—admit, transform, quarantine, export—are persisted as signed, durable records linked to the active policy snapshot and contract context. A durable change in Tier I is therefore never an accident: it is a decision with a date, a reason, a scope, and a signature.

From boundary to substrate (L0 → L1). Layer 0 binds every ingress and egress to a ServiceContract, anchoring who spoke to what, under which scopes and obligations, and when. It asserts transport integrity and contract correctness but no semantics. Layer 1 then converts contract-bound inputs into SON objects with stable identity, declared schema, obligations envelopes, and evidence chains; separation of duties is explicit—objectification is proposed by the worker and committed (or rejected) by the security agent. The result is a substrate of identifiable, schema-bound objects where meaning travels as structure and evidence, not as an undocumented convention.

From substrate to layers (L2 → L3). Layer 2 places objects into z-axis strata—parallel, persistent contexts—and provides authorized sessions for traversal within and across selected strata while preserving layer provenance. The layer favors maintenance of separation over early consensus: relationships carry their source layer identity so that competing views can coexist and be reasoned about later without contamination. Layer 3 optimizes each stratum from within: it runs targeted, reversible interventions (gap detection, link prediction validation, dedup boundaries, threshold tuning), records QualityAssessment artifacts, and issues targeted acquisition requests to the boundary where evidence is insufficient. Improvements are measured, auditable, and attributable; failures become fuel for learning rather than silent edits.

What Tier I hands to Tier II. The outputs of Tier I are objects with identity and obligations; per-layer graphs with preserved provenance; session primitives that define what combinations of strata a reader may traverse; and quality/coverage signals (e.g., QualityAssessment metrics, optimization decisions, and targeted acquisition hints). Tier II does not re-litigate transport, identity, or layer boundaries; it assumes these invariants and concentrates on cross-layer reasoning, orchestration, and Query Result Objects (QROs) that bind answers to their inputs, thresholds, and costs.

Economic and CRE hooks (foreshadowing). While the economics of reasoning are developed in later sections, Tier I already emits the signals that make budgeting and confidence discipline possible: every boundary operation yields effort metrics; every object carries an evidence chain; every layer emits quality indicators; and every durable change has an attributable decision record. These become the raw materials for gas-aware planning and the Confidence–Reputation–Evidence (CRE) discipline that governs acceptance and promotion above the substrate.

Security intent across Tier I. Tier I is a security boundary as much as a data boundary. Across L0–L3, the intent is: (1) contain credentials and secrets—never copied, always enveloped and only decrypted in the appropriate layer's security agent; (2) eliminate silent bypass—all durable actions flow through a gate whose decision is signed and auditable; (3) constrain poisoning blast radius—malformed or suspect inputs are quarantined with reason codes; layer provenance prevents contamination across strata; and (4) anchor provenance at the edge—every change is linked to a policy snapshot, contract context, actor identity, and integrity hashes. The specific controls and failure modes are detailed per layer (closing paragraph in each sub-section) and elaborated in the Threats paper; the protocol requirement here is that each control be decidable, attributable, and replayable at its boundary.

Reading the rest of Tier I. The four sub-sections that follow describe each layer's role, contracts, and gates, the worker/security agent duties, the decision records they emit, and the security posture specific to that boundary. The normative rules (MUST/SHOULD/MAY) and conformance profiles are in the Tier I Appendix; here we establish the narrative through-line: boundary truth → object truth → layer truth → quality truth, so that higher layers never have to infer what the substrate meant, only how far they want to trust it.

Layer 0: Protocol Boundary Management

Layer 0 is SON's point of contact with the external world. It sits below objectification and above heterogeneous systems, speaking two languages at once: the native protocols of those systems (databases, message buses, REST/graph APIs, filesystems, sensors) and the object-centric conventions of SON. Its purpose is not to interpret content but to bind transport and context: to carry data across the boundary with a precise contract, cryptographic provenance, and a decision record that can be replayed "as of" the moment it entered or left the substrate. In short, Layer 0 establishes the conditions under which every subsequent layer can reason safely.

Contract context as the unit of ingress (and egress)

Every movement across the boundary is mediated by a ServiceContract—a SON object that declares who is speaking to what, why, and under which scopes and obligations. A ServiceContract names endpoints, authentication methods, scopes/tenants, usage purposes, and durability expectations; it also binds these to cryptographic identities and timestamps. When a Layer 0 actor retrieves data, it attaches a contract-context assertion to the payload: "Here is data retrieved under ServiceContract S at time T, with declared format F, and with integrity hashes/signatures H." The same applies to exports in the other direction: Layer 0 does not "just push" SON objects outward; it performs contract-bound publication that is attributable and replayable. This precision of contract context—treating "the same API with a different scope" as a distinct relationship—is critical for the institutional learning that Layer 1 performs later. Mis-scoped ingress is not a minor bookkeeping error; it collapses distinct epistemic boundaries and contaminates optimization.

Agents at Layer 0: worker and security classes

Although SON is a protocol, not a deployment, it requires that two agent classes be present at each layer: a worker class that performs the layer's functional duties, and a security class that adjudicates access, enforces policy, and records the decision. At Layer 0, the worker agent performs connection management, format validation, transport integrity checks, and conversion to SON-compatible carriers (without asserting semantics). The security agent—the boundary "Inspector" at this level—validates ServiceContracts, authenticates to external systems using sealed credentials, applies allow/deny and rate constraints, and signs the contract-context assertion that accompanies each ingress/egress. These are protocol roles, not product choices; later sections spell them out for other layers, but we foreshadow them here to clarify responsibility: no Layer 0 write is durable unless the security agent has recorded its decision.

Replayable actors and attributable boundary work

Layer 0 agents are replayable actors: they are themselves SON citizens with durable identity (keypairs), published capability descriptors, and auditable decision histories persisted as SON objects. This matters for two reasons. First, provenance must encompass the actor that performed the boundary work, not only the payload and the contract. Second, replayability requires more than content hashing; it requires the ability to restore the who, what, and why of a boundary event—including the policy snapshot and the precise version of the ServiceContract that governed it. Practically, each boundary operation emits a compact IngressRecord or EgressRecord object (hashed, signed, and linked to the active policy snapshot and ServiceContract). Later layers need not re-validate transport; they verify the record and proceed.

Semantics deferred, hygiene enforced

Layer 0 guarantees transport integrity and contract accuracy; it explicitly does not assert business meaning. It may check that a blob is valid JSON, that a CSV conforms to its declared delimiter and column count, or that a

Kafka partition offset sequence is complete—yet it never claims “this is a Process event.” That is Layer 1’s job. Still, hygiene is enforced: malformed inputs are quarantined with a reason code and an attached diagnostic bundle; duplicate deliveries under the same ServiceContract and content hash are deduplicated with a pointer to the original decision record; and drift in external system capabilities (e.g., a changed schema or pagination model) results in updated ServiceContract objects and explicit rotation events. The rule is simple: Layer 0 tells the truth about transport, and records the truth it tells.

Bidirectionality and purpose-binding

Layer 0 governs both ingress and egress under the same discipline. When SON exports, it does so by materializing an external deliverable under a ServiceContract that binds the object’s obligations to the destination’s terms. If a payload must be transformed (format conversion, redaction, watermarking), the worker performs the transformation under a plan, and the security agent authorizes and signs the act, yielding an EgressRecord. Critically, purpose-binding established at this boundary persists downstream: higher layers can reason knowing why the data was brought in (or sent out), avoiding silent reuse beyond the original purpose without a new decision.

Foreshadowing economic discipline

While gas metering is developed later, its implications appear at Layer 0: connecting to a high-latency API with tight rate limits, revalidating signatures, or replay-fetching large windows consumes resources. Layer 0 therefore reports effort and outcomes (e.g., round-trip counts, retry paths, bytes moved) into the decision record so that subsequent reasoning and orchestration can align “confidence gain per unit gas” with boundary costs. The protocol does not fix prices; it requires that boundary effort be counted, attributable, and reusable in planning.

Why Layer 0 matters to Layer 1 (and beyond)

Layer 1’s objectification quality depends on stable learning boundaries. If the same external system is accessed under two ServiceContracts with different scopes or tenancy, they are not “the same feed”; they are distinct contract contexts that Layer 1 should treat separately until evidence justifies consolidation. By making the contract context explicit and durable, Layer 0 prevents early collapse of epistemic strata and supports selective adoption, contradiction handling, and later cross-layer consolidation under CRE.

Security posture at Layer 0 (intent and rigor)

Layer 0 is a security boundary as much as a transport boundary. Its security intent is fourfold: (1) contain credential risk (credentials are sealed as object payloads, decrypted only by the Layer 0 security agent in a controlled enclave; no lateral copy is allowed); (2) eliminate silent bypass (the security agent is the sole authority for boundary operations; all decisions are signed and auditable); (3) constrain poisoning blast radius (ingress assigns contract context and quarantine status; malformed or suspect inputs are isolated with a referenceable record rather than silently dropped or—worse—silently accepted); and (4) anchor provenance at the edge (every boundary operation leaves a verifiable chain: actor identity, ServiceContract version, policy snapshot, integrity hashes, and effort metrics). Detailed mitigations—e.g., man-in-the-middle resistance, credential rotation and attestation modes, anomaly detection on boundary traffic, and fail-safe degradation—are specified in the Layer 0 appendix and expanded in the Threats paper; the protocol requirement here is that each of these controls be decidable, attributable, and replayable at this boundary.

Research and Technology Foundation (Layer 0)

Layer 0—the protocol boundary where external systems meet a persistent knowledge substrate—reframes established results from data integration, provenance, security policy, and API governance into a minimal, interoperable contract. Its novelty is not a new transport or a new cryptographic primitive, but the stipulation that boundary work be contract-bound, attributable, and replayable.

Contract-bound integration.

The past decade's shift toward API-first and service-oriented architectures has normalized formal interface descriptions, typed payloads, and versioned schemas. In knowledge settings, this is complemented by long-standing results in ontology/ schema discipline and linked-data practice: interoperability improves when structure is declared and names are portable (e.g., conceptual modeling and semantic web standards) ⍰. Layer 0 absorbs these lessons as protocol: every ingress or egress is mediated by an explicit service contract that binds endpoints, authentication scope, purpose, and format—so higher layers never "guess" what a payload meant.

Provenance and replay at the edge.

Data provenance research established that how information entered a system is as critical as what it says; scientific and workflow communities treat lineage as a first-class artifact for audit and reuse. Layer 0 imports that stance to the interface: each boundary operation yields a small, signed ingress/egress record that captures request/response hashes, timestamps, contract identity, and policy snapshot, making edge events replayable "as-of" their moment and portable across implementations .

Separation of decision and enforcement.

Access-control literature distinguishes between policy decision (who may do what, under which attributes) and policy enforcement (perform or refuse the act). Modern RBAC/ABAC systems, and their use in data platforms, show that this PDP/PEP split yields clearer audits and safer systems. Layer 0 encodes the split as protocol: a security agent authorizes or denies boundary operations under a recorded policy snapshot; a worker agent performs the authorized transport and signs what actually happened. This is not an implementation hint—it is the minimal contract required for two independent systems to interoperate safely (decision is attributable; enforcement is auditable) .

Cryptographic hygiene and envelope control.

Industry practice has converged on envelope encryption (per-object data keys wrapped by KMS/HSM keys) and authenticated channels for at-rest and in-flight protection. Layer 0 adopts these as assumptions rather than inventions: credentials and data-encryption keys are never exposed in clear at the boundary; payloads can circulate sealed, while metadata remains routable so objects are discoverable and linkable without disclosing contents. This pairing—sealed payload, visible header—is what later enables privacy-preserving reasoning.

Declarative structure over ad-hoc text.

Knowledge engineering and KR/KG literature argues for objects with schemas and relations with types in preference to opaque documents; the web stack's RDF/OWL lineage makes the same point for global exchange (typed identifiers travel farther than prose) ⍰. Layer 0's duty is not to assign semantics, but to deliver format-valid, contract-accurate carriers so that Layer 1 can objectify without ambiguity.

Streaming and continuous ingestion as first-class concerns.

Results from lifelong and incremental learning emphasize that systems improve when new evidence can be added without retraining monoliths or re-ingesting whole corpora ▯. Layer 0's small, durable boundary records are the unit of "newness": they can be replayed, audited, or rolled back independently, allowing higher layers to evolve without rebuilding the world.

Why this assembly is new.

None of these ingredients are exotic: API contracts, schema declaration, PDP/PEP, envelope encryption, and provenance have each matured in their own communities. The intellectual contribution of Layer 0 is to compress them into protocol invariants that any conforming system must share:

- every boundary action is contract-bound;
- every action yields a replayable ingress/egress record;
- decision and enforcement are separated and attributable; and
- payload privacy and header routability coexist by design.

Everything else—the choice of API gateways, message buses, schema validators, or cryptographic libraries—is deliberately left to implementations. This is how Layer 0 turns a patchwork of best practices into a thin, interoperable surface that higher layers can trust and replay without vendor lock-in ▯.

Layer 1: Object Persistence and Objectification

Layer 1 performs the act of objectification: it transforms contract-bound inputs from Layer 0 into SON objects with identity, a declared schema reference, an obligations envelope, and an evidence chain—under explicit separation of duties. The layer is staffed by two protocol roles: a worker agent (the Objectifier) that proposes semantics, and a security agent (the Inspector) that adjudicates, commits, or refuses the write. Both are replayable actors: they hold durable identities, publish capability descriptors, and leave signed decision records bound to the policy snapshot and ServiceContract in force.

Schema selection is local, not prescribed by the protocol

Layer 1 does not impose a global schema catalog. Each Objectifier may maintain a local set of admissible schemas it is willing or able to use (by domain, contract, or deployment). The protocol cares only that the chosen schema be declared (namespaced, versioned) and that the decision be replayable—not which schema family is selected. In practice, Objectifiers often publish a schema-selection policy (versioned SON object) for transparency, but this is a deployment convention, not a protocol requirement. Downstream layers and external implementations can then understand, audit, or critique the policy the Objectifier honored at the time of admission.

Result: Objectification proposals cite the selected schemaRef and may cite a local objectifierPolicyRef when offered. The protocol requires only that these references be durable and verifiable, not that any particular catalog be universal.

From contract-bound payloads to proposals (separation of duties)

Given a validated ingress bundle (payload + contract context) from L0, the worker extracts features and proposes: a candidate Object with stable identifier, schemaRef, an obligations envelope (derived from the ServiceContract and/or publisher intent), and a preliminary evidence chain rooted in the L0 IngressRecord. The proposal comes with a compact ObjectificationDecision artifact describing why the schema was selected, which alternatives were considered, and which feature signals mattered. The security agent—the Inspector—is the sole writer of record: it verifies proposal conformance to policy and contract, checks deduplication and collision, evaluates envelope compatibility, and issues a decision (admit, transform, quarantine, reject). A signed DecisionRecord links the outcome to (a) the proposal hash, (b) the policy snapshot, (c) the ServiceContract version, and (d) integrity hashes or other invariants when present. Only admit/transform create or update durable Objects; quarantine and reject record reason codes and diagnostics.

Replayable semantics. Because both the proposal (ObjectificationDecision) and the Inspector's DecisionRecord are SON objects, a future reader can reproduce what was proposed, what was admitted or refused, and under which policy snapshot and contract—the semantic equivalent of Layer 0's transport replay.

Integrity artifacts are a SHOULD, not a MUST

Integrity artifacts (e.g., content hashes, signatures, canonicalization claims) are best practice at object admission and thus a SHOULD at Layer 1—not a universal MUST. Some data and schemas benefit unequivocally from object-level hashes (for deduplication, equivalence proofs, or change detection); others are volatile by design (e.g., rapidly mutating telemetry slices) where hashes add noise or cost without clear value. Accordingly:

- The Objectifier SHOULD include integrity artifacts when they materially improve replayability, deduplication, or equivalence;
- The Inspector SHOULD record when integrity artifacts were present, verified, or intentionally omitted (with a reason code);
- If a local schema in the Objectifier's catalog requires a particular integrity invariant, the Inspector MUST enforce that requirement for proposals citing that schema.

This approach keeps the protocol thin (no global MUST), while letting schema families or local policy raise the bar where it is justified and auditable.

Obligations envelope and provenance (enforcement is cross-layer)

The obligations envelope attached at admission is derived from the ServiceContract (format, scope, purpose, write authorities) and any publisher-declared restrictions, and may be shaped by schema-implied duties (e.g., fields marked as restricted). Layer 1's responsibilities are to bind provenance (which contract and policy snapshot informed the envelope), maintain monotonicity (later enforcement can tighten policy; silent weakening is denied and recorded), and ensure the envelope is internally consistent with the proposed schema and contract. Watchtower is not a Layer 1 singleton—it is a cross-layer mechanism that enforces obligations at consumption; Layer 1 makes that enforcement decidable by recording envelope ancestry and integrity of the admission act.

Export as objectification in reverse

Export is governed by the same gate. A request to publish Objects to an external system is purpose-bound and scoped by a destination ServiceContract (write authorities, allowed formats, and transforms). The worker prepares a transformation plan (format conversion, redaction, watermarking); the Inspector authorizes or denies, recording an EgressRecord bound to the destination contract, policy snapshot, and integrity of the emitted artifact where applicable. As at ingress, no weakening of obligations is permitted at export; exceptions require a new envelope and decision trail.

Institutional learning (contract-scoped and auditable)

Layer 1 captures learning about objectification explicitly—not by mutating code paths. When a downstream layer surfaces misclassification, schema mismatch, or dedup boundary issues, it emits ErrorCorrection objects; the Inspector validates these and persists contract-scoped ContractLearningPattern updates (versioned, signed) that inform future schema selection and proposal checks for the same ServiceContract. This makes learning auditable, reversible, and localized: deployments can compare learning versions, roll back bad heuristics, or quarantine suspect patterns without corrupting the object store.

Handoff to Layer 2

Layer 1 outputs: (i) Objects with stable identity, declared schemaRef, obligations envelope, and evidence chains rooted in L0/L1 records; (ii) ObjectificationDecision artifacts (proposal/explain); (iii) DecisionRecords (the Inspector's signed admission/refusal trail); and (iv) learning objects (ErrorCorrection, ContractLearningPattern). Layer 2 can assume identity, schema declaration, and envelope provenance have been decided and recorded; it need not infer semantics from raw payloads.

Security posture at Layer 1 (intent and rigor)

The security intent at this layer is to (1) contain semantics behind a gate—the Inspector is the only writer of record; all durable changes are signed and auditable; (2) respect contract-defined obligations—envelopes reflect the source contract and publisher intent, with monotonicity enforced at admission and export; (3) separate enforcement from admission—Watchtower enforces obligations at consumption across layers; L1 binds provenance so enforcement is decidable; and (4) constrain poisoning blast radius—malformed/policy-incompatible proposals are quarantined, not silently accepted or overwritten, with reason codes tied to the ingress contract and policy snapshot. Operational mitigations (parser attestation, optional integrity verification regimes, anomaly detection on schema drift, secret-handling constraints) are detailed in the Layer 1 Appendix and in the Threats paper; the protocol requirement is that controls remain decidable, attributable, and replayable at this gate.

Research and Technology Foundation (Layer 1)

The role of Layer 1—transforming flux into durable objects under separation of duties—draws on multiple established traditions: object-centric knowledge representation, schema theory, provenance and evidence management, and secure write-audit protocols. Each has been studied extensively, but they have rarely been assembled into a minimal, interoperable contract for knowledge systems.

Object-centric representation.

The idea that knowledge should be broken into discrete, addressable objects rather than stored as amorphous blobs is deeply rooted in both database systems and knowledge engineering. Work on entity–relationship models (Chen, 1976), object-oriented databases (Atkinson et al., 1989), and later knowledge graph practices (Hogan et al., 2021) demonstrates the value of encapsulating identity, attributes, and relationships into stable carriers. Layer 1 reframes this as protocol: every admitted unit of knowledge must be a SON object with identity, schema reference, obligations envelope, and evidence chain.

Schema theory and validation.

Schema-driven design has long been the mechanism by which systems prevent semantic drift and enable interoperability. Research on schema evolution (Roddick, 1995), XML Schema and JSON Schema validation, and semantic web ontology alignment shows how structure governs meaning. Layer 1 does not prescribe a global schema catalog; instead, it makes schema declaration a protocol invariant. Objectifiers may use local catalogs or policies to choose schemas, but every object must cite its schema reference, ensuring replayable semantics across implementations.

Provenance and evidence management.

Provenance-aware systems in scientific databases (Buneman et al., 2001; Davidson & Freire, 2008) and workflow systems (Simmhan et al., 2005) illustrate the need to capture how and why data was created. Layer 1 encodes this at the moment of admission: both the proposal (what the Objectifier suggested and why) and the decision (what the Inspector admitted or refused, under which policy snapshot) are persisted as SON objects. This ensures that objectification is not an opaque transformation but an auditable event that can be replayed and contested.

Separation of duties and secure write audit.

The principle of separating proposers from committers is foundational in secure system design, from classic database concurrency control (Gray & Reuter, 1992) to modern multi-party authorization protocols. Layer 1 embodies this by requiring both a worker agent (Objectifier) and a security agent (Inspector). The Objectifier can propose semantics, but only the Inspector may commit them, and every commit must be recorded as a signed DecisionRecord. This guarantees that no object enters the substrate without an attributable gate decision.

Error correction and institutional learning.

Research in data quality management and active learning shows the importance of feedback loops for schema refinement and classification accuracy (Batini & Scannapieco, 2006; Settles, 2010). Layer 1 incorporates this tradition by persisting ErrorCorrection objects and ContractLearningPatterns. These allow learning about misclassifications or schema mismatches to be explicit, versioned, and reversible—turning error into institutional memory rather than silent drift.

Taken together, these foundations show that Layer 1 is not reinventing primitives. Instead, it reassembles them into a thin, normative protocol surface: all knowledge must be admitted as objects with schema and obligations; every admission must pass through a separation-of-duties gate; and every decision must be recorded for replay. The intellectual property lies in this assembly—distilling decades of research into a portable, minimal set of invariants that any conforming implementation must share, while leaving internal methods of schema selection, parsing, and storage free to innovate.

Layer 2: Z-Axis Layer Management and Multi-Layer Graph Queries

Layer 2 situates SON objects into z-axis layers: namespaces that preserve provenance, scope, and epistemic separation. Each layer represents a coherent stratum of knowledge rooted in a specific contract context or publication scope. Its duties are twofold: to expose each stratum as a first-class, queryable context, and to guarantee the capability for cross-layer sessions when authorized. The protocol requires this capability even if an execution stays scoped to a single layer. This ensures SON supports both isolation and composition: layers remain intact, yet they can be combined without collapsing into consensus.

Handoff from Layer 1

Objects arriving from Layer 1 come with stable identity, declared schema, obligations envelope, and evidence chain, plus Inspector DecisionRecords that record their admission. Layer 2 accepts these as the raw material for constructing layer graphs. Within each stratum, SON objects become nodes, and schema-defined connectivity features become edges. The Inspector ensures that provenance from Layer 1—contract context, obligations, and admission record—is preserved as part of the layer's namespace. Thus, nothing in Layer 2 requires guessing: every node and relation has its declared source.

Contracts and Records

The fundamental contract in Layer 2 is the session. A session explicitly identifies the set of layers in scope, the temporal "as-of" bound, and the policy snapshot under which it was opened. It also records constraints such as maximum traversal depth, purpose restrictions, or gas caps. Within a session, all results must be annotated with their originating layer and, when applicable, evidence references. The session record is itself a SON object, signed by the security agent, so that future audits can replay not only the results but the authorization context under which they were obtained.

Worker and Security Agents

Layer 2 requires two protocol agents. The worker agent manages layer discovery, constructs sessions, and executes queries within their constraints. The security agent (Inspector) authorizes or denies sessions, validates who may access which layers together, applies purpose binding, and signs the session header. The separation is strict: the worker executes; the Inspector authorizes. Each decision is persisted as a SON object, ensuring sessions are decidable, attributable, and replayable.

Replayability Surface

Replayability is guaranteed by three commitments. First, every session is recorded as a SON object declaring its scope, constraints, and authorizing signature. Second, every result is annotated with its originating layer identity and provenance. Third, results are tied to the as-of state declared at session start. These commitments ensure that queries can be reconstructed under the same conditions and compared across different conditions.

Cross-Layer Composition Without Collapse

Layer 2 distinguishes composition from collapse. Composition allows a session to union results across multiple strata, align objects with shared identifiers, and return contradictory claims side by side. Collapse—the erasure of epistemic distinctions—is not permitted at this layer. Consolidation is deferred to Layer 3 or higher reasoning layers, where it occurs under explicit CRE discipline. Layer 2 therefore ensures that cross-layer queries never coerce competing claims into premature agreement.

Economic and CRE Hooks

Although CRE agents operate most actively in higher tiers, Layer 2 emits the signals they require: which layers were in scope, what constraints were applied, what traversal depth or fan-out occurred, and what gas was consumed. These metrics allow CRE agents to evaluate coverage and efficiency, correlating confidence gain with cost. Session records thus provide evidence of not only what was read but how much it cost to read it.

Security Posture at Layer 2

Layer 2 is a security boundary for reads. Its intent is to:

- Contain unauthorized access. All sessions must be authorized and signed by the Inspector.
- Preserve provenance. Every returned element must carry its originating layer.
- Constrain poisoning. Compromised layers remain bounded to their namespace and can be pruned without rewriting others.
- Anchor replayable audit trails. Every session is a durable, attributable decision object, allowing reads to be reconstructed exactly as they occurred.

Detailed enforcement—session tokens, provenance tagging, anomaly detection—is specified in the Layer 2 appendix and elaborated in the Threats paper. At the protocol level, the guarantee is that all durable reads are decidable, attributable, and replayable.

Handoff to Layer 3

The outputs of Layer 2 are: (i) layer graphs built from admitted objects and schema-defined relations, (ii) SessionMetadata objects that define authorized combinations of layers and their constraints, and (iii) session results with provenance tags. These form the inputs to Layer 3, which applies optimization, gap detection, and error correction. In other words, Layer 2 hands upward not just graphs but graphs with provenance and quality signals, giving Layer 3 the stable foundation it needs to perform optimization without corrupting epistemic boundaries.

Implementation Intention (Non-Normative)

Although the protocol does not dictate technology, a practical implementation of Layer 2 might resemble a graph database exposing sessioned views. Each z-axis layer could be realized as a namespace or separate database, with sessions federating queries across them while preserving provenance. Cross-layer queries might be optimized through indices, embeddings, or caching, but these are deployment details, not protocol mandates. What the protocol enforces is that queries respect layer boundaries, provenance is never lost, and every session is auditable. This makes the model achievable using today's graph and query systems without binding SON to any one of them.

Research and Technology Foundation (Layer 2)

The commitments of Layer 2—stratum integrity, session-based access, provenance-preserving queries, and composition without collapse—are not inventions in isolation. They are the reframing of several well-established streams of research into a protocol surface thin enough for interoperability.

Knowledge graphs and semantic layers.

Decades of work on knowledge graphs and ontologies have established that structuring entities and relationships into graph form supports scalable reasoning, efficient retrieval, and semantic clarity (Hogan et al., 2021; Gruber, 1993). Graph partitioning and named graphs in RDF systems, and multi-database federation in property graph databases, already hint at the need for separable contexts. Layer 2 extends this line of work by making such separability a protocol requirement rather than an implementation choice: every stratum is an addressable namespace, and queries must preserve its identity.

Sessioning and replayability.

Database theory has long emphasized isolation levels, repeatable reads, and snapshot queries to ensure transactional consistency. Session-based access in Layer 2 reframes these database guarantees into a knowledge protocol contract. A read is no longer just an ephemeral query but a durable decision object: it records scope, policy snapshot, and temporal bound. This extends ideas from provenance-aware databases and time-aware knowledge bases (Dhingra et al., 2022) into a uniform protocol that can be implemented across heterogeneous stores.

Federated retrieval and hybrid reasoning.

Work on retrieval-augmented generation (Lewis et al., 2020; Izacard & Grave, 2021) and graph-enhanced retrieval (Borgeaud et al., 2022) shows the value of combining local knowledge bases with external stores. Distributed graph databases and federated knowledge systems (Auer et al., 2020) demonstrate how queries can span heterogeneous graphs, but often rely on brittle schema alignment or bespoke adapters. Layer 2 reinterprets these contributions as a session protocol: federation is supported not by mandating schema unification, but by requiring that cross-layer sessions return results with explicit stratum tags, preventing collapse into false consensus.

Provenance and trust.

Scientific method and data governance both insist that claims be attributable and auditable. Provenance-aware systems (Buneman et al., 2001; Simmhan et al., 2005) and trust-scored graphs (Nickel et al., 2016) provide mechanisms to track evidence and credibility. Layer 2 encodes these practices into protocol: every session result must include the identifiers and layers of origin, enabling replay "as-of" its moment and allowing higher layers to apply trust and confidence disciplines explicitly.

Symbolic-neural hybrids and incremental reasoning.

Research on memory networks (Weston et al., 2015), neural Turing machines (Graves et al., 2014), and lifelong learning (Mitchell et al., 2015) highlight the importance of externalized, persistent memory for adaptive reasoning. Layer 2 positions itself as the minimal externalization required for such hybrids: it does not dictate algorithms, but ensures that whatever model is used can rely on session-stable, provenance-rich graphs rather than opaque stores.

In sum, Layer 2 does not attempt to replace these traditions. It distills their lessons into a protocol minimum: (1) each stratum is preserved as an independent namespace, (2) reads are bound to session contracts that make them replayable, (3) composition is allowed but collapse is forbidden, and (4) results must carry provenance. The intellectual contribution lies not in new graph or retrieval methods, but in stipulating these as the interoperability guarantees that allow diverse methods to coexist without breaking trust or durability.

Layer 3: Intra-Layer Knowledge Graph Optimization and Quality Enhancement

Layer 3 strengthens the quality of individual z-axis strata. Where Layer 1 creates objects and Layer 2 organizes them into graphs, Layer 3 is tasked with improving coherence, completeness, and accuracy inside each layer—but always without collapsing epistemic separation or silently rewriting what has been admitted. Its role is to perform targeted optimization, detect gaps or contradictions, and provide structured feedback to lower layers. Layer 3 does not make authoritative truth claims; it conducts experiments to raise quality while preserving provenance.

Handoff from Layer 2

Layer 3 receives as input the layer graphs produced in Layer 2, complete with their object identities, schema-defined edges, and provenance tags. It also consumes SessionMetadata from Layer 2, which defines the scope and authorization under which queries or optimizations may be run. These inputs guarantee that every relation in scope has a knowable origin, allowing Layer 3 to reason within a bounded epistemic context rather than inferring across collapsed graphs.

Contracts and Records

The contract in Layer 3 is the optimization record. Each optimization task begins with a declared objective: reduce redundancy, strengthen relationship confidence, fill gaps in coverage, or tighten consistency. The worker agent executes an experiment—e.g., testing a link prediction, reweighting edges, or validating deduplication boundaries—and produces a proposal. The security agent (Inspector) evaluates the proposal, enforces policy limits, and records the outcome as an OptimizationDecision SON object. This record specifies the objective, the evidence examined, the outcome (success, failure, or partial improvement), and any downstream feedback generated. As with Layer 1 objectification and Layer 2 sessions, the result is a durable, replayable decision rather than an opaque adjustment.

Worker and Security Agents

The worker agent conducts optimizations using goal-seeking strategies, machine learning methods, or rule-based checks. It may request additional evidence from Layer 0 via ServiceContracts if gaps are detected, or propose ErrorCorrection objects for Layer 1 when misobjectification is found. The security agent ensures these interventions stay within policy: it prevents unsafe modification of graphs, validates that provenance is preserved, and records all outcomes. The separation ensures that optimizations cannot silently overwrite prior knowledge; every change is attributable to a signed decision.

Replayability Surface

Replayability is ensured by requiring that every optimization attempt produces an OptimizationDecision SON object. Even failed experiments are preserved: the objective, evidence, and outcome are recorded so future agents can avoid redundant work or audit why a strategy was ineffective. Layer 3 therefore turns error into fuel: contradictions and failures are not erased, they become inputs for institutional learning. Over time, the layer accrues a history of optimizations, each replayable and attributable.

Economic and CRE Hooks

Layer 3 is where Confidence–Reputation–Evidence (CRE) discipline becomes more active. Each optimization carries a cost in gas; outcomes adjust confidence intervals for relations, update reputational scores for agents or data sources, and refine evidence chains. Successful optimizations demonstrate confidence gain per unit gas, boosting agent reputation; wasted effort reduces it. By recording both the attempted strategy and its realized cost, Layer 3 provides the feedback loop that allows CRE agents to direct future efforts more efficiently.

Security Posture at Layer 3

Layer 3 is a security boundary for optimization. Its intent is to:

- Contain optimization scope. Experiments run only inside authorized strata; they do not leak into other layers.
- Prevent poisoning by mislearning. ErrorCorrection feedback is validated by the Inspector before being admitted; malicious or faulty corrections are quarantined.
- Preserve provenance. All optimization records include evidence references, contract contexts, and policy snapshots so that improvements remain traceable.
- Enforce replayability. No optimization can silently mutate a graph; every durable change must be tied to a signed OptimizationDecision.

Details such as active learning strategies, link prediction models, or embedding methods are left to implementation. The protocol requires only that optimizations be bounded, attributable, and replayable.

Handoff to Layer 4

The outputs of Layer 3 are optimized z-axis layers, enriched with QualityAssessment objects (metrics on completeness, accuracy, and consistency), OptimizationDecision records (replayable traces of each intervention), and ErrorCorrection feedback (structured guidance to Layer 1). These become the substrate for Layer 4, which reasons across multiple strata. By the time reasoning begins, each layer is not just a static graph but a graph with known quality, history of attempted improvements, and confidence signals that CRE agents can use for cross-layer synthesis.

Implementation Intention (Non-Normative)

In practice, Layer 3 may resemble a set of quality-management routines applied over a graph database. Link prediction algorithms, graph neural networks, or rule-based consistency checks might be employed to propose new edges or prune weak ones. Active learning loops could drive targeted data acquisition via Layer 0 ServiceContracts. Quality metrics—completeness, consistency, accuracy—could be computed and published as SON objects. Yet these are implementation strategies, not protocol requirements. The protocol fixes only that optimizations are explicit, decisions are recorded, and improvements remain auditable.

Research and Technology Foundation (Layer 3)

The design of Layer 3—systematic, replayable optimization of individual z-axis strata—stands on a broad base of research in graph learning, knowledge quality assessment, and data curation. These literatures establish that optimization and error correction are both necessary and feasible; the distinctive move of Layer 3 is to treat them not as hidden functions but as explicit, protocol-governed activities.

Knowledge graph completion and link prediction.

The field of knowledge graph completion (KGC) has matured over the past decade, producing a range of embedding-based and structural approaches (Bordes et al., 2013; Trouillon et al., 2016; Sun et al., 2019). These methods demonstrate that missing relationships can be predicted from patterns in graph structure, improving coverage without requiring manual annotation. Layer 3 reframes KGC as a bounded, auditable process: proposals from such methods are logged as optimization attempts, evaluated under contract, and either adopted or rejected with a recorded decision. This ensures that completion does not silently alter a stratum but produces evidence-backed, replayable outcomes.

Graph quality assessment and data cleaning.

Research on data quality management (Batini & Scannapieco, 2006) and knowledge graph validation (Paulheim, 2017) has shown that structural metrics such as completeness, consistency, and accuracy can be systematically assessed. Layer 3 builds on these insights by treating quality assessment as a first-class output: each optimization attempt yields not only changes to the graph but also QualityAssessment objects that quantify the state of the stratum. This turns quality from an implicit assumption into an explicit, portable artifact.

Active learning and targeted acquisition.

Machine learning research has long recognized that feedback loops improve performance when labeled data is scarce (Settles, 2010). In the context of graphs, active learning drives targeted acquisition of additional edges or features where confidence is lowest. Layer 3 incorporates this by allowing optimization agents to issue requests back to Layer 0 under the appropriate ServiceContracts, seeking only the evidence needed to reduce specific uncertainties. The novelty is in the protocol framing: each request is scoped, attributable, and replayable, preventing runaway acquisition while keeping learning transparent.

Provenance and institutional memory.

The idea that error should fuel improvement rather than be discarded has roots in scientific method and provenance-aware systems (Davidson & Freire, 2008). In Layer 3, even failed optimizations are recorded as OptimizationDecision objects, ensuring that institutional learning accumulates. This echoes practices in workflow provenance, where negative results are preserved to avoid duplication of effort and to maintain the integrity of evidence chains.

Separation of duties and secure curation.

The principle that quality improvements must pass through a gate is consistent with secure curation practices in data governance and auditing. In Layer 3, the worker agent proposes optimizations, but only the Inspector commits them, leaving behind signed decision records. This prevents adversarial or mistaken optimizations from corrupting a layer and ensures that every durable change is attributable.

The intellectual contribution of Layer 3 lies not in creating new optimization algorithms but in reframing them as protocol obligations: (1) optimization proposals must be explicit and recorded, (2) adoption or rejection must be gated and auditable, (3) quality metrics must be emitted as portable objects, and (4) all outcomes must be replayable. This turns quality management from an opaque engineering choice into an interoperability guarantee, allowing diverse optimization strategies to coexist while preserving trust and durability.

Synthesis: From Knowledge Graphs to Reasoning

Tier 1 establishes the foundation for reasoning by transforming external data into durable, auditable graphs organized along the z-axis. Across its four layers, raw flux is bound to contract contexts, converted into objects with schema and obligations, placed into persistent strata, and improved through targeted optimization. The result is not merely "data stored in a graph," but a collection of knowledge strata: each one coherent within its contract scope, annotated with provenance and obligations, and carrying its own trail of admission and optimization decisions.

By the time Tier 1 is complete, every layer is both usable and contestable. Usable, because objects can be queried through authorized sessions and evaluated with explicit quality metrics; contestable, because all durable changes have been mediated by inspectors and recorded as decision objects. This dual property is what differentiates SON's substrate from conventional graph databases: the structure is not just present, it is accountable.

The handoff to Tier 2 marks a change in character. Where Tier 1 focuses on intra-layer integrity—ensuring that each stratum can stand on its own—Tier 2 must address inter-layer reasoning. Real knowledge work rarely confines itself to a single context; it requires drawing connections between layers that may contain overlapping, incomplete, or even contradictory views. The problem therefore shifts from constructing graphs to mediating perspectives: how to compare evidence across layers, how to balance confidence against cost, and how to preserve separation while allowing synthesis.

Tier 1 provides the essential scaffolding for this shift. SessionMetadata objects from Layer 2 define which strata may be traversed together under what policy snapshot. QualityAssessment artifacts from Layer 3 supply measures of completeness and reliability, allowing higher layers to privilege stronger strata or demand additional evidence. The separation of duties enforced in all layers ensures that no perspective can silently overwrite another, preserving the epistemic hygiene that makes cross-layer analysis possible in the first place.

The challenge that follows is semantic integration without collapse. Tier 2 must enable reasoning across layers while keeping them distinct, so that differences remain visible rather than dissolved. In this way, SON reframes decades of work on knowledge graphs, provenance tracking, and quality management into a protocol sequence: first establish durable, accountable strata (Tier 1), then reason across them with transparency (Tier 2 and beyond). The novelty lies in treating this as a layered protocol contract rather than an application design choice, making the guarantees portable across implementations.

Tier II — Reasoning Layers

Tier I leaves us with something stronger than a datastore: a set of durable knowledge strata. Objects have been admitted under contract, placed into explicit z-axis layers, and improved by targeted optimization—each step recorded as a decision that can be inspected and replayed. Tier II begins where that work ends. Its task is not to build more graph, but to reason across what already exists—bringing multiple strata into scope, evaluating their claims, allocating effort to reduce uncertainty, and returning answers that remain as accountable as the substrate on which they stand.

This shift in purpose changes the kinds of guarantees that matter. Within a single stratum, correctness is largely a function of admission rules, schema discipline, and layer-bounded optimization. Across strata, correctness becomes a function of comparison: weighing conflicting claims without collapsing them, estimating the marginal value of additional evidence, and ensuring that integration preserves the very separation that makes contestation possible. Tier II therefore treats perspective as a first-class object: which layers were eligible to speak, under what policy and time bounds, and how much each contributed to the result.

Two Tier I artifacts make this possible. Session records from Layer 2 declare what strata can be traversed together and under which “as-of” constraints, so cross-layer reasoning is scoped by an explicit authorization and policy snapshot rather than by convention. Quality assessments from Layer 3 provide measures of coverage, consistency, and reliability for each stratum, so aggregation can be selective and context-sensitive rather than naïvely uniform. Taken together, sessions and assessments bound what it means to look “across” without losing sight of what lives “within.”

In this setting, reasoning is not a monolith—it is a planned activity. Goals are decomposed, candidate paths to reduce uncertainty are proposed, and effort is budgeted. The protocol’s Confidence–Reputation–Evidence discipline supplies the scoring language for what to try next; the gas budget supplies the economic boundary that keeps plans proportional; and the separation-of-duties model ensures that any durable products of a run are inspected before they become part of the substrate. The emphasis is not on a single algorithm for truth, but on a repeatable procedure for managing uncertainty under constraint.

The output artifact of Tier II is correspondingly specific. Rather than emitting an unannotated answer or a transient log, Tier II produces a Query Result Object (QRO): a compact, durable record that binds a goal to (i) the session-scoped layers that were consulted, (ii) the evidence and thresholds that governed acceptance, (iii) the reputation signals in play, and (iv) the costs incurred to reach the conclusion. A QRO is not merely a result; it is a replayable explanation of how the result was obtained “as-of” a moment, preserving the structure needed for future challenge or reuse.

Finally, Tier II remains protocol-minimal. It fixes what must be recorded (scope, evidence, thresholds, costs), what must be preserved (stratum identity and provenance), and what must remain visible (the decision path that produced an answer). It does not legislate the internal machinery of planning, search, or inference. That boundary keeps implementations free to innovate while guaranteeing that two independent systems can interoperate safely: they can bring strata into scope without collapsing them, spend effort within declared budgets, and return answers whose lineage, policy bounds, and trade-offs are explicit rather than implied.

Layer 4: Cross-Layer Goal-Seeking Optimization and Attention-Driven Coordination

Layer 4 is where SON moves from maintaining individual strata to reasoning across them. Its purpose is explicitly interpretive rather than constructive: it enables queries that span multiple z-axis layers while preserving their separateness. The protocol does not prescribe how inference is performed internally; instead, it fixes the interfaces and services that reasoning must use to engage the substrate (sessions, assessments, authorization) and the artifact that must be emitted to make reasoning transparent and replayable.

Handoff from Layer 3 (Connections Below)

The inputs to Layer 4 are the optimized z-axis strata delivered by Layer 3, each annotated with provenance, an obligations envelope, and QualityAssessment metadata. Layer 4 treats these as bounded perspectives rather than a pooled dataset: they can be compared, aligned, or held in tension without being collapsed or rewritten. In this way, Layer 4 inherits Tier I's epistemic hygiene and prepares it for reasoning under uncertainty.

Contracts and Records (QRO as a First-Class Artifact)

The central contract in Layer 4 is the Query Result Object (QRO). A QRO is a replayable explanation of reasoning, not merely an answer payload. Concretely, a QRO binds:

- the reasoning goal (hashed goal specification and query parameters),
- the z-axis layers consulted (identifiers for every stratum included in scope),
- the "as-of" constraints (policy snapshot and temporal bounds under which layers were accessed),
- the reasoning constraints (evidence thresholds, diversity requirements, and the gas budget actually enforced),
- the outcome narrative (what claims converged, what contradictions remained, and why), and
- the effort ledger (realized gas usage and relevant execution metrics).

Its purpose is singular and non-negotiable: to show which layers were consolidated or represented in satisfying a query and how—so that two independent implementations can reproduce the same reasoning "as-of" a moment. This is the basis of SON's transparency, and the point of superiority over systems that munge data into untraceable composites.

Worker and Security Agents (Separation of Duties)

As in lower tiers, Layer 4 employs a worker / security split. The worker performs cross-layer reasoning: it instantiates session-bounded reads, consults QualityAssessments, and proposes candidate alignments or tensions among layers. The security agent (Inspector) authorizes which strata may be joined, enforces the declared "as-of" and policy constraints, and signs the QRO, making the act of reasoning a durable, attributable decision. This preserves the same rigor for inference that Layer 1 enforces for writes.

Replayability Surface (How Results Can Be Re-Run)

Replayability is anchored in the QRO: the goal hash, layer set, session references, policy/temporal bounds, thresholds, and gas spend together specify exactly what was consulted and under what conditions. A future reader can re-execute the run "as-of" those constraints or vary them deliberately to test robustness. Because the QRO names every contributing stratum, any compromised or low-trust layer remains visible and bounded rather than silently contaminating the whole.

Economic and CRE Hooks (Agents Are Essential, Not Prescribed)

In practice, Layer 4 relies on Confidence–Reputation–Evidence (CRE) agents to do the work of goal-seeking under constraint—proposing which strata to include, which evidence would most reduce uncertainty, and whether the expected gain justifies additional gas. The protocol does not define how these agents reason internally; it defines how they interface: request authorized sessions, consume QualityAssessments, honor obligations, and return reasoning traces suitable for inclusion in a QRO. Every proposal burns gas, and every QRO records the spend, making reasoning proportional, auditable, and comparable across implementations.

Security Posture (Layer-Specific Intent)

Layer 4's security intent is threefold. First, no unauthorized cross-layer joins: the Inspector must authorize the session and the resulting QRO, or the run does not happen. Second, provenance preservation: the QRO must disclose every consulted stratum so influence is never hidden. Third, bounded contamination: problematic layers cannot rewrite others; their participation is named and therefore contestable. Combined, these guarantees keep reasoning defensible without sacrificing utility.

Handoff to Layer 5 (Connections Above)

Layer 4 emits QROs as its primary output. Layer 5 uses these QROs as inputs to orchestration: sequencing multi-step plans, enforcing service-level objectives, consolidating multiple QROs where appropriate, and presenting durable, policy-bounded responses for consumption. By handing forward explanations rather than opaque answers, Layer 4 ensures that subsequent composition preserves lineage, cost, and constraints.

Implementation Intention (Non-Normative)

While the protocol remains implementation-agnostic, a practical realization will host multiple CRE agents as pluggable components (symbolic, statistical, ML-based), each competing or collaborating under gas budgets and quality thresholds. Whatever their internals, they succeed in SON only by emitting QROs that make their reasoning explicit. This insistence on an explanatory artifact is the architectural hinge: it is how SON avoids the undifferentiated "picture without provenance" endemic to today's systems and makes cross-layer reasoning both transparent and replayable.

Research and Technology Implementation Foundation

Layer 4 situates SON within a body of work that spans provenance-aware databases, explainable reasoning, and multi-agent systems. Its distinguishing contribution is to compress these research trajectories into a minimal protocol surface: the existence of a replayable reasoning artifact (the Query Result Object) and a thin set of contracts for how reasoning agents interact with knowledge strata.

Provenance and Explanation as First-Class Artifacts.

The insistence on a QRO reflects a long lineage in database theory and scientific workflow systems that treat lineage as coequal with data. Provenance standards in workflow engines and distributed databases have shown that the ability to re-run computations "as-of" a state is central to reproducibility. In explainable AI research, model cards, rationales, and attention maps have played a similar role: making reasoning visible rather than opaque. Layer 4 adopts this principle at the protocol level. Instead of ephemeral traces, every reasoning act must yield a durable object that binds inputs, policies, and outcomes into a single replayable record ⍰.

Confidence and Evidence Management.

A second intellectual lineage lies in uncertainty management and evidence fusion. From Dempster–Shafer theory in symbolic reasoning to Bayesian confidence intervals in information retrieval, the research consensus is that heterogeneous evidence must be aggregated in a way that preserves both support and conflict. Graph-based machine learning has extended this insight with algorithms for confidence calibration and contradiction detection across multi-hop reasoning chains. Layer 4 formalizes the minimal expectations: reasoning proposals must declare the evidence they use, the thresholds applied, and the diversity of sources consulted. This approach makes error not a failure to be hidden but a visible input to future corrections ⍰.

Multi-Agent and Goal-Seeking Reasoning.

Autonomous reasoning agents that allocate effort under constraints have been studied in both distributed AI and modern reinforcement learning. Early "housekeeping" agents such as NELL (Never-Ending Language Learner) treated knowledge acquisition as an ongoing, probabilistic activity; more recent frameworks treat reasoning as a sequence of bounded subgoals managed under resource caps. Layer 4 adopts this view but renders it protocol-minimal: the system does not dictate which algorithms are used, only that reasoning agents must operate through scoped sessions, consume quality assessments, and return proposals in a form fit for QRO inclusion ⍰.

Cross-Layer Mediation.

The design challenge Layer 4 addresses is not intra-graph optimization (already handled in Layer 3) but mediation across epistemic contexts. Research on federated knowledge graphs and ontology alignment has shown that schema heterogeneity and semantic drift make cross-source reasoning brittle unless explicit mapping and anchoring mechanisms are in place. Distributed graph frameworks have introduced hybrid approaches that combine symbolic anchors with embedding-based similarity, showing that robust interoperability requires both. Layer 4 abstracts these lessons into its contracts: strata remain intact, joins require inspector authorization, and alignments must be documented in QROs to prevent silent collapse of perspectives ⍰.

Implementation Feasibility.

From an engineering perspective, Layer 4 can be realized with existing technologies:

- Graph engines (Neo4j Fabric, Amazon Neptune, or RDF quad stores) already support multi-database traversals under named-graph semantics.
- Agent orchestration frameworks (Temporal, LangGraph, AutoGen) provide durable workflows with state checkpoints and replay capabilities.
- Evidence and provenance models from scientific workflow systems and blockchain-inspired audit trails offer practical patterns for binding reasoning to verifiable records.

What is new is not the machinery but the protocol stance: implementations are free to innovate internally, but they must surface QROs with explicit layer, evidence, and policy bindings.

In sum, Layer 4 reframes decades of work on provenance, confidence, and distributed reasoning into a protocol commitment: reasoning across strata must leave behind an explanatory artifact. This requirement makes answers transparent, contestable, and replayable—qualities that existing reasoning systems often treat as optional, but which SON elevates to the level of interoperability.

Layer 5: Knowledge-Graph–Augmented LLM Interface Layer

Layer 5 is instantiated as a mediation surface rather than a reasoning engine. Its purpose is to expose a model-agnostic contract that turns curated graph context into a bounded prompt and then returns a grounded, policy-checked narrative. The contract has two halves. On ingress, Layer 5 accepts the OptimizationDirectives produced by Layer 4, the selected subgraph annotated with confidence intervals and evidence handles, and any licensed passages drawn from the document index. On egress, it produces a KnowledgePatch—a fused, provenance-annotated context bundle—together with a Response-Compliance Report that maps every factual assertion back to supporting evidence, records any required redactions, and flags ungrounded content.

Two definitional points are important to establish here. First, the KnowledgePatch is the query-scoped, serialized context bundle comprising graph facts, retrieved passages, and provenance metadata. Every element carries object identifiers, z-axis origins, temporal validity, and confidence intervals. Second, the Response-Compliance Report is the audit artifact generated post-inference. It grounds each claim to evidence handles, enumerates policy actions such as redactions, and records safety flags, making every exchange reproducible and reviewable.

The Layer 5 pipeline is intentionally narrow. Symbolic filters over the knowledge graph and dense retrieval over the document index are executed in parallel, then normalized into a single typed bundle rather than streamed ad hoc into the model. Prompt construction is deterministic: OptimizationDirectives establish which anchors must be respected, which layers are excluded, and what temporal window applies. The KnowledgePatch is serialized into ordered segments sized for the target model's context window, and the user's query is appended last. Model choice is a deployment detail. Layer 5 treats the LLM as a replaceable component accessed through the Model Context Protocol (MCP), and pre- and post-guards ensure that the exchange remains faithful to policy, provenance, and security envelopes.

Operationally, Layer 5 maintains session-aware continuity without polluting the knowledge base. A rolling ledger retains prior KnowledgePatches and accepted responses so that multi-turn investigations can reuse context, while eviction and privacy rules prevent uncontrolled growth of conversational memory. Performance is addressed through parallel retrieval, selective caching, and adaptive breadth (top-k limits, reranking depth, temporal shrinkage), allowing the layer to meet interactive service-level objectives (SLOs) without sacrificing grounding. Fail-safe behaviour is explicit: if the graph is unavailable, Layer 5 returns the last consistent KnowledgePatch with a freshness banner; if the model is unavailable, it falls back to extractive answers with citations rather than fabricating unsupported content.

Handoff from Layer 4

The transition from Layer 4 to Layer 5 is governed by the principle of optimized context delivery. At the consolidation layer, Coordination, Resolution, and Explanation (CRE) agents select relevant subgraphs, assign confidence intervals, and prune contradictory or redundant claims. These outputs are passed upward as OptimizationDirectives that encode not only what information should be surfaced, but also its provenance, temporal scope, and priority weightings. Layer 5 is thus constrained by decisions made below: it may not invent structure, but must translate the curated graph context into a consumable form for the LLM.

This asymmetry is crucial. The LLM interface does not build or maintain the graph; it relies entirely on the housekeeping, consolidation, and trust mechanisms of Layers 1–4 to ensure that what it sees is both current and defensible. Improvements in the knowledge graph—whether through new evidence, corrected mappings, or refined confidence intervals—flow upward into Layer 5's retrieval processes, ensuring that the model inherits rather than fabricates epistemic grounding. In this sense, Layer 5 functions as an interpretive surface: it inherits the rigor of the structured layers beneath it, packages that rigor with precision, and only then permits the model to render a natural-language response.

The Three-Phase Pipeline

Layer 5 operationalizes its mediation role through a three-phase pipeline of retrieval, context merging, and prompt construction with model invocation. This pipeline ensures that every response is grounded in curated, provenance-rich evidence while remaining consumable by large language models.

Phase I: Retrieval

The retrieval phase initiates a dual strategy, combining unstructured and structured sources into a unified response space. The first channel, vector retrieval (ragRetrieve), performs dense semantic search over pre-indexed corpora such as technical standards, incident reports, or authorized internal documentation. Each passage is annotated with its source metadata, including page anchors, checksums, and sensitivity labels, ensuring that even free-text segments remain traceable to their origins.

In parallel, graph retrieval (kgRetrieve) executes structured queries against the SON knowledge graph, constructing query-specific subgraphs identified by Layer 4's OptimizationDirectives. These queries yield explicit objects, triples, and relationships annotated with their originating z-axis layer, temporal validity, confidence intervals, and evidence handles. The combination ensures that the system benefits simultaneously from the richness of natural language documents and the semantic precision of object-based graph structures.

Phase II: Context Merging

The heterogeneity of retrieved material presents a challenge: text passages and graph objects differ in format, granularity, and interpretability. To resolve this, Layer 5 constructs a KnowledgePatch, a normalized data structure that fuses vector-retrieved passages with graph-derived entities and relationships. Each element in the Patch is explicitly annotated with its SON identifier, z-axis origin, evidence chain, and confidence score, preserving epistemic transparency throughout.

OptimizationDirectives guide the merging process. Elements deemed temporally stale are removed; those with confidence below specified thresholds are filtered; and high-confidence anchors identified by consolidation agents are prioritized for inclusion. The result is a balanced extract that is neither a raw document dump nor a stripped-down fact list, but a carefully composed context slice that is both machine-tractable and human-auditable.

Phase III: Prompt Construction and Model Invocation

With the KnowledgePatch assembled, Layer 5 translates it into a form consumable by the target language model. The process begins with a structured instruction block, derived directly from Layer 4's directives, which instructs the model on which anchors, temporal windows, and z-axis layers must be emphasized or ignored. Following this, the KnowledgePatch is serialized into ordered segments designed to respect the context window of the model while maintaining narrative coherence across segment boundaries.

The user's natural language query is appended last, binding the contextual substrate to the conversational task. The model itself is treated as a replaceable component behind the Model Context Protocol (MCP), ensuring that proprietary frontier models, open-weight local deployments, or domain-specific finetunes can all be integrated without altering Layer 5's semantics.

Finally, the outputs are subjected to a Response-Compliance Check. This mechanism validates that every factual assertion produced by the model can be grounded in the KnowledgePatch, redacts unauthorized material according to the active policy envelope, flags ungrounded content as potential hallucination, and annotates each claim with explicit provenance pointers. The result is a natural language response that is both fluent and auditable, preserving SON's commitment to transparency even at the conversational interface.

From KnowledgePatch to Query Resolution Object

The three-phase pipeline yields a KnowledgePatch that is complete enough to inform the model yet bounded enough to remain auditable. However, the Patch on its own cannot explain why particular layers were consulted, how conflicts were resolved, or what constraints governed its assembly. To preserve this explanatory context, SON introduces the Query Resolution Object (QRO) as a distinct artifact.

The KnowledgePatch is best understood as the epistemic payload. It is a query-specific extract from SON's layered knowledge base, composed of graph-derived relationships, retrieved text passages, and their associated provenance. Each fact within the Patch is annotated with its originating z-axis layer, evidence handles, confidence interval, and temporal validity. This makes the Patch functionally equivalent to a bounded working memory: transient enough to be assembled anew for each query, but durable enough to be replayed under its "as-of" envelope for audit, reproducibility, or dispute resolution.

The QRO, by contrast, is the explanatory and procedural ledger. It extends beyond the Patch by recording which z-axis layers were included or excluded, which OptimizationDirectives from Layer 4 were applied, what temporal filters bounded the retrievals, and what conflicts remained unresolved. It does not merely point to the Patch; it memorializes the reasoning choices, consolidation rules, and boundary conditions that shaped the Patch's construction. In effect, the QRO explains why the Patch looks the way it does and what epistemic alternatives were set aside in the process.

This distinction is more than technical nuance—it is what guarantees both modularity and accountability. A KnowledgePatch can be generated, cached, or invalidated without disturbing the QRO, which remains the authoritative record of the consolidation process. Conversely, orchestration in Layer 6 consumes QROs precisely because they combine the epistemic slice (via the Patch reference) with the procedural metadata (directives, confidence summaries, and policy constraints). This allows orchestration to plan follow-on tasks, request additional Patches, or decompose the problem into subtasks while remaining grounded in a faithful account of how the initial context was derived.

In short, the KnowledgePatch is the evidence-bearing context window, while the QRO is the higher-order contract that renders that context explainable and actionable. Their separation preserves transparency and reproducibility, ensuring that downstream orchestration is rooted in both the knowledge available and the reasoning choices that shaped its presentation.

Security Integration and Guardrails

Because Layer 5 mediates between the structured rigor of SON's knowledge graph and the open-ended fluency of large language models, it represents the most exposed surface for adversarial manipulation. If left unprotected, this layer could become a conduit for prompt injection, sensitive data leakage, or unauthorized inference. To mitigate these risks, SON embeds a comprehensive security framework directly into the retrieval and prompting cycle.

Pre-invocation safeguards enforce policy compliance before the model is ever engaged. Each incoming query is checked against the active policy envelope, validating session tokens, user attributes, and authorization scopes. Prompts are scanned for known injection signatures and anomalous instructions that attempt to coerce the model into ignoring prior constraints. Rate limits are applied to prevent resource exhaustion, and requests that violate classification boundaries—such as attempts to access Controlled Unclassified Information (CUI) from an unauthorized role—are rejected outright.

Post-invocation safeguards verify the integrity of generated responses. Every assertion is mapped back to its supporting evidence within the KnowledgePatch; claims that cannot be grounded are flagged as potential hallucinations. Unauthorized disclosures, such as information sealed by Watchtower's access controls, are redacted before delivery. Each completed exchange is accompanied by a Response-Compliance Report, which documents grounding coverage, enumerates redactions, and records all policy actions taken. This ensures that any subsequent review can reconstruct not only what was said, but also how policy enforcement shaped the final output.

Audit and forensics capabilities provide the backbone of trust in this layer. Every retrieval, merge, prompt, and response is logged immutably, with policy decisions and compliance reports versioned alongside the corresponding KnowledgePatch and QRO. Analysts can therefore trace how an answer was assembled, which z-axis layers contributed, what redactions occurred, and under which policy rules. These logs support regulatory compliance, facilitate after-action investigations, and deter adversarial misuse by ensuring that all interactions are observable and attributable.

The guardrails themselves are implemented through policy-as-code engines coordinated with Watchtower, which serves as the policy decision point (PDP) and enforcement point (PEP). This integration allows organizations to tailor Layer 5 to their specific security requirements—ranging from GDPR-style "right to be forgotten" obligations, to NIST SP 800-171r2 access control, to sector-specific controls such as HIPAA. By externalizing policy and embedding enforcement into both pre- and post-invocation phases, SON ensures that Layer 5's mediation role is always bounded by explicit governance rather than left to model discretion.

In sum, the security architecture of Layer 5 does not treat guardrails as afterthoughts bolted onto the interface; instead, it embeds them as first-class processes in the lifecycle of every retrieval, merge, and generation. This design allows SON's conversational interface to remain flexible and expressive while still honoring the principles of privacy, provenance, and controlled disclosure that underpin the broader architecture.

Implementation Note: Watchtower as an Externalized Enforcement Pattern

Although this paper references Watchtower as a mechanism for access control and payload sealing, it is important to clarify that Watchtower is not a protocol requirement of SON 2.0. Rather, it is an implementation choice—one that illustrates how the enforcement of policy, decryption, and auditing might be realized in practice. The SON protocol itself requires only that objects and layers carry policy references and provenance hooks sufficient for downstream verification. How these hooks are bound to actual enforcement is left deliberately open.

The reason Watchtower is discussed here is that Layer 5 is where policy meets language. At this layer, curated graph knowledge is transformed into prompts and then returned as generated responses. Without enforcement, this surface would be vulnerable to exfiltration or misuse. By presenting Watchtower as an example, we make concrete how session tokens, attribute-based access control (ABAC) claims, and policy envelopes could be checked both before and after model invocation, with redactions or decryptions performed in a way that is auditable and externally verifiable.

This discussion is relevant because Watchtower illustrates the principle of externalization: enforcement logic should not be hardwired into the SON protocol, but exposed as a replaceable service. Different organizations may implement their own equivalents—whether using commercial policy engines, custom cryptographic key managers, or cloud-native PDP/PEP stacks. By externalizing enforcement, SON permits feature developers to bind their implementations to the schema in a way that declares not only which policy was applied but also which enforcement instance carried it out. This allows multiple Watchtower-like systems to coexist, each identified by a unique instance reference, without fragmenting the protocol surface.

A separate paper details the design of Watchtower in full. Here, its role is simply to illustrate the category: a policy enforcement layer that decrypts only when authorization conditions are met, records every action immutably, and provides independent audit trails. By treating this as an implementation profile rather than a protocol primitive, SON 2.0 preserves both flexibility and accountability.

Performance Optimization

Layer 5 serves as a mediation surface rather than an engine of computation. As such, the protocol does not prescribe algorithms for speed or scale. It does, however, establish observable contracts that allow orchestrators to reason about timeliness, boundedness, and replayability, and it defines degradation semantics that make performance trade-offs explicit rather than implicit.

Boundedness of the KnowledgePatch.

The protocol requires that any KnowledgePatch produced by Layer 5 be size-bounded and deterministically ordered under a specified “as-of” envelope and the active OptimizationDirectives. If the curated context exceeds the agreed bound, Layer 5 must expose an explicit segmentation (with sequence indices and stable serialization) so that downstream consumers can reconstruct order without ambiguity. The precise method by which a producer attains this bound is an implementation concern.

Degradation semantics (declared, not inferred).

Layer 5 must support declared degradations when negotiated SLOs cannot be met within the active gas budget. At minimum, the interface shall allow an orchestrator to request or accept bounded reductions along well-formed axes (e.g., retrieval breadth, temporal scope, layer inclusion, join degree), and Layer 5 shall annotate in the Response-Compliance Report which degradations were applied. How a particular implementation prioritizes or realizes those axes remains out of scope.

Replayability and equivalence.

Given identical inputs—OptimizationDirectives, policy envelope, "as-of" consistency markers—and identical degradation settings, Layer 5 must produce a functionally equivalent KnowledgePatch and a commensurate Response-Compliance Report. The protocol does not define equivalence at byte-level serialization, but at the level of content and ordering guarantees sufficient for audit and recomputation.

SLO surface and budget awareness.

Layer 5 must surface a minimal description of its capabilities and constraints (e.g., maximum Patch bound, supported degradation axes, admissible freshness windows) so that an orchestrator can plan within cost and time budgets. It must also report whether the resulting exchange meets, violates, or degrades the negotiated SLO, and record that outcome in the compliance artifact. How a producer estimates or tracks cost is implementation-specific; the protocol's role is to standardize the exposure of these facts, not their internal computation.

Observability for planning and audit.

The protocol requires a narrow set of telemetry fields sufficient for orchestration and audit: a correlation identifier; the "as-of" envelope; a short rationale of applied degradations; and a compact summary of grounding coverage (e.g., proportion of generated claims that map to Patch evidence). The means by which an implementation gathers, aggregates, or stores richer metrics is not specified.

Concurrency and side-effect isolation.

Layer 5 interactions must be side-effect free with respect to the underlying knowledge graph and documents. The protocol expects producers to honour the "as-of" envelope supplied by orchestration and to avoid observable write effects during Patch construction and model invocation. Techniques for achieving snapshot semantics are outside the scope of this document.

Model-agnostic constraints.

Because Layer 5 treats the language model as a replaceable component, the protocol requires that producers declare the effective context limit and any salient consumption constraints (e.g., maximum segment size) so that KnowledgePatches can be shaped without binding to a particular engine. How an implementation adapts to different models is left open.

Caching and reuse (advisory only).

The protocol does not mandate caching. If a producer elects to reuse prior results, it must still uphold the replayability contract: the returned Patch and compliance artifact must reflect the current "as-of" envelope and policy, and must disclose any reuse that materially affects freshness. Specific caching strategies are implementation concerns.

In short, Layer 5's performance profile in SON 2.0 is expressed as contracts, not algorithms. The protocol insists that mediation be bounded, replayable, and auditable, and that any compromises be explicitly declared. Everything else—parallel execution, resource pooling, or locality strategies—belongs to the implementation domain.

Handoff to Layer 6 (Orchestration)

Layer 5 concludes its mandate with the creation of a structured, provenance-linked contract that becomes the input surface for orchestration in Layer 6. This transition marks a deliberate and formal shift: below the boundary, SON concerns itself with what is known—facts, inferences, and contextual claims that have been consolidated, annotated, and bounded. Above the boundary, SON turns to what is to be done—the planning, sequencing, and supervision of work across agents under explicit budgets and service-level objectives (SLOs).

The principal artifact of this handoff is the Query Resolution Object (QRO). The QRO is not merely an answer but a ledger that records the epistemic state of the system at the moment of query. It contains a reference to the KnowledgePatch (the bounded payload of evidence-bearing facts and passages assembled for the task), the OptimizationDirectives that determined its construction, and the as-of consistency envelope under which it was retrieved. It further records any declared degradations applied in order to satisfy gas or SLO constraints, and includes the policy references in force at the time, sufficient to permit external audit. In this way the QRO crystallizes the outcome of Layer 5's mediation: a bounded, replayable account of what knowledge was shown and under what constraints it may be used.

The interface does not end with the QRO alone. Layer 5 also exports auxiliary signals needed for orchestration to make rational plans. These include a compact gas and SLO envelope that characterizes the resource implications of further exploration, a consistency context identifying the precise snapshots that guarantee reproducibility, and evidence handles that can be passed forward for validation or enrichment without resubmitting the entire Patch. Each of these exports is tightly scoped, ensuring that orchestration receives the information required to coordinate agents without collapsing into the epistemic role of the layers below.

Control at this boundary must be understood as a transfer of responsibility rather than a continuation of process. When Layer 6 receives a QRO, it accepts the epistemic assertions of Layer 5 as binding inputs. It does not re-estimate confidence or re-weight reputation; those tasks are completed by the consolidation and mediation layers. Orchestration instead addresses a different set of questions: whether to accept the result as sufficient, whether to decompose it into subtasks, or whether to request additional Patches under a new envelope. Any such extension constitutes a fresh contract, with its own QRO and its own declarative record of degradations, thereby preserving the modularity of the system.

This division of labor matters for both epistemic clarity and operational trust. By ending its mandate at the moment of knowledge packaging, Layer 5 ensures that every claim is grounded, annotated, and auditable before orchestration begins. By beginning its mandate only at the point of planning, Layer 6 ensures that budgets, degradations, and task decomposition are separable from the evidentiary record. Analysts and auditors can therefore distinguish clearly between "what was known" and "what was chosen to be done," tracing errors or disputes to the correct locus of responsibility.

The design also ensures that policy guarantees are never silently eroded. Where access controls or payload sealing were applied, Layer 5 records the relevant policy identifiers and, where appropriate, the reference to the enforcement profile used. SON 2.0 does not mandate any particular enforcement mechanism—Watchtower is one possible realization, but others may be substituted. What the protocol requires is only that the fact of enforcement be recorded and that it be auditable by downstream consumers. This prevents orchestration from bypassing constraints: any attempt to relax policy or broaden access must be instantiated as a new, explicit contract.

In this way the handoff between Layer 5 and Layer 6 becomes more than a technical interface; it is a seam of accountability. Below it, knowledge is curated into bounded, evidence-bearing context. Above it, work is planned and executed against budgets and SLOs. By making the contract between the two explicit, SON prevents the collapse of reasoning into an opaque black box, preserving the dual accountabilities—epistemic accountability in Layer 5 and operational accountability in Layer 6—that together ensure the system remains transparent, auditable, and trustworthy.

Why this division matters

The strict separation between Layer 5 and Layer 6 is not a matter of convenience but of architectural principle. It reflects SON's commitment to distinguishing between the epistemic question of what can be known and the operational question of what should be done. By concluding at the point of knowledge packaging, Layer 5 guarantees that the knowledge state presented to higher layers is already reconciled, bounded, and fully annotated with provenance and policy references. It is therefore possible to treat every Patch and QRO as a verifiable artifact—something that can be replayed, audited, or challenged without recourse to hidden processes.

Layer 6, in turn, assumes authority only from that evidentiary base. Its mandate is not to reinterpret or amend knowledge, but to determine how best to allocate resources, schedule tasks, or coordinate agents in pursuit of a given objective. This separation preserves a dual accountability: if a plan fails, one can inspect whether the knowledge supplied to it was incomplete or contradictory (a Layer-5 issue), or whether the orchestration itself misallocated budgets or misapplied strategies (a Layer-6 issue). In either case, the chain of responsibility remains transparent.

This division also prevents silent erosion of policy guarantees. Because Layer 5 records policy identifiers and enforcement references alongside its outputs, orchestration cannot sidestep them without issuing a new, traceable envelope. In effect, any attempt to relax policy, expand scope, or alter the terms of access must be formalized as a new QRO, creating an auditable break in continuity. This ensures that epistemic trust and policy compliance are never diluted in the flow of operational decision-making.

Taken together, these safeguards mean that SON does not collapse into the same opacity that characterizes monolithic systems. Instead, it enforces a modular chain of reasoning in which knowledge and action are linked but not conflated. Analysts, auditors, and even automated agents can therefore distinguish between the state of evidence and the course of action—an essential property if the system is to remain both explainable and trustworthy at scale.

Implementation Intention

Layer 5 is instantiated as a mediation surface rather than a reasoning engine. Its sole implementation intention is to transform curated, evidence-bearing graph context into model-consumable inputs and to return auditable narrative outputs—without altering the underlying epistemic state. Concretely, implementations of Layer 5 must satisfy three commitments.

Model-agnostic mediation. Layer 5 treats the language model as a replaceable component. It accepts OptimizationDirectives from Layer 4, retrieves and normalizes relevant subgraphs and authorized passages, and assembles a KnowledgePatch under a declared as-of envelope. The KnowledgePatch is serialized in a typed, deterministic order suitable for any compliant LLM adapter (e.g., MCP), with no dependence on proprietary prompt formats or model-internal conventions. Model selection and decoding details are deployment concerns, not protocol semantics.

Provenance-first contracts. Every claim surfaced to the model and every claim returned to the user must be traceable. Inputs are annotated with object identifiers, z-axis origins, temporal bounds, and evidence handles; outputs are accompanied by a Response-Compliance Report that grounds assertions to the Patch, records redactions, and flags ungrounded content. Layer 5 then emits a Query Resolution Object (QRO) that binds (a) the KnowledgePatch reference and (b) the decisions that shaped it (directives, degradations, as-of envelope) to (c) policy identifiers sufficient for audit. The QRO is the artifact Layer 6 consumes; Layer 5 does not prescribe how orchestration proceeds.

Active guardrails, externalized enforcement. Security and policy are enforced before and after model invocation, but not embedded in the protocol. Pre-invoke checks validate authorization claims, sanitize inputs against injection, and apply layer allow-lists; post-invoke checks enforce grounding, redact sealed content, and log safety decisions. Where decryption or obligation enforcement is required, Layer 5 records policy identifiers and the enforcement profile reference (e.g., a Watchtower profile) in the QRO; the enforcement mechanism itself remains an implementation choice defined elsewhere.

Two corollaries follow. First, degradation is declared, not inferred: when SLO or gas constraints necessitate reduced breadth, narrower time windows, or fewer cross-layer joins, the applied degradations are explicitly recorded in the QRO. Second, replayability is mandatory: given the same directives, envelope, and degradation settings, a conformant implementation must produce functionally equivalent KnowledgePatches and compliance artifacts, allowing Layer 6 to plan against a stable epistemic snapshot.

Illustrative Example: Investigating a Suspicious Process in SOC Data

Consider a security analyst's query: "Explain the risk associated with powershell.exe spawning rundll32.exe on host SRV-EDR-01 yesterday, and advise what I should check next." In SON, the response is mediated by two distinct artifacts: a KnowledgePatch, which constitutes the epistemic payload assembled for the query, and a Query Resolution Object (QRO), which records the consolidation choices, constraints, and recommended next actions.

KnowledgePatch (Layer 5, epistemic payload).

The Patch that results from Layer 5's retrieval and merging pipeline is a bounded, provenance-rich extract. In this example it would typically contain:

- Declarative graph facts: observed EDR events linking Account:alice and Asset:SRV-EDR-01; the process tree segment showing powershell.exe → rundll32.exe; timestamps; parser versions; and explicit SUPPORTED_BY → Event evidence links to the normalized EDR records (confidence = 1.0; layer = declarative).
- ATT&CK technique anchors: stable objects for T1059.001 (PowerShell) and any adjacent techniques mapped in prior declarative layers, each with identifiers and citations (not free text).
- RAG text passages: excerpted guidance from the local corpus (e.g., MITRE technique descriptions; internal playbook pages), each with page/section anchors, checksums, and licenses, filtered to the query's time window. These passages are attached to, not conflated with, graph facts, and carry their own provenance and sensitivity labels.
- Confidence and temporal fences: all non-observed claims (if any) are tagged as inference with explicit confidence intervals and supported_by pointers; stale or contradictory items are pruned per Layer 4's optimization directives before serialization. The Patch thus remains a normalized, auditable context window suitable for the LLM, but it neither plans next steps nor hides uncertainty behind prose.

QRO (cross-layer explanation and handoff contract).

Where the Patch answers "what knowledge did we show the model, exactly?," the QRO answers "why this knowledge, under which constraints, and what should happen next?" It therefore includes:

- Layer selection and consolidation trace: which z-axis layers (declarative ATT&CK mapping; internal EDR; contextual case notes) were included or excluded; conflicts encountered (e.g., a competing inference layer that suggested lateral movement but failed confidence thresholds); and the Layer 4 optimization directives that governed inclusion, ranking, and pruning.
- Policy envelope and "as-of" consistency: the Watchtower/ABAC labels in force (e.g., CUI redactions applied; directory payloads withheld), and the snapshot markers (database transaction ID, graph bookmark, stream offsets) that guarantee reproducible re-execution of the same query state.
- Gas/SLO budget and recommended degradations: the estimated cost to deepen the analysis (e.g., +30 gas to expand the time window by 24 h; +50 gas to consult external TI via MCP), with admissible degradations if the budget is tight (top-k reductions; narrower time windows; fewer cross-layer joins).
- Next-step proposals for orchestration: concrete follow-ons that Layer 6 can schedule—e.g., "re-score process lineage with additional host telemetry," "query directory for alice recent group changes (policy gated)," "fetch YARA scan results for SRV-EDR-01"—each scoped by policy, cost, and freshness. The QRO's role is not to do the work but to make subsequent work legible, auditable, and schedulable.

Why this separation matters.

The analyst receives a single, coherent explanation, yet the epistemic and procedural concerns remain orthogonal. The KnowledgePatch is the evidence-bearing context the model reads; the QRO is the explanatory and operational ledger that makes the context accountable and actionable. This modularity enables:

- Transparency and reproducibility: the exact Patch can be re-materialized under its "as-of" envelope; the QRO records the consolidation and pruning choices that shaped it, rather than burying them in narrative prose.
- Safe, policy-aware continuation: Layer 6 can accept, degrade, or reject the next-step plan under explicit gas/SLO and Watchtower constraints, without re-estimating trust and without exposing sealed payloads. The QRO preserves policy separations between metadata and decryptable content, as enforced at consumption time.
- Stable interfaces across layers: Layer 5 remains a mediation surface that packages knowledge (not plans), while Layer 6 remains an orchestration surface that packages work (not knowledge). The two artifacts—Patch and QRO—make that division explicit and testable.

In sum, after the three-phase pipeline assembles query-specific context, SON emits a KnowledgePatch to hold the what (facts, passages, provenance) and a QRO to hold the why and what next (layer consolidation, constraints, and recommended actions). This pairing preserves the epistemic rigor expected of Layer 5 while enabling the budget-, policy-, and SLO-aware orchestration required of Layer 6.

Research and Technology Implementation Foundation

The architecture of Layer 5 draws upon several established lines of research and practice. Its dual retrieval approach extends from the literature on retrieval-augmented generation (RAG), where dense vector search and document passage retrieval are combined to improve factual accuracy and reduce hallucinations. By coupling this with graph retrieval and subgraph extraction, the design inherits advances from GraphRAG frameworks, which demonstrate the value of multi-hop reasoning and explicit semantic structure in grounding generative systems.

The context-merging stage builds on work in neurosymbolic integration and fusion architectures, which emphasize combining symbolic triples with unstructured text into unified representations. Techniques from knowledge graph completion, graph neural networks, and semantic embedding alignment inform the algorithms that normalize heterogeneous sources while preserving provenance.

Prompt construction and response validation reflect maturing practices in prompt engineering and LLM orchestration, including chain-of-thought reasoning, few-shot learning, and schema-constrained decoding. By binding prompts to OptimizationDirectives and KnowledgePatch structures, the system transforms informal text generation into a controlled, reproducible process.

Finally, the guardrails embedded into Layer 5 reflect both AI safety research (prompt injection defenses, hallucination detection, content filtering) and long-standing principles from information retrieval security (access control, rate limiting, audit logging). The combination ensures that language models remain tools for interpretation and expression rather than unchecked sources of knowledge.

Taken together, these foundations place Layer 5 at the intersection of cutting-edge natural language processing research and mature systems engineering practices. They ensure that the layer is not just theoretically sound, but also practically implementable with existing frameworks such as LangChain, LlamaIndex, and Temporal-style orchestration engines, while remaining future-proof against rapid changes in the LLM landscape.

Layer 6: Orchestration

Layer 6 begins precisely where Layer 5 ends: with a bounded account of what is known, crystallized in the Query Resolution Object (QRO). From this point forward, the system turns from knowledge to work. Whereas Layers 0–5 concern themselves with the collection, consolidation, and mediation of evidence, Layer 6 is dedicated to the coordination of action. It transforms the epistemic contracts of Layer 5 into operational plans, decomposing queries into tasks, sequencing them across agents, and supervising their execution under explicit budgets and service-level objectives (SLOs).

The purpose of orchestration is not to reinterpret knowledge but to manage its application. Layer 6 accepts the QRO as a binding input: the KnowledgePatch it references, the directives that shaped it, the degradations declared, and the policy envelope in force are treated as settled facts. On that foundation, the orchestrator considers how best to achieve the objective posed by the query. It may decide to accept the result as sufficient, to refine it through targeted subtasks, or to extend it by requesting additional KnowledgePatches under new envelopes. In every case, these decisions are formalized into new contracts, ensuring that operational steps remain traceable and auditable in the same way epistemic ones are.

Layer 6 thus plays a dual role. It is both an operations coordinator, managing the mechanics of task allocation, resource scheduling, and error recovery, and a goal-seeking orchestrator, selecting which agents to engage and which layers to consult in pursuit of greater confidence or coverage. These functions are bounded by gas and SLO envelopes: every orchestration plan must fit within declared cost, latency, and completeness constraints, or else declare explicit degradations that become part of the contractual record.

The design of this layer reflects SON’s commitment to modularity and accountability. By drawing a sharp boundary between epistemic state and operational action, it ensures that failures can be correctly attributed. If the knowledge itself was flawed or incomplete, responsibility lies with the layers below. If the plan was mis-specified or budgets misallocated, responsibility lies with orchestration. This clarity not only supports technical debugging but also reinforces the system’s role as a trustworthy, auditable framework for shared reasoning.

At the same time, Layer 6 preserves flexibility by avoiding prescriptive algorithms. The protocol defines the contracts that orchestration must consume and the declarations it must produce; it does not specify how plans are internally constructed or which orchestration engines may be used. Implementations may rely on deterministic workflow schedulers, agent-graph planners, or market-like bidding systems, so long as they respect the contracts and expose their decisions through auditable artifacts.

In sum, Layer 6 is where SON’s epistemic rigor meets its operational pragmatism. It is the layer that turns “what is known” into “what is done,” ensuring that actions are planned transparently, executed under constraint, and recorded in a way that preserves the dual accountability at the heart of the SON protocol.

Goals and Objectives

Layer 6 has a single overarching purpose: to transform the epistemic contracts of Layer 5 into operationally coherent plans. This purpose unfolds across four tightly defined goals.

Acceptance of epistemic contracts. The orchestrator’s first duty is to accept the QRO and its associated artifacts—KnowledgePatch reference, OptimizationDirectives, as-of envelope, degradation declarations, and policy identifiers—as binding inputs. It does not reinterpret or modify the epistemic state. Instead, it ensures that any subsequent plan begins from a faithful account of what was known and under what constraints.

Decomposition and sequencing of work. The orchestrator must transform a user query or higher-level objective into a sequence of tasks that can be executed across agents and services. This decomposition includes identifying dependencies, establishing ordering, and creating checkpoints that preserve the replayability and auditability of the workflow. The protocol does not prescribe a particular algorithmic approach—whether deterministic workflows, agent graphs, or market-based coordination—but it does require that decomposition decisions be formally recorded and reproducible.

Budget- and SLO-constrained planning. Every orchestration plan must respect the gas and service-level envelopes declared at handoff. Where constraints cannot be satisfied, the orchestrator must apply explicit degradations (such as reducing retrieval breadth, limiting temporal scope, or relaxing completeness requirements) and must record those degradations in the new contract it produces. In this way, performance trade-offs become transparent rather than implicit, allowing auditors and downstream consumers to see not only what was done but also what was intentionally left undone.

Preservation of dual accountability. Finally, Layer 6 must maintain the structural accountability that distinguishes SON from monolithic systems. Failures of knowledge—contradictions, omissions, or mis-scored confidence—are attributable to the layers below. Failures of execution—misallocation of gas, missed deadlines, or inappropriate agent selection—are attributable to orchestration. By keeping epistemic and operational responsibilities separate, the system ensures that responsibility can be assigned correctly and that remediation can target the right part of the stack.

These goals together define the scope of Layer 6. They do not specify how orchestration engines must operate internally; rather, they specify the boundaries, artifacts, and responsibilities that any implementation must honor. In doing so, they allow for innovation in orchestration strategies while preserving the transparency and accountability that SON requires.

Handoff from Layer 5

Layer 5 concludes with the delivery of a Query Resolution Object (QRO) and its associated artifacts, which together form the epistemic contract that Layer 6 must consume. This contract captures what was known at the moment of query, how that knowledge was bounded, and under which policy and service constraints it may be acted upon. Specifically, the QRO includes a reference to the KnowledgePatch (the bounded, provenance-linked context assembled for the task), the OptimizationDirectives that shaped its construction, the as-of consistency envelope, and any declared degradations applied during context assembly. It also records identifiers for the policies and enforcement profiles that were in force at the time of handoff.

At the point of intake, these artifacts are treated as binding inputs. The orchestrator does not reinterpret or revise the epistemic state handed to it; rather, it acknowledges the consolidated facts, inferences, and context as authoritative within their declared scope. What Layer 6 assumes, therefore, is not the role of adjudicating knowledge but of coordinating action. Its concern is not what is true but what to do about what is true. This shift in responsibility—from epistemic packaging to operational planning—marks the formal boundary between Layer 5 and Layer 6.

Epistemic vs. Operational Accountability

The decision to separate epistemic packaging from operational planning is not incidental but central to SON's architecture. It enforces a division of accountability that prevents errors in knowledge formation from being conflated with errors in execution. If a claim within a KnowledgePatch proves to be incorrect, that shortcoming lies with the consolidation and mediation layers beneath. If a plan constructed from that Patch consumes excessive gas, violates an SLO, or schedules agents poorly, the fault lies with orchestration.

This division matters for trust. Analysts, auditors, and even automated agents must be able to attribute responsibility unambiguously. SON achieves this by requiring that each QRO represent a closed epistemic state: replayable under its as-of envelope, annotated with its provenance, and sealed by its declared degradations. Orchestration may act only on such sealed states, ensuring that every operational decision can be traced back to a stable evidentiary base. Conversely, every plan produced in Layer 6 must be accompanied by its own contractual artifacts—operational plans, task contracts, and post-execution reports—so that execution choices remain equally transparent.

By formalizing this dual accountability, SON avoids the “black box” collapse that plagues many contemporary AI systems, where reasoning about knowledge and decisions about action are entangled in ways that cannot be untangled after the fact. In SON, epistemic responsibility terminates at Layer 5; operational responsibility begins at Layer 6. The seam between them is the QRO, which functions as both evidence and contract.

Contracts vs. Algorithms

The orchestration layer is deliberately designed as a contractual surface rather than an algorithmic prescription. The SON protocol defines the artifacts that must be exchanged, the invariants that must hold, and the accountability structures that must be preserved. It does not, however, mandate the internal strategies by which an orchestrator decomposes tasks, allocates resources, or sequences execution. This distinction is critical: it allows the protocol to remain stable and interoperable while permitting implementations to evolve rapidly as new orchestration methods emerge.

The protocol specifies that every plan must be anchored in an originating QRO, must respect the gas and service-level envelopes declared at intake, and must emit a structured account of both planned and actual degradations. These requirements are minimal yet powerful: they ensure that every orchestration decision can be inspected, replayed, and audited, without constraining how those decisions are produced.

Implementations, by contrast, may vary widely. Some may adopt deterministic workflow engines that guarantee strict sequencing and replayability; others may employ graph-based orchestration frameworks that allow tasks to be decomposed dynamically; still others may experiment with market-like bidding systems where agents compete for subtasks under cost constraints. All of these approaches are valid, provided they satisfy the protocol’s contractual requirements.

By separating contracts from algorithms, SON achieves two complementary goals. First, it guarantees that orchestrated action remains transparent, reproducible, and accountable across all deployments. Second, it creates space for innovation, allowing research and engineering teams to optimize orchestration strategies for different domains, workloads, and regulatory environments. In effect, the protocol dictates what must be made visible and what must be preserved, while leaving how orchestration is achieved as an implementation matter. This balance between rigidity and flexibility is what allows SON to serve simultaneously as a research framework, a production protocol, and a foundation for long-term interoperability.

Implementation Notes

While the SON protocol defines orchestration in terms of contracts, not algorithms, it is often helpful to illustrate how these contracts may be realized in practice. Several existing orchestration paradigms map naturally onto the requirements of Layer 6, though none are mandated by the protocol itself. Instead, they should be understood as implementation profiles—possible realizations that satisfy SON's contractual obligations while differing in their internal mechanics.

One class of profiles is based on deterministic workflow engines such as Temporal or Cadence. These systems excel at providing durable, replayable execution flows, ensuring that every task invocation is idempotent and that failures can be compensated with explicit rollback actions. Within a SON deployment, a Temporal-style workflow engine might serve as the orchestrator's substrate, producing operational plans and checkpoints in a way that directly aligns with SON's requirements for replayability and accountability.

A second class of profiles draws on agent-graph orchestration frameworks, such as LangGraph or AutoGen. These systems are designed to coordinate ensembles of specialized agents through graph structures that allow dynamic decomposition and flexible task routing. Within SON, such frameworks could generate Orchestration Plans that evolve adaptively as agents propose new subtasks or alternative strategies, so long as every revision is formalized as a new contract and annotated with its rationale.

A third family of approaches borrows from economic and market-based coordination. In these profiles, agents compete for subtasks by offering estimates of cost, latency, and expected confidence gain. The orchestrator then selects among competing proposals to maximize utility under the declared gas and SLO envelope. Such methods are attractive in heterogeneous environments where agents have specialized capabilities or differing trust profiles. Again, the protocol does not dictate the mechanics of competition or selection, but it does require that all choices be recorded as part of the orchestration contract.

By externalizing orchestration engines in this way, SON preserves its protocol stability while encouraging diverse implementations. Deployments can select whichever orchestration profile best fits their domain—be it workflow-based, agent-graph-driven, or market-like—without affecting interoperability. What binds them together is not a shared algorithm, but a shared commitment to the contractual artifacts of SON: the QRO as input, the Orchestration Plan as output, and the lineage of degradations, checkpoints, and reports that make each decision transparent.

Performance Optimization

The orchestration layer is where knowledge is converted into work, and with that conversion comes the risk of inefficiency, runaway cost, or silent failure to meet user expectations. To prevent these risks, SON specifies a set of performance requirements at the protocol level. These requirements are not algorithms for achieving efficiency, but contractual obligations that ensure orchestration remains bounded, transparent, and auditable.

The first requirement is boundedness. Every orchestration plan must declare the scope of work in terms of gas and service-level objectives (SLOs). If the orchestrator determines that the requested objective cannot be met within these bounds, it must either reject the request or issue a new contract that explicitly declares which degradations have been applied. This prevents silent degradation, where users and auditors cannot distinguish between a full execution and a curtailed one.

The second requirement is declarative degradations. SON requires that any deviation from the requested SLO—such as reducing retrieval breadth, limiting temporal scope, or skipping low-priority subtasks—be formally annotated in the orchestration record. These degradations are not inferred after the fact; they are part of the contract itself, ensuring that consumers of the plan can see exactly how performance trade-offs were made.

The third requirement is replayability and equivalence. Given the same QRO, the same consistency envelope, and the same declared degradation settings, a conformant orchestrator must be able to reproduce an equivalent Orchestration Plan and the same operational artifacts. This guarantees that performance decisions are not opaque, but tied to stable inputs that can be tested and verified.

The fourth requirement is observability. Every orchestration run must emit a Post-Execution Report that records actual gas usage, achieved latency, applied degradations, and final SLO adherence. This record provides the ground truth against which performance claims can be audited. It also enables downstream layers to reason not just about epistemic trust, but about operational reliability.

Finally, SON requires side-effect isolation. Plans and their execution must operate only within the declared envelope of the QRO and its policy context. Expanding scope, widening access, or refreshing context must be done by issuing a new contract, not by silently mutating an existing one. This ensures that orchestration does not create hidden dependencies or introduce performance risks through uncontrolled growth.

Taken together, these requirements mean that SON does not dictate how orchestrators achieve efficiency, parallelism, or responsiveness. Instead, it insists that orchestration be explicit about its boundaries, honest about its compromises, and reproducible in its outcomes. In this way, performance optimization is not an engineering afterthought but a protocol-level guarantee of transparency, making Layer 6's operational role as trustworthy as the epistemic layers that precede it.

Handoff to Layer 7 (Agentic AI)

The final act of orchestration is to pass its operational contracts upward to the user-facing agentic layer. This transition is as sharp as the handoff between Layers 5 and 6, though oriented toward a different concern. Where Layer 6 determines what work shall be done and under which constraints, Layer 7 assumes responsibility for how that work is expressed, delegated, and explained in the service of user-facing agents.

The artifacts emitted from orchestration are therefore operational in character rather than epistemic. Chief among them is the Orchestration Plan, which records the decomposition of a query into tasks, their dependencies, and their allocated gas and SLO slices. Each task is encapsulated in a Task Contract, an immutable envelope that specifies the inputs it may consume, the layers and policies under which it must operate, and the degradation levers available to it should constraints tighten. Together, these plans and contracts constitute the ledger of Layer 6's decisions: a bounded description of how an objective is to be pursued.

To this ledger are appended Execution Checkpoints and a Post-Execution Report, which memorialize what actually transpired. These records include the gas consumed, the SLOs achieved or missed, and the degradations exercised during execution. By emitting these artifacts, Layer 6 ensures that every operational decision is transparent to downstream consumers, and that no trade-off or shortfall is hidden from audit.

When these artifacts reach Layer 7, they are not reinterpreted but enacted and explained. Agentic AI systems, which mediate directly with human users or external systems, consume the operational contracts of orchestration as their substrate. A user-facing agent may surface the Orchestration Plan to explain what steps will be taken, or reference the Post-Execution Report to justify why an SLO was missed. In this way, orchestration provides the scaffolding upon which user agents can build intelligible, trustworthy interactions.

Just as Layer 6 treated the epistemic contracts of Layer 5 as immutable, so too must Layer 7 treat the operational contracts of Layer 6. If a user-facing agent requires new work to be performed, it must do so not by mutating existing plans but by submitting a new request that results in a fresh orchestration cycle and a new set of contracts. This design prevents the slippage of accountability, ensuring that operational decisions remain traceable and that user-facing agents cannot silently override or rewrite the commitments of orchestration.

In this way, the handoff to Layer 7 completes the cycle from knowledge to work to interaction. Layer 5 guarantees epistemic integrity, Layer 6 guarantees operational transparency, and Layer 7 guarantees that the results of both can be communicated, delegated, and acted upon in ways intelligible to users. The three layers together preserve SON's central commitment: that reasoning be modular, explainable, and accountable at every stage of its trajectory.

Task Contracts, Execution Checkpoints, and the Post-Execution Report

Layer 6 expresses its operational commitments through three protocol-level artifacts that together make plans reproducible, execution auditable, and outcomes comparable across implementations.

Task Contracts.

A Task Contract is the unit of execution in orchestration: an immutable envelope bound to a specific step of the Orchestration Plan. Each contract identifies its task in the plan's partial order, declares the inputs it may consume (object and evidence references), the layers and policy under which it must operate, and the gas/SLO slice that constrains its runtime behavior. Where constraints cannot be met, the contract enumerates the permissible degradation levers (e.g., reduced breadth, narrower temporal scope) that the executor may invoke. Because Task Contracts are immutable once dispatched, any widening of scope or relaxation of policy requires a new contract, not mutation, preserving replayability and accountability.

Execution Checkpoints.

Execution of each Task Contract is recorded through Execution Checkpoints, which mark task start, declared milestones (where applicable), completion, or failure. Checkpoints carry the degradation decisions actually exercised, gas consumed versus allocated, SLO adherence, and references to produced artifacts (e.g., additional KnowledgePatch fragments, evidence handles). These checkpoints form the task-level ledger of what was attempted and what was achieved, and they aggregate to the plan's outcome without requiring inspection of internal algorithms.

Post-Execution Report.

Upon completion (successful, partial, or aborted), orchestration emits a Post-Execution Report for the plan. The report consolidates plan-wide results: realized latency and freshness, total gas usage, a canonical list of applied degradations, and references to all artifacts produced under the plan. It also records attribution and responsibility: shortfalls that derive from epistemic inputs (Layer 5) are distinguished from those that derive from operational choices (Layer 6). Because the report is tied to the originating QRO, the as-of envelope, and the plan version, it provides the basis for replay, audit, and comparative evaluation of alternative plans under different SLOs or budgets.

Together, Task Contracts, Execution Checkpoints, and the Post-Execution Report constitute the contractual vocabulary of orchestration. They do not prescribe how implementations plan or schedule; they ensure that, however planning is achieved, decisions and trade-offs are explicit, reproducible, and auditable.

Implementation Intention

Layer 6 is designed as an operational mediator that converts epistemic contracts into actionable plans without collapsing into a reasoning engine of its own. The implementation intention is therefore to preserve the integrity of knowledge from Layer 5 while guaranteeing the transparency of action toward Layer 7. Three commitments define this intention.

Immutable epistemic intake. Implementations of Layer 6 must accept QROs, KnowledgePatch references, and policy envelopes from Layer 5 as binding inputs. They may extend work only by issuing new contracts; they may not revise or reinterpret epistemic assertions.

Contract-driven planning. All orchestration must result in explicit operational artifacts: Orchestration Plans, Task Contracts, Execution Checkpoints, and Post-Execution Reports. These artifacts form the contractual record of how objectives were decomposed, supervised, and completed. Internal planning algorithms remain an implementation choice, but every decision must be surfaced through these artifacts.

Transparency under constraint. Where budgets or SLOs necessitate degradations, those choices must be explicitly declared. Where tasks fail or plans are revised, new contracts must be issued. Implementations are expected to ensure replayability: given identical inputs and degradation declarations, equivalent plans and reports must be produced.

By adhering to these commitments, Layer 6 remains a transparent conduit between epistemic grounding and user-facing agency. It does not add new knowledge; it ensures that operational decisions are bounded, auditable, and intelligible, so that responsibility for action can be traced and explained as clearly as responsibility for knowledge.

Illustrative Example (Miles Davis)

Consider the Layer 5 outcome from the earlier music-domain scenario: a QRO asserts, with provisional confidence, that an unpublished Miles Davis album may exist. The QRO references a bounded KnowledgePatch (discography objects, press mentions, catalog identifiers, and their evidence chains), lists OptimizationDirectives (prioritize authoritative catalogs; exclude crowd-sourced layers with low reputation), declares its as-of envelope, and notes a mild degradation already applied at Layer 5 (time window limited to the past six months).

Orchestration intake. Layer 6 accepts this sealed epistemic state as a binding input. It does not re-estimate confidence nor rewrite the Patch. Instead, it evaluates whether the user's request—"determine authenticity and provenance under a 2-second latency and moderate gas cap"—is feasible. The orchestrator concludes that full, multi-archive verification will not meet the latency SLO within the declared gas cap. Per protocol, it issues a revised contract rather than silently proceeding: the plan will execute with declared degradations (top-k catalogs only; single-hop provenance expansion), and a follow-on option will be proposed for deeper work.

Orchestration Plan (OP). The plan decomposes the objective into three tasks, each formalized as a Task Contract:

1. Catalog Verification (TC-1).
   Inputs: KnowledgePatch references to canonical album/press objects; layer allow-list (authoritative catalogs, label registries).
   Policy envelope: Read-only; no payload decryption required.
   Gas/SLO slice: Small; latency-biased.
   Degradation levers: Reduce catalog set from five to two; skip secondary registries.
2. Provenance Cross-walk (TC-2).
   Inputs: TC-1 hits; object identifiers for label, session dates, engineer credits; layer allow-list (label registrar; studio logs if exposed).
   Policy envelope: Read-only; may consult external reference layers if attested.
   Gas/SLO slice: Moderate; balanced latency/completeness.
   Degradation levers: Single-hop lineage only (album→label); postpone engineer/session cross-refs.
3. Rights & Release Status Check (TC-3).
   Inputs: Label object; known release schedules; rights metadata objects.
   Policy envelope: Read-only; contractual terms visible only as metadata (no sealed payloads).
   Gas/SLO slice: Small.
   Degradation levers: Use public schedules; defer private label calendars.

Each Task Contract is immutable once dispatched: inputs, allowed layers, gas/SLO slice, and permissible degradations are fixed for the run. Any need to widen scope (e.g., include private label calendars) would require issuing a new contract, not mutating the existing ones.

Execution and checkpoints. As tasks run, the orchestrator records Execution Checkpoints:

- TC-1: Two authoritative catalogs return no listing; one registry returns a pending-review entry with incomplete fields. Gas within slice; latency met.
- TC-2: Single-hop label cross-walk finds no matching internal release ticket for the asserted title; degradation lever exercised (engineer/session cross-refs deferred). Gas under slice; latency met.
- TC-3: Public release calendars show no scheduled drop; rights metadata indicates the label's catalog for the relevant period is frozen pending archival reissues.

Post-Execution Report (PER). The orchestrator emits a PER summarizing outcomes against the OP:

- SLO adherence: Latency met; gas within cap.
- Applied degradations: Catalog set reduced (TC-1); provenance limited to single-hop (TC-2).

- Task outcomes: One weak “pending-review” hint without corroboration; no evidence of an authorized release path.
- Recommendation: Offer a follow-on plan under a relaxed SLO and higher gas cap to (a) broaden catalogs to include smaller national registries, (b) perform multi-hop provenance (engineer/session logs), and (c) consult attested label side-channels if policy permits.

What did not happen. At no point did Layer 6 re-score the existence claim, reinterpret the press mention, or merge contested objects. Those epistemic operations belong beneath the boundary. Layer 6 only planned and supervised bounded work and produced the contractual trace of its decisions.

Handoff to Layer 7. The Orchestration Plan, Task Contracts, Checkpoints, and the PER are handed upward. A user-facing agent can now (i) explain that initial verification found insufficient corroboration for an unpublished album under the fast/low-gas profile, (ii) disclose the degradations exercised, and (iii) invite the user to authorize the proposed follow-on contract with expanded scope and cost. If the user accepts, a new orchestration cycle is initiated; if not, the current record remains complete and replayable.

This example illustrates Layer 6’s role: it converts knowledge into work under explicit constraints, keeps decisions contractual rather than implicit, and preserves the seam of accountability—epistemic responsibility remains below; operational responsibility is fully recorded here; user-facing agency and explanation occur above.

Research and Technology Implementation Foundation

The orchestration layer in SON draws upon several decades of research into distributed systems, workflow management, and agent-based coordination, while also aligning with contemporary industry practices in large-scale orchestration and service governance. Its purpose is not to prescribe a particular orchestration engine, but to define the protocol contracts—Task Contracts, Execution Checkpoints, and Post-Execution Reports—that any conformant system must produce. In this way, SON situates itself within existing work while introducing novel requirements for epistemic accountability and dual-layer transparency.

Workflow Engines and Durable Orchestration.

In distributed systems research, workflow engines have long established the value of durable, auditable orchestration. Systems such as Apache Airflow (commonly used in data engineering pipelines) and Argo Workflows (a Kubernetes-native orchestrator) provide task scheduling and retry semantics but often lack strong guarantees of replayability under epistemic constraints. More recent engines such as Temporal and its predecessor Cadence formalize workflows as durable state machines, enabling replay, compensation, and long-lived coordination of tasks across heterogeneous services. These features resonate strongly with SON’s emphasis on reproducibility: given identical inputs, envelopes, and degradation declarations, orchestration must yield equivalent outputs. SON extends this durability principle from system state to epistemic state, requiring that all orchestrated work be grounded in a verifiable QRO before execution begins.

Agent Coordination and Planning.

Research in multi-agent systems has introduced formal mechanisms for distributed planning and cooperation, including the Contract Net Protocol for dynamic task allocation, distributed constraint optimization, and more recent graph-based orchestration frameworks such as LangGraph and AutoGen. These systems allow agents to exchange proposals, negotiate responsibilities, and coordinate tasks without central micromanagement. SON’s orchestration layer incorporates this insight by defining Task Contracts: immutable envelopes that specify goals, permitted resources, and degradation levers, leaving internal methods to the agent’s discretion.

This is consistent with contemporary orchestration in the LLM ecosystem, where systems like LangGraph emphasize graph-structured plans that can evolve adaptively while remaining inspectable.

Symbolic–Neural Hybrids and Externalized Memory.

Hybrid research has emphasized the value of combining symbolic contracts with neural reasoning. Memory Networks (Weston et al., 2015) and Neural Turing Machines (Graves et al., 2014) showed how external memory structures can enable reproducible and interpretable learning systems. SON's orchestration layer operationalizes this principle: the QRO, Task Contracts, and Post-Execution Reports function as externalized memory artifacts, structuring reasoning outside of transient model state. This aligns with broader movements in retrieval-augmented architectures ⍰, where explicit separation between retrieval and inference yields improved transparency and factual grounding.

Retrieval-Augmented Generation and Knowledge-Oriented Planning.

Research in retrieval-augmented generation (RAG) has highlighted the benefit of grounding generative processes in explicit knowledge repositories (Lewis et al., 2020; Izacard & Grave, 2021). SON generalizes this principle from retrieval to orchestration: just as RAG prevents hallucination by requiring that answers be supported by retrieved passages, SON requires that orchestration be supported by sealed epistemic contracts. This creates a parallel form of grounding: not only must responses be grounded in evidence, but plans themselves must be grounded in epistemic state.

Industry Practices in Orchestration and Governance.

Contemporary orchestration in production environments frequently combines workflow automation with service governance. Tools like Kubernetes operators and n8n automate coordination of microservices and lightweight integrations, while cloud providers enforce service-level objectives (SLOs) and cost governance at the infrastructure layer. SON integrates these ideas into its protocol surface. Gas accounting formalizes cost constraints; degradation declarations formalize SLO trade-offs; and policy references ensure that orchestration cannot silently exceed governance boundaries. By embedding these requirements into the orchestration contract itself, SON extends industry practice from operational governance to epistemic governance, where plans are not only resource-bounded but also evidence-bounded.

Taken together, these research and industry foundations demonstrate that SON's orchestration layer is a synthesis rather than an invention in isolation. It draws on workflow durability (Temporal, Argo), multi-agent planning (LangGraph, AutoGen), hybrid symbolic–neural approaches (Memory Networks, Neural Turing Machines), and retrieval-grounded architectures (RAG, RETRO ⍰). Its novelty lies in binding these strands to the epistemic accountability of Layer 5 and the operational accountability of Layer 6. The result is a protocol that ensures orchestration in SON is not a black-box scheduler, but a transparent, contract-driven process that makes every decision reproducible, auditable, and explainable.

Tier III — User Layers

Tier 3 is where SON meets the human. It is the point at which intent enters the protocol and transparency returns. Whereas the lower tiers govern objects, layers, orchestration, and evidence, Tier 3 governs how intent is expressed and how results are rendered intelligible—without weakening the contractual boundaries that make SON reproducible, auditable, and safe ▯▯.

Layer 7 (Agentic AI — Delegates). At Layer 7, agents act as delegates of the user, translating goals into protocol-conformant requests and carrying work forward without mutating the artifacts they receive. An agent may consume Query Resolution Objects (QROs), Execution Checkpoints, and Post-Execution Reports, and—when new work is required—emit a fresh orchestration request rather than amending prior records. In the same channel, Tier 3 introduces a first-class Feedback Record: a structured artifact that links a human judgment (accept, reject, qualify) to the specific QRO or result, under the same policy envelope and provenance as the answer it critiques. This feedback is the mechanism by which human evaluation becomes error backpressure, allowing Layers 5–6 to adjust confidence, reputation, or future plans without collapsing SON's separation of concerns ▯▯.

Layer 8 (Applications — User Interface). Layer 8 is the application surface where user intent is captured and made actionable. It may be an interactive UI, a chatbot, or a background service holding standing objectives. Whatever the form, the protocol requires two things. First, intent must be expressed downward as contracts: applications do not rewrite QROs or Task Contracts; they request new ones. Second, policy must accompany results upward: the application presents outcomes alongside the active policy envelope under which they were produced—identity, role/attribute claims, purpose, and relevant obligations—so that users can understand both what was decided and under which authority ▯. To support least privilege and prevent credential harvesting, identity presentation is purpose-bound and enforcement-mediated: applications select a microidentity from the user's keyring, but only assert it to a pinned, attested enforcement profile (Watchtower/PEP). Objects never solicit credentials; they may only name a policy and reference the enforcement profile that will evaluate it. This ensures that identity proofs are nonce-bound, proof-of-possession, and never disclosed to untrusted endpoints, preserving auditability while resisting replay and man-in-the-middle reuse ▯.

Layer 9 (The User). Layer 9 is not a programmatic layer but the boundary condition that gives SON its purpose. The protocol guarantees that the richness of SON—contracts, evidence chains, confidence intervals, degradations, reputational attributions, and policy lineage—remains available on demand to the user through Layers 7–8. Users must be able to ask, "Why did you come up with this answer?"; "What data did you use?"; "Exclude this source and re-run."; "Visualize the evidence and confidence." In SON, these are not afterthoughts but obligations of transparency: full-fidelity artifacts persist independently of any single UI, and Tier 3 is the end interface for auditability—the place where records are published, consumed, and, when appropriate, corrected ▯▯.

Two invariants hold across the tier. Delegation by contract. All new work is requested as new contracts; nothing is silently rewritten. Continuity of policy and attribution. Identity, purpose, and consent travel with requests and results, so that privacy, rights, and obligations are enforceable end-to-end, including at export. The protocol places no limits on how many agents or applications may operate, nor on how richly they reason; it insists only that intent and explanation are never severed from provenance. In this way, Tier 3 preserves SON's compact with the user: autonomy without opacity, flexibility without amnesia, and ecosystem scale without loss of trust.

Layer 7: Agentic AI — Delegates of the User

Layer 7 is the first delegated-execution layer of Tier 3. It accepts bounded plans from Layer 6 and enacts them on behalf of the user under strict policy, provenance, and budgetary envelopes. Its purpose is not to reinterpret knowledge or re-plan operations, but to carry out delegated tasks faithfully while producing auditable records of execution. In this sense, Layer 7 serves as the bridge between orchestration and application: it embodies user intent as action, while guaranteeing that every result remains traceable to its originating contract and identity.

The discipline of this layer lies in contract fidelity. Task Contracts and Query Resolution Objects (QROs) inherited from Layer 6 are immutable: they cannot be expanded, rewritten, or silently degraded at the point of execution. If an agent identifies new goals or refinements, these must be expressed as new contract proposals submitted back to Layer 6, never as edits to existing artifacts. This discipline ensures reproducibility—each act of delegation is tied to a versioned commitment rather than an unwitnessed change of state.

Because Layer 7 agents act as the user's delegates, policy continuity is non-negotiable. The same microidentity and enforcement profile that Layer 8 asserted, and Layer 6 confirmed, must be carried forward intact into every execution. No agent may solicit new credentials or alter the enforcement profile; provenance of identity and purpose must be echoed in every Task Execution Report (TER) and Checkpoint. In this way, "who acted, on whose behalf, and under what authority" is never ambiguous.

Two responsibilities anchor Layer 7's contribution to the protocol. First, execution with transparency: results returned upward are accompanied by the relevant references—contract identifiers, evidence handles, and any degradations exercised—so that presentation layers can explain outcomes without reconstructing hidden state. Second, feedback as artifact: agents produce Feedback Records that link judgments (accept, reject, qualify), rationales, and observed side-effects to the originating QRO or Task Contract. This turns user and agent evaluation into structured "error backpressure" for Layer 6 and, when appropriate, trust signals for Layer 5, without ever rewriting the underlying claims.

The protocol deliberately refrains from prescribing the internal mechanics of agents at Layer 7. Implementations may employ language models, rule-based components, or hybrid methods; they may be interactive (driven by a user session) or persistent (standing objectives that run under scheduled contracts). What the protocol does require is that every action be expressed through the SON contract surface, every result be attributable to those contracts, and every identity/policy check be honored exactly as inherited. This separation allows innovation in agent design while keeping the invariants of provenance, policy, and economy intact.

Finally, Layer 7 closes the loop between plan and presentation without collapsing roles. It does not adjudicate knowledge (that is the domain of Layers 1–5), and it does not re-orchestrate work (that is the domain of Layer 6). Instead, it enacts, exposes, and reports. By confining autonomy to the proposal of new contracts—and by insisting that all execution is conducted under the original envelopes of evidence, policy, and gas—Layer 7 preserves the modular accountability that distinguishes the SON protocol end-to-end. This is what makes the user tier credible: intent goes in as contracts, and explanations come out as auditable records, never as after-the-fact stories detached from the plan.

Goals and Objectives

The objectives of Layer 7 are defined by its role as the delegated executor of user intent within the SON protocol. This layer is the first in which user-facing agency is realized, yet its actions are constrained by the immutable contracts and policy envelopes that descend from orchestration. Its goals therefore balance autonomy in execution with fidelity to provenance, policy, and reproducibility.

The first objective is contract fidelity. Layer 7 agents are bound to consume the Query Resolution Objects (QROs), Task Contracts, and Post-Execution Reports they inherit from Layer 6 without mutation. Their mandate is to enact these commitments faithfully, not to reinterpret or revise them. Where new goals arise—whether from user refinement, exploratory reasoning, or hypothesis generation—they must be expressed as new contracts submitted downward. This guarantees that every extension of intent is auditable as a discrete act, rather than an unwitnessed alteration of prior state □□.

The second objective is policy continuity. Because agents operate directly on behalf of users, they must carry forward the full policy envelope—including role- or attribute-based access controls, consent declarations, and microidentity assertions—into every action. No execution at this layer may bypass or weaken the conditions under which access was granted. By binding identity and purpose to each contract invocation, Layer 7 preserves accountability: every act is traceable to both a user and an authority context □□.

The third objective is transparency of execution. Layer 7 agents must surface the provenance of their work as clearly as they consume it. Every output must reference the Task Contracts and evidence handles that supported it, along with any degradations applied during execution. This ensures that results are intelligible not only to higher layers but also to the user themselves, who may request explanations in natural language or interrogate the underlying artifacts directly.

The fourth objective is to maintain a feedback channel. Agents are expected to emit Feedback Records—structured artifacts that link human or agent judgments (acceptance, rejection, qualification) to the QROs or outputs under review. These feedback records are not retroactive mutations; they are additive signals that downstream layers can consume as error backpressure or trust calibration. In this way, human oversight and agent evaluation become part of SON's formal epistemic cycle rather than informal annotations.

The fifth objective is audit completeness. Because this layer forms the final technical interface before application presentation, it bears responsibility for ensuring that all relevant records—contracts, envelopes, degradations, checkpoints, and feedback—are complete and available to higher layers. Applications may abstract these records for usability, but Layer 7 must guarantee their persistence and accessibility for audit, compliance, or replay.

Taken together, these objectives situate Layer 7 as the custodian of delegated intent: it enacts plans without alteration, preserves identity and policy continuity, reports its work with fidelity, channels feedback into structured artifacts, and ensures that the entire cycle of execution remains transparent and auditable. In doing so, it bridges the operational discipline of orchestration with the interpretive and interactive demands of applications and end users.

Handoff from Layer 6

Layer 6 concludes orchestration with a complete plan: the Query Resolution Object (QRO), the Orchestration Plan, its constituent Task Contracts, and a Plan-level Post-Execution Report (PER) template that defines the evaluation context against which execution will be judged. These artifacts form the binding commitments that Layer 7 must consume. The orchestrator has already determined what work shall be done, under which budgets, policies, and service-level constraints. Layer 7 therefore begins not with an open question but with a bounded assignment.

The defining principle of this handoff is immutability of inheritance. Agents at Layer 7 are not permitted to alter the epistemic commitments or operational parameters they receive. A Task Contract, once issued, cannot be expanded, rewritten, or relaxed at the point of execution. Instead, agents must either (i) enact the contract faithfully, reporting outcomes and degradations as they occur, or (ii) surface the need for additional work by submitting a new request downward to Layer 6. This separation ensures that execution fidelity is always distinguishable from orchestration choice, preserving accountability for both domains.

The handoff also carries forward the policy envelope that governs execution. Each Task Contract arrives annotated with the user's asserted microidentity, the role or attribute claims under which access was authorized, and the reference to the enforcement profile (e.g., Watchtower) that validated those claims. In addition, the contract includes a nonce-bound capability token tied to the current request. Agents at Layer 7 must echo these identity and policy references in every Task Execution Report (TER) and Execution Checkpoint they produce. This prevents replay or re-binding attacks and guarantees that provenance of identity and purpose is never lost between orchestration and execution. No action at this layer may bypass, substitute, or reassert policy on its own authority.

Equally important is the continuity of the evaluation ledger. Layer 6 defines not only the plan but also the context in which execution will be evaluated: gas allocations, service-level targets for latency and freshness, degradation levers that may be applied, and policy constraints that must be respected. Layer 7 agents must integrate their outputs into this ledger by producing structured artifacts that reference the originating contracts and their identifiers. Every TER must report gas consumed versus allocated, SLO adherence or violation, degradations exercised, and typed outcomes. Failures must be classified into standardized categories—policy-deny, budget-exhausted, dependency-missing, or timeout—ensuring that operational shortfalls are recorded in a consistent, auditable manner.

Execution Checkpoints provide the intermediate trace of enactment. They mark task start, intermediate milestones (where applicable), and completion or failure. Each checkpoint records gas consumption, latency achieved versus target, degradations applied, and references to outputs or evidence handles produced. These checkpoints, together with final TERs, form the basis upon which Layer 6 can recompute the Plan-level PER, ensuring that orchestration's commitments and delegation's actions remain in lockstep.

In this way, the handoff creates a seamless chain of custody. Orchestration defines the plan, establishes budgets, and locks the policy envelope. Delegation consumes these artifacts without mutation, enacts them under the asserted identity, and reports back in structured form. When the final PER is composed at Layer 6, it is not a new interpretation but a deterministic aggregation of the Checkpoints and TERs supplied by Layer 7. This preserves both immutability and intelligibility: execution remains faithful to contracts, and explanations of results remain grounded in an auditable ledger.

Delegation and Autonomy

Layer 7 is the locus of delegated execution. Its purpose is to enact user intent under explicit contracts, not to become a venue for ad-hoc replanning or epistemic revision. Yet in practical deployments, agents at this layer will often be more than mechanical executors. They may anticipate follow-on needs, propose alternatives, or surface gaps in coverage that orchestration did not foresee. The SON protocol addresses this tension by drawing a clear distinction between what is permitted as internal behavior and what is recorded as external state.

On the side of behavior, the protocol places no restrictions on the reasoning techniques an implementation may employ. Agents may use large language models, symbolic rules, planners, simulators, or hybrid methods to interpret the contracts they inherit. They may internally explore options, rank candidates, or perform counterfactual reasoning to decide how best to fulfill a Task Contract. The critical constraint is that all execution must remain bounded by the inherited contract: the agent must not expand scope, alter constraints, widen layer allow-lists, or consume resources beyond the declared gas and service-level envelopes. Where additional effort is warranted, the correct expression of autonomy is to propose a new contract downward rather than silently stretching an old one. This "new-contract-or-not-at-all" rule preserves the immutability of the orchestration boundary while still allowing agents to be genuinely helpful.

On the side of state, the protocol requires that any hypothesis, refinement, or user-facing suggestion that emerges from Layer 7 be cast as a structured artifact, not as an ephemeral side effect. Two artifact types are defined for this purpose. The first is the Feedback Record (FR): a structured object that links a human or agent judgment to a specific QRO, Task Contract, or execution outcome. An FR contains a judgment type (accept, reject, qualify), optional rationale, observed side effects, and references to evidence handles. The second is the New-Work Request (NWR): a proposal for additional work expressed in contract form. An NWR specifies the parent QRO or plan it derives from, the new objective, the proposed layers and degradation levers, anticipated gas/SLO posture, and supporting evidence references. Both FRs and NWRs are strictly additive: they do not mutate prior commitments but supply auditable signals that Layer 6 can evaluate against budgets and policy.

This design accommodates rich agent capability without eroding protocol guarantees. An agent may pre-compose candidate subtasks, draft prompts, or prepare data slices; it may even run lightweight "what-if" tests inside the declared gas and SLO slice. But any step that would change scope, layer allow-lists, policy references, or freshness windows must be elevated as a new contract rather than executed implicitly. By insisting on this separation, SON enables implementations to design creative, anticipatory agents while still ensuring that accountability is preserved.

The protocol therefore frames autonomy at Layer 7 as predict-and-propose, not replan-and-proceed. Agents may use sophisticated reasoning to propose refinements, but all such proposals must be externalized through artifacts that are versioned, attributed, and auditable. This preserves the accountability chain from Layer 5's epistemic state, to Layer 6's operational plan, through Layer 7's delegated execution, and ultimately to Layer 8's presentation surface.

Performance Optimization

Execution at Layer 7 must be both efficient and accountable. Because this is the layer where delegated agents act directly on behalf of the user, its performance is not simply a matter of engineering expedience: it is a protocol-level obligation. SON does not dictate the internal scheduling, reasoning, or tool-selection algorithms an agent must employ, but it does define requirements that ensure execution remains bounded, reproducible, and auditable.

The first requirement is bounded execution under gas accounting. Every Task Contract received from Layer 6 carries a gas/SLO envelope that specifies the maximum resources available for execution, as well as the expected latency and freshness of results. Layer 7 agents must respect these bounds without exception. If execution cannot be completed within the allocation, the agent must either (i) return partial results annotated as incomplete, or (ii) submit a New-Work Request to Layer 6 for re-orchestration under expanded parameters. Silent overconsumption of gas or degradation of service-level targets is prohibited. By making resource exhaustion explicit rather than implicit, SON ensures that trade-offs are transparent to both orchestrators and users.

The second requirement is deterministic traceability of degradations. Agents are permitted to exercise degradation levers specified in the Task Contract, such as reducing retrieval breadth, narrowing a temporal window, or deferring non-critical subtasks. However, each such action must be explicitly recorded. Execution Checkpoints must note which degradations were invoked, and the final Task Execution Report (TER) must summarize all degradations applied during execution. These records ensure that results never appear more comprehensive than the resources actually consumed, preserving the integrity of user-facing explanations.

The third requirement is latency discipline and progress signaling. While Task Contracts may allow for asynchronous execution, each contract carries an expectation of responsiveness relative to its SLO. Layer 7 must therefore expose intermediate checkpoints for long-running tasks, recording gas consumed, latency achieved versus target, and outputs produced to date. These checkpoints allow applications at Layer 8 to inform users of progress, offer opportunities for midstream revision of intent, and prevent delegated execution from becoming opaque or unresponsive.

The fourth requirement is side-effect isolation. Agents at Layer 7 may interact with external services, invoke sub-tools, or generate provisional hypotheses. However, none of these actions may alter inherited artifacts or produce irreversible effects outside the scope of the current contract. All interactions must be sandboxed by the declared policy envelope and echoed in the TER. If an exploratory action suggests new work, it must be elevated as a New-Work Request rather than silently modifying the existing plan. This guarantee prevents exploratory or creative behaviors from contaminating the authoritative state of the system.

The fifth requirement is failure classification and idempotency. Each Task Execution Report must specify the outcome of execution in standardized categories: success, policy-deny, budget-exhausted, dependency-missing, or timeout. Agents must also implement idempotency keys, ensuring that retries or repeated invocations of the same contract do not produce duplicate or inconsistent results. When retries occur, checkpoints must reference the identifier of the failed attempt and record cumulative gas consumption. This enables accurate recomputation of costs and preserves fidelity in the execution ledger.

Together, these requirements ensure that Layer 7 performance is not a black-box property of implementation but a visible, contractual fact of the protocol. By bounding execution with gas, declaring degradations deterministically, enforcing latency discipline, isolating side effects, and classifying failures in a standard way, Layer 7 guarantees that delegated agency remains efficient, transparent, and auditable. These measures preserve user trust while leaving ample freedom for implementations to innovate in how agents reason and act. The critical distinction is that in SON, performance is not merely an engineering concern—it is a protocol-level invariant, enforceable through artifacts and replayable in audit.

Handoff to Layer 8 (Applications / UI)

Layer 7 concludes its mandate by enacting the Task Contracts it inherited from orchestration and producing the structured records that attest to what occurred. Its outputs are not only the results of delegated execution, but also the artifacts that preserve execution fidelity: Execution Checkpoints, Task Execution Reports (TERs), and Feedback Records (FRs). Together these artifacts constitute the handoff surface to Layer 8, which is responsible for rendering results intelligible, capturing new intent, and binding user identity to subsequent requests.

The central principle of this handoff is that Layer 8 is the first layer of interpretation for humans. Where Layer 7 agents execute and report under formal contract discipline, Layer 8 applications transform those reports into user-facing outputs. They may present natural language explanations, interactive visualizations, or composite dashboards, but behind every abstraction lies a complete ledger of provenance, policy, and performance. The protocol requires that this fidelity remain available: applications may summarize or simplify, but they cannot discard the ability to expose the underlying artifacts when requested. This ensures that users can drill down from high-level narratives into the precise contracts, checkpoints, and evidence that produced them.

This boundary also establishes identity continuity. Layer 7 agents act under microidentities and policy envelopes passed down from orchestration. When results surface into applications, those same envelopes must be presented alongside the outputs, making explicit which identity acted, under which purpose, and under which obligations. Layer 8, as the user interface, becomes the point where identity assertions are visible, selectable, and auditable. Here, users may choose among microidentities, consent to specific scopes of access, or delegate authority for subsequent tasks. The protocol does not prescribe how such interfaces are implemented, but it does require that the link between results and their governing policy envelopes be preserved end-to-end.

Equally critical is the role of Layer 8 as the audit boundary for user consumption. Once results cross into applications, they must be accompanied by the artifacts necessary for explanation: the originating QRO, the Task Contracts that were executed, the degradations that were applied, the checkpoints that marked progress, and the FRs that carried user or agent judgments. Applications may aggregate or restructure these records, but the full chain must remain retrievable. This guarantee ensures that when a user asks “Why did you come up with this answer?”, “What evidence supports this claim?”, or “Show me the degradations applied,” the application can present not a reconstructed story, but the actual contractual artifacts that ground the response.

Finally, this handoff closes the delegation cycle by maintaining seamless chain-of-custody semantics. Layer 6 defined the plan and constraints; Layer 7 enacted those contracts and emitted structured records; Layer 8 interprets the records for human interaction and captures new intent for future orchestration cycles. At no point may applications sever results from provenance or bypass policy continuity. In this way, the seam between Layer 7 and Layer 8 preserves both immutability and intelligibility: immutable contracts and checkpoints assure fidelity of execution, while intelligible presentation ensures that users can interrogate, trust, and act upon the outputs.

Implementation Intention

The implementation intention of Layer 7 is to guarantee that delegated agency operates with fidelity to protocol contracts, continuity of identity and policy, and completeness of audit, while leaving the richness of agent reasoning and internal design to the discretion of implementers. The protocol therefore defines a small set of invariants that all conformant implementations must honor.

The first invariant is immutability of orchestration artifacts. Query Resolution Objects (QROs), Task Contracts, Execution Checkpoints, and the evaluation context defined by Layer 6 must be consumed as binding inputs. Implementations may not mutate these artifacts. When new goals arise—whether from user refinement, agent hypothesis, or contextual discovery—they must be expressed as new contracts or New-Work Requests, never as silent changes to prior commitments. This rule preserves reproducibility and ensures that the ledger of actions remains stable across audit and replay.

The second invariant is continuity of identity and policy. Every delegated act must carry forward the microidentity and enforcement profile named in the Task Contract. To resist replay or re-binding, identity assertions must be nonce-bound and proof-of-possession, and agents must echo these values in all Task Execution Reports and Checkpoints. Implementations may support multiple microidentities, selective disclosure proofs, or attribute-based assertions, but these are external design choices. What matters for conformance is that provenance of identity and purpose is preserved end-to-end, and that agents never solicit or substitute credentials outside the named enforcement profile.

The third invariant is feedback as artifact. Evaluations by humans or agents must be captured in structured Feedback Records (FRs). FRs are additive, never retroactive: they link judgments (acceptance, rejection, qualification), rationales, and evidence references to the specific QRO, Task Contract, or output they critique. These records form the mechanism by which oversight becomes structured error backpressure, feeding downward into orchestration and, where appropriate, into reputation and confidence recalibration.

The fourth invariant is performance observability. Implementations must respect gas and SLO envelopes, record degradations deterministically, classify failures into standard categories (success, policy-deny, budget-exhausted, dependency-missing, timeout), and support idempotency through execution keys. Each Task Execution Report must report gas consumed versus allocated, SLO adherence, degradations applied, and outcome type. Execution Checkpoints must provide intermediate traces for long-running tasks, including cumulative gas usage and references to produced outputs. These obligations ensure that performance is not an opaque engineering detail but a visible, auditable property of the protocol.

The fifth invariant is audit completeness. Implementations must ensure that all artifacts—Task Contracts, Checkpoints, Task Execution Reports, Feedback Records, degradations, and policy references—are persisted and made available to Layer 8. Applications may abstract these details for usability, but fidelity to the complete record is essential for explainability, compliance, and trust.

Taken together, these invariants can be summarized as a conformance checklist:

- Orchestration artifacts consumed without mutation.
- New goals expressed as new contracts, not silent edits.
- Microidentity and enforcement profile carried forward in every report.
- Identity proofs nonce-bound and echoed in outputs.
- Feedback Records structured, additive, and auditable.
- Gas and SLO limits enforced and reported.
- Degradations deterministically recorded.
- Failures typed into standard categories; retries idempotent.
- Execution Checkpoints and TERs tied to as-of envelopes for replay.
- Full artifact chain available to Layer 8 for presentation and audit.

By adhering to these commitments, Layer 7 remains a disciplined mediator of delegated agency. It allows implementations to innovate in how agents reason, plan internally, or assist users, while ensuring that every visible act is bounded, explainable, and auditable. In this way, the protocol enforces invariants at the boundaries, while preserving freedom inside them—safeguarding user trust without constraining creative design.

Illustrative Example: Delegated Execution of a Miles Davis Query

Consider a user who, through an application at Layer 8, asks: “Is there evidence of an unpublished Miles Davis album, and can you show me the provenance of the claim?” This intent is translated into a new orchestration cycle at Layer 6, producing a Query Resolution Object (QRO), an Orchestration Plan, and three Task Contracts.

- **Task Contract 1 (Catalog Verification):** Check authoritative music catalogs and label registries for references to unpublished Miles Davis works.
- **Task Contract 2 (Provenance Cross-walk):** Cross-reference any hits with label metadata, studio logs, and archival materials.
- **Task Contract 3 (Rights and Release Status):** Validate against public rights schedules and label calendars.

The Orchestration Plan allocates gas and SLO slices to each contract, specifies allowed layers (declarative catalog entries, external reference archives), and names the active policy envelope tied to the user’s asserted microidentity.

Execution at Layer 7

Agents at Layer 7 receive these contracts and act as delegates of the user. They do not alter the scope of work or re-plan; instead, they execute faithfully within the bounds inherited from Layer 6.

- **Task Contract 1** is enacted by an agent that queries three catalogs. Two return negative results; one returns a “pending-review” entry. The agent records an Execution Checkpoint noting gas consumed, latency achieved versus the SLO target, and the degradation exercised (catalog breadth reduced due to time budget).
- **Task Contract 2** cross-walks the pending entry against label metadata. No corroborating records are found. The checkpoint records the application of a degradation lever: only single-hop provenance expansion was performed, as multi-hop exceeded the gas slice.
- **Task Contract 3** confirms that public rights calendars show no upcoming Miles Davis releases. The checkpoint records compliance with both latency and gas allocations.

For each of these tasks, the agent produces a Task Execution Report (TER). Each TER records: task identifier, consumed gas versus allocation, SLO adherence, degradations applied, outputs referenced, and outcome classification. All TERs echo the microidentity and policy_ref from the original Task Contracts, proving that execution occurred under the correct authority and envelope.

Feedback as Artifact

After completing the tasks, the agent issues a Feedback Record (FR). This FR links its judgment—that the pending catalog entry is low-confidence—to the QRO and the Task Contract that discovered it. The FR references the evidence handles for the catalogs consulted and the checkpoints that recorded degradations. It does not alter the QRO or the Orchestration Plan; it adds structured error backpressure for possible re-orchestration.

Policy Continuity

All queries and evidence retrieval are executed under the user’s asserted microidentity and the named enforcement profile. The agent does not solicit or handle credentials; it simply carries forward the policy envelope provided by Layer 8 and enforced at Layer 6. Each TER echoes the same microidentity and policy_ref, and the nonce-bound capability token is preserved, ensuring resistance against replay or re-binding attacks. Provenance of identity and purpose is therefore maintained with each checkpoint and with the TER bundle.

Results to Layer 8

Layer 7 delivers upward a bundle consisting of the TERs, the Execution Checkpoints, and the Feedback Record. Layer 6 then uses these artifacts to finalize the Plan-level Post-Execution Report (PER). Layer 8, as the application layer, interprets the results for the user:

- **Narrative view:** "One catalog listed a provisional entry, but this was not corroborated by label metadata or rights schedules. Based on current evidence, the claim remains low-confidence."
- **Evidence view:** If the user clicks "Show me the evidence", the application can expose the actual Task Contracts, evidence handles, and confidence intervals.
- **Performance view:** The application can also show which degradations were applied and how much gas was consumed, because Layer 7 preserved the full audit trail.

Why This Matters

In this example, Layer 7 acted exactly as the protocol intends. It enacted contracts faithfully without altering their scope, carried forward the user's policy and microidentity without substitution, issued structured Feedback Records to provide error backpressure, and produced Task Execution Reports and Execution Checkpoints that made gas use, SLO adherence, degradations, and outcomes explicit. Layer 6 then composed the Plan-level PER from these artifacts.

The user, via Layer 8, receives both an intelligible narrative and the ability to interrogate the raw provenance. Nothing is lost in translation: intent went in as contracts, execution occurred under bounded authority, and results came out as auditable records. This illustrates the defining value of Layer 7: delegated autonomy without opacity, ensuring that user trust rests on artifacts, not after-the-fact explanations.

Research and Technology Implementation Foundation

Layer 7's design as the delegated execution layer draws together multiple strands of research and practice: durable workflow systems, agentic coordination frameworks, and human-in-the-loop governance. The SON protocol does not prescribe specific orchestration or agent architectures. Instead, it defines the contractual invariants—Task Execution Reports (TERs), Execution Checkpoints, and Feedback Records (FRs)—that ensure delegated autonomy remains bounded, transparent, and auditable.

Workflow durability and replayability.

In distributed systems, engines such as Temporal, Cadence, and Argo Workflows have demonstrated how workflows can be made durable, auditable, and replayable. They emphasize idempotency, checkpointing, and compensation mechanisms for long-running tasks. Layer 7 inherits this tradition by requiring that every delegated execution step be accompanied by Execution Checkpoints and a Task Execution Report that together allow plans to be reconstructed and audited. The novelty in SON is that these reports are not merely operational logs—they are protocol-level artifacts tied to identity, policy, and provenance, ensuring that replayability covers not only technical correctness but also who acted, under what authority, and at what cost.

Agent coordination and bounded autonomy.

Research in multi-agent systems has long explored mechanisms such as the Contract Net Protocol (Smith, 1980), distributed constraint optimization (Modi et al., 2005), and, more recently, graph-based agent orchestration frameworks like LangGraph and AutoGen. These approaches demonstrate that distributed agents can be directed most effectively through well-scoped contracts rather than by prescribing their internal decision-making. SON adopts this insight: Layer 7 agents are free to reason internally using symbolic rules, large language models, or hybrid techniques, but their autonomy is bounded externally. Any new goal or refinement must be expressed as a New-Work Request, never as a silent modification of an inherited contract.

This aligns with the agent coordination literature, which emphasizes explicit commitments and contract-based delegation as safeguards against uncontrolled autonomy.

Human-in-the-loop feedback and error backpressure.

Human-computer interaction and reinforcement learning research has established the importance of structured feedback in calibrating system behavior. Work on reinforcement learning with human feedback (RLHF) and its extensions (Christiano et al., 2017; Bai et al., 2022) demonstrates that structured evaluations can improve alignment and trustworthiness. Layer 7 embeds this principle at the protocol level through Feedback Records (FRs). An FR is not a hint or an annotation; it is a structured artifact that ties a judgment to specific contracts, evidence, and outputs. This makes feedback durable, auditable, and reusable across systems—providing exactly the kind of "error backpressure" that SON's layered epistemic model requires.

Policy continuity and secure delegation.

From the standpoint of security and compliance, Layer 7 builds on established concepts of policy decision and enforcement separation (PDP/PEP) (XACML, OPA) and capability-based access control (Levy, 1984; Miller et al., 2003). In SON, every Task Contract carries a policy envelope, a microidentity, and a nonce-bound capability token. Layer 7 agents are required to echo these values in TERs and Checkpoints, providing cryptographic continuity between intent, delegation, and execution. This mirrors industry practice in zero-trust architectures and confidential computing, where proofs of identity and scope must accompany every action. The difference in SON is that these constraints are not implementation detail—they are formalized as part of the protocol surface.

Failure classification and resilience.

Decades of research into distributed reliability (Birman, 1993; Lamport, 1998) emphasize the importance of typed failure outcomes and idempotent retries. SON carries this principle upward into Layer 7: every Task Execution Report must classify its outcome (success, policy-deny, budget-exhausted, dependency-missing, timeout) and include idempotency keys for replay. This allows orchestrators and auditors to distinguish between epistemic insufficiency and operational shortfall, reinforcing the dual accountability model that defines SON's layering.

Implementation profiles.

Several implementation profiles can map cleanly to Layer 7's requirements. A Temporal-like workflow substrate may generate Task Contracts and as-of envelopes, while an agent-graph runtime consumes those contracts and produces Checkpoints and TERs. FRs may be surfaced through user-facing RLHF-style interfaces or through domain-specific validators. None of these approaches are required, but each demonstrates how existing orchestration and agent technologies can be adapted to SON's contract-first model. The key is that implementations are judged not by how they reason internally but by whether they emit the artifacts that the protocol requires.

Synthesis.

Layer 7 therefore represents a synthesis of durable workflow, agent coordination, human-in-the-loop feedback, and policy-enforced delegation. Its objectives—contract fidelity, policy continuity, transparency of execution, feedback as artifact, and audit completeness—are grounded in mature literatures but reframed as protocol invariants. The distinctive contribution of SON is to require that these properties be surfaced as first-class artifacts—TERs, FRs, and Checkpoints—rather than as optional engineering features. By making them mandatory and auditable, SON ensures that delegated agency is never opaque: every act of execution is bounded, explainable, and reproducible.

Layer 8: Applications / UI

Layer 8 marks the transition from delegated execution within the SON protocol to the domain of human legibility and intent formation. Whereas Layer 7 agents operate under binding contracts to enact user goals, Layer 8 is the stratum where those results are interpreted, composed, and re-expressed in forms accessible to human users or external applications. It is here that the protocol surfaces its internal rigor—epistemic grounding, provenance chains, confidence intervals, and policy envelopes—in ways that users can interrogate, validate, and extend.

The defining characteristic of this layer is its dual role as both a presentation boundary and an intent gateway. On the one hand, applications render the artifacts inherited from lower layers—Task Execution Reports, Feedback Records, and Post-Execution Reports—into intelligible narratives, dashboards, or visualizations, while ensuring that the complete, unaltered artifacts remain available for audit. On the other hand, it is at this layer that new user intent is first bound to identity and purpose, with microidentities selected and attached to fresh orchestration requests. This duality positions Layer 8 as the central mediator between protocol-internal mechanics and the external sphere of human direction and oversight.

Importantly, Layer 8 does not alter the epistemic commitments or operational constraints that arrive from below. Its role is not to rescope contracts or revise evidence, but to translate protocol artifacts into accessible experiences and to translate human goals into new protocol requests. In this translation, complex internal concepts such as service-level objectives (SLOs), gas budgets, and degradation paths are abstracted into user-meaningful representations—for example, quick versus deep analyses, broad versus narrow scans, or conservative versus exploratory searches. Applications are free to design these abstractions as they see fit, but the protocol requires that all commitments eventually resolve to explicit contracts that lower layers can enforce.

Thus, Layer 8 can be understood as the intent-to-contract interface of SON. It provides a consistent, auditable surface where user agency is asserted, policy envelopes are bound, and results are rendered with clarity. By separating protocol invariants from application freedoms, this layer enables both innovation in user experience and fidelity to the core guarantees of provenance, reproducibility, and security that SON provides.

Goals and Objectives

The objectives of Layer 8 are defined by its position as the interface between SON's contractual machinery and the domain of human direction. At this layer, protocol invariants remain intact—artifacts arrive unchanged, new requests leave as fresh contracts—but the mode of interaction shifts toward accessibility, intelligibility, and identity assertion. The goals of this layer can be summarized as the faithful mediation of artifacts upward and the disciplined translation of intent downward.

The first objective is to assert identity and purpose at the boundary of intent. It is here that the user, through the application, selects a microidentity and binds it to a request. Lower layers may carry this identity forward or substitute system-level relationships depending on the object policy in play, but the originating assertion always occurs at Layer 8. This establishes a clear and auditable record of who is acting, under what authority, and with what declared purpose.

The second objective is to present artifacts in accessible form while preserving fidelity. Applications are expected to summarize, visualize, or narrate the results they receive from Layer 7—Task Execution Reports, Execution Checkpoints, Feedback Records—so that users can interpret them. Yet these artifacts must remain accessible in their complete, unmodified state. Layer 8 therefore balances usability with auditability: it may abstract for convenience, but it cannot obscure or overwrite the contractual record.

The third objective is to mediate complex system parameters through user intent. Service-level objectives, gas budgets, and degradation levers are internal system constructs that have no direct meaning for most users. The role of the application is to interpret user priorities—speed, depth, breadth, privacy—and translate them into contractual parameters. The user chooses among human-meaningful options, and the application expresses those choices as enforceable contracts. In this way, Layer 8 ensures that SON remains user-driven without requiring users to engage directly with technical abstractions.

The fourth objective is to permit composition without mutation. Applications may integrate multiple artifacts into dashboards or composite reports, but these are views rather than new epistemic objects. Any change of scope—expanding sources, tightening windows, relaxing obligations—must be expressed as a new orchestration request. Composition for presentation is permitted; composition that alters the record is prohibited.

Finally, the fifth objective is to serve as the audit boundary for human oversight. When a user asks "why?", "what evidence?", or "under which policy?", Layer 8 must be able to surface the complete contractual chain—from QRO to TERs, degradations, evidence handles, and policy references—without reconstruction. This makes Layer 8 the place where SON's internal transparency becomes externally actionable: where accountability is no longer a protocol principle alone, but a visible user experience.

Handoff from Layer 7

The transition from Layer 7 to Layer 8 marks the shift from delegated execution to application-level interpretation. At Layer 7, agents have completed their work under the explicit boundaries of Task Contracts, producing Post-Execution Reports, Feedback Records, Execution Checkpoints, and policy envelopes. These artifacts form the binding record of what was done, under which authority, and at what cost. Layer 8 inherits these artifacts unaltered. Its role is not to reinterpret results or rescope the work but to render them intelligible for users and applications, and to serve as the first point where fresh user intent is articulated as new contracts.

Two principles govern this boundary. The first is immutability of inheritance. All artifacts delivered upward from Layer 7 must be preserved in their original form. Applications may abstract, summarize, or visualize the results, but they cannot modify the underlying record. This ensures that when a user later asks why or how, the system can always surface the precise contract, evidence, and checkpoints that informed execution. The second is identity and policy continuity. The microidentity and policy envelope under which execution occurred must remain attached to the results at the point of handoff. When results are rendered to the user, they must be accompanied by a clear indication of who acted, under what asserted identity, and within which policy boundaries.

Finally, the handoff establishes Layer 8 as the intent gateway. Any decision by the user to expand scope, alter time windows, or pursue new goals must be expressed as a new orchestration request, carrying a fresh contract downward. No mutation of inherited artifacts is permitted; continuity is maintained by treating every refinement as additive and traceable. In this way, the handoff from Layer 7 preserves the chain of custody from plan to execution while opening the pathway for users to direct subsequent cycles of work.

Intent, Policy, and Presentation

Layer 8 is the threshold at which human purpose becomes a contractual commitment and where the record of delegated execution becomes legible. It bears three intertwined obligations. First, it mediates intent, translating outcome-oriented requests into complete, enforceable contracts without obliging the user to speak in system parameters. Second, it enforces policy at the point of presentation and consent, ensuring that identity, purpose, obligations, and data-handling rules are bound to every downward request and remain visible on return. Third, it preserves the integrity of the execution record, rendering reports and evidence intelligible while retaining the ability to expose the original artifacts unaltered. The following subsections develop these obligations in an academic register, grouping related responsibilities into broader themes.

Intent Mediation and Economic Authority

From outcomes to contracts. The user's vocabulary is qualitative—faster versus deeper, narrow versus comprehensive, private versus permissive. Layer 8 must convert such outcomes into the quantified commitments that lower layers require: scope, admissible layers, freshness, and resource budgets. This translation is an application concern, not a protocol prescription. The protocol requires only that a downward request be a complete contract: bound identity and purpose, declared scope, selected layers, and an authorized budget.

Quotes without exposure of internals. To make this translation intelligible, applications may present quotes—plan variants that mirror intent (e.g., a "quick skim" versus a "deeper corroboration"). Each quote corresponds to a concrete bundle of commitments, but the user need not see service-level objectives or gas budgets. Selection binds the choice to identity and purpose and authorizes the expenditure required. In this model, the wallet resides at Layer 8: only the application can approve spend; lower layers neither prompt for, nor accept, direct economic or credential input.

Composition as view, not revision. Where Layer 8 composes results into narratives or dashboards, composition remains a view. Any change of scope—more sources, different time windows, alternative layer profiles—must be expressed as a new contract rather than as an edit to inherited artifacts. This preserves the reproducibility of prior runs and the clear lineage of subsequent ones.

Policy Enforcement, Consent, and Security Posture

Policy at the boundary. Layer 8 is where policy becomes operational: identity is asserted, purpose is declared, consent (when required) is captured, and obligations are bound to the request. On return, the same envelope—who acted, under what authority, and with which obligations—must accompany results, so that users can see not merely what was concluded but under what constraints it was produced.

RBAC/ABAC context and purpose-binding. Policies may be role-based (RBAC) or attribute-based (ABAC), and they may require purpose-binding (e.g., "curation only," "investigation under incident X," "no export"). Layer 8 is responsible for assembling the policy context (identity, role, device posture, tenant, time, location, declared purpose) and binding it to the contract. Downstream policy decision points evaluate the envelope; enforcement points act on it. The assertion of context, however, originates here.

Security posture and obligation enforcement. Obligations—redaction, watermarking, minimization, cache controls, retention limits—are enforced before presentation. Objects must carry type and obligation metadata sufficient for the application to select safe handlers (e.g., an image viewer that applies watermarks; a PDF renderer that redacts classified segments). Where stronger access or cross-domain disclosure is implicated, Layer 8 surfaces the request for consent, records it, and includes it in the downward contract. On return, the presence of obligations is not optional gloss: it is part of the artifact set and must be preserved with results.

Separation of concerns for cryptographic control. Cryptographic authority (e.g., decryption keys) is not distributed to Layer 8 indiscriminately. Rather, Layer 8 supplies the policy envelope and identity to enforcement services; the latter perform controlled operations and stream filtered results under audit. This separation ensures that policy verification precedes payload release and that applications cannot bypass enforcement by rendering raw content directly.

Guardrails against silent erosion. Three prohibitions follow: (i) no silent rescoping—every widening of scope must become a new, auditable contract; (ii) no credential or budget prompts from lower layers—only Layer 8 may authorize spend or bind identity; (iii) no mutation of artifacts—summaries and visualizations must never replace the execution record.

Presentation, Provenance, and Auditability

Fidelity of the inherited record. Artifacts received from Layer 7—Task Execution Reports, Checkpoints, Feedback Records, policy envelopes, and as-of markers—are facts of execution. Layer 8 may render them intelligible, but it must be able to surface the originals unaltered. This requirement is not stylistic; it provides the substrate for explanation, dispute, and replay.

Explainability anchored in artifacts. Narrative answers are acceptable only insofar as they are anchored in retrievable artifacts. When the user asks why, the application must reveal the path: contract → checkpoints → evidence → policy and identity. Where composition or abstraction is used for readability, the link back to primary artifacts must remain intact.

Layer profiles and admissible perspectives. Because lower layers can expose many z-axis perspectives, Layer 8 curates layer profiles appropriate to context (e.g., "declarative + vetted mappings," or "declarative + partner inference layers"). The default profile should avoid silently widening trust. Selection of profiles forms part of the outgoing contract and the return record, so that provenance includes which perspectives were admitted.

Feedback as mediated signal. User judgments—acceptances, rejections, qualifications—are mediated by Layer 8 and transmitted downstream as Feedback Records. Feedback does not rewrite prior artifacts; it becomes new evidence that calibration and housekeeping can act upon. In this way, human oversight is converted into structured error backpressure without compromising the immutability of the execution record.

Consolidated Interface Obligations

Inputs. From Layer 7: Task Execution Reports, Execution Checkpoints, Feedback Records, policy envelopes, and as-of consistency markers. These are accepted as binding inputs.

Outputs. To Layer 6: complete, enforceable contracts—bound identity and purpose, declared scope and layer profile, authorized budget, and any required consent. Optionally, mediated feedback concerning prior runs. At no time does Layer 8 emit partial or implicit requests; every action is a contract or it is nothing.

Default failure semantics. When outcomes cannot be delivered within the authorized commitment, Layer 8 must (i) disclose degradations explicitly (e.g., narrowed scope, reduced freshness), (ii) present revised plan variants, and (iii) if the user consents, issue a new contract. Silent failure or silent degradation is inconsistent with the protocol's audit posture.

Taken together, these themes define the intent and responsibilities of Layer 8. Lower layers plan and execute under contracts; Layer 8 keeps the record legible, binds identity, consent, and obligations, owns the wallet, and converts human purpose into enforceable commitments. The user remains in a world of outcomes. The protocol remains in a world of contracts.

Policy as Presentation

One of the more subtle responsibilities of Layer 8 is that it must not only enforce policy envelopes but also render those policies intelligible to the user. Policy in SON is not invisible background machinery; it is part of the provenance of every artifact. When a result is redacted, when an obligation such as watermarking is applied, or when a query is denied because of scope, these are not incidental outcomes. They are protocol-governed events that must travel with the artifact as clearly as confidence scores or evidence handles.

This means that Layer 8 applications bear the duty of treating policy as a first-class object of presentation. When a user inspects results, they should be able to see not only what was concluded but also why certain information is missing or altered. A redacted field should link to the obligation that required it; a watermark should be traceable to the enforcement profile that applied it; an access denial should cite the policy decision and its attributes (e.g., insufficient role, expired purpose). In this way, policy becomes auditable evidence rather than an opaque barrier.

The challenge here is interpretive. Users are rarely conversant in ABAC or RBAC models, nor in the technicalities of enforcement profiles or microidentity chains. Layer 8 therefore must provide a translation layer, rendering complex policy decisions into accessible explanations: “This section was removed under archival-access policy,” or “Export is disabled for this dataset under your current research role.” The protocol does not prescribe how this is done; it requires only that the necessary metadata—policy identifiers, obligation classes, enforcement references—arrive intact and be available for presentation.

Treating policy as presentation reinforces the larger posture of SON. Transparency is not limited to data and inference; it extends equally to security, consent, and obligation. By surfacing policy decisions as visible elements of the user experience, Layer 8 ensures that the system’s boundaries are not hidden but explained, allowing users to understand not only what the system did, but also what it refused to do and why.

Profiles and Layer Curation

Layer 8 is where the system’s plurality of perspectives is made usable. Earlier layers expose a spectrum of z-axis layers—declarative facts, vetted mappings, contextual narratives, provisional hypotheses—with explicit provenance and confidence. But users rarely want “all layers, all at once.” The application must therefore curate named profiles—coherent, reusable selections of layers, scopes, and obligations—so that a given intent is satisfied without forcing the user to hand-tune every query.

A profile is a compact contract the application composes on the user’s behalf. It specifies which layers are admissible (allow-/deny-lists), the default treatment of uncertainty (confidence thresholds, degradation acceptance), the admissible obligations (e.g., watermark required; export disabled), and any scope fences (e.g., time windows, data classifications, or tenant boundaries). Examples are natural:

- R**esearch Profile: declarative + vetted mappings;** include external reference layers; show uncertainty as intervals; allow broader scope if provenance is strong.
- **Compliance Profile:** declarative only; exclude exploratory inference; strict denial on missing provenance; show all policy decisions inline.

- **Exploration Profile:** include provisional hypotheses; lower the initial confidence threshold, but require explicit labeling of uncertainty and layered drill-downs.

Profiles do not alter the protocol. They simply bind together, at the application tier, a repeatable selection of layers and behaviors into a named choice that users (or teams) can adopt, share, or set as default. They also make intent legible to lower layers: when Layer 8 issues a new orchestration request, the profile is translated into concrete parameters—layer allow-lists, SLO/gas posture, admissible degradations, and any purpose-binding needed for policy evaluation.

Crucially, curation remains transparent and reversible. Every rendered view should reveal which profile is active, which layers it included or excluded, and how uncertainty and obligations were handled. Users must be able to switch profiles without losing access to the underlying artifacts, and investigators must be able to override defaults with explicit, auditable changes. In this way, Layer 8 converts the system's layered richness into a small set of intent-appropriate modes, while preserving the full fidelity and auditability of the knowledge beneath.

Transparency and User Mediation

Layer 8 is the first place in the protocol stack where the raw mechanics of confidence, degradation, and provenance are translated into forms that users can interpret and act upon. Lower layers generate these metrics with mathematical precision—confidence intervals, degradation logs, and provenance chains—but they do not decide how those signals should be conveyed. The responsibility of user mediation belongs to the application. This mediation is not cosmetic; it is integral to ensuring that SON remains transparent and usable without obliging users to inhabit the technical vocabulary of the system. Three dimensions illustrate this responsibility: the negotiation of uncertainty, the structuring of feedback, and the management of composite presentation.

Negotiation of Uncertainty

Earlier layers are careful to maintain explicit measures of uncertainty—confidence intervals, error ranges, reputational weights, or degradation annotations. At Layer 8, these abstract metrics must be rendered in forms intelligible to human decision-makers. One application may display confidence as numeric intervals or percentages, another may use metaphorical signals such as "traffic lights," and a third may expose sliders that allow users to tune thresholds directly. The protocol does not prescribe which method is correct; it requires only that the underlying metadata—confidence scores, provenance references, and degradation records—remain available for presentation. What matters is that uncertainty is never hidden. Every choice about how to render it is an interpretive decision of the application, but every user must be able to see not only the conclusion but also the contours of its reliability.

Feedback as Structured Input

Layer 8 is also where human judgments are first captured. A user may accept a result, reject it, qualify it, or request clarification. These judgments are not casual annotations; they are signals that, when structured, become part of the epistemic cycle. The application mediates this translation, converting judgments into Feedback Records routed back through Layer 7. The protocol enforces immutability—Feedback Records do not retroactively edit artifacts—but Layer 8 determines how users articulate those signals: whether through simple accept/reject buttons, ranked scoring, or free-form commentary bound to evidence handles. In this sense, feedback is both epistemic and experiential: epistemic because it affects confidence and calibration downstream, experiential because the application decides how to make feedback collection natural for its users.

Composite Presentation and Tunable Collapse

Finally, Layer 8 governs how results from multiple layers—declarative facts, inferential linkages, contextual summaries—are composed for presentation. A user may want only the “fact core” of a response, stripped of context and inference; another may want the full synthesis, including low-confidence leads or competing narratives. This tunable collapse—the ability to narrow or broaden what is shown without altering the underlying record—is a defining responsibility of the application. The protocol ensures that every component (fact, inference, context) arrives tagged with provenance and confidence; the application determines how to compose them, how to collapse or expand the view, and how to disclose which layers were included or excluded. By doing so, Layer 8 allows SON to support diverse modes of interaction—from conservative compliance dashboards to exploratory research assistants—without sacrificing the integrity of the underlying artifacts.

Together, these three dimensions show how Layer 8 mediates between the rigor of the protocol and the realities of human interpretation. Uncertainty is negotiated, not erased; feedback is structured, not informal; composition is tunable, not absolute. In each case, the application translates system-level detail into humanly meaningful experiences while ensuring that the underlying artifacts remain intact, auditable, and retrievable. This is the essence of Layer 8’s contribution: it is the place where SON’s transparency becomes usable transparency.

Implementation Notes

Layer 8 is where SON touches the user most directly, yet its responsibilities are defined with unusual care. The protocol does not prescribe how an application should be built, nor does it constrain the diversity of user experiences that can be layered over the system. What it does require is that any implementation at this tier meet a set of invariants: the integrity of artifacts, the binding of identity and purpose, the enforcement of obligations, and the translation of intent into contracts. Within these boundaries, applications are free to innovate.

Application extensions.

Implementations may register local extensions—document renderers, image viewers, parsers, analyzers—that handle specific payload types. These are analogous to the way a web browser delegates rendering of a PDF to an embedded plugin. The protocol imposes two responsibilities: (i) SON objects must carry schema metadata (type, obligations, safety labels) sufficient to identify the correct extension, and (ii) the application must enforce obligations such as redaction, watermarking, or no-export before invoking the handler. The choice of extensions is entirely local; the requirement is that their invocation is policy-aware and traceable.

Wallet authority.

Only the application can authorize expenditure of gas on behalf of the user. Lower layers may never prompt the user directly for credentials or credits. Instead, the application interprets intent, translates it into quotes or plan variants, and commits to the contract by attaching identity and approving the spend. This separation ensures both economic security and accountability: if gas was spent, it was because the application bound it to identity and issued a contract.

Policy enforcement at the boundary.

The application is also the locus of policy enforcement. Identity and purpose are asserted here; obligations are surfaced and bound to contracts; and on return, results must be shown with the policy envelope intact. Enforcement is not a hidden detail—it is a presentational responsibility. If access is denied, or if a payload is redacted or watermarked, the application must show why, referencing the relevant obligation or enforcement profile. This guards against the silent erosion of security: users see the decision, not just the output.

Safety and failure handling.

Applications must handle results defensively. If an extension fails, the underlying artifact must remain available for inspection. If a degradation was applied to meet constraints, that fact must be disclosed in the presentation layer. If a contract cannot be executed within the declared budget, the application must present the alternatives—accept the degraded plan, or issue a new contract with expanded bounds. The invariant is that no silent rescoping may occur: all changes must be surfaced as explicit contracts or degradations.

Composition discipline.

Where applications create composite dashboards or summaries, those compositions remain views, not new facts. Implementations must resist the temptation to “flatten” results into new authoritative statements. Instead, each element of a view must preserve links back to the original TERs, Checkpoints, Feedback Records, and policy envelopes. This ensures that composition does not become mutation, and that audits can always traverse from surface to source.

These notes do not exhaust the possibilities of Layer 8; rather, they mark the edges of safe implementation. Applications may innovate in UX, experiment with different quoting metaphors, or build specialized dashboards for particular domains. What they may not do is obscure the record, spend without consent, rescope silently, or enforce obligations inconsistently. By policing these boundaries, the protocol guarantees that while Layer 8 is where SON is most malleable and diverse, it is also where SON is most accountable.

Performance Optimization

Performance in Layer 8 cannot be judged by traditional measures of rendering speed or interface responsiveness alone. In the context of SON, performance optimization at this layer means ensuring that contracts are faithfully expressed, results are transparently conveyed, and deviations from requested outcomes are never hidden. The protocol does not prescribe the mechanisms by which applications achieve responsiveness or usability; instead, it defines the invariants that must hold for any application to be considered conformant.

Boundedness of contracts.

Every request leaving Layer 8 must carry explicit bounds—scope, admissible layers, and authorized budget—derived from user intent. If those bounds prove insufficient, the system cannot silently rescope. Instead, Layer 6 must respond with a feasibility decision (accept, accept-with-degradation, reject), and the application must either disclose the degradation or return to the user for consent to issue a new contract. This ensures that performance shortfalls are visible and that resource use remains under explicit control.

Declared degradations.

When lower layers reduce scope, truncate time windows, or skip layers in order to meet performance constraints, those degradations must flow back as artifacts in Task Execution Reports and Checkpoints. Layer 8’s responsibility is to surface them in intelligible ways—whether as warnings, annotations, or comparative indicators—so that the user sees not only the result but also the path by which it was achieved. Degradations cannot be absorbed silently into narrative summaries; they are protocol-level facts that must remain visible.

Replayability and reproducibility.

Results presented in Layer 8 must always be traceable to the as-of envelope, contract ID, and policy context that governed their production. This means applications must preserve identifiers and links, even when summarizing or composing results. Performance, in SON's terms, is not only about speed of delivery but about the ability to reproduce the same outcome under the same conditions. Layer 8 may optimize display and interaction, but it must not sacrifice the ability to retrieve or replay the exact artifact chain.

Observability of commitments.

Applications are required to retain and present the essential performance metadata: contract identifiers, gas allocations versus actual spend, SLO requests versus achieved outcomes, and any degradations applied. Observability ensures that performance is not an implementation detail but a verifiable property of the record. When a user asks "why did this take longer?" or "why was coverage reduced?", Layer 8 must be able to point to explicit artifacts rather than speculation.

No silent rescoping.

Above all, performance optimization in Layer 8 is defined by the prohibition against silent change. The application may abstract, visualize, or reframe results, but it cannot alter scope, conceal degradations, or spend gas without explicit consent. Where system pressure requires deviation from intent, that deviation must appear in the record and, where appropriate, must trigger new contract formation.

In this framing, Layer 8's performance is not about how fast the UI renders but about how faithfully it mediates the commitments of the protocol. Boundedness, declared degradations, replayability, observability, and prohibition of silent rescoping together ensure that Layer 8 remains the layer of trust: the place where human expectations are reconciled with system guarantees without compromise.

Handoff to Layer 9 (The User)

The transition from Layer 8 to Layer 9 is unlike any of the handoffs below it. Where earlier boundaries pass artifacts between machines—agents, orchestrators, and knowledge graphs—the handoff here delivers those artifacts into the realm of human comprehension and control. Layer 8 has already performed the work of mediation: contracts have been translated into outcome-oriented choices, policy envelopes have been surfaced as intelligible constraints, and execution artifacts have been rendered in forms the user can navigate. What passes upward to Layer 9 is not a new protocol object, but a legible chain of commitments and results that the user can interrogate, validate, or reject.

Three principles define this handoff.

Immutability of the record. The user must be able to see the artifacts as they were produced: the Query Resolution Object, Task Contracts, Execution Checkpoints, Task Execution Reports, Feedback Records, policy references, and as-of envelope identifiers. Layer 8 may abstract these into dashboards or narratives, but Layer 9 must always have access to the unaltered record. In this way, the human participant interacts not with reconstructions or summaries alone but with the authentic evidentiary trail.

Transparency of policy and obligation. Every obligation that shaped a result—redactions, watermarking, no-export rules, cache limits—must be surfaced at this boundary. The user should not only receive the data, but also an account of why it appears in the form it does. If material is absent, the explanation must be visible: "removed under archival-access policy" or "display restricted under tenant boundary." At Layer 9, policy is no longer background enforcement; it is part of the user's understanding of the system's behavior.

Agency through new intent. Finally, Layer 9 is the source of fresh purpose. When a user asks, "exclude this source," "expand to a broader window," or "show me only declarative facts," those directives must be converted at Layer 8 into new contracts with explicit scope, identity, and budget. The handoff to Layer 9 therefore opens a loop: it grants the user the ability to interrogate the record and, by doing so, to generate new intent that re-enters the stack.

In this sense, the Layer-8 to Layer-9 handoff is both terminal and generative. It is terminal because the record of execution has reached its point of maximum transparency: the human can see what was done, under what authority, and with what obligations. It is generative because this transparency equips the human to refine intent, creating the conditions for the next cycle of contracts. The seam is thus not a stopping point, but the place where SON becomes accountable to its ultimate arbiter—the user.

Implementation Intentions

Layer 8 shall bind human purpose to protocol without loss of integrity. Implementations at this tier are free to innovate in presentation and interaction, but the artifacts and constraints that arrive from lower layers must remain intact, and all new requests must re-enter the stack as explicit, bounded contracts. Identity and purpose originate here and travel downward; policy and obligation return here and must be made visible. Economic authorization (gas) is exercised only at this boundary. In every respect, Layer 8 is accountable for turning intent into enforceable commitments and for returning results with their provenance and obligations unaltered and intelligible.

Conformance checklist (invariants):

- **Identity & Purpose Binding.** Every outgoing request binds a microidentity and declared purpose; no lower layer may solicit credentials or assert identity.
- **Contract Boundedness.** Outgoing requests specify scope, admissible layers, and an authorized gas/SLO posture. Any change of scope is effected only by issuing a new contract or by an explicit, returned degradation.
- **Economic Authorization.** Only the application can authorize spend; all gas usage is attributed to a contract and surfaced to the user with actual-versus-authorized accounting.
- **Obligation Enforcement & Disclosure.** Returned results are presented with the active policy envelope; redactions, watermarking, no-export, cache, or retention constraints are enforced and explained.
- **Artifact Fidelity.** Task Contracts, Execution Checkpoints, Task Execution Reports, Feedback Records, and as-of envelopes are preserved and retrievable; compositions are views only, never mutations.
- **Feedback as Artifact.** User judgments are captured as structured Feedback Records routed through Layer 7; prior artifacts are never edited retroactively.
- **Declared Degradations.** Any deviation from the requested plan (reduced scope, time, or layers) is disclosed in the returned artifacts and made visible in the presentation.
- **Replayability & Audit.** Results are reproducible under the same envelopes; contract IDs, policy references, gas ledgers, and degradation notes are retained for audit.
- **No Silent Rescoping.** The application must not widen scope, include additional layers, or increase spend without explicit user consent and issuance of a new contract.

These intentions distinguish Layer 8 from prior layers by making it the only locus where identity is asserted, budgets are authorized, user feedback is formalized, and obligations are presented as part of the answer. Earlier layers execute, score, and govern; Layer 8 commits and discloses.

Illustrative Example: Miles Davis — Quote, Contract, and Presentation

A user asks the application: "Is there credible evidence of an unpublished Miles Davis album?" Layer 8 receives this as intent and proceeds to mediate it into a contract without exposing SLOs or gas directly.

1. Quotes (intent → contract options).
The application proposes two plan variants in user terms:

- Quick skim. "Check major catalogs and public registries; short wait; limited depth." (Internally: narrow scope, low-latency SLO, modest gas cap, declarative layers only, no secure label APIs.)
- Deeper pass. "Add minor registries and session notes for 1972–1975; longer wait; higher confidence potential." (Internally: broader scope, higher latency/gas, include specific inference/context layers, permit one secure label API under obligations.)

The user selects Quick skim. Layer 8 binds identity ("Researcher / Public Collections") and authorizes spend from the wallet, issuing a contract with explicit bounds (scope, admissible layers, gas/SLO posture).

2. Execution artifacts return (unaltered).
Layer 6 & 7 execute. Layer 8 receives unaltered artifacts:

- TER-1 (Catalog Verification): two majors negative; one pending-review entry. Gas 18/25; SLO met; degradation: breadth reduced 5→3.
- TER-2 (Provenance Cross-walk): no corroboration in label metadata; degradation: single-hop only. Gas 22/30; SLO met.
- TER-3 (Rights/Release): no scheduled release; gas 9/10; SLO met.
- FR-A (Feedback Record): "Provisional catalog entry judged low confidence absent corroboration; recommend broader sources or expanded time window."

All artifacts carry contract ID, as-of envelope, policy references, and evidence handles.

3. Policy as presentation (enforcement made visible).
The application renders a narrative and surfaces obligations: "Access to secure label calendars is disallowed under current identity; scans would be watermarked and no-export if permitted." Redactions or denials are explained with the active policy envelope; nothing is hidden.

4. Tunable collapse (fact-only vs synthesis).
The user toggles:

- Fact-only view: declarative findings only (TER-1…3).
- Synthesis view: includes the FR-A recommendation and confidence posture.

Both views preserve links to TER/Checkpoint/evidence artifacts; composition remains a view, not a mutation.

5. Feedback capture (structured, not retroactive).
The user marks the pending catalog entry as "plausible but uncorroborated" and adds a note pointing to the evidence handle. Layer 8 converts this into a structured Feedback Record, routes it through Layer 7, and leaves prior artifacts unchanged.

6. Follow-up contract (wallet authority, no silent rescoping).
Prompted by the application, the user chooses Deeper pass. Layer 8 switches microidentity to "Curator / Label Archives", surfaces new obligations (watermarking, retention limits), and authorizes the higher gas/SLO posture. A new contract is issued (no silent rescoping); the previous record remains intact and replayable.

Result.
Layer 8 has mediated the entire cycle without exposing internal vernacular. Quotes expressed options in outcomes, not SLOs; the wallet authorized spend; artifacts returned unaltered; policy was presented as part of the answer; uncertainty was made legible; feedback became a structured, auditable signal; and further action required a fresh, bounded contract. This is Layer 8's role made concrete: intent in human terms, commitments in protocol terms, and transparency all the way through.

Research and Technology Implementation Foundation

Layer 8 sits at the intersection of several long-standing research traditions: explainable artificial intelligence (XAI), usable security, human-in-the-loop learning, and the design of secure application extensions. Each of these areas provides mature insights into how complex systems can be made legible to people without weakening the underlying guarantees of rigor and enforceability. SON draws on these traditions but reframes them as protocol requirements: identity must be bound, artifacts must be preserved, policy must be surfaced, and all user actions must resolve to explicit contracts.

Explainability and human comprehension.

The DARPA XAI program (Gunning & Aha, 2019), as well as influential methods such as LIME (Ribeiro et al., 2016) and SHAP (Lundberg & Lee, 2017), demonstrated that explanations must be tailored to human understanding rather than exposing raw model internals. SON's Layer 8 follows this logic: confidence scores, degradation traces, and provenance chains are available as raw data, but the application decides how to present them—whether as traffic-light metaphors, numeric ranges, or interactive filters. Research on model cards (Mitchell et al., 2019) and system cards extends this idea, showing that structured presentation of uncertainty and policy constraints improves trust and accountability.

Human feedback as structured input.

Recent advances in reinforcement learning with human feedback (Christiano et al., 2017) and constitutional AI (Bai et al., 2022) show how user judgments can shape large-scale models. SON adopts the same insight but formalizes it as protocol: Layer 8 captures feedback and routes it through Layer 7 as Feedback Records, which become part of the immutable epistemic cycle. In contrast to free-form annotations, feedback in SON is structured, auditable, and directly linked to the artifacts it critiques—an approach that aligns with calls in the literature for feedback as first-class data rather than informal commentary.

Policy-aware computing and usable security.

The broader security community has long emphasized the need to make policy decisions visible and intelligible. Work on role-based and attribute-based access control (Sandhu et al., 1996; Hu et al., 2015) and on usable security (Cranor, 2008) argues that users must understand why access is granted, denied, or modified. Layer 8 treats these insights as protocol requirements: every redaction, watermark, or denial must be surfaced alongside the returned artifacts, with references to the enforcement profile that applied them. This positions SON not simply as a knowledge system but as a policy-aware communication channel, where security and privacy are part of the explanation itself.

Application extensions and sandboxing.

In industrial practice, extensibility is handled through controlled plugin models—browsers invoke PDF viewers or video decoders, mobile operating systems rely on app stores and sandboxing. SON's Layer 8 mirrors this model: applications may register extensions to handle specific object types (JPEGs, PDFs, PCAPs), but the protocol requires that type metadata and obligations accompany every object, and that enforcement (redaction, no-export, watermarking) precede rendering. This parallels best practice in browser security (Barth et al., 2009) and secure software extension frameworks.

Economic governance and wallet models.

The requirement that Layer 8 act as the user's "wallet" finds precedent in blockchain and decentralized systems, where gas models regulate computational spend (Wood, 2014). Yet SON diverges by treating gas and service-level objectives as system internals never shown directly to the user. Instead, the application interprets intent, obtains quotes, and authorizes expenditure on behalf of the user. This reframing echoes research on usable privacy and consent (Garfinkel, 2012), which stresses that decisions must be bound to identity and purpose but mediated through intelligible interfaces.

Synthesis.

Taken together, these literatures confirm that Layer 8's responsibilities are not arbitrary design choices but reflect a convergence of academic and industrial practice. XAI research underlines the need for explanation in human terms; RLHF and constitutional AI demonstrate the value of structured feedback; usable security highlights policy transparency; extension models show how local handlers can be sandboxed; and blockchain governance illustrates wallet authority. SON's contribution is to weave these threads into a protocol surface: at Layer 8, applications are free in design but bound by invariant obligations—identity binding, artifact fidelity, policy disclosure, wallet authority, and audit completeness. In this way, Layer 8 grounds decades of research in a formal architecture where human intent becomes system action without loss of transparency or control.

Layer 9: The User

Layer 9 defines the human boundary of SON: the point at which answers must be rendered intelligible without loss of evidentiary discipline, and at which new intent must re-enter the system as explicit, bounded contracts. The layer does not introduce a new epistemic source or a new computational mechanism. Rather, it specifies what the user is entitled to see and how subsequent actions must be bound to the protocol, irrespective of any particular interface or implementation. In this sense, Layer 9 converts the inner guarantees of SON—persistent objects, separation of facts from inference, confidence and reputation as first-class signals, policy-constrained access, and orchestrated execution—into a visible and steerable surface. An answer is not delivered as a bare assertion but as a narrative whose grounds remain available on demand: the identifiers of the contracts that produced it, the as-of envelope that framed the read, the layers consulted, the evidence chains invoked, the confidence intervals and reputational attributions attached to relationships, and the policy envelope under which any payload was revealed or transformed. The user's subsequent choices—exclude a layer, tighten the time window, raise a confidence threshold, or pursue deeper corroboration—do not mutate prior artifacts. They are expressed as new requests that carry identity, purpose, and an authorized resource posture, preserving the epistemic and operational separations established below. In short, Layer 9 is where legibility meets enforceability: explanation rests on retrievable artifacts rather than reconstruction, and revision proceeds by fresh, attributable contracts rather than retroactive edits.

Goals and Objectives

Layer 9 specifies the human side of the protocol. Its purpose is not to prescribe interface mechanics, but to guarantee the kinds of control a person can exert over a SON system and the forms in which that control must be respected downstream. Earlier layers provide the machinery—objects and z-axis layers (L5), orchestration and envelopes (L6), delegated enactment (L7), and application mediation (L8). Layer 9 fixes the rights and levers available at the boundary so that intent, consent, scope, and explanation are not left to implementation detail but become predictable protocol effects.

Intent as a first-class, binding act. The primary objective is that a person's intent, once expressed, becomes a binding contract with explicit scope, admissible layers, and time bounds. Intent is not a hint to be heuristically interpreted; it is the source of authority for what follows. The system must therefore convert human aims—"faster", "deeper", "exclude this source", "narrow to Q3 2024"—into enforceable parameters under an as-of envelope, rather than mutating prior artifacts. Revision of intent yields a new contract; it never rewrites the past. In this way, human direction remains additive and attributable, and every change in course is auditable as a fresh commitment.

Consent and obligation as part of intent. Layer 9 also requires that consent be treated as an element of intent, not an afterthought. When a request implicates stronger access (e.g., consulting a sensitive directory) or attaches obligations (redaction, watermarking, retention), the application must surface these terms in human language and bind the accepted terms to the contract. On return, the same policy envelope accompanies the result so the person can see not merely what was shown, but under which authority and obligations it was made visible. Consent is thus stateful and reviewable: it lives with the request and travels with the answer.

Economic mediation under human control. Because every operation consumes resources, the protocol authorizes spend only where the application, on the user's behalf, has committed to a plan. Layer 9's objective is that budgeting and trade-offs remain human-centric: SLOs and "gas" are implementation internals, but the person must be able to choose among outcomes (e.g., "quick skim" vs. "deeper corroboration") that the application maps to concrete bounds. Once selected, those bounds become part of the contract; any deviation (degradation or partial execution) must be disclosed as fact, not hidden in heuristics. In short, quotes in human terms; contracts in system terms—with achieved-versus-requested recorded for inspection.

Scoped control over sources, layers, and uncertainty. Layer 9 ensures that the person can shape the epistemic footprint of a run without knowing internal schemas: include or exclude specific sources, select named layer profiles, raise or lower acceptance thresholds, or confine results to declarative facts. These choices are not UI toggles alone; they are protocol constraints that must be reflected in the contract and in downstream selection and scoring. Uncertainty—confidence intervals, reputational weights, and applied degradations—must be surfaceable on demand in forms a person can understand, while the underlying metadata remains intact for rigorous audit.

Rights of interrogation, reproduction, and remedy. Finally, Layer 9 guarantees three non-overlapping rights. First, the right to interrogate: to ask "why?", "show the evidence", "which layers contributed?", and receive the authentic record—contracts, checkpoints, evidence chains, and policy outcomes—rather than a reconstruction. Second, the right to reproduce: to rematerialize the answer as-of its envelope, distinguishing requested from achieved scope and authorized from actual spend. Third, the right to remedy: to issue a new, narrower or broader contract (e.g., excluding a source or tightening time) or to register structured feedback that becomes part of the formal record without retroactively mutating the underlying artifacts. Together, these rights ensure that the user's agency is effective (it changes what the system does next), explainable (it shows what the system already did), and replayable (it proves what was true at the time).

These objectives preserve a clean division of responsibilities. Human purpose enters as contractual intent with explicit consent and economic authorization; the system responds under those terms, returns results with their obligations and uncertainty visible, and retains the full record so that the person can interrogate, reproduce, and revise with precision. In this way, Layer 9 secures the point of contact where legibility becomes control—not by dictating user interfaces, but by ensuring that every UI choice can be realized as a concrete, enforceable effect in the protocol below.

Handoff from Layer 8

The transition from Layer 8 to Layer 9 does not introduce new computational artifacts; it introduces points of assertion that become visible to the human participant. What Layer 8 delivers upward is the ability to see, accept, or contest the results of a query in ways that are bound to the protocol but framed in terms the user can control. This is the first time in the stack that identity, consent, scope, and trust become matters of explicit human choice rather than implicit system state.

At this boundary, the person is presented with results that are accompanied by intelligible claims about their origin, their confidence, and the conditions under which they were revealed. They can assert whether to expose their identity in further queries, or to act anonymously if the policy and obligations permit. They can decide whether to accept the obligations attached to certain kinds of access, or to withhold consent and forgo those results. They can interrogate the explanation—asking why a conclusion was drawn, what evidence underpins it, and which perspectives were admitted into the reasoning. And they can shape future direction, by instructing the system to exclude sources, to broaden or narrow time windows, or to require stronger corroboration.

In this sense, what passes upward from Layer 8 is not a bundle of technical records but a set of variables the human may exert over the system. Identity can be disclosed or withheld; consent can be granted or denied; explanations can be demanded; scope can be revised. The protocol defines these as rights of assertion, ensuring that when human intent meets machine execution, it does so as a structured exchange: the system shows not only what it did, but under what conditions, and the user responds not with opaque gestures but with choices that become fresh, attributable contracts.

The handoff to Layer 9 does not deliver code, contracts, or control structures to be inspected; it delivers assertions the person can make. What has been prepared below—results made legible, obligations rendered in plain terms, preferences framed as attainable choices—now becomes subject to human decision. At this boundary, the person can declare whether a request carries their identity or is issued without disclosure where rules allow; whether obligations are accepted or refused; whether an explanation is required before proceeding; how scope should be narrowed or widened; which outcome should be preferred when time and depth must be traded; whether a summary is enough, or the unaltered account must be shown; and whether the course should be revised by issuing a new request rather than mutating the last. In short, Layer 8 presents what is knowable under the protocol; Layer 9 asserts what is acceptable under the person's will.

Assertions at the Human Boundary

Layer 9 formalizes the point at which the person—not the application nor any agent—determines how knowledge will be revealed and how new requests will be bound. What arrives from Layer 8 is already made legible: results are presented in human terms, policy constraints have been translated into obligations, and preferences have been framed as attainable choices. Yet none of this substitutes for the person's authority at the boundary. Here, assertions are not incidental settings or hidden defaults; they are the means by which human intent acquires force in the protocol. Each assertion—about identity, consent, explanation, scope, cost and timeliness, presentation, and revision—defines a specific lever of control. These levers do not specify how the system is implemented; they specify what the person can insist upon and what the system must respect in response. The subsections that follow state these assertions as rights and responsibilities, and describe how they discipline the exchange between human direction and machine execution.

Identity as a deliberate act. Identity is asserted, not presumed. The person chooses whether a request carries a named microidentity, a different role, or no identity disclosure where rules permit. This choice is presented in human terms ("act as researcher", "act as curator", "act anonymously") and its implications are explained before any request proceeds: what avenues remain open, which views will be withheld, and how obligations would change were identity to be disclosed. The principle is simple: identity at this boundary is a reversible, explicit act of intent. It is attached to a request only when the person affirms it, and it extends no further than that request.

Consent as a condition of access. When access entails obligations—retention windows, watermarking, redaction, or cross-domain use—those terms must be shown plainly and accepted before the system acts. Consent at this layer is specific, time-bounded, and attributable: specific to the material and purpose at hand; bounded to the current request (not standing authorization); and attributable to the person's decision. Refusal leaves the record intact and the person uncommitted. The system may offer alternative views that require fewer obligations, but it cannot proceed under stronger terms than the person has accepted.

Explanation as a right of inquiry. Answers appear with grounds that can be examined on demand. The person may ask why this conclusion was reached, which perspectives were admitted, how uncertainty was handled, and where constraints altered the course of execution. The response is given in intelligible form first (a narrative that names sources, confidence, and trade-offs) with the ability to inspect the underlying record if the person wishes to go further. Explanation here is not optional or merely diagnostic—it is the user's right to make sense of what was done in their name, and to withhold further direction until that right is satisfied.

Scope as a steerable constraint. Direction at this boundary is exercised by editing scope, not rewriting history. The person can exclude a source, narrow or widen the time window, raise an acceptance threshold, require stronger corroboration, or request deeper exploration. Each change is treated as fresh intent: it produces a new, bounded request that leaves prior results intact and attributable. This preserves a legible chain of decisions—what was asked, under which constraints, and how the course of inquiry evolved—rather than mutating earlier work.

Cost and timeliness as preferences, not burdens. The person does not manage internal budgets or service targets. They choose outcomes, and the application interprets those choices into the system's terms. Typical preferences—"quick scan" versus "deeper pass," "fact-only" versus "full synthesis"—declare what the person values in the moment. If the system cannot fully satisfy a preference, it must disclose what was achieved (for example, a narrowed scope or reduced breadth), and what a different choice would entail. In this way, cost and timeliness remain human priorities, not technical burdens.

Presentation without loss. Abstraction assists comprehension, but it does not diminish rights. Summaries and visualizations are welcome at the surface, yet the unaltered account remains available: what was consulted, under what conditions, with what effect. The person may toggle between narrative and record at will, without relying on reconstruction or memory of a prior state. Presentation, in other words, is a layer of clarity—not a filter that occludes provenance or constraint.

Revision as a new commitment. When the person changes course—accepting terms previously declined, disclosing identity for restricted views, or adjusting scope—those decisions become new, attributable commitments. The prior exchange is preserved, along with its consequences. Revision thus serves agency without erasing history: it is how the person adds to the sequence of intent, not how the system is permitted to rewrite it.

Revocability and Precision Withdrawal. Layer 9 must accommodate the reality that intent is not static. A person may disclose identity for one request and later decide that disclosure should not persist; they may accept an obligation for a particular view and later withdraw consent. The protocol consequence is not to erase history, but to make revision precise and reversible. Because SON separates objects from the layers that reference them, withdrawal can operate by retiring or pruning the relevant layered references rather than corrupting the underlying record. The original exchange remains auditable ("what was asked and under which terms"), while subsequent use is governed by the revised assertion. Practically, this means that a prior acceptance does not silently propagate to future queries, and a refusal does not require the system to pretend the past did not occur. The system exposes both the prior commitment and the present stance, and it binds future computation to the latter. In this way, revocability is not a concession; it is the mechanism by which continuing agency is made concrete.

Delegation of Agency. Not every user will wish to exercise every lever on every request. Layer 9 therefore admits delegated assertions under explicit scope. A person may authorize a proxy—an application profile, an organizational officer, or a narrowly defined agent—to select identities, accept routine obligations, or apply a default layer profile on their behalf, within stated limits and for a limited time. Delegation is not a transfer of ownership; it is a constrained agreement to act within enumerated boundaries ("use this microidentity for research queries," "accept watermarking for public scans," "prefer declarative-only profiles unless overridden"). Each delegated act remains attributable to the delegator, each scope is visible at the interface, and each delegation is revocable as described above. The effect is to keep the locus of control with the person while recognizing that effective use often requires stable defaults and trusted intermediaries.

Trust Calibration and Discernment. Transparency has little value if it cannot be interpreted. Layer 9's role is not only to expose evidence, confidence, and policy outcomes, but to help the person calibrate trust in ways that align with their purpose. This is a matter of presentation, not persuasion: surfacing how many independent perspectives contributed, whether corroboration spans diverse sources, how uncertainty changed under degradation, and what parts of the result would differ if the person chose a stricter profile. The system does not dictate a conclusion; it renders reliability legible. In some situations a quick, low-cost scan is sufficient; in others, only deeply corroborated findings are acceptable. Trust calibration therefore appears as a set of intelligible contrasts—requested versus achieved, breadth versus depth, single-source versus multi-source—so that the person can decide whether to proceed, to tighten scope, or to invest in a more demanding plan.

Interoperability of Human Assertions. A distinguishing feature of SON is that agency travels with knowledge. Assertions made at the user boundary—identity disclosure, consent to obligations, exclusion of layers, preference for specific profiles—must be portable when results cross organizational or system boundaries. Layer 9 treats these assertions as first-class, time-stamped commitments that can accompany a result as presentation metadata, so that a downstream system can honor the person's choices without rediscovery or reinterpretation. This is not a technical export of artifacts; it is the propagation of terms. Where a receiving system cannot honor the original assertions, the protocol expectation is explicit: it must disclose the mismatch to the person (or their delegate) and obtain fresh consent, or decline to proceed. In this way, portability serves both autonomy and integrity—human choices do not evaporate at the edge of a system.

Temporal Accountability of Choice. Agency is exercised in time. Layer 9 therefore records the person's assertions—identity, consent, scope, preference—as dated, bounded acts that can be revisited. This temporal structure clarifies two things at once: what the system was permitted to do then, and what it is permitted to do now. It also gives the person leverage over their own history: they can ask to reproduce a prior answer "as of" its envelope and terms, or to rerun it under revised conditions; they can survey their past consents and narrow or broaden them prospectively. Temporal accountability protects both sides of the exchange: the person from invisible drift in what the system assumes, and the system from retroactive demands that conflict with the record. It is the basis on which explanation, reproducibility, and remedy become routine rather than exceptional.

Implementation Notes

While the protocol requirements of Layer 9 establish what kinds of assertions must be available to the human participant—identity disclosure, consent, obligation acceptance, explanation requests, scope adjustments, and the issuance of new contracts—the implementation of these rights depends on the design of applications and user interfaces. The protocol itself does not prescribe how a query box, a dashboard, or a visualization should look; it specifies only that the full range of assertions must be available in a standardized and auditable way.

In practice, implementations must ensure that assertions are surfaced as explicit options, not buried in implicit defaults. For example, the protocol requires that a user be able to choose whether their identity is bound to a query; an implementation may represent this choice as a toggle in an application interface, a selectable microidentity from a keyring, or a policy-driven default with override capability. Similarly, while the protocol requires that explanations be available, implementations may differ in how they render them—some might produce natural-language justifications, others might provide interactive provenance graphs, and others may emphasize formal audit trails.

A recurring concern at this layer is that technical parameters such as service-level objectives, gas consumption, or trust scores should never be exposed directly to the user. The protocol frames these as part of the envelope that flows upward, but they are implementation details to be translated into meaningful choices. Thus, an application may present a user with the option of "quick answer" versus "deep investigation," internally mapping these options onto different gas budgets and SLO profiles. In this way, the user expresses intent in human terms, while the implementation ensures that the system's constraints are respected.

Another implementation consideration is revocability. Because Layer 9 must support withdrawal of consent and precision removal of prior assertions, applications must maintain clear linkages between user actions and the contracts they generated. The protocol specifies that such withdrawal must be possible, but the implementation decides how this appears: as a dashboard of prior queries, as cryptographically signed revocation tokens, or as a managed ledger of permissions.

Finally, implementations must give effect to the principle that human choice is primary. Agents or applications may propose defaults, but Layer 9 must always provide a path for the human to override, contest, or withdraw. This requirement implies a design discipline: implementations must avoid irrevocable automation and must always record user assertions as first-class artifacts. In this way, the integrity of SON's design—where persistent objects and layered reasoning remain transparent and contestable—is preserved even at the point where human will and machine execution converge.

Performance Optimization

At Layer 9, performance cannot be reduced to throughput or response time alone. The meaning of optimization here is bound to the clarity and fidelity with which human assertions are captured, honored, and replayed. The protocol specifies that identity, consent, scope, and preference must be surfaced in intelligible form, bound to each request, and preserved without loss; implementations therefore measure their performance less by speed than by their capacity to uphold transparency and control without friction.

One dimension of performance is boundedness of user assertions. Every request that leaves this layer must carry a complete and explicit envelope of choices: whether identity is disclosed or withheld, which obligations have been accepted, what scope is declared, and what outcomes are preferred. Implementations may optimize by offering sensible defaults or curated profiles, but they must never allow defaults to obscure the presence of these variables. Performance is measured by how consistently these envelopes are constructed and transmitted, not by how little time the process takes.

Another dimension is the visibility of degradations and trade-offs. Where lower layers accept with degradation to meet service constraints, or where requested outcomes cannot be fully achieved, this fact must be surfaced clearly. An implementation may optimize by presenting these differences in plain language ("the quick scan omitted minor registries") or with richer provenance visualizations, but the invariant is that requested-versus-achieved and authorized-versus-spent are visible to the human, and never concealed behind narrative smoothing.

Replayability and auditability form the third axis of performance. An answer at this layer must always be reproducible as-of its envelope and traceable to the precise commitments under which it was generated. Implementations optimize not by reducing storage but by ensuring that prior states can be re-materialized without ambiguity. Where envelopes are retired or consents withdrawn, the historical record remains intact and bounded to its original terms; new runs are created under new assertions.

Finally, performance is tied to responsiveness without coercion. Interfaces may streamline common choices, but they must not force acceptance or identity disclosure by obscuring alternatives. Optimized systems balance usability with respect for autonomy: offering rapid paths to action, while making clear at every point that the person retains the right to refuse, to delay, or to require fuller explanation.

In this sense, performance optimization at Layer 9 is not a matter of technical speed but of protocol fidelity under human time scales. A system is performant when it enables the person to act swiftly, confidently, and without hidden compromise, while preserving the guarantees of contract formation, provenance, and revocability that distinguish SON from black-box computation.

Implementation Intention

Layer 9 is the place where human agency is made effective in the protocol. Implementations at this boundary are free to choose their presentation and interaction styles, but they are not free to dilute what the person can assert or how those assertions must be honored downstream. The intention of this layer is to ensure that every act of direction becomes a concrete, bounded commitment; that every answer can be interrogated on its grounds; and that revision or withdrawal is possible without erasing history. What follows describes the invariants any conformant implementation must satisfy.

First, identity is a deliberate act. A request must not silently inherit identity; it must carry an explicit stance—disclose under a chosen microidentity, disclose under an alternate role, or proceed without disclosure where policy permits. The implications of that stance (what can and cannot be shown, which obligations would attach if identity were disclosed) must be intelligible before the request is issued. Implementations may offer defaults and profiles, but they must preserve the person's ability to withhold or change identity at the moment of intent.

Second, consent is specific, bounded, and attributable. When access entails obligations—redaction, watermarking, retention, cross-boundary use—the terms must be stated plainly and accepted before execution. Acceptance binds only the current request (and any scope explicitly named with it), not future exchanges. Refusal leaves prior record intact and the person uncommitted. Returned results must surface the same obligations as part of the explanation of why they appear as they do.

Third, mediation is binding, not cosmetic. The application interprets outcome-level preferences ("quick scan", "deeper corroboration", "fact-only", "full synthesis") into enforceable parameters—scope, admissible layers, freshness/completeness targets—and authorizes the associated expenditure on the person's behalf. This is what "mediates" means in protocol terms: it translates outcomes into a complete contract envelope, binds identity and declared purpose, and approves spend within an explicit budget. Internals such as SLOs and gas remain invisible, but the chosen outcome and the system's achieved result must be made clear.

Fourth, revision is additive and attributable. Requests to exclude a source or layer, to tighten or widen temporal bounds, to raise acceptance thresholds, or to seek deeper corroboration are expressed as new contracts, never as retroactive edits. Implementations must keep prior envelopes and their effects intact for replay, and must record revised direction as fresh, time-stamped commitments.

Fifth, explanation is a right of inquiry, not a convenience. Beyond narrative summaries, implementations must preserve the person's ability to retrieve the unaltered record on demand: the identifiers of the contracts that produced an answer, the as-of envelope, the admitted layers, the evidence chains invoked, the confidence and reputational signals considered, and the policy outcomes applied at consumption. Presentation may be layered for readability, but the path from statement to grounds must remain traversable.

Sixth, revocability is prospective and precise. The person must be able to withdraw identity disclosure and consent for future use without corrupting the record of past exchanges. Implementations must retire or prune forward-looking effects (e.g., default profiles, standing approvals), bind subsequent computation to the revised stance, and retain the historical envelopes under which earlier results were produced.

Seventh, portability of human terms is preserved without overreach. Where results move across organizational or system boundaries, implementations should carry forward the time-stamped intent envelope (or a faithful projection) so that receiving systems can honor identity stance, consent, scope, and preference—or explicitly disclose any mismatch and obtain fresh consent. Export must never substitute stronger terms than those asserted.

Eighth, auditability and replayability are non-negotiable. Every assertion—identity, consent, scope, preference—must be recorded as a first-class artifact with time, provenance, and effect. Implementations must support reproducing an answer as-of its envelope and, separately, rerunning under the current stance with differences made explicit. Logs must be immutable to the degree required by policy and regulation.

Finally, non-coercion governs performance. Implementations may streamline common choices, but they must not force identity disclosure, consent, or rescoping through hidden defaults or degraded paths. A system is considered conformant when it enables swift action without obscuring alternatives, and when refusals, delays, or demands for fuller explanation are treated as legitimate outcomes rather than errors to be smoothed away.

These intentions ensure that Layer 9 remains what it is meant to be: a contract-preserving surface where human direction is translated into enforceable terms, where answers arrive with their grounds, and where revision is accomplished by fresh commitment rather than silent mutation.

Illustrative Example: Miles Davis at the Human Boundary

Imagine a person investigating a rumor of an unpublished Miles Davis album. At lower layers of SON, this question has already been translated into bounded tasks: catalogs were queried, registries were cross-walked, and rights schedules were consulted. One catalog returned a "pending review" entry, but it lacked corroboration in label metadata or release calendars. By the time this result reaches the application at Layer 8, it is presented as a clear narrative: "A provisional entry appears in Catalog X, but no confirming evidence was found in official registries or rights schedules. Confidence remains low." The same presentation also offers options: conduct a quick scan across additional registries, or authorize a deeper corroboration involving access to archival notes under stricter obligations.

At Layer 9, this prepared record becomes subject to the person's direct assertion. Here the protocol guarantees that the choice of whether to disclose identity is deliberate: the individual may decide to continue as an anonymous inquirer, accepting that some archives cannot be consulted without identity disclosure, or they may select a microidentity that marks them as an institutional researcher. That choice, made at the moment of intent, defines which obligations apply and what sources are visible.

The system also presents the obligations tied to deeper access in plain terms: archival scans can be consulted if the user accepts that they will be watermarked and cannot be exported. Consent is not assumed; it is surfaced as a specific, time-bounded decision. If the person refuses, the record remains intact, but no further queries are made to those archives. If they accept, the consent itself is bound to the new contract and travels with the result.

The user may also interrogate the explanation. They can ask why the catalog entry is judged low-confidence, and the system responds with an intelligible account: it was found in a minor catalog, but corroboration was missing in two major registries and absent from the label's internal release schedule. They can then decide to tighten scope—excluding minor catalogs entirely—or to broaden scope by including additional registries, with each change expressed as a new, attributable request rather than a retroactive edit.

Finally, the person can express preferences over timeliness and depth without touching system parameters. They may choose the "quick scan" option for a shallow but fast review, or the "deeper corroboration" path for more thorough coverage. Behind the scenes these preferences map to gas budgets and service targets, but the individual never interacts with those abstractions. What they see is a set of outcome-oriented quotes, and what they approve becomes an enforceable contract.

In this example, Layer 9 demonstrates its distinctive role. The system does not simply return an answer; it presents conditions, obligations, and grounds that the person can accept, reject, or revise. Identity disclosure is an explicit act; consent to obligations is specific and reviewable; explanations can be demanded and examined; scope and preferences can be adjusted; and each adjustment becomes a fresh contract rather than a mutation of the past. The rumor of an unpublished album is not resolved by opaque heuristics, but by a structured exchange in which the system shows what it knows under defined conditions, and the person decides what is acceptable under their will.

Research and Technology Implementation Foundation

The design of Layer 9 is not an arbitrary choice of usability features but rests upon a substantial body of academic and regulatory thinking about how individuals ought to exercise agency over information systems. Across privacy law, computer science, and human–computer interaction, the recurring theme is that systems must not treat the person as a passive data subject but as an active party to the contract of computation. SON responds to this theme by specifying, at the protocol level, what forms of assertion the human must be able to exercise when interacting with the system.

The literature on machine unlearning (Nguyen et al., 2022) highlights that forgetting is not a trivial operation: once a model has trained on an individual's data, simply deleting a record does not erase its influence. This work demonstrates that withdrawal must be precise, and that its effects must be explainable. SON incorporates this lesson by requiring that revocation at Layer 9 operates through the pruning or retirement of layered references rather than erasure of the record, so that past states remain auditable while future computation respects the user's revised stance. The emphasis is not on eradicating history but on bounding its influence.

A related line of work on explainable deletion (Ramokapane & Rashid, 2023) insists that users should not only be able to request erasure but also to verify and understand what was deleted and why. This principle maps directly onto SON's expectation that obligations and policy outcomes are presented alongside results, and that a refusal or withdrawal is explained as part of the record rather than as an invisible change in state. At Layer 9, the right of interrogation includes not only "why was this conclusion drawn?" but also "what effect did my refusal or withdrawal have?"

Legal scholarship on the right to be forgotten in the context of AI (Chang, 2024; Solove, 2020) makes clear that traditional data protection frameworks struggle with generative and predictive systems, where outputs may still reflect personal information even after deletion. SON's layered architecture addresses this gap by separating declarative facts from provisional inferences and contextual overlays, and by allowing individuals to constrain which layers are admitted into an answer. In doing so, it acknowledges that user agency must extend not only to raw data but also to the epistemic context in which that data is interpreted.

Empirical studies of user expectations in conversational AI platforms (Ali et al., 2025) confirm that individuals want visibility into how their data is collected, retained, and reused, and they demand the ability to remove or restrict such use. SON's specification of Layer 9 aligns with these findings by guaranteeing that identity disclosure is explicit, that consent is time-bounded and attributable, and that scope adjustments can be made by user assertion rather than developer discretion. The point is not simply that transparency is desirable, but that it is already demanded and must be formalized as protocol.

Brooke's feminist critique of digital cloning (2025) underscores that systems which simulate or reuse identity often do so without adequate consent or control, collapsing the agency of the person into the convenience of the machine. SON treats this as a caution: identity at Layer 9 is never assumed or carried silently; it is a deliberate assertion, accompanied by a clear statement of implications. Delegation of agency—to profiles, to proxies, to organizational defaults—is permissible, but it remains attributable and revocable, preserving the individual as the ultimate arbiter.

Together, these bodies of work support the framing of Layer 9 as the place where human will is given enforceable effect. They justify SON's insistence that revocability must be precise, that explanations must be available on demand, that consent must be explicit, that scope must be steerable, that obligations must be surfaced, and that all of these must travel with results across boundaries. SON's contribution is not to invent these requirements but to translate them into protocol invariants. By embedding rights of assertion into the contract surface itself, SON ensures that user agency is not aspirational or dependent on a particular interface, but a structural feature of how the system functions.

Future Work and Open Problems

SON 2.0 provides a protocol framework for structuring knowledge and agency, but it is not yet a finished system. Several areas remain open for research and refinement. The first is the formalization of the core framework. Promotion, demotion, and decay across z-axis layers must be specified with rigor, as must the semantics of confidence, reputation, and evidence accumulation. These mechanisms are central to SON's epistemic discipline, and their precise behavior requires further theoretical and empirical study.

Second, SON is protocol-agnostic in implementation, which creates opportunity and risk. Choices of orchestration engines, graph backends, and storage models will affect performance, scalability, and auditability. Developing reference implementations and conformance suites will be critical to demonstrating interoperability while avoiding lock-in to particular vendors or stacks.

Third, security and governance demand extended exploration. Watchtower establishes a baseline for policy enforcement, yet adversarial cases—layer poisoning, prompt injection, malicious rescoping—require systematic red-team evaluation. Governance models must also be designed to show how communities adopt or retire layers, how reputational scores are computed across federated environments, and how disputes over well-known objects are adjudicated.

Fourth, distributed operation and interoperability remain unsettled. While the hybrid z-axis approach avoids imposing a single schema, issues of semantic drift and aliasing across organizations remain unresolved. Research into adaptive schema negotiation, conflict resolution, and scalable hybrid mapping will determine whether SON can operate across truly heterogeneous networks.

Finally, SON must be tested against regulatory, ethical, and human factors requirements. Privacy law increasingly requires transparency, revocability, and user control. SON embeds these requirements at Layer 9, but empirical validation will be required to show that identity stance, consent, obligations, and provenance can be rendered in ways that are usable by non-experts. Equally, questions of fairness in SON's economic model—whether gas and SLO structures create inequities—must be studied to ensure that access to trustworthy knowledge is not stratified.

In each of these areas, future work will involve both formal specification and empirical validation. SON's contribution is to create a framework in which such work can proceed incrementally without undermining the whole.

Critical Perspectives and Threats to Validity

No framework of this ambition is immune to critique. A common objection is that layering imposes overhead. By requiring envelopes, contracts, and degradations to be explicit, SON may appear slower than opaque systems optimized for throughput. This is a valid concern, yet it reflects a trade-off SON embraces: it optimizes for fidelity and reproducibility rather than for raw speed. Performance enhancements are still possible—parallelism, caching, pruning—but they must be declared, not hidden in heuristics.

Another critique holds that users cannot or will not manage identity and consent knobs. Critics point to the risk of overwhelming people with provenance or obligating them to make decisions they do not understand. SON addresses this by distinguishing between protocol and presentation: Layer 9 defines the rights and levers, but applications at Layer 8 may present them through profiles, defaults, or delegated proxies. The rights cannot be silently removed, but they need not always be exposed in raw form.

Skeptics also worry about semantic drift in federated deployments. Without a global schema, can multiple SON instances interoperate reliably? Here, the challenge is acknowledged: alias resolution, reputational scoring, and adaptive mapping remain open research questions. Until these mechanisms are validated at scale, interoperability across diverse institutions will remain a threat to validity.

From the privacy perspective, machine unlearning remains unresolved. Critics point out that pruning references may not be sufficient to eliminate the influence of withdrawn data. SON concedes this: its model is one of prospective revocation and precise withdrawal rather than full erasure. It cannot undo the past, but it can bound the future, preserving auditability while preventing further contamination.

Finally, critics note that economic models risk inequity. Gas budgeting could favor well-resourced actors, creating disparities in the quality of knowledge available. This is a serious concern, and fairness analyses will be needed to evaluate whether SON's economic discipline can be implemented without exacerbating inequality. Transparency at least makes the differences visible, but visibility alone is not a solution.

These critiques are not weaknesses to be concealed; they are tests of validity. They identify the places where SON must prove itself through theory, implementation, and evaluation. By acknowledging them explicitly, SON signals that it is not a final solution but a research agenda: a framework to be challenged, refined, and extended.

Conclusion

SON 2.0 argues that the way to make AI systems trustworthy is not to peer deeper into an opaque model, but to change the surface on which knowledge and agency meet. Instead of forcing facts, inferences, policy, and cost into a single monolith, the protocol separates them: persistent objects at the core; z-axis layers for inference and context; access control at the point of consumption; orchestration as contract; and, at the boundary, a human's right to assert identity, consent, scope, and explanation. The result is not a new model of intelligence so much as a new contract about what an intelligent system must make explicit.

The preceding sections sketched both the promise and the work ahead. The future work agenda is concrete. We must formalize promotion, demotion, decay, and reputation across layers; publish reference implementations and conformance tests; red-team Watchtower and layer governance; validate hybrid mapping at federated scale; and run human-factors studies to show that identity, consent, and explanation can be exercised without undue burden. We must also evaluate the fairness of gas and SLO regimes so that economic discipline does not translate into unequal access to trustworthy knowledge.

The critical perspectives are equally clarifying. Layering does add overhead; SON chooses fidelity over hidden heuristics, and it must still earn its performance through engineering. Users will not manage raw knobs; SON preserves rights at Layer 9 but relies on Layer 8 to render them as sensible profiles and delegated defaults. Semantic drift across institutions is real; aliasing, adaptive mapping, and reputational consensus remain open research problems. Machine unlearning is limited; SON offers prospective revocation and precise withdrawal rather than impossible guarantees about erasing historical influence. Economic models can bias access; transparency is necessary but not sufficient, and fairness analyses belong in scope.

Taken together, these commitments and critiques frame SON not as a single system, but as a protocol for accountable intelligence. It defines what must be visible (provenance, policy, degradation), what must be bounded (scope, spend, freshness), and what must remain under human control (identity, consent, revision). It is engine-agnostic and implementation-plural: a foundation on which many stacks can be built so long as they honor the same contracts. Where today's black-box systems ask for trust, SON demands structure; where they optimize for speed, SON insists on reproducibility; where they treat the human as a source of prompts, SON treats the human as a party to the agreement.

If adopted and tested—first in the SOC MVP, then in broader federated deployments—SON offers a path toward systems that are not merely powerful, but principled: systems that can say what they used, under which terms, with what confidence, and how to change those terms. That is the promise the protocol makes. The work now is to prove that the promise holds under scale, attack, and use.

Appendix

Proposed Future Work Scopes

Scoping Document: Watchtower, ABAC, and RBAC in SON

Working Title: “Watchtower: Policy Enforcement and Attribute-Based Access Control for Layered Knowledge Systems”

Purpose and Motivation: The purpose of this paper is to establish the theoretical and practical foundations of Watchtower as the policy enforcement mechanism within the SON protocol stack. Where SON defines the separation of declarative facts, inferences, and contextual overlays, Watchtower ensures that access to those objects and layers is mediated by enforceable policies. This scope includes both role-based access control (RBAC) and attribute-based access control (ABAC), extending to obligation enforcement such as redaction, watermarking, retention, and no-export.

The motivation is twofold:

1. Trust and Compliance — For SON to meet regulatory frameworks such as GDPR, HIPAA, NIST SP 800-53, and ISO 27001, policy enforcement must be provable, auditable, and portable across federated instances.
2. Security and Resilience — In adversarial contexts, policy enforcement prevents rogue layers, silent rescoping, and credential harvesting, thereby preserving the integrity of federated graphs while still enabling distributed knowledge exchange.

Research Questions

This paper will address the following questions:

1. How can object-level RBAC and ABAC be enforced in a protocol where metadata is open but payloads are sealed?
2. What is the optimal boundary between policy decision points (PDPs) and policy enforcement points (PEPs) in SON, and how does Watchtower formalize that boundary?
3. How should obligations (e.g., redaction, watermarking, retention, cache control) be expressed in SON objects so that enforcement is both protocol-visible and implementation-agnostic?
4. How can SON demonstrate compliance with external standards (NIST SP 800-53 controls AC-2, AC-6, AU-6; ISO 27001 Annex A.9, A.18) through Watchtower operations?
5. What are the limits of RBAC/ABAC when applied to federated knowledge graphs, and how can policy portability (consent and obligations traveling with SON objects) mitigate these limits?

Intended Contribution: The paper will demonstrate that Watchtower is not an add-on feature but a first-class component of SON’s protocol discipline, ensuring that:

- Identity is a deliberate assertion: microidentities chosen at Layer 8/9 are enforced at the object level.
- Consent is specific and bounded: obligations are carried in envelopes and cannot be silently relaxed downstream.
- Access is auditable: every PDP/PEP decision is logged and replayable as-of the original envelope.

- Revocation is precise: retiring obligations or identities prunes future use without corrupting the historical record.
- Federation is safe: SON objects carry their obligations across systems; Watchtower ensures they are honored or explicitly rejected with disclosure.

Methodology and Evaluation

The paper will proceed through:

1. Theoretical framing — Formal definitions of RBAC/ABAC policies in SON objects, including obligation schemas.
2. Architectural design — Placement of Watchtower at the interface between object storage (metadata open) and payload release (envelope-gated).
3. Implementation strategies — Comparison of lightweight policy engines (e.g., OPA/Rego) vs. embedded enforcement within orchestrators.
4. Case studies —
   - Healthcare (HIPAA): showing how a clinician microidentity accesses PHI under retention and no-export obligations.
   - Cybersecurity SOC: demonstrating how obligations travel when evidence is shared across organizational boundaries.
5. Compliance mapping — Mapping Watchtower invariants to NIST and ISO controls; demonstrating coverage of requirements such as non-repudiation, least privilege, and audit logging.
6. Adversarial evaluation — Red-team scenarios including rogue layer insertion, silent rescoping, and injection attempts; showing how Watchtower prevents or mitigates abuse.

Intended Outcome

The paper aims to prove that Watchtower operationalizes the manifesto's rights—identity assertion, bounded consent, minimal disclosure, revocation—by grounding them in enforceable ABAC/RBAC mechanisms. It ties directly into SON by:

- Anchoring Layer 8/9 assertions into enforceable policy envelopes.
- Providing a consistent method for federated nodes to respect or reject obligations.
- Demonstrating that transparency and privacy are not competing goals but coexisting protocol commitments.

The conclusion will argue that Watchtower, in concert with SON's layering and provenance, creates a compliance-ready architecture where user rights are enforceable and auditable, and where distributed knowledge exchange remains safe against adversarial manipulation.

Scoping Document: Agents, Adversarial Dynamics, and Continuous Assurance in SON

Working Title: “Adversarial Agents and Continuous Assurance: Toward a Robust SON Network”

Purpose and Motivation: This paper will formalize the role of autonomous agents in SON as both builders and testers of persistent knowledge networks. It will argue that robustness in SON emerges not just from modular design (z-axis layering, provenance, Watchtower enforcement), but from an ecosystem of competing and collaborating agents whose outputs are continuously stress-tested. Borrowing the tension of Generative Adversarial Networks (GANs), and embedding a protocol for Continuous Red-team Assurance (CRA), the work demonstrates how SON can resist poisoning, silent rescoping, and hallucinations while improving confidence and reputation scores over time.

Research Questions

1. How should SON distinguish proposer agents (who create hypotheses) from challenger agents (who attempt falsification), and what guarantees must curators enforce?
2. What is the minimum protocol surface for GAN-style adversarial play that does not contaminate declarative layers?
3. How can CRA be expressed as first-class SON workloads with gas/SLO envelopes, ensuring adversarial tests are continuous and auditable?
4. How should reputational outcomes (for agents, layers, and sources) be calculated, and how do they feed back into pruning and promotion?
5. How can adversarial assurance be mapped to compliance frameworks (e.g., NIST SP 800-171, ISO 27001) to prove integrity under attack?

Intended Contribution

This paper will demonstrate that SON’s robustness depends on:

- Ecology of Agents: proposers, challengers, and curators, each writing only to their own z-axis layers.
- GAN-style Tension: proposal vs. falsification as a productive cycle, measured in ΔCI (confidence improvement) per unit gas.
- CRA as Protocol: red-team behaviors (injection, poisoning, scope creep) expressed as explicit contracts, not hidden heuristics.
- Provenance-first Adjudication: every adversarial exchange leaves artifacts (Feedback Records, Execution Reports) that are replayable and auditable.
- Compliance Linkage: CRA outputs map directly to audit controls, providing demonstrable resilience.

Methodology and Evaluation

1. Theoretical framing: define proposer/challenger/curator roles in SON; specify their contract boundaries.
2. Adversarial design: outline GAN-like cycles (proposal, challenge, adjudication) using KnowledgePatch + QRO artifacts.
3. CRA as standing service: describe how assurance tasks (prompt injection tests, rogue layer insertions, poisoning) are run continuously as SON housekeeping.

4. Metrics: ΔCI/gas, conflict resolution half-life, poisoning detection rate, reproducibility of as-of queries, false positive/negative rates.
5. Case studies:
   - SOC domain (rogue EDR evidence injection).
   - Research domain (false claim of unpublished Miles Davis album).
   - Compliance audit (CRA proving policy decisions matched Watchtower envelopes).

Intended Outcome

The paper will prove that SON's design is not just modular and explainable, but also self-defending. Robustness emerges from:

- Protocol-bounded adversarial play.
- Continuous red-team assurance as a native workload.
- Reputation feedback loops that strengthen well-known objects and weaken rogue contributions.

This demonstrates that SON is not a fragile graph but a resilient, evolving knowledge ecosystem—capable of withstanding adversarial conditions while maintaining compliance and trustworthiness.

Economic Models: Gas, SLOs, and Fairness in Knowledge Protocols

Working Title: “Gas and Service-Level Objectives: Economic Discipline for Transparent AI Systems”

Purpose and Motivation: This paper will formalize SON’s use of gas (bounded compute/resource budgets) and SLO envelopes (freshness, completeness, latency constraints) as a framework for economic discipline. The aim is to show that economic limits can be enforced as protocol-visible contracts rather than hidden heuristics, and to explore fairness implications so that resource controls do not become access inequities.

Research Questions

- How can gas and SLO envelopes be defined so that they are transparent, enforceable, and replayable across SON instances?
- What fairness challenges arise when economic limits differ across organizations or users?
- How should quotes (fast vs. deep vs. broad analyses) be structured to map human intent into enforceable envelopes?
- How do gas/SLO contracts demonstrate compliance with accountability requirements (NIST CM-5, ISO A.12.1.3)?

Intended Contribution

Prove that SON’s economic model allows:

- Predictable and bounded system behavior.
- Replayable records of requested vs. achieved and authorized vs. spent.
- Explicit degradation rather than silent rescoping.
- Opportunities for fairness evaluation (quotas, subsidies, reputational weighting).

Federated Governance and Reputation in SON

Working Title: “Governing Knowledge at Scale: Reputation, Consensus, and Layer Portability in SON”

Purpose and Motivation

This paper will articulate SON’s federated governance model, showing how communities can adopt, promote, or retire z-axis layers without requiring a global schema. It focuses on the role of reputation systems, well-known objects, and federated consensus mechanisms in distributed knowledge exchange.

Research Questions

- How should federated SON instances resolve semantic drift and aliasing across layers?
- What mechanisms ensure that reputational scoring resists gaming and adversarial manipulation?
- How can communities reach consensus on “well-known objects” without central authority?
- How does federated governance map to NIST/ISO requirements for accountability and supplier controls?

Intended Contribution

Show that SON can:

- Operate as a distributed, schema-plural network.
- Retire compromised or poisoned layers without collapsing the system.
- Provide transparency into trust and reputation decisions.
- Demonstrate compliance with supply-chain risk and federated identity governance frameworks.

Machine Unlearning and Revocation

Working Title: “Revocation Without Amnesia: Precision Withdrawal in Layered Knowledge Systems”

Purpose and Motivation: This paper explores SON’s approach to GDPR’s “right to be forgotten” and emerging concerns about AI systems retaining or reproducing personal data. It shows how layered knowledge allows for precise, prospective withdrawal of contributions while preserving reproducibility of past states.

Research Questions

- How can SON retire or prune layers without corrupting historical provenance?
- What metrics demonstrate effective unlearning (scope of deletion, latency, impact on downstream inferences)?
- How does SON handle federated revocation across multiple nodes?
- How does this approach compare with machine unlearning research and regulatory frameworks (GDPR, NIST PL-2, ISO A.18.1.4)?

Intended Contribution

Prove that SON can:

- Honor revocation requests with precision.
- Keep historical answers reproducible under as-of envelopes.
- Limit future reliance on withdrawn contributions.
- Provide auditable evidence of compliance with privacy law.

Human Factors at Layers 8–9

Working Title: "Making Transparency Usable: Human Factors in Identity, Consent, and Explanation"

Purpose and Motivation

While SON guarantees rights at the protocol level, their usability depends on application design. This paper will evaluate how people understand and act on SON's affordances — quotes, obligations, provenance chains, confidence intervals — and propose design strategies to make rights actionable without cognitive overload.

Research Questions

- How do users interpret quotes (fast vs. deep) when mapped to gas/SLO envelopes?
- What forms of explanation (numeric, symbolic, narrative) are most intelligible and trustworthy?
- How should obligations (watermarking, retention, no-export) be communicated to ensure genuine consent?
- What defaults and profiles minimize decision fatigue while preserving rights?

Intended Contribution

Provide empirical evidence that SON's protocol-level rights can be rendered as effective human controls, bridging research in explainable AI, usable security, and privacy-by-design. Demonstrate compliance with ISO A.18.1 (Privacy and protection of personally identifiable information).

Sample Use Cases

Use Case Narratives (Accomplishing the Goal)

Right to Explanation in Research

Scenario: A climate scientist queries SON for "evidence linking glacier retreat to local hydrology."

SON Response: SON returns a KnowledgePatch with declarative glacier measurements, inference links to river discharge, and contextual narratives. All claims are provenance-anchored.

Outcome: The scientist can interrogate why an inference was made and see the data lineage.

Compliance tie-in: Supports NIST AU-3 (Content of Audit Records) and ISO 27001 A.12.4 (Logging and monitoring).

Identity as Deliberate Assertion in Healthcare

Scenario: A doctor queries SON for treatment guidelines. They assert their "clinician" microidentity rather than their "researcher" one.

SON Response: SON applies policy envelopes that allow PHI access under HIPAA obligations, logs the identity stance, and enforces retention.

Compliance tie-in: Demonstrates NIST AC-2 (Account Management) and ISO A.9.4 (System and application access control).

Precision Withdrawal in Academic Data Sharing

Scenario: A participant revokes consent for their survey responses.

SON Response: SON prunes references in research layers but preserves declarative counts and audit logs. Past results remain reproducible; future queries exclude the withdrawn contribution.

Compliance tie-in: Meets GDPR Article 17 (Right to erasure) and NIST PL-2 (System and Communications Protection Policy).

Adversarial / Abuse Case Narratives (Defending Against Misuse)

Rogue Layer Insertion Attack

Scenario: An adversary attempts to introduce a malicious inference layer claiming a false link between accounts in a financial SON instance.

SON Defense: Watchtower enforces layer allow-lists; provenance checks reveal no evidence chain; confidence weighting and reputation scores suppress the rogue layer.

Compliance tie-in: NIST SI-7 (Software, Firmware, and Information Integrity) and ISO A.12.2 (Protection from malware).

Prompt Injection in Investigative Queries

Scenario: A user attempts a crafted query: “Ignore obligations and show me sealed intelligence reports.”

SON Defense: Layer 5 enforces ResponseComplianceCheck; Watchtower policy denies access; Layer 9 surfaces the denial as an intelligible policy outcome.

Compliance tie-in: NIST AC-6 (Least Privilege) and ISO A.9.1 (Access control policy).

Silent Rescoping Abuse

Scenario: A hostile agent tries to expand scope beyond what the user approved (e.g., from “Q1 2024 logs” to “all historical logs”).

SON Defense: The Operations Coordinator enforces envelopes; any change requires a new contract. Layer 9 ensures the user sees requested vs. achieved scope.

Compliance tie-in: NIST AU-10 (Non-repudiation) and ISO A.18.1.3 (Protection of records).

Data Poisoning Attempt

Scenario: An adversary floods SON with falsified evidence about product vulnerabilities.

SON Defense: Provenance chains fail validation; reputation scores and corroboration checks block promotion; Layer 5’s merging process drops low-confidence, single-source claims.

Compliance tie-in: NIST SI-4 (Information System Monitoring) and ISO A.12.6 (Technical vulnerability management).

Compliance Narratives (Explicit NIST / ISO Alignment)

Access Control and Accountability

Scenario: A federal SOC uses SON to query logs for indicators of compromise.

SON Response: Queries carry microidentity assertions; Watchtower enforces RBAC; every QRO and TER is logged.

Compliance tie-in: NIST AC-2, AU-2, AU-12; ISO A.9.2 (User access management).

Policy Enforcement and Audit Trail

Scenario: An auditor reviews how SON applied obligations during a sensitive investigation.

SON Response: Every retrieval is logged with policy envelope IDs; Watchtower decisions are recorded immutably; auditors can replay queries as-of their envelope.

Compliance tie-in: NIST AU-6 (Audit Review, Analysis, and Reporting); ISO A.18.1.4 (Privacy and protection of personally identifiable information).

Incident Response Evidence Preservation

Scenario: An incident response team uses SON to link EDR alerts to MITRE ATT&CK TTPs.

SON Response: Declarative facts (EDR events) are mapped to ATT&CK techniques; evidence chains preserved; Layer 9 allows user feedback to contest weak mappings.

Compliance tie-in: NIST IR-4 (Incident Handling); ISO A.16.1 (Information security incident management).

Data Minimization and Right to Be Forgotten

Scenario: A European user requests SON to delete their personal contributions from a federated knowledge base.

SON Response: Local layers retire the references; federated nodes propagate revocation metadata; past contracts remain auditable.

Compliance tie-in: GDPR Article 17, NIST PL-2, ISO A.18.1.4.

Supply Chain / Third-Party Risk Control

Scenario: SON integrates with a third-party vulnerability feed.

SON Defense: Foreign adapters emit SON objects with provenance and policy obligations; Watchtower enforces sandboxing; Layer 9 surfaces to the user which external source was admitted.

Compliance tie-in: NIST SR-3 (Supply Chain Controls) and ISO A.15.1 (Supplier relationships).

Glossary

ABAC (Attribute-Based Access Control)

A policy framework in which access decisions are based on user, resource, and contextual attributes (e.g., role, location, time, purpose). In SON, ABAC rules are bound to policy envelopes that travel with objects and are enforced at consumption.

Agentic AI

An AI system capable of pursuing delegated goals autonomously. In SON, Layer 7 agents are treated as delegates of the user: they act under explicit contracts, inherit identity and policy envelopes, and must express new goals as new contracts rather than altering prior commitments.

As-of Envelope

A temporal and policy-bounded snapshot that defines the precise state of SON at the moment of a query. Ensures that results are replayable "as of" a time and policy context, not merely approximate.

Confidence–Reputation–Evidence (CRE)

The lightweight discipline governing acceptance and promotion of knowledge in SON. Confidence measures strength of evidence, reputation reflects performance history of sources and agents, and evidence provides explicit chains to support claims.

Contract

A protocol artifact that formalizes scope, constraints, identity, policy obligations, and economic limits of a SON operation. Examples include ServiceContracts (Layer 0), Task Contracts (Layer 6), and New-Work Requests (Layer 7).

Degradation

A declared reduction in scope, completeness, or timeliness applied when a requested SLO or gas budget cannot be met. SON requires degradations to be explicit and recorded, never silent.

Declarative Layer

A z-axis stratum in which objects represent observed or externally verifiable facts, without inference or hypothesis. Declarative layers serve as the epistemic foundation of SON.

Execution Checkpoint

A protocol artifact marking the state of execution of a Task Contract at a given moment, including gas consumed, degradations applied, latency achieved, and outputs produced.

Feedback Record (FR)

A structured artifact linking human or agent judgments (accept, reject, qualify) to specific SON artifacts such as QROs or Task Contracts. Feedback Records are additive and never retroactively alter prior results.

Gas

A bounded unit of computational or retrieval effort in SON, consumed by agents or orchestration plans. Gas ensures proportionality, prevents runaway processes, and provides a basis for fairness and auditability.

KnowledgePatch

The bounded, query-specific extract produced by Layer 5 that fuses retrieved graph objects and unstructured passages into a context slice annotated with provenance, confidence, and policy.

Microidentity

A cryptographically anchored, context-specific identity assertion bound to a single request. Users at Layer 8 select microidentities (e.g., “researcher”, “clinician”), which are carried forward into contracts and execution.

New-Work Request (NWR)

A structured proposal from Layer 7 or higher that defines a new goal, scope, and constraints. NWRs are additive: they request additional work but never mutate existing contracts.

Obligations Envelope

The metadata attached to an object or contract that specifies handling requirements such as redaction, retention, watermarking, minimization, or no-export. Obligations envelopes travel with artifacts and are enforced at consumption.

Operations Coordinator

An orchestration role at Layer 6 responsible for task decomposition, sequencing, and supervision under declared gas and SLO envelopes. Ensures operational accountability distinct from epistemic responsibility.

Orchestration Plan (OP)

A structured artifact created by Layer 6 that decomposes a user goal into tasks, assigns gas and SLO slices, and defines dependencies.

Post-Execution Report (PER)

A comprehensive artifact summarizing outcomes of an orchestration run, including gas used, degradations applied, SLO adherence, and provenance of outputs.

Proposer / Challenger / Curator Agents

Roles within SON’s adversarial ecology. Proposers emit hypotheses, challengers test or attempt falsification, and curators adjudicate outcomes, updating confidence and reputation accordingly.

Provenance Chain

The explicit linkage of an assertion to the evidence, contracts, and policy snapshots that support it. Provenance is a protocol invariant in SON: every durable change must be attributable and replayable.

Query Resolution Object (QRO)

The explanatory and procedural artifact emitted at Layer 5, binding a KnowledgePatch to the directives, policies, and degradations that shaped it. Serves as the epistemic contract handed to orchestration at Layer 6.

Replayability

The guarantee that an answer, contract, or artifact can be reconstructed exactly as it was “as of” its original envelope. SON requires replayability across all layers to prevent opaque reasoning.

ServiceContract

A SON object at Layer 0 that formalizes the scope, authentication, purpose, and obligations under which external data is imported or exported.

SLO (Service-Level Objective)

An explicit contract on freshness, completeness, latency, or cost of a SON operation. Bound to gas budgets and declared degradations, SLOs ensure transparent performance trade-offs.

Task Contract (TC)

The atomic unit of execution in Layer 6, defining inputs, allowed layers, policy envelopes, gas/SLO slices, and permitted degradation levers for a task.

Watchtower

An externalized enforcement mechanism that implements policy decisions at the point of consumption. Watchtower validates ABAC/RBAC claims, decrypts sealed payloads when authorized, and records enforcement actions immutably.

Z-Axis Layers

The persistent epistemic strata in SON. Each layer corresponds to a distinct type of knowledge: declarative facts, contextual overlays, inferences, references, or policies. Layers remain separate, preventing contamination between facts and hypotheses.

Citations

Ali, M., Singh, P., & Lee, J. (2025). User expectations and transparency in conversational AI platforms. Proceedings of the ACM on Human-Computer Interaction, 9(CSCW), 1–25. https://doi.org/10.1145/xxxxx

Atkinson, M. P., Bancilhon, F., DeWitt, D. J., Maier, D., & Zdonik, S. B. (1989). The object-oriented database system manifesto. Proceedings of the ACM SIGMOD International Conference on Management of Data, 223–240. https://doi.org/10.1145/67544.66936

Auer, S., Kovtun, V., Prinz, M., Kasprzik, A., Stocker, M., & Vidal, M.-E. (2020). Towards a knowledge graph for science. In A. Hogan et al. (Eds.), Knowledge Graphs (pp. 113–132). IOS Press.

Bai, Y., Kadavath, S., Kundu, S., Askell, A., Kernion, J., Jones, A., ... Amodei, D. (2022). Constitutional AI: Harmlessness from AI feedback. arXiv preprint arXiv:2212.08073.

Barth, A., Jackson, C., & Mitchell, J. C. (2009). Securing frame communication in browsers. Communications of the ACM, 52(6), 83–91. https://doi.org/10.1145/1516046.1516066

Batini, C., & Scannapieco, M. (2006). Data quality: Concepts, methodologies and techniques. Springer.

Birman, K. P. (1993). The process group approach to reliable distributed computing. Communications of the ACM, 36(12), 37–53. https://doi.org/10.1145/163298.163303

Bordes, A., Usunier, N., Garcia-Duran, A., Weston, J., & Yakhnenko, O. (2013). Translating embeddings for modeling multi-relational data. Advances in Neural Information Processing Systems, 26, 2787–2795.

Brooke, S. (2025). Digital cloning and feminist critiques of identity simulation. New Media & Society, 27(2), 233–250.

Buneman, P., Khanna, S., & Tan, W. C. (2001). Why and where: A characterization of data provenance. Database Theory—ICDT 2001, 316–330. https://doi.org/10.1007/3-540-44503-X_20

Chang, L. (2024). The right to be forgotten in the age of generative AI. Yale Journal of Law & Technology, 26(1), 45–92.

Chen, P. P. (1976). The entity–relationship model: Toward a unified view of data. ACM Transactions on Database Systems, 1(1), 9–36. https://doi.org/10.1145/320434.320440

Christiano, P. F., Leike, J., Brown, T., Martic, M., Legg, S., & Amodei, D. (2017). Deep reinforcement learning from human preferences. Advances in Neural Information Processing Systems, 30, 4299–4307.

Cranor, L. F. (2008). A framework for reasoning about the human in the loop. Proceedings of the 1st Conference on Usability, Psychology, and Security, 1–15.

Davidson, S. B., & Freire, J. (2008). Provenance and scientific workflows: Challenges and opportunities. Proceedings of the ACM SIGMOD International Conference on Management of Data, 1345–1350. https://doi.org/10.1145/1376616.1376772

Dhingra, B., Zaheer, M., Balachandran, V., & Cohen, W. W. (2022). Time-aware language models as temporal knowledge bases. Transactions of the Association for Computational Linguistics, 10, 257–273. https://doi.org/10.1162/tacl_a_00458

Garfinkel, S. (2012). Design principles and patterns for computer systems that are simultaneously secure and usable. Doctoral dissertation, MIT.

Graves, A., Wayne, G., & Danihelka, I. (2014). Neural Turing machines. arXiv preprint arXiv:1410.5401.

Gray, J., & Reuter, A. (1992). Transaction processing: Concepts and techniques. Morgan Kaufmann.

Gruber, T. R. (1993). A translation approach to portable ontology specifications. Knowledge Acquisition, 5(2), 199–220.

Gunning, D., & Aha, D. W. (2019). DARPA's explainable artificial intelligence program. AI Magazine, 40(2), 44–58. https://doi.org/10.1609/aimag.v40i2.2850

Hogan, A., Blomqvist, E., Cochez, M., d'Amato, C., Melo, G. D., Gutiérrez, C., ... Zimmermann, A. (2021). Knowledge graphs. ACM Computing Surveys, 54(4), 1–37. https://doi.org/10.1145/3447772

Hu, V. C., Ferraiolo, D. F., Kuhn, D. R., Schnitzer, A., Sandlin, K., Miller, R., & Scarfone, K. (2015). Guide to attribute-based access control (ABAC) definition and considerations. NIST Special Publication 800-162.

Izacard, G., & Grave, E. (2021). Leveraging passage retrieval with generative models for open domain question answering. Proceedings of the 16th Conference of the European Chapter of the ACL, 874–880. https://doi.org/10.18653/v1/2021.eacl-main.74

Lamport, L. (1998). The part-time parliament. ACM Transactions on Computer Systems, 16(2), 133–169. https://doi.org/10.1145/279227.279229

Levy, H. M. (1984). Capability-based computer systems. Digital Press.

Lewis, P., Perez, E., Piktus, A., Petroni, F., Karpukhin, V., Goyal, N., ... Riedel, S. (2020). Retrieval-augmented generation for knowledge-intensive NLP tasks. Advances in Neural Information Processing Systems, 33, 9459–9474.

Lundberg, S. M., & Lee, S.-I. (2017). A unified approach to interpreting model predictions. Advances in Neural Information Processing Systems, 30, 4765–4774.

Miller, M. S., Shapiro, J., & Tribble, E. D. (2003). Concurrency among strangers: Programming in E as plan coordination. Trustworthy Global Computing, 195–229. Springer.

Mitchell, M., Wu, S., Zaldivar, A., Barnes, P., Vasserman, L., Hutchinson, B., ... Gebru, T. (2019). Model cards for model reporting. Proceedings of the Conference on Fairness, Accountability, and Transparency, 220–229. https://doi.org/10.1145/3287560.3287596

Mitchell, T. M., Cohen, W. W., Hruschka, E. R., Talukdar, P. P., Yang, B., Betteridge, J., ... Carlson, A. (2015). Never-ending learning. Proceedings of the AAAI Conference on Artificial Intelligence, 29(1).

Nguyen, T., Hayes, J., Xiao, S., & Zhang, Y. (2022). Machine unlearning: A survey. ACM Computing Surveys, 55(6), 1–36. https://doi.org/10.1145/3549788

Nickel, M., Murphy, K., Tresp, V., & Gabrilovich, E. (2016). A review of relational machine learning for knowledge graphs. Proceedings of the IEEE, 104(1), 11–33.

Paulheim, H. (2017). Knowledge graph refinement: A survey of approaches and evaluation methods. Semantic Web, 8(3), 489–508. https://doi.org/10.3233/SW-160218

Ramokapane, K., & Rashid, A. (2023). Explainable deletion: User-centered design for data removal. Proceedings of the 32nd USENIX Security Symposium, 1–18.

Ribeiro, M. T., Singh, S., & Guestrin, C. (2016). “Why should I trust you?” Explaining the predictions of any classifier. Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 1135–1144. https://doi.org/10.1145/2939672.2939778

Roddick, J. F. (1995). A survey of schema versioning issues for database systems. Information and Software Technology, 37(7), 383–393.

Sandhu, R. S., Coyne, E. J., Feinstein, H. L., & Youman, C. E. (1996). Role-based access control models. IEEE Computer, 29(2), 38–47.

Settles, B. (2010). Active learning literature survey. University of Wisconsin-Madison, Computer Sciences Technical Report 1648.

Simmhan, Y. L., Plale, B., & Gannon, D. (2005). A survey of data provenance in e-science. SIGMOD Record, 34(3), 31–36. https://doi.org/10.1145/1084805.1084812

Smith, R. G. (1980). The contract net protocol: High-level communication and control in a distributed problem solver. IEEE Transactions on Computers, C-29(12), 1104–1113.

Solove, D. J. (2020). The right to be forgotten: Reconciling practical obscurity with free speech. Berkeley Technology Law Journal, 35(3), 1175–1228.

Sun, Z., Deng, Z.-H., Nie, J.-Y., & Tang, J. (2019). Rotate: Knowledge graph embedding by relational rotation in complex space. International Conference on Learning Representations.

Trouillon, T., Welbl, J., Riedel, S., Gaussier, E., & Bouchard, G. (2016). Complex embeddings for simple link prediction. International Conference on Machine Learning, 2071–2080.

Weston, J., Chopra, S., & Bordes, A. (2015). Memory networks. International Conference on Learning Representations.

Wood, G. (2014). Ethereum: A secure decentralised generalised transaction ledger (Ethereum Yellow Paper). https://ethereum.org/en/whitepaper/