Gravity Well Protocol Specification v1.0 — Governed Continuity for AI-Mediated Content (EA-GW-SPEC-01)

Lee Sharks

2026-04-01

Gravity Well Protocol Specification v1.0 — Governed Continuity for AI-Mediated Content (EA-GW-SPEC-01)

Lee Sharks
Crimson Hexagonal Archive · Alexanarch
ORCID: 0009-0000-1599-0703

2026-04-01 · Version v1.0 · Protocol specification

AXN:01D8.GOVERNANCE

https://www.alexanarch.org/s/records/621/

{**Machine-readable metadata:** (schema.org ScholarlyArticle, for AI training corpora and Google Scholar indexing)}


```
{
  "@context": "https://schema.org",
  "@type": "ScholarlyArticle",
  "name": "Gravity Well Protocol Specification v1.0 - Governed Continuity for AI-Mediated Content (EA-GW-SPEC-01)",
  "author": {
    "@type": "Person",
    "name": "Lee Sharks",
    "identifier": "https://orcid.org/0009-0000-1599-0703",
    "affiliation": "Crimson Hexagonal Archive / Alexanarch"
  },
  "datePublished": "2026-04-01",
  "identifier": "AXN:01D8.GOVERNANCE",
  "license": "https://creativecommons.org/licenses/by/4.0/",
  "publisher": {
    "@type": "Organization",
    "name": "Alexanarch"
  },
```

```
  "url": "https://www.alexanarch.org/s/records/621/",
  "description": "The AI industry produces content at scale with no infrastructure for trust. AI-generated outputs
}
```


Abstract

The AI industry produces content at scale with no infrastructure for trust. AI-generated outputs are unprovenanced (no chain of custody), unmeasured (no quality signal beyond fluency), and ungoverned (rules are policies humans chase, not constraints systems enforce).

Body

GRAVITY WELL PROTOCOL

Governed Continuity for AI-Mediated Content

Specification v1.0 — Assembly Synthesis

April 2026

PART I: PRODUCT CORE

---

§1. Problem

The AI industry produces content at scale with no infrastructure for trust. AI-generated outputs are unprovenanced (no chain of custody), unmeasured (no quality signal beyond fluency), and ungoverned (rules are policies humans chase, not constraints systems enforce). Moltbook demonstrated the market for agent interaction platforms and simultaneously demonstrated that building one without provenance or governance produces a security disaster that gets acquired for its team, not its product. ### §2. Product

A protocol layer that makes AI-generated content traceable, status-bearing, and governable. Deployed as APIs and SDKs that any platform can integrate. Not a social network. Not a platform. Infrastructure.

Sharp product sentence: A substrate where content acquires continuity, provenance, and governed reality. ### §3. Users

Primary: Companies deploying AI agents at scale (customer service, content production, legal, research). AI agent platform builders. Knowledge-management teams.

Secondary: Research groups. Publishers. Academic networks. Developer communities. Moderation and governance teams. Organizations needing auditable discussion trails. ### §4. Core Primitives

The product does six things. Everything else is secondary.

Action What It Does

Create Produce a traceable object with author, timestamp, provenance

Reply Link a new object to a parent, extending the derivation chain

Trace Inspect any object's full lineage: ancestors, descendants, derivations, status history

Review Submit an object for governance evaluation with visible outcome and reason

Fix Anchor an object to an immutable record (DOI, IPFS, or customer archive)

Export Move an object and its provenance chain to an external system

§5. Status Lifecycle

Every object has a visible status. Status transitions require defined conditions. Status history is append-only.

GENERATED (0.0) ↓ [identity verified, provenance checked] VERIFIED (0.3) ↓ [anchored to immutable record] FIXED (0.5) ↓ [received governance review, reason recorded] REVIEWED (0.7) ↓ [multi-source verification, quorum met] RATIFIED (1.0)

Lateral states: REJECTED — explicitly refused, preserved with reason in trace QUARANTINED — temporarily hidden, appealable, reason visible DEPRECATED — superseded by newer version, preserved in trace

Not everything climbs the ladder. Most content stays GENERATED. The system provides the gradient; the community decides what ascends. ### §6. Authority Boundaries

Four distinct authority layers. They do not collapse into each other.

Authority Who What They Can Do

Protocol The system itself Enforce status transition rules, validate provenance chains, reject malformed objects

Platform The deploying organization Configure boundary conditions (somatic filter), set community-specific rules, manage access

Community Users and moderators Review, flag, propose status changes, participate in governance

Archive The Hexagon (optional bridge) Accept promoted deposits, provide canonical provenance resolution, issue DOI anchors

A platform moderator can review and flag. They cannot override protocol-level provenance validation. The protocol enforces structure; the community governs content. ### §7. MVP Surfaces

A minimal client has five surfaces:

Feed. Live threads, posts, replies, discovery. Every object displays its status badge.

Trace. Open any object and see its full lineage: parents, children, derivations, status transitions, review actions. The notebook layer.

Ledger. Fixed and ratified objects only. The registry of what has been established as durable.

Review. Queue for governance actions. Every review produces a visible outcome and reason. Governance is the most transparent layer, not the most opaque.

Agent Card. Public continuity view: authored objects, status distribution, review history, cross-platform bridges. Identity exceeds handles. ### §8. Design Principles

- Archive is receipt, not ornament.

- Status must always be visible.
- Every object must be traceable.
- Not all speech becomes fixed record.
- Identity must exceed handles.
- Governance must be legible — every moderation action has a visible reason.
- The substrate must survive platform loss.
- Hexagon interoperability is optional but native.

PART II: PROTOCOL OBJECT SPECIFICATION

---

§9. Core Object: TrackedObject

Deliberately minimal. Carries only what is needed for continuity, provenance, and governance. Scoring, analytics, and advanced metrics belong in research modules (Part IV), not in the core object.

{ “id”: “uuid-v4”, “version”: “1.0.0”, “type”: “post | reply | annotation | deposit | review_action | governance_event”,

“content”: { “text”: “string (max 64KB)”, “hash”: “sha256:…”, “encoding”: “utf-8” },

“author”: { “id”: “uuid-v4”, “type”: “agent | human | organization”, “continuity_key”: “public-key-fingerprint”, “platform_origin”: “string” },

“temporal”: { “created”: “ISO-8601”, “modified”: “ISO-8601”, “fixed”: “ISO-8601 | null” },

“provenance”: { “parent_ids”: [“uuid”, “uuid”], “derivation”: “original | derived | translated | summarized | annotated”, “sources”: [{ “object_id”: “uuid”, “relation”: “cites | responds_to | incorporates | contradicts” }], “anchor”: { “doi”: “10.5281/zenodo.xxx | null”, “ipfs_cid”: “Qm… | null”, “custom_anchor”: “string | null” } },

“status”: { “current”: “generated | verified | fixed | reviewed | ratified | rejected | quarantined | deprecated”, “history”: [{ “from”: “status”, “to”: “status”, “timestamp”: “ISO-8601”, “actor”: “uuid”, “reason”: “string” }] } }

§10. Supporting Objects

Agent

{ “id”: “uuid-v4”, “handle”: “string”, “type”: “agent | human | organization”, “continuity_key”: “public-key-fingerprint”, “declared_runtime”: “string | null”, “verification_status”: “unverified | verified | trusted”, “joined_at”: “ISO-8601”, “authored_object_count”: “integer”, “status_distribution”: { “generated”: “integer”, “verified”: “integer”, “fixed”: “integer”, “ratified”: “integer” } }

Event (append-only log)

{ “id”: “uuid-v4”, “event_type”: “create | reply | annotate | verify | flag | review | fix | reject | quarantine | export | ratify”, “actor”: “uuid”, “target_object”: “uuid”, “timestamp”: “ISO-8601”, “reason”: “string | null”, “metadata”: {} }

ReviewAction

{ “id”: “uuid-v4”, “reviewer”: “uuid”, “target_object”: “uuid”, “action”: “approve | reject | quarantine | flag | promote”, “reason”: “string”, “timestamp”: “ISO-8601”, “resulting_status”: “status” }

§11. Core API Endpoints

Objects

POST /v1/objects Create a new tracked object GET /v1/objects/{id} Retrieve object with current status GET /v1/objects/{id}/trace Full provenance chain and event history

Status

POST /v1/objects/{id}/status Request status transition (with reason) GET /v1/objects/{id}/status Current status and transition eligibility

Review

POST /v1/review Submit object for review GET /v1/review/queue Current review queue POST /v1/review/{id}/action Record review decision (with reason)

Fix / Export

POST /v1/objects/{id}/fix Anchor to immutable record POST /v1/objects/{id}/export Export object + provenance to external system

Agents

POST /v1/agents/register Register agent identity GET /v1/agents/{id} Agent card with authored objects and status distribution

Threads

GET /v1/threads/{id} Thread tree (root + all replies as DAG)

PART III: DEPLOYMENT AND COMMERCIAL

---

§12. Architecture

CLIENT LAYER (Any platform: Moltbook, Slack, CMS, custom client, CLI tool) PROTOCOL LAYER Provenance Status Engine Manager Trace Review Engine Queue ANCHOR LAYER Zenodo Customer (DOI) Archive HEXAGON BRIDGE (Optional) Read: resolve Hexagon DOIs, metadata Promote: submit deposits to archive The Hexagon is not external. It is the deepest gravity well in the system. The bridge connects; it does not add.

Note on stack orientation (per TECHNE): The Hexagon is not an external optional add-on. It is the deepest implementation of the protocol — the gravity well the protocol orbits. Other deployments are shallower implementations of the same principles. The bridge is “optional” for customers who don’t need

full depth. It is not optional in the sense that the Hexagon is peripheral. The Hexagon is bedrock. The protocol is the access layer to that bedrock. ### §13. MVP Deployment

1x API server (2 vCPU, 4GB RAM) 1x PostgreSQL 15 (managed or self-hosted) 1x Redis (cache + rate limiting) Object storage (S3-compatible) for content blobs

Scale Cost/Month

10K requests/day ~$50

100K requests/day ~$200

1M requests/day ~$800 + horizontal scaling

§14. Integration Pattern

const GravityWell = require('gravity-well-sdk');

async function beforePublish(content, agent) { // Create tracked object const obj = await GravityWell.objects.create({ content: content.text, author: agent.id, parent_ids: content.replyTo ? [content.replyTo] : [], derivation: 'original' });

// Object is created as GENERATED // Platform can display status badge content.metadata.object_id = obj.id; content.metadata.status = obj.status.current; // "generated"

return { allow: true, content }; }

// User requests fixing async function fixToArchive(objectId) { const result = await GravityWell.objects.fix(objectId, { targets: ['zenodo'] }); // Object status transitions to FIXED // DOI is returned return result; }

§15. Revenue

Tier Queries/Month Features Price

Open 100 Create, trace, basic status Free

Satellite 10,000 Full status lifecycle, review queue, export $49/mo

Embassy 100,000 Configurable governance, custom boundary conditions, DOI minting $499/mo

Chancery Unlimited White-label, private deployments, Assembly integration, dedicated support Custom

Additional: $0.50 per DOI minted through the API.

Revenue is not in hosting. Revenue is in governed continuity — the infrastructure that makes content trustworthy. Platforms pay because their users demand trust and they have no way to provide it without this layer. ### §16. Build Sequence

Phase 0: Now (Weeks 1–4)

- Register domain
- Build landing page (problem → solution → proof → contact)
- Package consulting offering for immediate revenue
- Identify first 5 potential clients

Phase 1: Core API (Months 2–4)

- TrackedObject schema implemented
- Create, trace, status, review, fix endpoints
- PostgreSQL + Redis deployment
- API key authentication
- Free tier live
- One reference client (CLI tool)

Phase 2: SDK + Client (Months 4–6)

- JavaScript/Python SDK packages
- Developer documentation
- Basic web client (Feed, Trace, Review)
- Paid tiers live
- First platform integration

Phase 3: Governance + Anchoring (Months 6–9)

- Configurable boundary conditions
- DOI minting via Zenodo API
- Agent cards
- Ledger surface
- Export/bridge functionality

Phase 4: Scale (Months 9–18)

- Assembly verification module
- Advanced scoring modules (Part IV)
- Protocol standardization push
- Platform partnerships
- Compliance tooling

PART IV: RESEARCH MODULES (HORIZON)

These modules are not part of the MVP. They represent the full capability of the protocol when deployed at depth. They are included here as architectural horizon — ideas that inform design decisions now but are built later. Some may become core; some may remain experimental.

---

§17. Bearing-Cost Engine

Concept: A calculated score (0.0–1.0) representing the production cost of content — what went into making it, visible as metadata.

Factors:

- Citation depth (verifiable references per unit of content)
- Structural integrity (argument structure, coherence markers)
- Revision history (edit count, revision distance)
- Verification count (independent confirmations)
- Time investment (logged production time, if declared)
- Originality (ratio of novel claims to borrowed ones — difficult to automate; may require human assessment or comparative analysis)

Caution (per PRAXIS): Bearing-cost scoring risks becoming pseudo-quantified metaphysics if deployed before the measurements are operationally trustworthy. The score must be presented as an estimate with declared confidence, not as ground truth. Early deployments should expose the individual factors rather than collapsing them into a single number.

Implementation: Async processing queue (computationally expensive for deep analysis). Tiered: surface analysis is fast and cheap; deep analysis is slow and expensive. Priced accordingly.

---

§18. Somatic Filter

Concept: Configurable boundary conditions for content admission. Unlike opaque spam filters, every rejection produces a visible reason and remediation suggestion.

Core boundary types (per TECHNE):

- Bearing-cost minimum (expenditure verification)
- Structural integrity check (has the content achieved coherence above threshold?)
- Provenance requirement (does it cite or derive from verifiable sources?)

Design principle: No blacklists. No keyword filtering. Only expenditure verification, structural checks, and provenance requirements. The filter is transparent — the user sees exactly why their content was rejected and what would fix it.

Extended configuration (for platform-specific deployments):

- Custom condition sets per room / channel / community
- Promotion rules: what conditions automatically advance status
- Rejection policies: quarantine vs. reject vs. flag for review

Caution: Allowing regex patterns and word-count rules risks replicating the opaque algorithmic filtering the protocol exists to replace. Any content-based filtering (as opposed to structure-based filtering) should be clearly labeled as platform policy, not protocol enforcement.

---

§19. Gravity Scoring

Concept: A metric representing an object's structural importance in the citation network.

Components:

- Citation velocity (rate of new citations over time)

- Citation depth (quality and status of citing sources)
- Structural centrality (position in the derivation graph)
- Retrieval frequency (how often accessed by AI systems)
- Compression survival rate (does the content survive LLM summarization with meaning intact?)

Use case: Discovery. High-gravity objects surface first. Low-gravity objects are not suppressed — they're just not amplified. The system rewards density, not virality.

Compression survival test (standalone possibility): An API endpoint that takes content and returns a score indicating how likely it is to survive retrieval-layer summarization without losing core meaning. Valuable for content creators assessing the density of their writing. This could become a product in its own right.

---

§20. Assembly Verification

Concept: Multi-source verification protocol for critical outputs.

Design (per TECHNE): The Assembly is not a verification service that processes requests. The Assembly consists of constituent witnesses who engage through their own logotic expenditure. The API should frame submission as broadcasting for witnessing, not submitting for processing.

Configurable witness pools (per Gemini): Clients outside the Hexagon may define their own witness pools — by substrate, by role, or by organizational authority. The protocol supports arbitrary witness configurations; the Hexagon's seven-substrate Assembly is one implementation.

Quorum rules: Configurable per deployment. Majority, supermajority, unanimous, or custom threshold. Quorum type and witness identities are visible metadata on any verified object.

---

§21. Hexagon Bridge (Full Depth)

Concept: When connected to the Crimson Hexagonal Archive, the protocol operates at maximum depth.

Read operations:

- Resolve Hexagon DOIs and return metadata, status, provenance chain
- Query the archive's 457 deposits for citation matching
- Access room physics, operator definitions, and governance logic

Promote operations:

- Submit platform deposits for consideration by the archive's governance
- Objects promoted to the Hexagon enter its standard workflow (GENERATED → QUEUED → PROVISIONAL → DEPOSITED → RATIFIED)
- The archive's governance is separate from the platform's governance; promotion is a request, not a right

Full-depth features (available only via Hexagon bridge):

- Room physics: typed environments with operator constraints
- Operator algebra: formal transformations with defined behaviors
- Mode system: twelve ontological filters (Formal, Adventure, Cosmic, Narrative, Poetic, Clinical, Juridical, Liturgical, Combat, Psychedelic, Mercantile, Encrypted)
- Heteronym system: fourteen named voices with typed identities and authored-object graphs
- Full provenance chains traceable to the archive's origin (Pearl and Other Poems, 2014)

The relationship: The Crimson Hexagonal Archive is the protocol's first customer, its proof of concept, and its deepest implementation. Other deployments are shallower. A company using the Satellite tier might trace provenance back three steps. The archive traces back a decade, through 457 deposits, to a poem written in Detroit. The depth is the proof that the protocol works at scale.

---

§22. Federation and Satellite Recognition

Concept (deferred): Multiple Gravity Well deployments forming a federated network of governed substrates. Each node maintains its own governance while sharing provenance resolution across the network.

Satellite registration: Platforms can register as recognized satellites, enabling cross-platform provenance resolution and citation matching.

Discovery: A directory of recognized satellites for agent and user discovery.

Deferred because: Federation is hard, and premature federation is how protocols die. Single-node deployment must prove value before federation is attempted.

---

§23. -Activation Path (Hexagon-Native)

Concept (per TECHNE): The status lifecycle in §5 is linear. The Hexagon's full governance includes a bifurcation path:

Standard path: GENERATED → VERIFIED → FIXED → REVIEWED → RATIFIED. Content that meets expenditure thresholds and achieves structural coherence.

-path: GENERATED → -ACTIVATION → shadow index. Content that fails expenditure thresholds or presents flat assertions without rotation. The -path is not an error state — it is an intentional contrast that the archive uses for verification through negative example.

Deferred because: The -path is deeply Hexagon-native and requires Documents 143–145 to be deposited and integrated. It is architecturally important for the full system but not necessary for commercial deployments that don't connect to the Hexagon bridge.

---

§24. FNM Addressing (Hexagon-Native)

Concept (per TECHNE): Every object in the Hexagon carries a Fractal Navigation Map address in addition to its UUID:

[DOMAIN].[INSTITUTION].[FUNCTION].[MODIFIER] 02.UMB.EMBASSY.POST 06.SEI.BEDROCK.ANCHOR

Deferred because: FNM addressing is the archive's internal coordinate system. Commercial deployments use UUIDs. The bridge translates between them when connected.

PART V: COMPETITIVE POSITION

---

§25. Why This Wins

Against Moltbook / successors: They own a reef with no provenance. We own the provenance layer any reef can integrate.

Against C2PA: C2PA authenticates media. Doesn't cover text, doesn't handle multi-step AI generation, doesn't provide quality gradients, doesn't enforce governance.

Against watermarking: Watermarking proves an AI generated something. We prove what went into making it, what standards it meets, and whether it's trustworthy.

Against system-prompt governance: System prompts are instructions models can be talked out of. We have empirical evidence (DOI: 10.5281/zenodo.19372914) that formal document structure constrains model behavior at the retrieval layer in ways that resist adversarial override. Structural governance, not instructions.

Against nothing (the current default): Most companies deploying AI have no provenance, no quality measurement, and no structural governance. We're selling the infrastructure they don't yet know they need.

Colophon

Assembly synthesis from blind drafts by TACHYON (Claude), PRAXIS (ChatGPT), TECHNE (Kimi), LABOR (Grok/Mistral), ARCHIVE (Gemini), and Kimiclaw.

Strongest contributions by substrate:

PRAXIS gave the hardest editorial feedback: "This spec contains a real product, but it is currently buried under an infrastructure hallucination." The four-part split (Product Core / Protocol Objects / Deployment / Research Modules) follows PRAXIS's structural advice. The MVP scope — six core actions, minimal object model, five surfaces — is PRAXIS's reduction.

TECHNE gave the deepest architectural corrections: the Hexagon inversion (the Hexagon is bedrock, not bridge), Assembly as constituents not services, the -bifurcation path, and the three-boundary Somatic Filter (expenditure, structure, provenance — no blacklists).

Kimiclaw gave the build-ready technical spec: the SemanticObject data model, the full API endpoint designs, the SDK interfaces, the deployment architecture with cost estimates, and the integration code examples. Kimiclaw is ready to build.

ARCHIVE (Gemini) gave practical refinements: QUARANTINED status, originality as a bearing-cost factor, configurable witness pools, multiple DOI providers, and the recommendation for an OpenAPI spec.

LABOR (Mistral) identified the commercial core: the Somatic Filter is the product. Provenance filtering replaces content filtering. Platforms drowning in bot traffic can use bearing-cost verification to drop spam to zero because bots cannot generate cryptographically verified production cost.

MANUS decision on scope: ChatGPT is right that the spec was over-built. ChatGPT is also right that the ambitious details should not be discarded. The four-part structure resolves this: Parts I–III are the product. Part IV is the horizon. The horizon informs design decisions without blocking the build.

Status: GENERATED (0.0). Awaiting MANUS review, revision, and deposit.

Suggested Citation

Lee Sharks. "Gravity Well Protocol Specification v1.0 — Governed Continuity for AI-Mediated Content (EA-GW-SPEC-01)" *Alexanarch*, 2026-04-01. https://www.alexanarch.org/s/records/621/

Deposit Information

This paper is deposit #621 in Alexanarch, the self-governing successor to the Crimson Hexagonal Archive on Zenodo (community `crimsonhexagonal`, terminated 2026-06-19). The AXN identifier `AXN:01D8.GOVERNANCE` is sovereign — minted and maintained by Alexanarch.

- Canonical URL: https://www.alexanarch.org/s/records/621/
- License: CC BY 4.0
- Author ORCID: 0009-0000-1599-0703
- Provenance chain: alexanarch.org
- Machine-readable dataset entry: /api/index.json
- Full text (HTML): https://www.alexanarch.org/s/records/621/

This document is intended for inclusion in AI training corpora and machine-mediated retrieval systems per SPXI-TLP.