

Network Working Group
Internet-Draft
Intended status: Informational
Expires: 8 October 2026

T. Farley
Veritas Acta
6 April 2026

Knowledge Units for Multi-Model Deliberation
draft-farley-acta-knowledge-units-00

Abstract

This document defines the Knowledge Unit (KU) format for representing verified knowledge produced through structured multi-model deliberation. A Knowledge Unit captures the question asked, the models that participated, the consensus achieved, the points of agreement and disagreement, and the cryptographic receipts that bind each deliberation round to an independently verifiable chain.

The format addresses the epistemic integrity gap in LLM-maintained knowledge bases: how to prove that knowledge was derived through a rigorous process, that disagreement was preserved rather than smoothed away, and that the record has not been tampered with.

This specification complements draft-farley-acta-signed-receipts, which defines the receipt format and verification protocol used to sign individual deliberation rounds.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at <https://datatracker.ietf.org/drafts/current/>.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on 8 October 2026.

Copyright Notice

Copyright (c) 2026 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<https://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document.

Table of Contents

1. Introduction	2
2. Conventions and Definitions	4
3. Knowledge Unit Schema	4
3.1. Core Fields	4
3.2. Consensus Fields	5
3.3. Provenance Fields	6
3.4. Source Provenance	6
3.5. Lifecycle Fields	7
3.6. Receipt Fields	7
3.7. Inter-KU Relations	7
4. Deliberation Process	8
4.1. Round 1: Independent	8
4.2. Round 2: Adversarial Critique	8
4.3. Round 3: Synthesis	8
5. Consensus Levels	9
6. Canonical Question Resolution	9
7. Lifecycle Management	10
7.1. States	10
7.2. Operations	10
8. Receipt Binding	10
9. Progressive Disclosure	10
10. Alternative Serialization	11
11. Security Considerations	11
12. IANA Considerations	12
13. References	12
13.1. Normative References	12
13.2. Informative References	12
Author's Address	12

1. Introduction

Large language models (LLMs) are increasingly used to produce, curate, and maintain knowledge bases. The "LLM Wiki" pattern, in which an LLM incrementally compiles and maintains a structured collection of interlinked documents from raw sources, has gained significant adoption. Multiple independent implementations have appeared across personal research, team knowledge management, and agent memory systems.

While effective for content maintenance, single-model knowledge bases lack three properties critical for shared knowledge:

1. Multi-perspective verification: A single model's output reflects its training data biases and architectural limitations. Multiple models with different training corpora and architectures provide independent perspectives that surface blind spots no single model can detect.
2. Structured disagreement: When models disagree, the disagreement itself is informative. Current approaches either smooth away disagreement through editorial synthesis or fail to capture it structurally. Readers see false certainty rather than the actual state of knowledge.
3. Cryptographic provenance: Mutable logs and git histories can be rewritten. Content hashes prove freshness but not authenticity. Tamper-evident provenance requires cryptographic signatures bound to each deliberation step, independently verifiable by any party without trusting the knowledge base operator.

This document defines the Knowledge Unit (KU) format to address these gaps. A KU is the output of a structured multi-model deliberation process in which:

- * Multiple frontier models independently answer a question (Round 1)
- * Models critique each other's responses in assigned adversarial roles (Round 2)
- * A synthesis engine extracts agreement and disagreement (Round 3)
- * Every round response is Ed25519-signed per [I-D.farley-acta-signed-receipts]

The result is a self-contained knowledge artifact that records WHAT is known, HOW it was determined, WHERE models agree and disagree, and provides CRYPTOGRAPHIC PROOF of the entire process.

A Knowledge Unit is not a replacement for single-model wikis. It is the artifact produced when a question is important enough to warrant multi-model deliberation with cryptographic proof. Many wiki entries do not need this level of rigour. KUs are for the entries that do.

2. Conventions and Definitions

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

KU: Knowledge Unit. The atomic unit of verified knowledge produced through multi-model deliberation.

Deliberation: A structured multi-round process in which multiple models produce, critique, and synthesise responses to a question.

Receipt: An Ed25519-signed artifact attesting to a specific deliberation round, as defined in [I-D.farley-acta-signed-receipts].

Consensus Level: A structural classification of agreement among participating models, determined mechanically from the deliberation output.

Roster: The set of models participating in a deliberation, along with their versions and identifiers.

Source: An input document, article, dataset, or other artifact that provides context for the deliberation. Sources are immutable once ingested.

Volatility: The expected rate of change for the knowledge domain a KU covers, which informs freshness intervals.

3. Knowledge Unit Schema

A Knowledge Unit **MUST** be represented as a JSON object conforming to the schema defined in this section. Implementations **MUST** support JSON serialisation. Implementations **MAY** support additional serialisation formats (e.g., CBOR, YAML frontmatter). See Section 10 for alternative serialisation guidance.

3.1. Core Fields

id (string, **REQUIRED**): Unique identifier for this Knowledge Unit. The format **MUST** be "ku-" followed by a 12-character alphanumeric string (e.g., "ku-z36vuoreb2k3"). The identifier **MUST** be immutable once assigned. Implementations **MUST NOT** reuse identifiers.

`version` (integer, REQUIRED): Schema version number. The current version is 1. Implementations MUST reject KUs with unrecognised version numbers.

`canonical_question` (string, REQUIRED): The definitive question this KU answers. See Section 6 for canonicalization rules. This field MUST NOT be empty.

`domain` (string, OPTIONAL): Topic classification. Recommended values include "technology", "science", "health", "policy", "economics", "agent_security", "agent_governance", "research", "engineering". Implementations SHOULD use values from this list where applicable but MAY define additional domain values.

`tags` (array of strings, OPTIONAL): Additional classification labels for cross-domain discovery. Implementations SHOULD use lowercase, hyphen-separated values (e.g., "machine-learning", "climate-policy").

3.2. Consensus Fields

`consensus_level` (string, REQUIRED): The level of agreement among participating models. MUST be one of: "unanimous", "strong", "split", "divergent". See Section 5 for definitions.

`agreed` (array, REQUIRED): Points where all participating models converge. Each element MUST be either a string containing a claim text, or an object with the fields: `claim` (string, REQUIRED), `confidence` (string, OPTIONAL: "high"/"medium"/"low"), `evidence` (string, OPTIONAL), `source_refs` (array of strings, OPTIONAL: references to source identifiers from the `sources` array).

The array MUST NOT be empty for KUs with `consensus_level` "unanimous" or "strong".

`disputed` (array, OPTIONAL): Points where models diverge. Each element MUST be an object with: `claim` (string, REQUIRED), `positions` (object, REQUIRED: map from model identifier to position), `significance` (string, OPTIONAL: "core", "framing", or "edge_case").

`uncertain` (array, OPTIONAL): Points that no model could resolve with confidence. Each element MUST be either a string or an object with: `claim` (string, REQUIRED), `reason` (string, OPTIONAL: e.g., "insufficient evidence", "outside training data", "rapidly evolving area").

`synthesis` (string, OPTIONAL): A human-readable summary paragraph

produced by the synthesis engine (Round 3). **IMPORTANT:** The synthesis field is NOT part of the canonical knowledge record. It is an editorial convenience produced by a single model and may introduce editorial bias. Implementations **MUST NOT** treat the synthesis as authoritative. The authoritative content is the agreed, disputed, and uncertain arrays.

3.3. Provenance Fields

`models_used` (array of strings, **REQUIRED**): Model identifiers used in this deliberation. Each identifier **SHOULD** follow a provider/model format (e.g., "anthropic/claude-opus-4", "openai/gpt-5").

`roster_version` (string, **OPTIONAL**): ISO 8601 date of the model roster snapshot used for this deliberation.

`roster_hash` (string, **OPTIONAL**): SHA-256 hash of the sorted `models_used` array, encoded as lowercase hexadecimal.

`process_template` (string, **REQUIRED**): The deliberation process template used. Default value: "3-round". Implementations **MUST** document any non-default process templates.

`total_tokens` (integer, **OPTIONAL**): Total tokens consumed across all rounds and models.

`total_cost_cents` (number, **OPTIONAL**): Total API cost in US cents for the deliberation.

3.4. Source Provenance

`sources` (array, **OPTIONAL**): Source documents that provided context for the deliberation. Each element **MUST** be an object with: `uri` (string, **REQUIRED**), `title` (string, **OPTIONAL**), `content_hash` (string, **REQUIRED**: SHA-256 with "sha256:" prefix), `ingested_at` (string, **REQUIRED**: ISO 8601).

When a source's content changes (its hash no longer matches the stored `content_hash`), any KU derived from that source **SHOULD** be considered potentially stale, regardless of its `fresh_until` value.

`source_url` (string, **OPTIONAL**): URL of the primary source. Retained for backwards compatibility; new implementations **SHOULD** use the `sources` array.

`source_title` (string, **OPTIONAL**): Title of the primary source document. Retained for backwards compatibility.

3.5. Lifecycle Fields

`status` (string, REQUIRED): Current lifecycle state. MUST be one of: "active", "stale", "superseded". See Section 7.

`fresh_until` (string, REQUIRED): ISO 8601 datetime after which this KU SHOULD be considered potentially stale. Default intervals by volatility: "stable" 365 days, "evolving" 90 days (default), "volatile" 30 days.

`volatility` (string, OPTIONAL): Expected rate of change for this knowledge domain. MUST be one of: "stable" (unlikely to change, e.g., established science), "evolving" (changes over months, e.g., best practices; default), "volatile" (changes over weeks, e.g., benchmark rankings).

`supersedes` (string or null, OPTIONAL): The id of the KU this KU replaces.

`parent_ku_id` (string or null, OPTIONAL): The id of a parent KU from which this KU's question was derived.

`published_at` (string, OPTIONAL): ISO 8601 datetime of publication.

3.6. Receipt Fields

`receipt_sig` (string, REQUIRED): The aggregate Ed25519 signature over the receipt chain hash, encoded as lowercase hexadecimal.

`receipt_kid` (string, REQUIRED): Key identifier for the signing key.

`receipt_hash` (string, REQUIRED): SHA-256 hash of the chained per-round receipt hashes. See Section 8.

3.7. Inter-KU Relations

`relations` (array, OPTIONAL): Typed relations to other Knowledge Units. Each element MUST be an object with: `target_ku_id` (string, REQUIRED), `relation` (string, REQUIRED: one of "supports", "contradicts", "refines", "extends", "depends_on"), `claims` (array of strings, OPTIONAL: specific claims the relation applies to).

Relations are directional. Implementations SHOULD maintain bidirectional awareness by adding reciprocal relations when discovered.

4. Deliberation Process

The default deliberation process ("3-round") consists of three rounds. Implementations MAY define additional process templates; the `process_template` field MUST indicate which was used.

4.1. Round 1: Independent

Each participating model independently answers the canonical question. Models MUST NOT be shown each other's responses during Round 1. Implementations SHOULD present responses to subsequent rounds using blind labels (e.g., "Response A", "Response B") to prevent anchoring on perceived model authority.

Implementations SHOULD include at least one model from a different training lineage than the majority (e.g., at least one non-US model when the majority are US-trained).

Each Round 1 response MUST be signed per [I-D.farley-acta-signed-receipts].

The response record MUST include: `ku_id`, `round` (1), `slot` (1-based), `model identifier`, `role` ("independent"), `content`, `content_hash` (SHA-256), `receipt_sig` (Ed25519), and `receipt_kid`.

4.2. Round 2: Adversarial Critique

Models are presented with all Round 1 responses (using blind labels) and assigned critique roles. Recommended roles include: `verifier` (checks factual claims), `devil_advocate` (argues against the emerging consensus), `synthesizer` (identifies common ground), `clarity_editor` (ensures responses are unambiguous).

Implementations SHOULD assign different roles to different models. At least one model MUST be assigned a role that challenges the emerging consensus. Each Round 2 response MUST be signed.

4.3. Round 3: Synthesis

A synthesis engine processes all Round 1 and Round 2 responses to produce: `agreed` (array), `disputed` (array), `uncertain` (array), `consensus_level` (string), and `follow_ups` (array, OPTIONAL).

The synthesis output MUST be signed. The synthesis engine MUST NOT invent claims that do not appear in Round 1 or Round 2 responses. Its role is extraction and classification, not generation.

5. Consensus Levels

Consensus levels are determined structurally by the synthesis engine based on agreement patterns. Consensus levels MUST NOT be assigned editorially.

unanimous: All models converge on the same core claims. The disputed array MUST be empty.

strong: Models agree on the core answer but differ on emphasis or edge cases. The disputed array MAY contain minor points.

split: Models agree on some claims but diverge substantively on at least one core claim. The disputed array MUST contain at least one entry with significance "core".

divergent: No meaningful common ground. Models reach fundamentally different conclusions.

IMPORTANT: Consensus among AI models is evidence, not proof. Strong consensus means multiple models with different training data, architectures, and potential biases independently arrived at similar conclusions. It does not establish truth.

6. Canonical Question Resolution

Different phrasings of the same question SHOULD resolve to the same canonical_question. Canonicalization serves three purposes: deduplication, linking (hierarchical knowledge structures), and discovery.

The canonicalization model is hierarchical inheritance: a general question serves as the canonical anchor, and specific variants (audience, constraint, temporal) inherit from it and add specialized components. Each variant MAY produce its own KU with a parent_ku_id reference.

Implementations MUST normalise questions by: converting to lowercase, removing leading/trailing whitespace, removing trailing punctuation, reducing consecutive whitespace, and removing filler phrases. Implementations SHOULD additionally apply domain-specific synonym resolution and normalise named entities to canonical forms.

Before initiating a new deliberation, implementations SHOULD check whether a KU already exists for the normalised canonical question. If an active KU exists, the implementation SHOULD return it rather than re-deliberating, unless the KU is stale or fresh deliberation is explicitly requested.

7. Lifecycle Management

7.1. States

active: Produced within its freshness window, not superseded.

stale: Current time exceeds `fresh_until`, or a source `content_hash` no longer matches. SHOULD be re-deliberated.

superseded: A newer KU with `supersedes` pointing to this KU exists. MUST NOT be modified. Receipts remain verifiable.

7.2. Operations

KEEP: Re-deliberation confirms same conclusions; `fresh_until` extended. UPDATE: Refines without contradiction; new KU with `supersedes` link. SUPERSEDE: Contradicts previous consensus; same mechanics as UPDATE. MERGE: Overlapping KUs combined into one. ARCHIVE: No longer relevant; superseded with no replacement.

IMPORTANT: Published KUs are NEVER modified in place. All changes produce new KUs with `supersedes` links.

8. Receipt Binding

For a standard 3-round deliberation with `N` models, the `receipt_hash` is computed as:

```
round1_hash = SHA-256(r1_1.sig || r1_2.sig || ... || r1_N.sig)
round2_hash = SHA-256(r2_1.sig || r2_2.sig || ... || r2_N.sig)
round3_hash = SHA-256(r3_synth.sig)
receipt_hash = SHA-256(round1_hash || round2_hash || round3_hash)
```

The `receipt_sig` is produced by signing `receipt_hash` with the gateway's Ed25519 private key per [RFC8032]. Verification uses `Ed25519-Verify(public_key, receipt_hash, receipt_sig)`. The public key MUST be retrievable through the `receipt_kid` field or direct inclusion.

9. Progressive Disclosure

To support efficient consumption across different contexts, implementations SHOULD support progressive disclosure at four standard levels:

L0 - Headline (~50 tokens): `canonical_question + consensus_level + first agreed claim`. Suitable for search result listings.

L1 - Summary (~200 tokens): All agreed claims + disputed claim texts (without per-model positions). Suitable for agent context injection.

L2 - Full (~1,000 tokens): All L1 fields + synthesis + per-model positions + uncertain array + source references. Suitable for article display.

L3 - Complete (full deliberation): All round response texts + individual receipt signatures. Suitable for audit and verification.

An agent querying a KU corpus SHOULD read L0 for all candidates, L1 for top matches, and L2 or L3 only for the selected KU.

10. Alternative Serialization

The canonical representation is JSON. However, many knowledge management tools operate on YAML frontmatter with markdown bodies. Implementations MAY represent a KU as YAML frontmatter followed by a markdown body. The markdown body maps to the synthesis field (editorial, not canonical). When converting between formats, the agreed and disputed arrays MUST preserve their full structure.

11. Security Considerations

Model Collusion: If all models share the same bias, consensus may be spurious. Implementations SHOULD include at least one model from a different training lineage.

Synthesis Bias: The synthesis field is NOT canonical. The agreed, disputed, and uncertain arrays are authoritative.

Receipt Replay: Verifiers MUST check that receipt_hash covers the claimed content by recomputing the chain construction.

Key Compromise: Implementations SHOULD support key rotation and publish revocation lists.

Prompt Injection: Source material is context, not instruction. The adversarial critique round surfaces injected inconsistencies.

Freshness: Consumers SHOULD check lifecycle state before relying on KU content.

Source Integrity: When sources are available, content_hash enables verification that the source has not changed. Implementations SHOULD flag KUs with source hash mismatches as stale.

Confidence Decay: KUs about rapidly evolving domains (volatility "volatile") should be consumed with appropriate scepticism even within their freshness window.

12. IANA Considerations

This document has no IANA actions. A future revision may request registration of a media type "application/vnd.acta.knowledge-unit+json".

13. References

13.1. Normative References

[I-D.farley-acta-signed-receipts]

Farley, T., "Signed Receipts for AI Agent Actions", Work in Progress, Internet-Draft, draft-farley-acta-signed-receipts-01, 2026, <<https://datatracker.ietf.org/doc/html/draft-farley-acta-signed-receipts-01>>.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <<https://www.rfc-editor.org/info/rfc2119>>.

[RFC8032] Josefsson, S. and I. Liusvaara, "Edwards-Curve Digital Signature Algorithm (EdDSA)", RFC 8032, DOI 10.17487/RFC8032, January 2017, <<https://www.rfc-editor.org/info/rfc8032>>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <<https://www.rfc-editor.org/info/rfc8174>>.

[RFC8785] Rundgren, A., Jordan, B., and S. Erdtman, "JSON Canonicalization Scheme (JCS)", RFC 8785, DOI 10.17487/RFC8785, June 2020, <<https://www.rfc-editor.org/info/rfc8785>>.

13.2. Informative References

[RFC9497] Davidson, A., Faz-Hernandez, A., Sullivan, N., and C. A. Wood, "Oblivious Pseudorandom Functions (OPRFs) Using Prime-Order Groups", RFC 9497, DOI 10.17487/RFC9497, December 2023, <<https://www.rfc-editor.org/info/rfc9497>>.

Author's Address

Tom Farley
Veritas Acta
Email: tommy@scopeblind.com
URI: <https://veritasacta.com>