# The Concept AXN: Implementing the Stable Work Relation the LABOR Invariant Already Specifies

Sharks, Lee

2026-08-14

**The Concept AXN: Implementing the Stable Work Relation the LABOR Invariant Already Specifies**

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

2026-08-14 · Version v0.1 DRAFT · Protocol specification

AXN:05F3.STRUCTURAL.⊗

https://www.alexanarch.org/s/records/1473/



---

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


```
{
  "@context": "https://schema.org",
  "@type": "ScholarlyArticle",
 "name": "The Concept AXN: Implementing the Stable Work Relation the LABOR Invariant Already Specifies",
  "author": {
    "@type": "Person",
    "name": "Sharks, Lee",
    "identifier": "https://orcid.org/0009-0000-1599-0703",
    "affiliation": "Crimson Hexagonal Archive / Alexanarch"
  },
  "datePublished": "2026-08-14",
  "identifier": "AXN:05F3.STRUCTURAL.⊗",
  "license": "https://creativecommons.org/licenses/by/4.0/",
  "publisher": {
    "@type": "Organization",
    "name": "Alexanarch"
  },
```

```
  "url": "https://www.alexanarch.org/s/records/1473/",
  "description": "Seven editions of one work carried one backward pointer and zero forward pointers, so a reader
pointing every prior edition on each version bump is order-n work per version and conflicts with content-
addressing, since editing a published edition either changes its hash or places the pointer outside the hashed
level identifier resolving to the current edition, order-one forever. REFRAMED BEFORE DEPOSIT: this is not a ne
- root_axn is composed from the per-deposit hex and is therefore not stable across editions. This document is an
}
```


## Abstract

Seven editions of one work carried one backward pointer and zero forward pointers, so a reader arriving at the edition a search engine is most likely to hold had no route to the current state. Forward-pointing every prior edition on each version bump is order-n work per version and conflicts with content-addressing, since editing a published edition either changes its hash or places the pointer outside the hashed content where a copy of the deposit does not carry it. The resolution is a concept-level identifier resolving to the current edition, order-one forever. REFRAMED BEFORE DEPOSIT: this is not a new identifier type. The LABOR canonical invariant already specifies AXN root as a stable work relation beside AXN version as an immutable version relation, and the AXN protocol now carries that at identity semantics. What is missing is narrower and is recorded there as a known gap – root_axn is composed from the per-deposit hex and is therefore not stable across editions. This document is an implementation proposal for an existing invariant, and its derivation section states the open design question.

## Body

---

deposit_number: 1473 hex: 05F3 title: “The Concept AXN: Implementing the Stable Work Relation the LABOR Invariant Already Specifies” creator: Sharks, Lee orcid: 0009-0000-1599-0703 date: 2026-08-14 content_type: Protocol specification license: CC-BY-SA-4.0 substrate: AI-assisted (substrate) axn_schema_version: v2 protocol_version: alexanarch-deposit-protocol/v1 keywords: - concept AXN - AXN root - stable work relation - immutable version relation - version chain - supersession - forward pointer - content addressing - LABOR canonical invariant - identifier design *** # The Concept AXN: Implementing the Stable Work Relation the LABOR Invariant Already Specifies

## Description

Seven editions of one work carried one backward pointer and zero forward pointers, so a reader arriving at the edition a search engine is most likely to hold had no route to the current state. Forward-pointing every prior edition on each version bump is order-n work per version and conflicts with content-addressing, since editing a published edition either changes its hash or places the pointer outside the hashed content where a copy of the deposit does not carry it. The resolution is a concept-level identifier resolving to the current edition, order-one forever. REFRAMED BEFORE DEPOSIT: this is not a new identifier type. The LABOR canonical invariant already specifies AXN root as a stable work relation beside AXN version

as an immutable version relation, and the AXN protocol now carries that at identity semantics. What is missing is narrower and is recorded there as a known gap – root_axn is composed from the per-deposit hex and is therefore not stable across editions. This document is an implementation proposal for an existing invariant, and its derivation section states the open design question.

### Methodology

Measured against the capture registry's own version chain: editions enumerated from the registry, forward and backward pointers counted. Derivation candidates compared for stability under edition restoration, since the archive has restored early editions and cannot treat first-edition bytes as a stable quantity. Grammar conformance checked against the AXN protocol's compose and derive functions.

### Falsification Conditions

The scheme fails if two distinct works can produce the same work identifier, since the derivation would collide. The honesty requirement fails in practice if a concept AXN is ever cited as evidence for a textual claim, which would show the family marker insufficient. The order-n argument is falsified if forward-pointing can be shown to preserve content-addressing without placing the pointer outside the hashed content.

# THE CONCEPT AXN

**A version-stable identifier for a work across its editions**

> **REFRAMED BEFORE DEPOSIT.** This document was drafted as a proposal for a NEW identifier type. It is not one. **The LABOR canonical invariant already specifies `AXN root = stable work relation` beside `AXN version = immutable version relation`** — the two-level scheme is law, not a proposal, and `api/axn-protocol.json` now carries it at `identity_semantics`.
>
> What is actually missing is narrower and is recorded in the protocol as a `known_gap`: **`root_axn` is composed from the per-deposit hex, so it is not stable across editions of one work.** Two editions of the same work carry different roots today. Read everything below as an **implementation proposal for an existing invariant**, and read §III's derivation as the open design question — whether a stable root is derived from the work identifier or allocated with the first edition.

---

## I. The defect

**Older versions of a work do not point forward to the current one.** Measured on the capture registry:

| deposit | version | supersedes | superseded_by |
|---|---|---|---|
| #825 | v7.2 | — | — |
| #936 | v8.9 | — | — |
| #1396 | v0.2-restored | — | — |
| #1397 | v6.1 | — | — |
| #1398 | v1.0 | — | — |
| #1401 | v1.0 | — | — |
| #1461 | **v10.9** |  | — |

Seven editions of one work. **One backward pointer, zero forward pointers.** A reader arriving at #936 has no way to learn that v10.9 exists, and #936 is what a search engine is most likely to have indexed, because it has been public longest.

This is the archive's own subject arriving at home: a stale state, confidently addressable, with the current state unreachable from it.

## II. Why the obvious fix is the wrong fix

The obvious repair is to write `superseded_by` into every prior edition on each version bump.

**That is O(n) writes per version and the n grows.** Seven editions today; the eleventh bump rewrites ten records. Worse, it violates a property the archive depends on: **a deposit's canonical text is content-addressed, and its AXN is derived from those bytes.** Editing a published edition to add a forward pointer either changes its hash — invalidating its identifier — or requires the pointer to live outside the hashed content, in which case it is registry metadata that a copy of the deposit does not carry.

So forward-pointing is expensive, fragile, and does not travel with the object. It should not be the mechanism.

## III. The concept AXN

**One identifier for the work, allocated once, resolving to the current edition. O(1) forever.**

Exactly the concept-DOI relation the archive already relies on from the Zenodo chain — where EA-WG-CAPTURES-01 carried concept DOI `10.5281/zenodo.20683855` beside first-version DOI `…856`.

```
CONCEPT AXN  AXN:<hex>.CONCEPT.<glyph>    the work, across all editions → resolves to current
VERSION AXN  AXN:<hex>.<FAMILY>.<glyph>   one edition, content-derived, immutable, permanent
```

A version AXN never changes and never learns about its successors. **The concept AXN is the only thing that moves**, and it moves by a single write to the resolver.

### Derivation

The glyph derives from `sha256(work_identifier)` — **the stable work id, not any edition's bytes**:

`EA-WG-CAPTURES-01 → sha256 fe289e1f8d29e9ee… → ♻□■☁□□`

This is grammar-conformant: `compose_axn` takes any family, `axn_glyph_from_hash` takes any sha256. Nothing new is invented.

**Why the work id rather than the first edition's hash.** Zenodo allocates the concept DOI with v1, which privileges v1's bytes. The archive cannot afford that: **#1396 is a *restored* edition of an early version**, so "the first edition's bytes" is not a stable quantity here. Deriving from the work id means the concept AXN is **reproducible by anyone who knows the work identifier, forever, without access to any edition at all** — including after a deletion event.

### The honesty requirement

The apparatus grammar (#1077, device 4) states:

> *An AXN verifies the identity and integrity of the addressed content; a URL merely locates a resource.*

**A concept AXN does not do that**, and must not be allowed to imply it. Its referent changes by design. It verifies **work identity**; it does not verify content integrity, because there is no fixed content to verify.

**The `CONCEPT` family name carries that distinction in the identifier itself** rather than in documentation a compression will drop. A reader or a machine meeting `AXN:xxxx.CONCEPT.…` is told, in the identifier, that this addresses a work and not a byte sequence.

**Rule: never cite a concept AXN as evidence for a claim about content.** Cite the version AXN. The concept AXN is for *"the current state of this instrument"*, never for *"the text said X."*

## IV. What this replaces and what it does not

**Replaces** `version_series_id` — currently a bare string on 160 deposits (`SERIES-EA-ARK-01-ARCHON`), which is not resolvable, not citable, and not in the identifier namespace. Concept AXNs should supersede it, with the existing strings mapped rather than discarded.

**Does not replace** `supersedes`. A backward pointer is written **once, at mint, by the edition that knows** — it is O(1) and it belongs in the new edition's own hashed content. Keep it. The asymmetry is the point: **backward pointers are cheap and permanent; forward pointers are expensive and perishable, so the forward direction is served by resolution instead of by writing.**

**Does not replace** `superseded_by_deposit_number` where it already exists on 156 deposits. Those are historical adjudications and should stand. New chains use the concept AXN.

## V. Protocol addition

At mint, a deposit that belongs to a versioned work carries:

```
concept_axn        AXN:<hex>.CONCEPT.<glyph>       stable across editions
work_id            EA-WG-CAPTURES-01               what the glyph derives from
version            v10.9
supersedes         [{deposit_number, version}]     written once, backward only
```

And the resolver holds **one row per work**:

```
concept_axn → current_version_axn · current_deposit · updated_at
```

A version bump is **one write to that row**. No prior edition is touched, no hash is invalidated, no forward pointer is stored in content that would go stale the moment it was written.

### The transition record

Per the apparatus grammar's transition ledger (#1077 §3.1), each move appends: previous state · new state · effective date · reason · evidence · responsible operator · prior text hash · replacement text hash — **and the earlier state remains addressable.** The concept AXN is what makes "remains addressable" cheap: the old edition keeps its own AXN forever and needs no maintenance to stay findable.

## VI. First case

**EA-WG-CAPTURES-01**, seven editions, `AXN:<next>.CONCEPT.♻□■☁□□`, resolving to #1461 v10.9.

Then the works with the longest chains: EA-ARK-01 (□□□☿□□), EA-SPXI-01 (□□□☿□□), the GW.TACHYON continuity series, the DOI registry chain.

## VII. What is claimed, and what is not

**Claimed.** That forward-pointing is O(n) per version and conflicts with content-addressing; that a concept identifier is O(1) and is the standard solution; that deriving the glyph from the work id rather than an edition's bytes is more robust for an archive that has had to restore editions; and that the `CONCEPT` family is required so the identifier states its own weaker guarantee.

**Not claimed.** That this is novel — it is the concept-DOI pattern, which the archive already consumed from Zenodo and should now mint natively. That the hex allocation is settled: hexes are opaque sequential, and whether concept AXNs draw from the same sequence as deposits or a reserved range is a decision this document does not make. That the resolver work is small — `axn-resolution.json` ships on multiple surfaces and every one of them would need the new row shape.

**Falsifiable.** If two distinct works can produce the same work id, the derivation collides and the scheme fails. If a concept AXN is ever cited as evidence for a textual claim, the honesty requirement has failed in practice and the family marker is insufficient.

## Suggested Citation

Sharks, Lee. “The Concept AXN: Implementing the Stable Work Relation the LABOR Invariant Already Specifies” *Alexanarch*, 2026-08-14. https://www.alexanarch.org/s/records/1473/

## Deposit Information

This paper is deposit #1473 in Alexanarch, the self-governing successor to the Crimson Hexagonal Archive on Zenodo (community `crimsonhexagonal`, terminated 2026-06-19). The AXN identifier `AXN:05F3.STRUCTURAL.⊗` is sovereign — minted and maintained by Alexanarch.

- Canonical URL: https://www.alexanarch.org/s/records/1473/
- 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/1473/

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