Ontology Induction: Learning the Organization's Nouns

N71 Research

Technical Report TR-2026-03 June 2026

Abstract

The largest hidden cost in organizational AI is the ontology — the object model defining what kinds of things an organization's knowledge consists of. The industry has two answers. The first builds it by hand: embedded engineers spending months per customer constructing a bespoke object model — accurate, slow, and so expensive that the ontology becomes both the product's moat and its margin problem. The second fixes one taxonomy for every customer — cheap and auditable, but structurally incapable of representing the edges of a business, which are precisely the parts that differentiate it. This report specifies a third answer: ontology induction. N71 seeds each workspace from a curated core vocabulary and extends it from the organization's own evidence, with every type-level change passing through the same promotion gate, provenance requirements, and temporal guarantees that govern every other object in the graph (TR-2026-01). The ontology is not a configuration file and not a consulting deliverable; it is a learned, governed, evidence-anchored artifact. We specify the seed-and-extension model, the type-adoption contract, the noise-control mechanisms that make open-ended induction safe — including two production failures that shaped them — and an evaluation methodology for ontology quality, for which no standard benchmark exists and which we therefore define.

1. The Ontology Cost Problem

Every system that structures organizational knowledge must answer a prior question: structured *into what*? The set of entity types, relation types, and their semantics — the ontology — determines what the system can represent at all.

The hand-built answer is exemplified by the forward-deployed model: engineers embedded at the customer, interviewing teams and encoding the organization's object model by hand. It produces accurate ontologies and has built at least one very large company. Its economics are its ceiling: the ontology is a months-long professional-services engagement per customer, deployment cost scales linearly with customers, and the artifact begins decaying the day the engineers leave, because organizations change and hand-built ontologies are updated by nobody.

The fixed-taxonomy answer inverts the trade. Recent memory architectures in the literature adopt a single small class system — on the order of half a dozen categories — applied uniformly to every customer. This is cheap, deterministic, and auditable. It is also a claim that every organization is made of the same kinds of things, which is false exactly where it matters: a pharmaceutical company's *protocols* and *adverse events*, a fund's *theses* and *positions*, a retailer's *planograms* and *shrinkage incidents* are not generic "facts." Forcing them into a universal taxonomy does not delete the information, but it deletes the structure — and structure is the entire argument for moving beyond similarity search.

Our position: the ontology should be a **learned artifact** — induced from the organization's own evidence — and because an induced ontology is a machine-written one, it must be **governed**: proposed, validated, promoted, and superseded under explicit contract, never silently accumulated. Induction without governance is type sprawl; governance without induction is the fixed taxonomy again. The contribution of this report is the combination.

2. The Seed-and-Extension Model

2.1 The seed

Every workspace initializes from a canonical core vocabulary maintained as a single source of truth: a compact set of entity types (person, account, product, project, decision, commitment, and peers — on the order of ten) and a richer set of relation types (on the order of forty) covering authorship, membership, dependency, mention, and lifecycle relations. The seed is deliberately small. Its job is not to describe any organization well; its job is to give extraction a reliable spine on day one and to give induced types something to attach to.

2.2 Constrained extraction with explicit overflow

Extraction (TR-2026-01 §3.3) types candidates against the active workspace vocabulary. The critical design decision is what happens at the vocabulary's edge. Open-ended extraction — letting the model emit whatever type string it likes — produces unbounded sprawl. Hard clamping — forcing every candidate into the nearest seed type — silently destroys the signal that the vocabulary is incomplete. N71 does both halves deliberately: out-of-vocabulary candidates are conservatively typed into a general class *and* the model's suggested type is recorded as a structured type proposal. Nothing novel enters the working ontology at extraction time, and nothing novel is lost. The proposals accumulate as evidence about what the vocabulary is missing.

2.3 Identity-conditioned extraction

Extraction does not run context-free. A per-workspace identity synthesis — a continuously maintained model of what this organization is, what it sells, and what it calls things — conditions the extraction pipeline, so that typing decisions are made with knowledge of the business rather than against a generic prior. The identity is itself a synthesized, evidence-anchored artifact, regenerated as the organization's evidence evolves.

3. The Type-Adoption Gate

Accumulated type proposals become ontology only by passing the promotion gate — the same typed contract architecture that governs every synthesis producer in the system (TR-2026-01 §3.5). The type-adoption contract evaluates a proposed type against the workspace's evidence and returns exactly one of four decisions, each with a typed reason payload:

Promoted. The type enters the workspace vocabulary. Promotion requires recurring, multi-source evidence — a type proposed once by one document is noise; a type proposed repeatedly across meetings, threads, and systems is the organization telling you its nouns.

Provisional. Held and logged, re-evaluated as evidence accumulates. Provisional is a first-class state, not a queue to nowhere: the proposal, its evidence, and its evaluation history are queryable.

Rejected. The candidate remains conservatively typed, and the rejection — including the coercion that extraction applied — is recorded with its reason. Rejections are data: a pattern of rejections in one evidence region is a signal about either the vocabulary or the extractor.

Deprecated. A previously promoted type is retired. Its entities are migrated or retagged under contract; the type's history survives.

Two structural properties matter more than the states themselves. First, **there is no ungated path**: direct writes to the workspace vocabulary are impossible by construction — every type-level mutation routes through the contract. Second, **the gate is the same gate**: type adoption is not a special subsystem with its own rules but another producer on the promotion architecture that already governs instance-level synthesis, which means it inherits that architecture's audit trail, its typed rejection reasons, and its tests.

4. Noise Control: What Production Taught Us

Open-ended induction fails in predictable ways. We know because we hit the failures in production before the controls existed, and we consider the lessons worth publishing.

Case 1 — Entity fragmentation. Early entity resolution used string-similarity fuzzy matching. In production it fragmented a single major account into 79 distinct entity records — same organization, dozens of surface forms, no consolidation. The replacement is graph-enriched LLM resolution: candidate entities are presented to the resolver *with their temporal profiles* (TR-2026-01 §3.4), so the decision "is this the same account?" is made against behavioral history, not string distance. Lexical matching survives only as a recall mechanism feeding candidates to the resolver, never as the decision-maker. Merges execute transactionally, and the pre-merge state is recoverable.

Case 2 — Minting from noise. Promotional email — webinar invitations, marketing blasts — name-drops people and companies. An extraction path that processed flagged low-signal sources minted fake entities from promotional copy: people who had never interacted with the organization, companies that were never counterparties. The fix is layered: source-level signal classification at ingestion (low-signal material is flagged and excluded from entity extraction on every extraction path), name blocklists enforced uniformly including in LLM paths, and — the structural backstop — the promotion gate's evidence requirements, under which an entity observed only in promotional material cannot accumulate the multi-source support that promotion demands.

Case 3 — The namespace lesson. The most consequential pollution we found was not exotic types but a category error: evidence-layer objects (documents, chunks) being represented in the same store as ontology-bearing entities, eventually outnumbering them several-fold. The lesson is architectural and general: **the evidence layer and the ontology layer must be namespace-separated by construction**, or every type-level metric — and eventually the vocabulary itself — drowns in bookkeeping objects. In the current architecture the separation is enforced at the storage layer.

A finding worth reporting against the sprawl fear: with constrained extraction and the overflow path in place, genuinely stray LLM-induced types in production number in the dozens of

types across a few hundred rows — out of hundreds of thousands of extracted entities. Sprawl is not an inevitability of induction; it is a symptom of induction without a contract.

5. The Ontology as Temporal Artifact

TR-2026-01 establishes that facts in N71 carry provenance and time. The ontology is held to the same standard, because an ontology that cannot account for itself reproduces the hand-built artifact's decay problem in automated form.

Type provenance. Every workspace type carries a registry record: when it was first proposed, the evidence chunks that motivated it, its decision history through the adoption gate, and its current status. "Why does this workspace have a *clinical-protocol* type?" has an evidence-anchored answer, exactly as "why does the graph believe this fact?" does.

Versioned, non-destructive evolution. The workspace ontology is never overwritten in place. Vocabulary changes are versioned writes; deprecation supersedes rather than deletes; a type's entities survive its retirement under explicit migration. The organization's type system has a history, because the organization does.

6. Evaluation Methodology

No standard benchmark exists for ontology quality in organizational AI, so we define the measurements and commit to publishing them:

M1 — Induced vs. seed extraction quality. Extraction precision and recall on the same corpus under (a) the seed vocabulary alone and (b) the workspace's induced ontology. The delta is the measured value of induction.

M2 — The sprawl curve. Distinct active (non-evidence-layer) types per workspace over deployment tenure. Healthy induction grows and plateaus as the organization's noun-space is covered; pathological induction grows without bound. We report the curve shape across workspace cohorts.

M3 — Correction decay. Human corrections — merges, retypes, deprecations — per thousand extractions, over tenure. Induction is working if the organization has to fix the system less the longer it runs. Corrections are captured through the audited correction path (TR-2026-01 §3.3), giving this series a durable, verifiable source.

M4 — Vocabulary coverage. The fraction of an organization's high-frequency domain nouns representable as first-class types rather than coerced into a general class — the direct measure of what fixed taxonomies cannot do.

First results — beginning with M2 sprawl curves and the M3 series — will be published alongside this report's companion benchmark results.

7. Limitations and Ongoing Work

Stated plainly. **Type hierarchy:** the current vocabulary is flat; subtype relations (a *clinical protocol* is a *document-governed process* is a *process*) are design work in progress, and queries today do not traverse type generality. **Point-in-time ontology:** entity state supports time-travel queries; the ontology's own history is recorded (§5) but a first-class "the workspace's type system as of March" query surface does not yet exist. **Continuous induction:** type proposals accumu-

late from extraction; scheduled, proactive re-evaluation of the provisional set is early. **Cold start:** like every capability in this architecture, induction compounds with tenure — a week-old workspace has a seed vocabulary and not much else, and we make no claim otherwise.

8. Disclosure

This report specifies the lifecycle, the gate semantics, the noise-control posture, and the evaluation definitions — the claims we expect to be held to. The seed vocabulary's contents, promotion thresholds and evidence-support criteria, consolidation scoring, and the extraction prompt architecture are proprietary.

Companions: TR-2026-01 (the substrate this ontology lives in), TR-2026-02 (the governance architecture the adoption gate inherits).