

An Objective Framework for Assessing the Quality of LLM-Generated Business Ontologies

Mostafa Mahdavihorrami, PhD

Benjamin Miraglio, PhD
Metagem BV, Belgium

August 24, 2026

DRAFT, NOT FOR DISTRIBUTION

Abstract

Agentic pipelines of large language models (LLMs) can now construct a business ontology (concepts with definitions, synonyms, supporting source quotes, and typed relationships) from an enterprise document collection in hours. In current practice, the quality of such an ontology is asserted by the party that generated it, using measurements of its own design. This paper specifies an assessment methodology intended to make such claims objective, reproducible, and explicit about their limits. The setting departs from classical ontology evaluation in three ways: coverage can only be measured against a hand-built reference that does not transfer between document sets; generation is nondeterministic, so the artifact varies between runs of the same pipeline over the same input; and several quality judgements cannot be computed arithmetically and must be delegated to a language model whose behaviour is itself a source of measurement error. We contribute (i) a catalogue of twenty-seven automated checks organised into five tiers by prerequisite, with a result vocabulary that prevents an unrun check from being read as a clean one; (ii) a protocol that treats the LLM judge as a measurement instrument to be characterised rather than trusted, combining deterministic sampling, repeated queries with median aggregation, recorded dispersion, and human-validated conversion of judge output; (iii) a two-level reporting scheme: an audited pass count and a composite quality score with explicit admission criteria for its dimensions and non-compensatory aggregation across them; and (iv) reproducibility measurement across nondeterministic runs together with a pinned, versioned coverage matcher against a gold-standard reference. We report observations from a production deployment that motivated the principal design decisions, including the finding that the choice of concept matcher alone shifted measured coverage by 22% relative, exceeding the 14% run-to-run variation of the pipeline it was measuring.

1 Introduction

Agentic LLM pipelines can construct a business ontology from an enterprise document collection of policies, process descriptions, specifications, and reports in hours, where manual construction by domain experts and information architects is typically a project of months. Recent work has demonstrated LLM-driven ontology and terminology construction across domains [2], and such generated ontologies increasingly serve as the grounding layer for downstream AI systems: retrieval, generative business intelligence, and autonomous agents whose answer quality is bounded by the quality of the context they are given.

This creates an assessment problem that current practice does not address. The quality of a generated ontology is, almost without exception, reported by the party operating the pipeline, using measurements of its own choosing, applied once, with no statement of measurement error. Such claims cannot be independently interpreted, compared across versions, or audited. The purpose of this paper is to specify, in full, a methodology under which they can: what is

checked, how each result is graded, how the error of model-mediated measurements is handled, and, as importantly, what the assessment cannot show.

Three properties of the setting shape the methodology, and most of what follows is a consequence of one of them. First, *coverage is reference-relative*: what a pipeline missed can only be established against a statement of what should have been found, and that statement is written by hand for one document set and does not transfer to another. A reference is therefore a resource to be spent deliberately, and the remainder of the assessment is arranged to function without one. Second, *the output is nondeterministic*: two runs of the same pipeline over the same documents produce different ontologies: different concepts, different links, identifiers minted fresh each time. This variation is informative, so we measure it rather than design around it. Third, *some judgements are model-mediated*: whether a definition is well formed, or whether a quote supports a relationship, cannot be settled by arithmetic, so those checks put the question to a language model. A model is not exact in the way a count is; we therefore treat its behaviour as something to be measured rather than assumed: we ask more than once, record how far the answers move, and read every result against that recorded spread.

The paper makes four contributions:

1. A catalogue of twenty-seven automated checks organised into five tiers by what each check requires before it can run, which places an explicit cost boundary between checks that spend computation and checks that spend expert time (Section 4). The accompanying result vocabulary separates what a check *is*, how a result *scored*, and why a result is *absent*, so that a check nobody ran can never be read as a check that found nothing wrong (Section 3).
2. A protocol for using an LLM judge as a measurement instrument: deterministic sampling rules, repeated queries aggregated by median, per-ontology dispersion recorded rather than assumed, robustness to individual judge failures, and human-validated conversion of judge output into verdicts (Section 5).
3. A two-level reporting scheme that separates measurement from communication: an audited count of checks passed over the full catalogue, and a composite score over four dimensions with explicit admission criteria for what may enter a dimension and non-compensatory (geometric) aggregation across dimensions (Section 6).
4. A reproducibility measure defined over N nondeterministic runs, and a coverage protocol in which the matcher itself is treated as part of the measurement: pinned, versioned, fingerprinted onto every result, and extended only through an explicit human-promotion loop (Section 7).

Section 2 positions the framework against prior work. Section 8 reports the observations from a production deployment that motivated the principal design decisions. Section 9 states the framework’s limits, and Section 10 concludes with the calibration agenda: connecting the intrinsic measurements defined here to extrinsic, task-based measures of downstream AI quality.

2 Related Work

Ontology evaluation. Classical ontology evaluation is a mature field; surveys group its methods into gold-standard comparison, application-based evaluation, data-driven evaluation, and human assessment [3]. OntoClean [9] audits taxonomic decisions against formal metaproperties such as rigidity and identity; it is a methodology for expert analysis rather than an automated assessment, and it addresses the logical discipline of a taxonomy rather

than the linguistic quality or evidential support of its content. OQuaRE [6] adapts the SQuaRE software-quality model to ontologies, organising automatically computable metrics into characteristics and subcharacteristics; it is the closest antecedent to our dimensional score, but its metrics are structural and presuppose a single, stable artifact. OOPS! [19] catalogues common pitfalls detectable in hand-authored OWL ontologies and is the closest antecedent to our defect checks; its checks are deterministic and carry no notion of evidential grounding, because a hand-built ontology carries no evidence trail to ground against. None of these frameworks addresses the three properties that define our setting: a nondeterministic artifact, provenance and evidence as first-class subjects of assessment, and measurements that are themselves model-mediated.

Gold-standard evaluation of ontology learning. The ontology-learning literature established lexical and taxonomic precision and recall against a gold standard [5, 14], and task-based evaluation as an alternative ground truth [18]. These protocols treat the procedure that aligns candidate concepts to reference concepts as a neutral instrument. Our observations contradict that assumption in the generated-ontology setting: the choice of matcher alone moved measured coverage by more than the run-to-run noise of the pipeline being measured (Section 8). We therefore pin and version the matcher as part of the measurement itself.

LLM-as-judge. Using an LLM to grade generated text is now standard practice [23], and its failure modes are increasingly well documented: surveys catalogue systematic biases [8]; large-scale evaluation shows that judges can be highly self-consistent without being valid [16]; and rubric-based judging exhibits position bias, in which the order of criteria in the prompt shifts the grade [22]. This applies here directly, since our judge grades against ordered criterion lists. Self-consistency aggregation [21] improves stability by voting over repeated samples. This literature predominantly asks how good judges are in general. We take the complementary, metrological position: within one assessment framework, the judge is an instrument whose dispersion is measured per use, whose failures are contained rather than averaged away, and whose outputs are converted to verdicts only through rules validated against human annotation.

To our knowledge, no prior framework combines assessment of nondeterministically generated ontologies, evidence-grounding checks, and a model judge treated as a characterised instrument into a single, versioned methodology.

3 Preliminaries and Terminology

3.1 The artifact under assessment

The ontology we assess is a set of business concepts. Each concept carries a definition, optional synonyms, quotes from the source documents that support it (its *evidence*), and a record of which documents it came from (its *provenance*). Concepts are linked by edges of defined types, including a broader/narrower taxonomy. A pipeline of LLM agents reading a document set produces one such ontology per run.

3.2 Result vocabulary

The framework points a set of checks at an ontology; each check looks for one problem and returns a rate together with the list of concepts or links involved: the rate for tracking, the list for remediation. Because several of the framework’s terms are close enough to be mistaken for one another, Table 1 fixes the vocabulary once.

Two distinctions in Table 1 carry the weight. A *defect* is what a check looks for; *deficient* is how a result scored: a defect check that finds nothing is graded excellent. And *errored*,

Table 1: Result vocabulary. A check’s type is fixed; a grade describes one result; a verdict (or the reason for its absence) describes one run.

<i>Check types (fixed per check)</i>	
profile	Reports a plain quantity with no good or bad direction (e.g. concept count, hierarchy depth). Cannot pass or fail; it is the base rate against which the rest is read: three broken loops in an ontology of fifty concepts and three in one of fifty thousand are different situations.
defect	Reports something that is wrong: a concept with no definition, a link no document supports.
advisory	Reports something a person must judge: a concept under two categories may be an error or correct modelling. Never graded, because a grade would settle the very question the check exists to hand over.
<i>Grades (how one result scored against that check’s target b)</i>	
excellent	At most half the target.
adequate	Within the target.
deficient	Past the target, within twice it.
unacceptable	Worse than twice the target.
<i>Verdicts (what happened on this run, or why there is no result)</i>	
passing / failing	For a defect check, the pass mark sits between adequate and deficient. A verdict describes the ontology as it stands, not an event during a run.
skipped	The check did not apply, or an input it required was absent. The missing input is named.
errored	The check should have run and failed to. The error is named.
never assessed	The check exists and has no result for this version of the ontology.
<i>Summary statistics</i>	
checks passed	How many defect checks landed in the top two grades.
composite score	One number out of 100, aggregated from four dimensions (Section 6).

skipped, and *never assessed* are neither passes nor failures: a check nobody ran must never be read as a check that found nothing wrong. Every result is stored against the exact version of the ontology it was measured on.

4 The Assessment Framework

4.1 Tier structure

Each check is assigned to one of five tiers. A tier answers one question about the ontology, and the tiers are ordered by what must exist before their checks can run at all (Table 2). That ordering is the load-bearing part of the design: it determines where care is needed (T2), where cost concentrates (T4), and where the boundary between two kinds of cost falls.

Table 2: The five tiers: the question each answers, and its prerequisites beyond the ontology itself.

Tier	Question answered	Prerequisites
T0: Size and shape	How large is the ontology? Nothing here passes or fails.	
T1: Structure	Is it structurally sound: broken hierarchies, missing definitions?	
T2: Content quality	Is the content fit for use: definitions, names, and links supported by the documents?	LLM judge, embedding index
T3: Repeatability	How closely do independent runs of the pipeline agree? Nothing here passes or fails.	N pipeline runs
T4: Coverage	Does the ontology cover what it should, and what is missing?	Gold-standard reference; LLM judge, embedding index

The significant boundary falls between T3 and T4. Every check up to and including T3 requires no expert input: below the boundary a check spends computing time, above it a check spends an expert’s time writing and approving a reference. Everything below the boundary therefore runs on an ontology one did not build. A further asymmetry follows: checks below the boundary name the concepts at fault, while the checks above it name what is absent.

4.2 The check catalogue

The framework comprises twenty-seven checks; Appendix A lists the complete catalogue. T0 contributes seven profile measures of size and shape: concepts and links per type, disconnected components, hierarchy depth, hubs and isolates, the generating agent’s own stored definition scores, and the fraction of concepts carrying a supporting quote. None of these pass or fail. Tables 3 and 4 list the T1, T2, and T4 checks. The defect checks among them form the denominator of the pass count. Several checks function as diagnostics of the generating pipeline rather than faults of the ontology alone: a concept discussed in twelve documents to which the pipeline assigned no links is, in effect, a defect report against the relationship-extraction agent.

The structural checks in T1 use standard graph algorithms: taxonomy cycles are found with a strongly-connected-components pass [20], and a redundant parent link is one that a transitive reduction would remove [1]. The similarity checks in T2 compare each concept only with its nearest neighbours in a vector index [15] rather than with every other concept: the blocking step of record linkage, with an embedding index in place of a blocking key [7, 4].

Coverage results move when the pipeline is re-run, because re-running it over the same documents produces a different ontology. Whether a movement is an improvement or merely the variation that repeatability measures is a question for T3; it is not a reason to withhold the T4 verdict.

Table 3: T1 checks: structure, computed from the ontology alone.

Check	Reports	Type	Dimension
taxonomy cycles	a concept that is its own ancestor, which invalidates every upward traversal	defect	connectedness
transitive redundancy	a parent link already reachable through another parent	defect	
orphans	a business concept with no link to anything else	defect	connectedness
definition presence	a concept with no definition	defect	language
evidence presence	a concept citing no source quote at all	defect	traceability
identical definitions	two distinct concepts sharing one definition verbatim	defect	language
unenumerated code lists	a reference list that names no values	defect	
document zero yield	a source document that produced no concepts	defect	
multiple parents	a concept under two categories, often deliberate	advisory	

Table 4: T2 and T4 checks: language, evidence, and coverage.

Check	Reports	Type	Dimension
<i>T2: language and evidence; requires a model or a search index</i>			
definition quality	definitions failing the seven criteria C1–C7	defect	language
name quality	names failing the seven criteria N1–N7	defect	language
definition score	the assessment grade disagrees with the writing agent’s own stored grade	defect	
divergence	a link that no quote in either concept’s evidence supports	defect	traceability
relationship grounding	two concepts that appear to denote the same thing	advisory	
near duplicates	the same kind of entity filed under two different types	advisory	
type consistency	a name that appears misspelled, frequently the source document’s own spelling	advisory	
suspected typo	one concept split into numbered slots (<i>Address Line 1..4</i>)	advisory	
numbered variants			
<i>T4: coverage; requires an expert reference</i>			
concept coverage	expected concepts the pipeline found, by name	defect	coverage
concept coverage, judged	the same, plus judge-proposed pairs awaiting a human ruling	advisory	
link coverage	expected links both of whose concepts were found	defect	coverage

4.3 The reporting unit

Each check examines one ontology at one moment and returns a rate together with the concepts or links behind it, so that a reader can inspect the individual cases. Every rate states its denominator, and denominators are defined once, globally: changing the population changes the number even when the ontology has not changed. Zero problems out of four concepts and zero out of four hundred are the same rate and very different evidence; Section 6 describes how thin denominators cap a grade.

5 The LLM Judge as a Measurement Instrument

Several T2 and T4 checks re-grade definitions and names against seven criteria each, or ask whether a quote supports a link. The criteria (Table 5) are our own, developed from the terminology and metadata-registry standards [10, 11, 12] rather than copied from them: those standards give principles for writing definitions and for developing naming conventions, not a list that can be graded against. Each criterion is graded on its own, so a result names the specific rule a failing definition breaks.

Table 5: The seven definition criteria and seven naming criteria, graded one by one.

A definition must		A name must	
C1	state what the concept is, not how it is used	N1	be singular or plural as the language expects
C2	be one short statement	N2	denote a thing, not an action or a sentence
C3	not repeat the term itself	N3	omit articles and filler
C4	be intelligible without its source document	N4	be capitalised as the language expects
C5	exclude examples and reasoning	N5	be recognisable without a glossary
C6	use vocabulary a business reader knows	N6	carry no stray punctuation
C7	give a category and a differentiator	N7	identify the concept, not merely its type

The protocol below treats the judge as an instrument to be characterised, not trusted. Models used as evaluators have documented biases [8, 23], one of which reaches this design directly: a judge grading against an ordered list favours some positions in the list over others [22], so the ordering of the seven criteria may itself move a grade. We have not yet audited our judge for this effect; Section 10 places it in future work.

Sampling is deterministic. Samples are selected by a stated rule, never at random. The repetition protocol below re-asks the same questions to measure the judge’s dispersion; a random sample would put different concepts in front of each repeat, and the resulting spread would measure our sampling rather than the judge.

One failed judgement must not invalidate the check. A judge that fails on an individual item with probability p fails on at least one of n items with probability $1 - (1 - p)^n$, which approaches certainty long before n reaches the size of a realistic audit. A failed item is therefore dropped, the rate is computed over what was actually graded, and the number of dropped items is reported: an item that could not be graded is not an item that passed.

Queries are repeated; the median is reported. Each judged check is run an odd N times and the median result reported. Findings are matched across repeats by the concepts they name rather than by wording, and each finding carries its recurrence, “found in k of N runs”: a triage ordering for the reader, and the raw material for a majority vote at $k > N/2$ [21] should one be wanted. One caution from the reliability literature applies verbatim: repetition

makes a measurement more repeatable, not more correct. A judge that misreads a rule the same way every time agrees with itself perfectly [16].

Observed dispersion is attributed to its source. Queries are issued at temperature zero, so an identical question returns an identical answer. Where a check reaches the judge through an approximate-nearest-neighbour index, however, two repeats are asked slightly different questions; the disagreement recorded is then the index’s, not the judge’s, and misattributing it sends an engineer to repair the wrong component. In one production run the concept matcher proposed twenty-two candidate pairs across three repeats, seventeen of which appeared in the median repeat (Section 8).

Input quality interacts with judgement. A vague definition is worse for the judge than no definition. Asked whether two concepts were identical given names alone, our judge answered correctly; given two vague definitions of the same pair, it did not. Evidence a judge cannot use is not neutral; it displaces the evidence the judge could have used.

Dispersion is context-bound. The spread across repeats is the baseline against which a later change is compared. It shrinks as $1/\sqrt{n}$, so it belongs to a check run on a particular ontology, never to the check in the abstract. Where it was never measured, it is recorded as unknown rather than zero.

6 Scoring and Aggregation

6.1 Grading

Each check declares a single number: its target b . The four grade boundaries follow by halving the distance on the good side and doubling it on the bad side. For a lower-is-better measure m : excellent at $m \leq b/2$, adequate at $m \leq b$, deficient at $m \leq 2b$, unacceptable above, mirrored where higher is better. A check that tolerates only zero has $b/2 = 2b = 0$ and collapses to pass-or-fail by that arithmetic alone, with no special rule. A grade is capped when too few items were judged: weak evidence makes a result not-the-best, never bad.

6.2 The two summary statistics

The first summary is a count: k of n defect checks passed, where n is the full catalogue rather than the checks that happened to run (otherwise two passes out of fourteen reads as “2 of 2”), and checks that could not run form a third, named group, neither passing nor failing.

The second is a composite score out of 100 over four dimensions: *coverage* (was anything missed), *language* (are the concepts written well enough to use), *connectedness* (is the ontology joined up), and *traceability* (can a claim be traced to a document). The count is a measurement; the score is a communication device; the two are never merged.

6.3 Admission of a check into a dimension

Most checks contribute to no dimension. A dimension must be legible, must have room to move, and must not overlap another; the set is kept small because a geometric mean over a dozen components renders every component negligible. Three tests govern admission, and none of them is topical affinity.

First, the check’s population must be disjoint from those of the dimension’s other checks, so no fault is charged twice. Coverage takes both halves of recall on exactly this basis: an ontology is vocabulary plus relations, and a reference link whose concept was never found is

excluded from the link denominator rather than counted as a second miss; that conditioning is what makes counting both halves sound.

Second, the check must move in the same direction as the property. Transitive redundancy stays out of connectedness although it is a structural check: deleting links raises the orphan rate while lowering redundancy, so the two would cancel and the dimension would measure less than either check alone.

Third, the check must have room to move, and this settles not whether it enters but *how*. Within a dimension,

$$d = g \cdot \prod_t (1 - r_t), \quad (1)$$

where g is the mean of the checks that vary in practice and r_t is the rate of each check that should never fire. Averaging-in a check that is almost always clean contributes a near-constant hundred and drowns the informative check beside it. On one measured ontology this contributed nineteen points of connectedness the graph had not earned (Section 8). As a multiplicative factor, such a check contributes nothing when clean and penalises in proportion when it fires; it can only ever cost. A target at or near zero is the marker that assigns a check to this factor form: the framework has already declared that it ought to read clean.

6.4 Weights and cross-dimension aggregation

Weights within a dimension are equal, because no calibration data exists to justify anything else and equal weighting asserts least. Coverage is the single exception (0.7 on concepts, 0.3 on links), and the link half counts only when its population is not thin, since it is measured over reference links both of whose concepts were found, which on a poor ontology is a handful. “Thin” is the same population floor the grading already applies (Section 4), so this gauge falls silent by a rule everything else obeys.

Across dimensions the score is a weighted geometric mean,

$$S = \prod_i v_i^{w_i/W}, \quad W = \sum_i w_i, \quad (2)$$

following the composite-indicator literature’s treatment of non-compensatory aggregation [17]. The choice of arithmetic within a dimension and geometric across dimensions is deliberate. Within a dimension, checks are substitutable evidence about one property: missing definitions and weak definitions are both “the definitions are poor”, so one may reasonably offset another. Across dimensions they are not substitutable: no quantity of good writing compensates for having found nothing, which is precisely the argument for a geometric mean and would be undone by an arithmetic one. The form also lets a dimension survive a check that could not run; under an earlier one-check-per-dimension design, a single skip deleted the dimension entirely. A measured zero is fatal to the score, except where a dimension declares an explicit floor, because *unverifiable* is severe yet survivable in a way that *meaningless* is not. A dimension that could not be measured at all is dropped and the weights rebalanced, which is not a zero and must never be read as one.

7 Measuring Reproducibility and Coverage

7.1 Reproducibility

Repeatability compares several ontologies between which nothing lines up: identifiers are generated fresh on every run, so this is a computation over N ontologies rather than a check on one. Concepts are matched on meaning (name, definition, synonyms, and evidence quotes together) and one-to-one, since three concepts of one run matching a single concept of another

indicates that the second run merged three. For each pair of runs we take agreed concepts over concepts either run found (the Jaccard index [13]), averaged over all pairs:

$$\bar{a} = \binom{N}{2}^{-1} \sum_{i < j} \frac{|M_{ij}|}{|R_i \cup R_j|}, \quad (3)$$

where R_i is the concept set of run i and M_{ij} the matched pairs between runs i and j . Only pairwise agreement remains stable as N changes. Anything generated mechanically (links derived from folder paths, for example) repeats perfectly and is excluded before the calculation, since including it would inflate agreement with content that carries no information about the pipeline.

7.2 Coverage: the matcher is part of the measurement

Coverage compares the generated ontology against a gold-standard reference, and the alignment procedure is not a neutral instrument: over one fixed pair of ontologies, matcher choice alone moved measured coverage from 0.444 to 0.542 (22% relative) against 14% actual run-to-run pipeline variation (Section 8). The matching procedure is therefore pinned: an ordered ladder of rules, each individually named and declared: exact match, the curator’s canonical name, either side’s synonyms, acronym expansion, and singular/plural equivalence. The ladder’s version is fingerprinted onto every result. Embedding similarity is excluded from the pinned ladder by design, having matched *Internal Audit* to *External Audit*.

7.3 Two coverage numbers, one of which scores

The pinned matcher systematically under-counts: a synonym the source documents never state cannot appear in the reference’s synonym list, so a genuine hit is recorded as a miss. Against those recorded misses, an embedding index proposes near neighbours and a judge classifies the closest into one of seven relations: *same*, *narrower*, *broader*, *part*, *attribute*, *related*, *unrelated*. Only *same* constitutes a match, and the framework’s code derives that from the relation label rather than from the judge’s own yes-or-no answer, which proved wrong in a way the label was not: asked directly whether two concepts were the same, the judge answered *yes* for a pair it had simultaneously labelled *part-of*. Over ninety hand-annotated pairs, the relation label was correct in every case and the direct verdict was not (Section 8).

The judged number never reaches the score. It is a list of questions, and a person answers them: an accepted pair is promoted to a rung of the pinned matcher, found deterministically from the next run onward with no model involved, and counted separately from the exact matches, so the share of a coverage figure that rests on human judgement remains visible. The human rulings are hashed onto every result for the same reason the ladder version is. This is the mechanism by which an estimate becomes evidence without ever being weighted as such: by the time it counts, a person has said so.

8 Observations from a Production Deployment

The framework was developed against, and is operated on, ontologies produced by our generation pipeline from the document sets of a large European industrial enterprise, assessed over repeated pipeline runs. The observations in Table 6 are working observations from that deployment, not controlled experiments; each is reported because it decided a design question, and Section 9 discusses their evidential weight.

Table 6: Production observations and the design decisions they motivated.

Observation	Measurement	Decision motivated
Matcher choice dominates pipeline noise	Over one fixed ontology pair, matcher choice moved coverage $0.444 \rightarrow 0.542$ (22% relative); run-to-run pipeline variation on the same measure was 14%	Pin and version the matcher; fingerprint it onto every result (§7)
ANN retrieval injects dispersion that is not the judge’s	The concept matcher proposed 22 candidate pairs across three repeats; 17 appeared in the median repeat, at temperature zero throughout	Attribute dispersion to its source before reading it as judge disagreement (§5)
The judge’s relation label outperforms its direct verdict	Over 90 hand-annotated candidate pairs, the seven-way relation label was correct in all cases; the simultaneous direct yes/no was not (e.g. <i>yes</i> on a pair labelled <i>part-of</i>)	Derive match verdicts from the label in code; never use the judge’s own verdict (§7)
Averaging-in an always-clean check inflates a dimension	On one measured ontology, arithmetic inclusion of a near-constant check contributed 19 points of connectedness score not reflected in the graph	Factor form $\prod(1 - r_t)$ for should-never-fire checks (Eq. 1)
Vague definitions degrade judgement below the name-only baseline	Identity questions answered correctly from names alone were answered incorrectly when two vague definitions of the same pair were supplied	Treat unusable evidence as displacement, not noise; grade definition quality as its own defect (§5)

9 Limitations and Threats to Validity

A measurement that does not state its limits invites being read past them. The following are the limits we know, and each is a reason the framework is built as it is rather than an apology for it.

Repeatable is not correct. A pipeline that is wrong the same way every time agrees with itself perfectly. Low agreement demonstrates a problem; high agreement is merely consistent with correctness and never evidence of it [16]. Repetition narrows variance and never bias; detecting bias requires a person reading the findings. This is why repeatability (T3) reports a number and grades nothing.

A stable score is not a stable finding set. A concept sitting near a grade boundary falls on either side of it under repetition without moving the total, so two equal scores can name different faults. Comparisons must be made over the findings, not only over the figure.

The framework measures intrinsic quality only. Nothing here establishes which faults damage the answers of the downstream systems the ontology grounds. That requires testing the ontology on the tasks it was built to serve (extrinsic evaluation in the sense of [18]), which is a distinct exercise and not a substitute for this one in either direction. Section 10 makes this the first item of future work.

Several targets are judgement rather than calibration. The targets b are versioned and fingerprinted, so any figure states which target produced it and two figures under different targets cannot be silently compared. But a versioned judgement is still a judgement, and we have not yet measured what better targets would be.

The judge is not audited for position bias. Rubric-order effects are documented for judges of this kind [22] and our criteria are presented as ordered lists; the audit (randomised criterion order over a fixed sample) has not yet been run.

The evidential base is one deployment. The observations of Section 8 decided design questions but were made in a single production context. They should be read as existence proofs rather than effect-size estimates: a matcher *can* out-move the pipeline, and a direct verdict *can* contradict its own label.

10 Conclusion and Future Work

We have specified an assessment methodology for LLM-generated business ontologies under the three conditions that make the setting non-classical: reference-relative coverage, nondeterministic generation, and model-mediated judgement. The framework’s design reduces to a small number of commitments: checks are typed, tiered by prerequisite, and versioned; absence of a result is never read as absence of a problem; the model judge is an instrument whose dispersion is measured per use and whose outputs are converted to verdicts only through human-validated rules; measurement (the pass count) and communication (the composite score) are kept separate; and every figure carries the fingerprint of the exact matcher, targets, and rulings that produced it.

Three lines of future work follow directly from the limitations.

Extrinsic calibration. The framework’s most consequential open question is predictive validity: which intrinsic measurements, at which thresholds, predict the quality of what the ontology is used for. The planned study correlates the four dimension scores and individual defect rates with task-based, results-driven measures on the same deployments: the accuracy of generative business-intelligence outputs grounded in the ontology, the answer quality of retrieval-grounded AI assistants, and multi-hop question-answering performance. The outcome would replace judgement-set targets b with calibrated ones, and would identify checks whose findings do not affect downstream quality and can be demoted to advisories.

Judge auditing. The position-bias audit of Section 9 (randomising criterion order over a fixed, deterministically sampled set) is mechanical to run under the existing repetition protocol, and its result belongs in the recorded dispersion of every judged check.

Broader validation. The observations that motivated the design should be re-measured across deployments and document domains, converting existence proofs into effect sizes, and accumulating the cross-deployment data from which calibrated targets can be derived.

Conflict of Interest

The authors are involved in Metagem BV, a company that builds the agentic ontology-generation pipelines whose output this framework assesses. The framework was developed in that context and the observations of Section 8 were made on the company’s own deployments. This is precisely the self-assessment condition described in Section 1; the methodology’s versioning, fingerprinting, and stated limits exist so that its results can be scrutinised, reproduced, and challenged by parties without that interest.

A The Full Check Catalogue

Table 7 lists all twenty-seven checks, including the seven T0 profile measures the body summarises. “Graded” and “counted” are not columns: a defect is both, while an advisory and a profile are neither, so the type already determines them.

References

- [1] Alfred V. Aho, Michael R. Garey, and Jeffrey D. Ullman. The transitive reduction of a directed graph. *SIAM Journal on Computing*, 1(2):131–137, 1972.
- [2] Hamed Babaei Giglou, Jennifer D’Souza, and Sören Auer. LLMs4OL: Large language models for ontology learning. In *Proceedings of the 22nd International Semantic Web Conference (ISWC)*, 2023.
- [3] Janez Brank, Marko Grobelnik, and Dunja Mladenić. A survey of ontology evaluation techniques. In *Proceedings of the Conference on Data Mining and Data Warehouses (SiKDD)*, 2005.
- [4] Peter Christen. *Data Matching: Concepts and Techniques for Record Linkage, Entity Resolution, and Duplicate Detection*. Springer, 2012.
- [5] Klaas Dellschaft and Steffen Staab. On how to perform a gold standard based evaluation of ontology learning. In *Proceedings of the 5th International Semantic Web Conference (ISWC)*, pages 228–241, 2006.

Table 7: Every check, by tier.

Check	Reports	Type	Dimension
<i>T0: size and shape; nothing here passes or fails</i>			
concepts	how many, and how many of each type	profile	
links	how many, and how many of each type	profile	
disconnected groups	how many islands the ontology falls into	profile	
hierarchy depth	how deep broader/narrower goes	profile	
hubs and isolates	the most-linked concept, and the un-linked ones	profile	
stored definition scores	the writing agent’s own grade for its definitions	profile	
quote coverage	how many concepts carry a supporting quote	profile	
<i>T1: structure, from the ontology alone</i>			
taxonomy cycles	a concept that is its own ancestor	defect	connectedness
transitive redundancy	a parent link already reachable another way	defect	
orphans	a business concept with no link to anything else	defect	connectedness
definition presence	a concept with no definition	defect	language
evidence presence	a concept citing no source quote at all	defect	traceability
identical definitions	two concepts sharing one definition verbatim	defect	language
unenumerated code lists	a reference list that names no values	defect	
document zero yield	a source document that produced no concepts	defect	
multiple parents	a concept under two categories, often deliberate	advisory	
<i>T2: language and evidence, needing a model or a search index</i>			
definition quality	definitions failing the seven criteria (C1–C7)	defect	language
name quality	names failing the seven naming criteria (N1–N7)	defect	language
definition score divergence	our grade disagrees with the writing agent’s own	defect	
relationship grounding	a link no quote in either concept’s evidence supports	defect	traceability
near duplicates	two concepts that appear to mean the same thing	advisory	
type consistency	the same kind of thing filed under two different types	advisory	
suspected typo	a name that looks misspelled	advisory	
numbered variants	one concept split into numbered slots	advisory	
<i>T4: coverage, needing an expert reference</i>			
concept coverage	expected concepts the pipeline found, by name	defect	coverage
concept coverage, judged	the same, plus pairs proposed for a human ruling	advisory	
link coverage	expected links whose two concepts were both found	defect	coverage

- [6] Astrid Duque-Ramos, Jesualdo Tomás Fernández-Breis, Robert Stevens, and Nathalie Aussenac-Gilles. OQuaRE: A SQuaRE-based approach for evaluating the quality of ontologies. *Journal of Research and Practice in Information Technology*, 43(2):159–176, 2011.
- [7] Ivan P. Fellegi and Alan B. Sunter. A theory for record linkage. *Journal of the American Statistical Association*, 64(328):1183–1210, 1969.
- [8] Jiawei Gu, Xuhui Jiang, Zhichao Shi, Hexiang Tan, Xuehao Zhai, Chengjin Xu, Wei Li, Yinghan Shen, Shengjie Ma, Honghao Liu, Saizhuo Wang, Kun Zhang, Yuanzhuo Wang, Wen Gao, Lionel Ni, and Jian Guo. A survey on LLM-as-a-judge. *arXiv preprint arXiv:2411.15594*, 2024.
- [9] Nicola Guarino and Christopher A. Welty. Evaluating ontological decisions with OntoClean. *Communications of the ACM*, 45(2):61–65, 2002.
- [10] ISO/IEC 11179-4: *Information technology — Metadata registries — Part 4: Formulation of data definitions*. International Organization for Standardization, second edition, 2004.
- [11] ISO/IEC 11179-5: *Information technology — Metadata registries — Part 5: Naming principles*. International Organization for Standardization, third edition, 2015.
- [12] ISO 704: *Terminology work — Principles and methods*. International Organization for Standardization, fourth edition, 2022.
- [13] Paul Jaccard. Étude comparative de la distribution florale dans une portion des Alpes et du Jura. *Bulletin de la Société Vaudoise des Sciences Naturelles*, 37:547–579, 1901.
- [14] Alexander Maedche and Steffen Staab. Measuring similarity between ontologies. In *Proceedings of the 13th International Conference on Knowledge Engineering and Knowledge Management (EKAW)*, pages 251–263, 2002.
- [15] Yury A. Malkov and Dmitry A. Yashunin. Efficient and robust approximate nearest neighbor search using hierarchical navigable small world graphs. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 42(4):824–836, 2020.
- [16] Justin D. Norman, Michael U. Rivera, and D. Alex Hughes. Reliability without validity: A systematic, large-scale evaluation of LLM-as-a-judge models across agreement, consistency, and bias. *arXiv preprint arXiv:2606.19544*, 2026.
- [17] OECD and Joint Research Centre, European Commission. *Handbook on Constructing Composite Indicators: Methodology and User Guide*. OECD Publishing, 2008.
- [18] Robert Porzel and Rainer Malaka. A task-based approach for ontology evaluation. In *Proceedings of the ECAI 2004 Workshop on Ontology Learning and Population*, 2004.
- [19] María Poveda-Villalón, Asunción Gómez-Pérez, and Mari Carmen Suárez-Figueroa. OOPS! (OntOlogy Pitfall Scanner!): An on-line tool for ontology evaluation. *International Journal on Semantic Web and Information Systems*, 10(2), 2014.
- [20] Robert Tarjan. Depth-first search and linear graph algorithms. *SIAM Journal on Computing*, 1(2):146–160, 1972.
- [21] Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models. In *International Conference on Learning Representations (ICLR)*, 2023.

- [22] Yuzheng Xu, Tosho Hirasawa, Tadashi Kozuno, and Yoshitaka Ushiku. Am I more pointwise or pairwise? Revealing position bias in rubric-based LLM-as-a-judge. *arXiv preprint arXiv:2602.02219*, 2026.
- [23] Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric P. Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. Judging LLM-as-a-judge with MT-Bench and Chatbot Arena. In *Advances in Neural Information Processing Systems (Datasets and Benchmarks)*, 2023.