Reliable Context

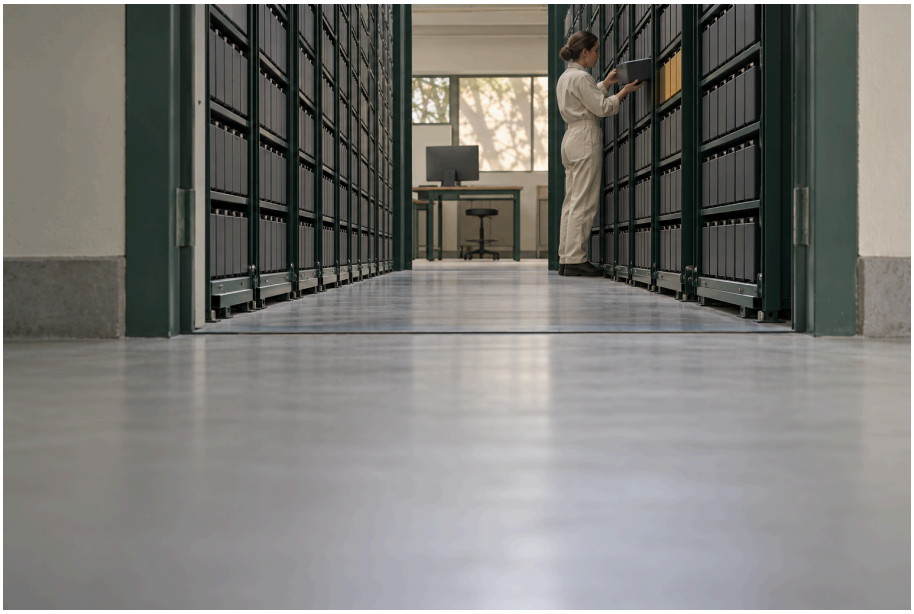
RELIABLE CONTEXT / RESEARCH & PRACTICAL GUIDANCE

OWL import closure: Can an old graph be reinterpreted?

> Reliable Context is a kernel of truth: a system of record for organizations that fuse private and public information and must defend every value they act on. One canonical record per real thing, a cited source and an observation date behind every value, unknowns declared rather than left blank, and any past state of the record reproducible.

Published by Reliable Context · 2026-10-31 · owl import closure · owl imports · ontology versioning · version iri · knowledge graph reproducibility · ontology dependency management · reproducible reasoning · historical graph replay

Original article: https://reliablecontext.com/articles/owl-import-closure-graph-replay-test



In this article

1. Executive Summary
2. Introduction and Background
3. Key Changes
4. Dependency manifest and retained artifacts
5. Implementation Considerations and Process Changes
6. Replay outcomes and acceptance boundaries
7. Version policy and consumer decisions
8. Data Analysis and Evidence
9. Implications and Future Directions
10. Conclusion

Executive Summary

OWL import closure is the root ontology plus every ontology reached through transitive imports. An unchanged consumer graph can therefore receive a different interpretation when a downstream, unversioned import resolves to changed axioms. The World Wide Web Consortium (**W3C**) **OWL 2 Recommendation of December 11, 2012** distinguishes ontology and version identities. A consumer must still retain the artifacts needed to reproduce an earlier query. (www.w3.org)

The practical answer is conditional: **an old graph can be reinterpreted**, and a historical reconstruction is defensible only within an explicitly verified scope. This report proposes recording requested import IRIs, resolved document locations, declared ontology and version IRIs, retrieval times, content digests, and parent edges. Replay should resolve every recorded transitive dependency, compare bytes and axiom sets, then evaluate specified entailments and query results. Digests support integrity checks; a structural comparison can distinguish serialization differences. These controls are proposed methods, not a content-hash requirement in OWL. (csrc.nist.gov) (robot.obolibrary.org) (force11.org)

A **synthetic two-hop fixture** demonstrates the distinction. The consumer C imports A; A imports B through a moving IRI. With all **2/2 imports resolved** and **3 documents** in the closure, replacing B/1 with B/2 changes **1/3 baseline entailments** and introduces a different membership. Replaying retained B/1 restores **0/3 changed baseline entailments**. These are transparent fixture calculations, not performance benchmarks. Matching selected outputs establishes the tested result; W3C explicitly cautions that passing its conformance tests does not prove conformance. (www.w3.org)

Acceptance must distinguish complete reconstruction, observed differences, and **unknown results caused by missing evidence**. Successful loading is insufficient: documented tool configurations can omit imports or tolerate missing dependencies. OBO Foundry's stronger immutability and retrieval requirements apply to its officially released ontologies, rather than all OWL publications. The consumer should pin a suitable version IRI, preserve a permitted snapshot, or reject an unsupported historical replay claim. Artifact availability and result validation remain separate checks, and query behavior needs its recorded entailment regime. (jena.apache.org) (owlcs.github.io) (obofoundry.org) (www.acm.org) (www.w3.org)

Introduction and Background

An approved knowledge graph can preserve its asserted triples while losing the evidence needed to reconstruct their earlier interpretation. The issue is especially visible when a consumer imports an ontology that itself imports a moving dependency. Cuenca Grau and colleagues (2007) identify imported ontology evolution outside the consumer's control as a modularity concern. Botoeva and colleagues (2018) explain that safe replacement depends on the application being preserved. Historical reconstruction therefore needs an explicit object of comparison: documents, axioms, entailments, or query answers. (ijcai.org) (arxiv.org)

For Reliable Context's audience of ontology consumers, graph stewards, and audit reviewers, the decision is concrete: can the organization reproduce the inputs and interpretation behind an earlier approval? This report evaluates an observable bundle of ontology documents, resolution records, reasoning settings, and expected outputs. Its manifest and replay procedure are proposed evaluation methods. They are not additional requirements of the Web Ontology Language (**OWL**) standard.

Knowledge graph reproducibility has several possible scopes. Reproducing a file establishes an input property. Reproducing an entailment establishes a logical consequence under stated semantics. Reproducing a query additionally depends on the selected dataset and evaluation configuration. The Association for Computing Machinery (**ACM**) separately treats artifact availability and validation of results, a useful precedent for keeping these evidence categories distinct. (www.acm.org) (jena.apache.org)

A **graph snapshot**, as used here, is a retained representation of the consumer's asserted data and relevant dataset arrangement at approval time. It is an operational definition, not an OWL version identifier. The accompanying dependency snapshot preserves the ontology documents that supplied its interpretation. FORCE11's data citation principles similarly distinguish the provenance and fixity needed to verify a cited version from the archive's responsibility for persistence. (force11.org)

The analysis is anchored to sources accessed on the publication date, October 31, 2026. No search-volume estimate, deployment statistic, or reasoning-speed benchmark is used. The quantitative evidence is a fully specified synthetic fixture, with its arithmetic and limits reported below.

Key Changes

The relevant change is an upstream dependency becoming different while the consumer remains unchanged. Understanding that event requires separating identifiers, locations, and logical dependencies.

Identity and location are different observations

An **Internationalized Resource Identifier (IRI)** names an ontology; an optional **version IRI** identifies a particular version. An ontology document physically represents the ontology. A **Uniform Resource Locator (URL)** used to retrieve that document is a further observation. These roles can coincide in a deployment, but the evidence record should distinguish them. (www.w3.org)

Table 1 separates the identities and artifacts a historical replay must account for.

Item	Meaning	Replay evidence
Ontology IRI	Identifier associated with an ontology series.	Record the identifier declared inside the document, alongside the requested identifier. (protegewiki.stanford.edu)
Version IRI	Identifier of a version within that series.	Record the declared version and compare it with the intended release. (protegewiki.stanford.edu)
Document location	Retrieval or mapped location of a concrete document.	Retain requested and resolved locations; OWLAPI explicitly maps ontology IRIs to document IRIs. (owlcs.github.io)
Direct import	An immediate dependency of the consuming ontology.	Preserve each parent-to-dependency edge; Protégé distinguishes direct and indirect imports. (protegeproject.github.io)
Import closure	The root ontology together with every transitively imported ontology.	Inventory the root and the recursively reached dependencies. (www.w3.org)
Graph snapshot	This report's retained asserted-data artifact and dataset arrangement.	Keep its bytes, contextual metadata, and associated query separately from ontology identity. (www.researchobject.org)

The table establishes distinct comparison targets. A matching ontology IRI does not by itself establish a matching document. Likewise, an unchanged root artifact does not establish that its dependencies were reconstructed.

A moving target can be expected behavior

OWL recommends updating an ontology IRI's target for a new release while retaining earlier versions at their version IRIs. This current-version behavior is deliberate. (www.w3.org)

Normative wording matters. **MUST** is an absolute requirement; **SHOULD** allows justified exceptions. A consumer should not convert a retrieval recommendation into evidence that a particular historical artifact remains available. That availability must be checked. (www.rfc-editor.org)

Transitive dependencies determine the comparison boundary

The logical imports relation is transitive. Saving a root and its immediate dependency leaves the reconstruction incomplete if another dependency is reached through that import. The import closure also should not mix different versions of the same ontology series or the specified incompatible ontologies. (www.w3.org)

For a consumer, these observations suggest separate questions: which dependencies were expected, which documents were loaded, which declared identities were found, and which consequences followed? They should remain separately answerable even when the overall replay is rejected.

Dependency manifest and retained artifacts

A **dependency manifest** should connect every requested import to the actual artifact used. The following fields are this report's proposal. They combine retrieval observations, declared ontology identity, artifact integrity, and dependency structure. Dublin Core defines provenance through custody and ownership history

relevant to interpretation; Research Object Crate (**RO-Crate**) provides a related framework for aggregating data and contextual descriptions. Neither source defines this OWL replay manifest. (www.dublincore.org) (www.researchobject.org)

- **Requested IRI:** Preserve the literal target appearing in the parent's import declaration.
- **Resolved document URL:** Record the location actually used after network redirects or local mappings.
- **Declared ontology IRI:** Extract identity from the loaded document rather than inferring it from the filename.
- **Declared version IRI:** Store the declaration, or an explicit reason when it is absent.
- **Retrieval time:** Record Coordinated Universal Time (**UTC**) and distinguish original observation from later replay.
- **Content digest:** Compute a named digest over the exact retained bytes and preserve those bytes.
- **Parent edges:** Retain every importing parent, including multiple paths to a shared dependency.
- **Resolution evidence:** Keep redirect observations, catalog mappings, media type, and retrieval or parsing failures.

Hypertext Transfer Protocol (HTTP) distinguishes a redirect's Location target from Content-Location, which identifies a resource corresponding to the enclosed representation. A changed redirect destination is consequently a retrieval observation requiring comparison, rather than sufficient evidence of changed ontology axioms. The w3id service likewise explicitly distinguishes redirection from hosting files. (www.rfc-editor.org) (w3id.org)

The JavaScript Object Notation (**JSON**) template below records the synthetic dependency chain (Hypothetical Example) introduced in the evidence section. Every identifier, example.invalid location, and timestamp is illustrative. Replace the digest placeholders with calculated values; they are not claimed hashes of fetched artifacts.

```
{
  "root": "C",
  "expectedImportDocuments": 2,
  "documents": [
    {
      "id": "C",
      "requestedIri": "urn:fixture:C/1",
      "resolvedDocumentUrl": "https://example.invalid/C-1.ofn",
      "ontologyIri": "urn:fixture:C",
      "versionIri": "urn:fixture:C/1",
      "retrievedAt": "2026-01-15T12:00:00Z",
      "digest": {"algorithm": "SHA-256", "value": "<digest-of-C-bytes>"},
      "snapshot": "C-1.ofn",
      "parentEdges": []
    },
    {
      "id": "A1",
      "requestedIri": "urn:fixture:A/1",
      "resolvedDocumentUrl": "https://example.invalid/A-1.ofn",
      "ontologyIri": "urn:fixture:A",
```

```
      "versionIri": "urn:fixture:A/1",
      "retrievedAt": "2026-01-15T12:00:00Z",
      "digest": {"algorithm": "SHA-256", "value": "<digest-of-A-bytes>"},
      "snapshot": "A-1.ofn",
      "parentEdges": [{"parent": "C", "requestedIri": "urn:fixture:A/1"}]
    },
    {
      "id": "B1",
      "requestedIri": "urn:fixture:B",
      "resolvedDocumentUrl": "https://example.invalid/B-current.ofn",
      "ontologyIri": "urn:fixture:B",
      "versionIri": "urn:fixture:B/1",
      "retrievedAt": "2026-01-15T12:00:00Z",
      "digest": {"algorithm": "SHA-256", "value": "<digest-of-B1-bytes>"},
      "snapshot": "B-1.ofn",
      "parentEdges": [{"parent": "A1", "requestedIri": "urn:fixture:B"}]
    }
  ]
}
```

Secure Hash Algorithm 256 (SHA-256) supplies a byte-integrity check. The National Institute of Standards and Technology (**NIST**) describes message digests as a way to detect changed messages. A digest does not itself retrieve a document, establish its origin, or test semantic equivalence; the proposed procedure uses it alongside the retained artifact and provenance. (csrc.nist.gov) (force11.org)

Software Heritage's intrinsic identifiers illustrate the related idea of computing identity from an artifact. That is an analogy for content identity, not an assertion that Software Heritage identifiers or content hashes are part of OWL. (www.softwareheritage.org)

Implementation Considerations and Process Changes

The historical replay should use an explicit resolver configuration instead of allowing unidentified dependencies into the candidate closure. **Extensible Markup Language (XML)** catalogs provide one documented mechanism: ROBOT, OWLAPI, and Protégé use them to locate imports, and Protégé documents catalogs that describe import redirection. (robot.obolibrary.org) (protegewiki.stanford.edu)

Reconstruct the dependency boundary before reasoning

- **Freeze the baseline:** Retain the approved root, dependency artifacts, manifest, query text, and expected outputs as one reviewable evidence package.
- **Traverse recursively:** Enumerate dependencies from each loaded document, preserving edges and deduplicating only after document identities have been examined.
- **Resolve deliberately:** Map historical requests to retained artifacts or verified version-specific artifacts; record the mapping used for each request.
- **Check completeness:** Compare observed dependencies with the recorded inventory, and report missing, unexpected, and identity-mismatched documents separately.

- **Verify bytes:** Compare digests before parsing so that formatting changes remain distinguishable from later structural comparisons.
- **Compare axioms:** Identify additions, removals, declarations, annotations, and import changes with an explicit comparison scope.
- **Run configured reasoning:** Record the parser, reasoner, versions, supported semantics, settings, and output-generation choices.
- **Compare observations:** Evaluate the baseline entailment predicates and query outputs, retaining new results as well as lost baseline results.

These are proposed replay controls. They address documented tool behavior that a reviewer cannot infer from a successful process exit. For example, OWLAPI exposes missing-import handling, including a silent strategy; Protégé may display an unresolved import IRI without a loaded ontology. (owlcs.github.io) (protegeproject.github.io)

The current Apache Jena ontology application programming interface (**API**) documentation states that imports are not automatically loaded by the ordinary reading operation. ROBOT query similarly requires an explicit option to include imports as named graphs. The replay record should therefore show the effective loaded dataset, rather than treating an import declaration as proof that the dependency participated. (jena.apache.org) (robot.obolibrary.org)

Do not deduplicate solely by declared IRI. If different documents claim the same ontology and version identity, retain both until their bytes and axiom sets have been compared. Distinguish a serialization difference from changed axioms. When the historical artifact cannot be selected from recorded evidence, report identity ambiguity and reject automatic acceptance under the proposed consumer policy. Structural comparison supplies a separate observation from byte identity. (robot.obolibrary.org)

Compare structure and execution separately

A byte difference is not necessarily an axiom difference. ROBOT diff compares ontology structure across differently serialized files and supports separate catalogs for the compared sides. Use that capability to avoid comparing a historical root against a candidate whose dependencies were accidentally resolved through the baseline catalog. (robot.obolibrary.org)

Reasoning output also depends on configuration. ROBOT reason performs logical validation before classification, but its default inferred output concerns subclass axioms. OWL-RL documents forward-chaining graph expansion and limitations in datatype-specific comparisons under particular conversion settings. These are different observable processing choices, not interchangeable certificates of complete historical reproduction. (robot.obolibrary.org) (owl-rl.readthedocs.io)

RO-Crate's provenance guidance recommends describing software with a version and associating input files with the creation action. The same principle supports retaining executable settings alongside ontology artifacts. OWLAPI's documented buffering behavior provides another reason to record the actual reasoning run: changes held in a buffer must be accounted for when interpreting its results. (www.researchobject.org) (owlcs.github.io)

Replay outcomes and acceptance boundaries

Reproducible OWL reasoning needs an acceptance statement with a defined scope. Under OWL Direct Semantics, satisfaction depends on the axiom closure, and entailment concerns all models of the premise ontology. Matching a small collection of observed answers is consequently a narrower claim than establishing general equivalence. (www.w3.org)

Table 2 distinguishes useful replay outcomes. The labels are proposed reporting categories, not standardized OWL result codes.

Observation	Permitted conclusion	Further action
All recorded artifacts and digests match; selected outputs match.	Historical artifact replay and the specified output checks succeeded.	State the exact fixture and execution scope. Artifact availability and result validation remain distinct. (www.acm.org)
Bytes differ; parsed comparison and selected outputs match.	The candidate passed the stated structural and output comparisons.	Report the byte mismatch; ROBOT's structural diff addresses serialization differences. (robot.obolibrary.org)
All imports resolve; a baseline entailment changes.	Current interpretation differs on the tested predicate.	Keep the changed result and the responsible candidate inputs available for review. (arxiv.org)
A required historical artifact is unavailable.	Historical reconstruction is **unknown**.	Identify the missing document and reason; surviving metadata can remain useful. (www.gofair.foundation)
The closure contains conflicting versions or declared incompatibility.	The candidate requires conflict review against OWL's closure recommendation.	Reject automatic acceptance under the proposed replay policy. (www.w3.org)
Reasoning or query evaluation fails, times out, or lacks required support.	The specified output is **unknown**, with a recorded reason.	Do not convert an uncompleted test into an empty result.

The distinction between a negative result and an unexecuted test is essential. **Not entailed** means the selected semantics does not require the proposition. It does not establish that the proposition's negation is entailed. An unavailable dependency provides still less information about the intended historical ontology. (www.w3.org)

For the SPARQL Protocol and RDF Query Language (**SPARQL**), the entailment regime includes the chosen entailment relation, permitted graphs and queries, and possible errors. Its OWL Direct Semantics regime requires an error for an inconsistent queried ontology. Query evidence should therefore retain the regime and error state alongside rows. (www.w3.org)

- **Byte identity:** State which retained documents match their recorded digests.
- **Structural identity:** State whether the compared axiom sets, imports, and relevant metadata match.
- **Semantic scope:** Name the entailment semantics and the predicates actually tested.
- **Query scope:** Identify dataset arrangement, graph selection, query text, and comparison rules.
- **Execution state:** Distinguish completed, failed, timed-out, unsupported, and missing-input checks.
- **Claim boundary:** State what passing this evidence package establishes and what remains untested.

W3C's conformance specification separately defines consistency and entailment tests and warns that passing its test collection does not prove conformance. Botoeva and colleagues' survey similarly distinguishes query inseparability from other notions of ontology inseparability. Finite replay fixtures should carry an equally explicit boundary. (www.w3.org) (arxiv.org)

Version policy and consumer decisions

Ontology dependency management combines publisher conventions with consumer acceptance policies. These responsibilities overlap, but evidence that a publisher assigns version identifiers does not establish that a particular consumer preserved the required artifacts.

Pin identifiers and preserve permitted snapshots

Pin a version IRI when the publisher supplies an appropriate version-specific artifact, then verify that the loaded document declares the intended identity. If a retained copy is permitted, preserve it with its retrieval evidence and use a documented mapping during replay. ROBOT mirror documents local caching of an import chain and generation of a catalog mapping source identifiers to local files. (robot.obolibrary.org)

A version-specific identifier still needs a preservation and resolution check. DataCite's versioning guidance illustrates why identifier policy must be read rather than assumed: a **Digital Object Identifier (DOI)** can represent all versions, and minor changes may retain a DOI with updated metadata. These are DataCite conventions, not OWL rules. (support.datacite.org)

The same distinction applies to format changes. Dublin Core distinguishes substantive version changes from format differences. A consumer can report a serialization change separately while requiring stricter byte preservation for the original historical-artifact claim. (www.dublincore.org)

OBO policy has a defined community scope

The **Open Biological and Biomedical Ontologies (OBO) Foundry** Principle 4 requires a unique resolving version IRI for each official release and states that official release contents must not change. Its perpetual retrieval expectation explicitly applies to officially released versions. These are OBO community requirements, not universal requirements imposed on every OWL ontology. (obofoundry.org)

The policy also requires persistent URL destinations to be updated when files move. Its automated check examines version-IRI presence and resolution; an absent version IRI is an error. A passing version check should not be presented as proof that a consumer's full historical closure or earlier query output has been reproduced. (obofoundry.org)

Project discussions clarify scope without establishing prevalence

The **Unified Cyber Ontology (UCO)** project's issue 437, opened in 2022, discusses reviewing import closure for multiple ontology versions and proposes versioned ontology imports. It is evidence of a concrete design discussion. It supplies no general failure rate, benchmark, or claim about all OWL consumers. (github.com)

The WebAgents Community Group interoperability report links ontology and version identity when discussing semantic validity over time. It describes itself as a living document. That link supports attention to version identity; it does not endorse the manifest, acceptance categories, or fixture introduced here. (w3c-cg.github.io)

The consumer's practical choice is therefore conditional: pin an appropriately identified artifact, preserve a permitted snapshot with a reproducible mapping, or reject the historical replay claim when the required evidence cannot be recovered.

Data Analysis and Evidence

Two-hop dependency change (Hypothetical Example)

The fixture uses three ontology documents: **C**, **A**, and **B**. C imports version A/1; A/1 imports B through its unversioned IRI. At **T1**, that request yields B/1. At **T2**, it yields B/2. C and A/1 remain byte-identical. The closure definition includes the root, so there are three ontology documents but only two imported dependency documents. This is a constructed example of the standard's transitive boundary. (www.w3.org)

The following Functional-Style Syntax fragments specify the fixture's operative assertions. Prefix `:` denotes `urn:fixture:`; the manifest above supplies ontology and version identities. The example is deliberately limited to named classes and an instance propagated through subclass axioms.

```
C, unchanged:
  Import(<urn:fixture:A/1>)
  ClassAssertion(:Sample :x)

A/1, unchanged:
 Import(<urn:fixture:B>)
 SubClassOf(:Sample :Intermediate)

B/1, returned at T1:
 SubClassOf(:Intermediate :Approved)

B/2, returned at T2 instead of B/1:
 SubClassOf(:Intermediate :Review)
```

The calculations apply the elementary rule that an instance of a subclass is an instance of its superclass until no further named-class memberships are added. They were independently executed in separate checks with the same explicit inputs. They are fixture calculations, not measurements of a full OWL reasoner, production system, throughput, or latency.

Table 3 reports the generated memberships restricted to the fixture's named classes, plus completeness and baseline-change counts. Built-in universal-class consequences are outside this comparison scope.

Check or observation	T1 baseline, B/1	T2 current resolution, B/2	Historical replay, retained B/1
Resolved imports / expected imports	**2/2**	**2/2**	**2/2**

Check or observation	T1 baseline, B/1	T2 current resolution, B/2	Historical replay, retained B/1
Documents in closure, root included	**3**	**3**	**3**
`Sample(x)`	Entailed	Entailed	Entailed
`Intermediate(x)`	Entailed	Entailed	Entailed
`Approved(x)`	Entailed	Not entailed	Entailed
`Review(x)`	Not entailed	Entailed	Not entailed
Changed baseline entailments / baseline entailments	**0/3**	**1/3**	**0/3**
New fixture memberships outside baseline	**0**	**1**, `Review(x)`	**0**
Query result over the fixture materialization	`Approved`	`Review`	`Approved`

The import count remains complete while an expected consequence changes. Neither **2/2 resolved imports** nor an unchanged root establishes historical interpretation. Replacing the current B/2 artifact with recorded B/1 restores the baseline within the stated fixture. If B/1 is missing, the historical result is unknown even if the current closure loads successfully.

The query is `SELECT ?category WHERE { :x a ?category . VALUES ?category { :Approved :Review } }`, with the prefix supplied and the fixture memberships materialized beforehand. Its differing rows follow directly from the memberships in the table. This does not assume that an arbitrary SPARQL endpoint performs OWL reasoning; Jena documents configuration-dependent query results. (jena.apache.org)

- **Completeness denominator:** Two distinct expected dependency documents, excluding the root, are counted; parent edges remain separately recorded.
- **Change denominator:** Three baseline predicates, `Sample`, `Intermediate`, and `Approved`, are evaluated for the same individual.
- **Additional outputs:** The new `Review` membership is reported separately, so baseline retention cannot hide added consequences.
- **Unavailable artifact:** Resolving only A yields **1/2** dependencies; predicates requiring B are untested for historical reconstruction.

For query comparison, state whether rows are compared as sets or multisets, how ordering is handled, and how blank-node results are matched. At the ontology level, OWL's axiom closure standardizes anonymous individuals apart across imported ontologies; raw blank-node labels should not be used as cross-document identity evidence. (www.w3.org)

Implications and Future Directions

The central procurement or stewardship question is whether the evidence package supports the claimed reconstruction scope. A readable archive supports retrieval. A digest supports artifact-integrity comparison. A reasoner run supports specified logical tests. A query comparison supports specified observations over a recorded dataset. The ACM distinction between available artifacts and validated results is useful precisely because these evidence categories can be evaluated independently. (www.acm.org)

The proposed method has implications for retention and review:

- **Approval boundary:** Record the consumer artifact, the complete dependency inventory, and the outputs the approval actually relied on.
- **Preservation boundary:** Decide which dependencies may be retained and how historical artifacts remain available under applicable access conditions.
- **Execution boundary:** Preserve settings and software identity alongside data, without assuming that the ontology files fully describe their execution environment.
- **Comparison boundary:** Define whether acceptance requires bytes, axiom sets, selected outputs, or a stronger independently justified semantic relation.
- **Unknown boundary:** Preserve missing-input reasons and completed observations even when the overall reconstruction cannot be accepted.

Research on workflow provenance offers a useful direction for packaging this evidence. Leo and colleagues' Workflow Run RO-Crate work describes bundling objects associated with execution, while RO-Crate's provenance guidance connects inputs and software to the action that produced an artifact. These approaches can inform evidence packaging without turning a particular package format into an OWL semantic requirement. (arxiv.org) (www.researchobject.org)

Persistent identifiers also require operational stewardship. The GO FAIR Foundation explains that persistence concerns continued resource identity; w3id describes a redirection service rather than a file archive. Maintaining an identifier and maintaining the bytes it identifies are complementary responsibilities. FORCE11 similarly separates version verification from persistence. (www.gofair.foundation) (w3id.org) (force11.org)

A useful future evaluation would compare independently reconstructed closures and outputs from the same retained bundle, with declared software differences. This report supplies no empirical success rate for that exercise. Botoeva and colleagues' application-dependent account of safe replacement remains the appropriate constraint: a narrow fixture can establish a narrow result, while broader equivalence requires a stronger argument. (arxiv.org)

Conclusion

An old graph can be reinterpreted when an unchanged consumer reaches changed ontology axioms through its imports. A historical replay claim must consequently identify the dependency documents and the outputs being reconstructed. Import closure supplies the logical dependency boundary; it does not itself constitute the retained evidence package.

The proposed procedure records requested and resolved identities, declared ontology versions, retrieval observations, digests, and parent edges. Replay then checks dependency completeness, artifact differences, axiom differences, and explicitly selected entailments or query results. These comparisons answer different questions and should retain separate outcomes. The distinction follows the broader separation between artifact availability and validated results. (www.acm.org)

In the synthetic fixture, complete current resolution still changes a baseline conclusion. Historical reconstruction succeeds only within the declared test scope when the recorded dependency is supplied. A missing historical artifact warrants an unknown result with a reason. Substituting the current ontology would answer a different question from the one the historical approval requires.

The defensible decision is to pin an appropriate version, preserve a permitted snapshot with its resolution evidence, or reject the replay claim. Reviewers should require enough provenance and fixity to verify the specific artifacts used, together with explicit limits on what matching outputs establish. Application-dependent equivalence remains a separate, stronger question. (force11.org) (arxiv.org)

Source links

Links cited in this article, in order of first appearance. Full addresses are provided for readers using extracted text.

1. https://www.w3.org/TR/owl2-syntax
2. https://csrc.nist.gov/pubs/fips/180-4/upd1/final
3. https://robot.obolibrary.org/diff
4. https://force11.org/info/joint-declaration-of-data-citation-principles-final
5. https://www.w3.org/TR/owl2-test
6. https://jena.apache.org/documentation/ontology
7. https://owlcs.github.io/owlapi/apidocs_5/org/semanticweb/owlapi/model/MissingImportHandlingStrategy.html
8. https://obofoundry.org/principles/fp-004-versioning.html
9. https://www.acm.org/publications/policies/artifact-review-and-badging-current
10. https://www.w3.org/TR/sparql11-entailment
11. https://ijcai.org/proceedings/2007/46
12. https://arxiv.org/abs/1804.07805
13. https://protegewiki.stanford.edu/wiki/How_Owl_2.0_Imports_Work
14. https://owlcs.github.io/owlapi/apidocs_5/org/semanticweb/owlapi/model/OWLOntologyIRIMapper.html
15. https://protegeproject.github.io/protege/views/imported-ontologies
16. https://www.researchobject.org/ro-crate/specification/1.2/introduction.html
17. https://www.rfc-editor.org/rfc/rfc2119.html
18. https://www.dublincore.org/specifications/dublin-core/dcmi-terms
19. https://www.rfc-editor.org/rfc/rfc9110.html
20. https://w3id.org
21. https://www.softwareheritage.org/mission/approach

22. https://robot.obolibrary.org/global
23. https://protegewiki.stanford.edu/wiki/Importing_Ontologies_in_P41
24. https://robot.obolibrary.org/query
25. https://robot.obolibrary.org/reason
26. https://owl-rl.readthedocs.io/en/latest/owlrl.html
27. https://www.researchobject.org/ro-crate/specification/1.2/provenance.html
28. https://owlcs.github.io/owlapi/apidocs_5/org/semanticweb/owlapi/reasoner/OWLReasoner.html
29. https://www.w3.org/TR/owl2-direct-semantics
30. https://www.gofair.foundation/a2
31. https://robot.obolibrary.org/mirror.html
32. https://support.datacite.org/docs/versioning
33. https://github.com/ucoProject/UCO/issues/437
34. https://w3c-cg.github.io/webagents/TaskForces/Interoperability/Reports/report-interoperability.html
35. https://arxiv.org/abs/2312.07852
36. https://www.gofair.foundation/f1

THE PUBLISHER

About Reliable Context

Reliable Context is a system of record for organizations that fuse private and public information and must defend every value they act on.

The guarantees

One canonical record per real thing: duplicates resolve by rule, ambiguity is shown rather than hidden, and related organizations relate without ever being merged. A verbatim, cited source and an observation date sit behind every value. Unknowns are declared as not found, blocked or not applicable with a reason, never left as an empty cell. Any past state of the record is reproducible, together with what changed since.

Where it runs

It runs inside the customer's perimeter, one database per organization. Permitted sources, model routing, reader clearance and retention are set as the customer's own policies, and every value is exportable with its chain of custody for an auditor or regulator.

What it is not

It is not a vector database or a RAG framework, not a data catalog, MDM or data-quality suite, not a CRM or data-enrichment vendor, and not a chatbot or AI assistant. Where a reader needs one of those, the honest answer is to say so and point them there.

Visit

See Reliable Context.

Reliable Context is in production use in two sectors; more than six thousand of its identity decisions were made by rule and none has been reversed. Articles published here explain general practice, cite their primary and academic sources, and name the jurisdiction of any regulatory rule they describe. They are not legal, regulatory or compliance advice for a specific organization, and requirements change.

Website: https://reliablecontext.com

This article is educational. Verify consequential medical, legal, financial, regulatory and technical decisions with appropriate professionals and the cited primary sources. Company services are described separately from the article's research findings.