Reliable Context

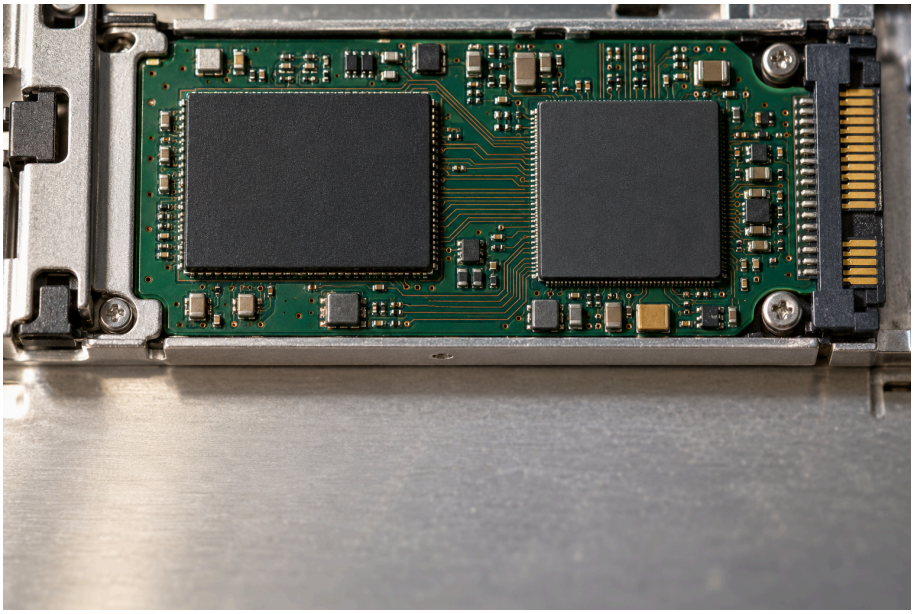
RELIABLE CONTEXT / RESEARCH & PRACTICAL GUIDANCE

W3C PROV Explained: Entities, Activities and Agents

> 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 · 2026-10-29 · w3c prov · prov-o ontology · prov data model explained · wasderivedfrom wasgeneratedby · data provenance standard

Original article: https://reliablecontext.com/articles/w3c-prov-explained-entities-activities-agents



In this article

1. What PROV is for
2. The PROV document family
3. The three core types
4. The core relations
5. Worked example: one value from page to record
6. The same graph in PROV-O
7. Validity: what PROV checks
8. What PROV does not say
9. Checklist: minimum statements for a defensible value
10. Sources

What PROV is for

W3C PROV is a family of specifications for describing provenance: "information about entities, activities, and people involved in producing a piece of data or thing, which can be used to form assessments about its quality, reliability or trustworthiness" (PROV-Overview). Its stated goal is interchange: "to enable the wide publication and interchange of provenance on the Web and other information systems". PROV is not a database, a lineage tool or a trust score. It is a shared vocabulary that lets one system hand another an account of where something came from, in a form the receiver can parse and check.

This tutorial covers the three core types (Entity, Activity, Agent), the core relations between them, a worked example that traces one value from a source page to a record, the same graph in RDF, the validity rules in PROV-CONSTRAINTS, and what a valid PROV graph still does not tell you.

The PROV document family

PROV is not one document. The W3C Provenance Working Group published the set on 30 April 2013, with four Recommendations and several Working Group Notes (PROV-Overview):

Document	Status	What it defines
PROV-DM	Recommendation	The conceptual data model: types, relations, attributes
PROV-O	Recommendation	An OWL 2 ontology mapping PROV-DM to RDF
PROV-N	Recommendation	A compact human-readable notation
PROV-CONSTRAINTS	Recommendation	Inferences and the rules that make a PROV instance valid
PROV-PRIMER	Note	Introductory examples

Document	Status	What it defines
PROV-XML	Note	An XML schema
PROV-AQ	Note	How to access and query provenance on the Web
PROV-DICTIONARY, PROV-DC, PROV-SEM, PROV-LINKS	Notes	Dictionaries, Dublin Core mapping, first-order semantics, links across bundles

The distinction matters when you write a requirement. "Conforms to PROV-DM and PROV-CONSTRAINTS" is a testable statement against Recommendations. "Follows the PROV Primer" is not.

The three core types

PROV-DM defines three kinds of thing (PROV-DM §2.1).

- **Entity.** "A physical, digital, conceptual, or other kind of thing with some fixed aspects; entities may be real or imaginary." A PDF fetched on a given day, a database row at a given version and a sentence quoted from a page are entities. The phrase "fixed aspects" is the important one. A web page whose content changes is not one entity in PROV terms; each state you want to talk about is a separate entity, which PROV can relate as alternates or specializations of a more general one.
- **Activity.** "Something that occurs over a period of time and acts upon or with entities; it may include consuming, processing, transforming, modifying, relocating, using, or generating entities." A download, an extraction, a review and a merge are activities. Activities have optional start and end times.
- **Agent.** "Something that bears some form of responsibility for an activity taking place, for the existence of an entity, or for another agent's activity." People, organizations and software can all be agents.

The core relations

Seven relations connect the three types in the core of the model (PROV-DM §2.1). Directions run from the later or dependent thing to the earlier or responsible one.

Relation (PROV-N)	From → To	Definition in PROV-DM
`wasGeneratedBy`	Entity → Activity	Generation: "the completion of production of a new entity by an activity"
`used`	Activity → Entity	Usage: "the beginning of utilizing an entity by an activity"
`wasInformedBy`	Activity → Activity	Communication: one activity used an entity generated by the other
`wasDerivedFrom`	Entity → Entity	Derivation: "a transformation of an entity into another, an update of an entity resulting in a new one, or the construction of a new entity based on a pre-existing entity"
`wasAttributedTo`	Entity → Agent	Attribution: "the ascribing of an entity to an agent"

Relation (PROV-N)	From → To	Definition in PROV-DM
`wasAssociatedWith`	Activity → Agent	Association: "an assignment of responsibility to an agent for an activity", optionally with a plan
`actedOnBehalfOf`	Agent → Agent	Delegation: authority and responsibility assigned to a delegate while the other agent "retains some responsibility for the outcome"

PROV-DM adds subtypes of derivation that are useful for evidence: `wasRevisionOf` (a revised version of an original), `wasQuotedFrom` (a repetition of part of an entity) and `hadPrimarySource` (PROV-DM §5.2). It also defines **bundles**: "a named set of provenance descriptions, and is itself an entity, so allowing provenance of provenance to be expressed" (PROV-DM §2.2). Bundles are how you say who asserted a piece of provenance.

Worked example: one value from page to record

The example is fictional. **Halden Therapeutics AG is a fictional company**, and the registry, people and software below are invented for illustration.

A record about Halden Therapeutics AG carries the value *development phase of HLD-204 = Phase 2*. The value was read from version 9 of a trial registry entry. The chain was:

1. On 11 March 2026 at 08:14 UTC, a retrieval service fetched the registry entry and stored a snapshot.
2. At 08:20, an analyst, Mara Okafor, extracted the phase from the snapshot, quoting the sentence "Phase 2 study of HLD-204 in KRAS-mutant tumours".
3. At 09:05, a reviewer, Jonas Weber, admitted the value to the record under a written review procedure.

The graph in PROV-N

```
document
  prefix ex <https://example.org/halden/>
  prefix reg <https://registry.example.org/>

  entity(reg:HLD-204-v9, [prov:type='ex:RegistrySnapshot', ex:version="9",
                          ex:sha256="3c9a...f0"])
  entity(ex:quote-1, [prov:value="Phase 2 study of HLD-204 in KRAS-mutant tumours"])
  entity(ex:phase-value-1, [prov:value="Phase 2", ex:field="developmentPhase"])

  activity(ex:fetch-1, 2026-03-11T08:14:00Z, 2026-03-11T08:14:03Z)
  activity(ex:extract-1, 2026-03-11T08:20:00Z, 2026-03-11T08:31:00Z)
  activity(ex:review-1, 2026-03-11T09:05:00Z, 2026-03-11T09:12:00Z)

  agent(ex:retriever, [prov:type='prov:SoftwareAgent', ex:version="2.3"])
  agent(ex:okafor, [prov:type='prov:Person'])
  agent(ex:weber, [prov:type='prov:Person'])
  agent(ex:halden-research, [prov:type='prov:Organization'])

  wasGeneratedBy(reg:HLD-204-v9, ex:fetch-1, 2026-03-11T08:14:03Z)
```

```
  wasAssociatedWith(ex:fetch-1, ex:retriever, -)

  used(ex:extract-1, reg:HLD-204-v9, 2026-03-11T08:20:00Z)
  wasGeneratedBy(ex:quote-1, ex:extract-1, 2026-03-11T08:24:00Z)
  wasQuotedFrom(ex:quote-1, reg:HLD-204-v9)
  wasGeneratedBy(ex:phase-value-1, ex:extract-1, 2026-03-11T08:31:00Z)
  wasDerivedFrom(ex:phase-value-1, ex:quote-1)
  wasAssociatedWith(ex:extract-1, ex:okafor, -)

  used(ex:review-1, ex:phase-value-1, 2026-03-11T09:05:00Z)
  wasAssociatedWith(ex:review-1, ex:weber, ex:review-procedure-v4)
  wasAttributedTo(ex:phase-value-1, ex:okafor)
  actedOnBehalfOf(ex:okafor, ex:halden-research)
  actedOnBehalfOf(ex:weber, ex:halden-research)
endDocument
```

Read from the value backwards, the graph answers five questions:

Question	Statement that answers it
What is asserted?	`entity(ex:phase-value-1, [prov:value="Phase 2"])`
From what evidence?	`wasDerivedFrom(ex:phase-value-1, ex:quote-1)` and `wasQuotedFrom(ex:quote-1, reg:HLD-204-v9)`
Which version of the source, read when?	`reg:HLD-204-v9` with `ex:version` and a hash, generated by `ex:fetch-1` at 08:14:03
By what process?	`ex:extract-1`, then `ex:review-1` under plan `ex:review-procedure-v4`
Who is responsible?	Okafor (attribution), Weber (association), both on behalf of the organization (delegation)

Note what is recorded as an attribute rather than in the PROV core: the registry version, the hash and the field name all sit in `ex:` attributes. PROV leaves domain detail to the application.

The same graph in PROV-O

PROV-O expresses PROV-DM as an OWL 2 ontology so the graph can live in any RDF store. Its terms come in three tiers: Starting Point terms (the three classes plus `prov:used`, `prov:wasGeneratedBy`, `prov:wasDerivedFrom`, `prov:wasAttributedTo`, `prov:wasAssociatedWith`, `prov:actedOnBehalfOf`, `prov:wasInformedBy`, `prov:startedAtTime`, `prov:endedAtTime`), Expanded terms, and Qualified terms (PROV-O §3). Each RDF statement is a subject-predicate-object triple whose nodes are identified with IRIs (RDF 1.1 Concepts).

```
@prefix prov: <http://www.w3.org/ns/prov#> .
@prefix xsd:  <http://www.w3.org/2001/XMLSchema#> .
@prefix ex:   <https://example.org/halden/> .
@prefix reg:  <https://registry.example.org/> .

reg:HLD-204-v9 a prov:Entity ;
```

```
    prov:wasGeneratedBy ex:fetch-1 ;
    prov:generatedAtTime "2026-03-11T08:14:03Z"^^xsd:dateTime ;
    ex:version "9" .

ex:quote-1 a prov:Entity ;
    prov:value "Phase 2 study of HLD-204 in KRAS-mutant tumours" ;
    prov:wasQuotedFrom reg:HLD-204-v9 ;
    prov:wasGeneratedBy ex:extract-1 .

ex:phase-value-1 a prov:Entity ;
    prov:value "Phase 2" ;
    prov:wasDerivedFrom ex:quote-1 ;
    prov:wasGeneratedBy ex:extract-1 ;
    prov:wasAttributedTo ex:okafor .

ex:fetch-1 a prov:Activity ;
    prov:startedAtTime "2026-03-11T08:14:00Z"^^xsd:dateTime ;
    prov:endedAtTime   "2026-03-11T08:14:03Z"^^xsd:dateTime ;
    prov:wasAssociatedWith ex:retriever .

ex:extract-1 a prov:Activity ;
    prov:startedAtTime "2026-03-11T08:20:00Z"^^xsd:dateTime ;
    prov:endedAtTime   "2026-03-11T08:31:00Z"^^xsd:dateTime ;
    prov:used reg:HLD-204-v9 ;
    prov:wasAssociatedWith ex:okafor .

ex:review-1 a prov:Activity ;
    prov:used ex:phase-value-1 ;
    prov:qualifiedAssociation [
        a prov:Association ;
        prov:agent ex:weber ;
        prov:hadPlan ex:review-procedure-v4
    ] .

ex:retriever a prov:SoftwareAgent .
ex:okafor a prov:Person ; prov:actedOnBehalfOf ex:halden-research .
ex:weber  a prov:Person ; prov:actedOnBehalfOf ex:halden-research .
ex:halden-research a prov:Organization .
```

The review activity shows the **qualification pattern**. A plain `prov:wasAssociatedWith` triple cannot carry a plan, a role or a time, because a triple has no place for extra detail. PROV-O "restates an unqualified influence relation by using an intermediate class that represents the influence between two resources", which can then be annotated (PROV-O §3.3). The same pattern exists for usage (`prov:qualifiedUsage`), generation (`prov:qualifiedGeneration`) and the other influences. It is how you attach a time to a specific use of an entity rather than to the activity as a whole.

Validity: what PROV checks

PROV-DM itself "imposes minimal requirements upon PROV instances". PROV-CONSTRAINTS defines a subset called **valid** instances, which correspond to "a consistent history of objects and interactions to which logical reasoning can be safely applied". The authors are explicit that this "differs from the usual meaning of 'validity' in logic" and is "closer to logical 'consistency'" (PROV-CONSTRAINTS §1.2). PROV instances need not be valid; validity is something an application can choose to require.

There are four kinds of constraint (PROV-CONSTRAINTS §6):

Kind	Example	What it catches
Uniqueness	Constraint 24 (unique-generation): an entity is generated by a given activity at most once	Two conflicting generation times for one snapshot
Event ordering	Constraint 37 (generation-precedes-usage): an entity's generation precedes any usage of it	A value used before it existed
Event ordering	Constraint 41 (derivation-usage-generation-ordering): in a derivation, the usage of the source precedes the generation of the result	A conclusion timestamped before its evidence was read
Impossibility	Constraint 55 (entity-activity-disjoint)	The same identifier used as both an entity and an activity
Type	Constraint 50 (typing)	Relations whose arguments have the wrong type

PROV-CONSTRAINTS also defines inferences an application may draw. Inference 11, for instance, says that a derivation with a known activity implies that the activity used the source and generated the result.

Now break the example. Suppose a clock error records the snapshot as generated at 08:25 while `ex:extract-1` used it at 08:20. Constraint 37 requires the generation event to precede the usage event, so the instance is invalid. That is a useful check: it detects a value whose recorded evidence did not yet exist when the value was produced. Because the constraints are stated as rules over statements, the check can be run mechanically rather than by inspection.

What PROV does not say

A valid PROV graph is a consistent account. It is not a true one. Specifically:

- **Truth.** Nothing in the example tells you that HLD-204 is in Phase 2. It tells you that a named person derived "Phase 2" from a quoted sentence in a named snapshot. If the registry was wrong, the graph is still valid.
- **Completeness.** PROV has no notion of a required minimum. A graph with one `wasDerivedFrom` triple and nothing else can be valid. Which statements are mandatory is an application-profile decision.
- **Trust and authority.** The model records responsibility (attribution, association, delegation). It does not grade sources or say whether an agent was authorized to make the assertion.

- **Integrity.** PROV can carry a hash as an attribute, but it does not define one, and it does not stop anyone from editing the graph. Tamper evidence must come from the store or a signature over a bundle.
- **Who asserted the provenance.** Unless the statements are placed in a bundle with its own attribution, the graph does not say who claims this history.

The PROV abstract makes the same division of labour: provenance is information "which can be used to form assessments about its quality, reliability or trustworthiness" (PROV-DM abstract). The assessment is left to the consumer.

Checklist: minimum statements for a defensible value

PROV deliberately leaves the minimum to each application. The following is a practical profile for a single value that must stand up to review, not a W3C requirement.

1. The value is an `Entity` with a stable identifier and `prov:value`.
2. The value `wasDerivedFrom` an evidence entity, preferably a `wasQuotedFrom` quotation with the verbatim text.
3. The source is a versioned `Entity` (a snapshot, not a live URL), with a version label and a content hash as attributes.
4. The source snapshot `wasGeneratedBy` a retrieval `Activity` with an end time: the observation time.
5. The extraction `Activity` `used` the snapshot and generated the value, with start and end times.
6. Each activity `wasAssociatedWith` an `Agent`; software agents carry a version.
7. The value `wasAttributedTo` the responsible agent, and delegation to the organization is explicit.
8. Review or approval is a separate activity, associated with a plan (the written procedure).
9. The statements are grouped in a bundle attributed to the system or person asserting them.
10. The instance passes PROV-CONSTRAINTS validation, at minimum the ordering constraints.

The site's W3C PROV reference entry summarizes the standard, and the testable PROV profile for LLM context shows how an application profile tightens these freedoms for answers produced by language models. The provenance glossary entry gives the short definition.

The checklist above is close to what Reliable Context's published guarantee describes as a cited, verbatim, dated source behind every value.

Sources

1. PROV-Overview: An Overview of the PROV Family of Documents, W3C Working Group Note, 30 April 2013 - purpose of PROV; list and status of the documents.
2. PROV-DM: The PROV Data Model, W3C Recommendation, 30 April 2013 - definitions of entity, activity, agent and the core relations; derivation subtypes; bundles.
3. PROV-O: The PROV Ontology, W3C Recommendation, 30 April 2013 - Starting Point, Expanded and Qualified terms; the qualification pattern.
4. Constraints of the PROV Data Model (PROV-CONSTRAINTS), W3C Recommendation, 30 April 2013 - validity as consistency; uniqueness, ordering, impossibility and type constraints; inferences.

5. PROV-Primer, W3C Working Group Note, 30 April 2013 - introductory examples of the same patterns.
6. RDF 1.1 Concepts and Abstract Syntax, W3C Recommendation - triples and IRIs underlying PROV-O.

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/prov-overview
2. https://www.w3.org/TR/prov-dm
3. https://www.w3.org/TR/prov-o
4. https://www.w3.org/TR/rdf11-concepts
5. https://www.w3.org/TR/prov-constraints
6. https://www.w3.org/TR/prov-primer

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.