

Talkoot AI provenance

Last updated October 6, 2026

1. Purpose and scope

This specification describes how Talkoot records cell-level AI provenance inside an XLSX workbook and exposes provenance properties on API content objects. It lets downstream software discover the records, interpret their meaning, and associate them with exported worksheet cells without changing visible worksheet content.

The namespace is a format identifier, not a website address, a certification, or an external industry standard. A reader does not need to resolve or contact that identifier to parse a workbook. The accompanying `ai-provenance-v1.xsd` defines the supported XML structure. The XML schema does not need to be embedded in each workbook.

The metadata records an AI-generation association held by Talkoot at export time. It does not independently inspect text to determine authorship. It does not establish that every word in a cell was generated by AI, that a person has reviewed the content, or that the exported content is unchanged from the original AI output.

2. Package discovery

An XLSX file is an Open Packaging Conventions (OPC) ZIP package. Discover metadata through package relationships, rather than assuming a filename such as `customXml/item1.xml`:

1. Read package relationships in `_rels/.rels` and locate the internal Office Document relationship to the workbook part.
2. Read that workbook part's relationship file. For example, the relationship file for `xl/workbook.xml` is `xl/_rels/workbook.xml.rels`.
3. Follow internal relationships whose type is `http://schemas.openxmlformats.org/officeDocument/2006/relationships/customXml`. Resolve each target relative to the workbook part.
4. Parse each target as XML and select the root by its expanded name: `{urn:talkoot:ai-provenance:v1}aiProvenance`.
5. Read the namespace-qualified `cell` children. Map each record's `sheet` and `address` to the corresponding workbook sheet name and A1 cell coordinate.

The current exporter creates at most one matching custom XML part. It omits that part when there are no qualifying records. The part's filename, relationship identifier, XML prefix, attribute order, and record order are not stable identifiers and have no semantic significance. Elements use the provenance namespace; attributes are unqualified. The current implementation produces Transitional OOXML workbooks; the example reader supports that profile.

Readers should reject invalid or ambiguous provenance rather than silently selecting one interpretation. The supplied reader rejects multiple matching parts and duplicate cell locations. It never follows external relationships or extracts ZIP entries onto disk.

3. XML structure

```
<?xml version="1.0" encoding="utf-8"?>

<aiProvenance xmlns="urn:talkoot:ai-provenance:v1">

  <cell sheet="Export" address="F12" label="artificially generated or manipulated"

    copyId="12345" generationId="67890"

    generatedAt="2026-09-21T16:42:07Z" />

  <cell sheet="Export" address="F13" label="artificially generated or manipulated"

    copyId="12346" generationId="67891" />

</aiProvenance>
```

This is illustrative data, not a production export. A present provenance root contains one or more **cell** elements. A record identifies one exported cell, not an entire row, column, sheet, or workbook.

Attribute	Required	Type	Meaning
sheet	Yes	Nonempty string	Worksheet name as written in the workbook, not the numeric sheet ID or part filename. Match the actual name exactly.
address	Yes	A1 coordinate	Uppercase column letters followed by a positive row number, such as F12 . No \$, sheet prefix, or range. Must refer to an existing cell within Excel's A1:XFD1048576 grid.
label	Yes	Enumerated string	Only permitted value: artificially generated or manipulated . Capitalization and spaces are significant. See section 4.
copyId	Yes	Positive 32-bit integer	Talkoot copy record associated with the exported cell. Treat as an opaque identifier in its originating Talkoot context.

Attribute	Required	Type	Meaning
			It is not a publicly resolvable URL or a global identifier.
<code>generationId</code>	Yes	Positive 32-bit integer	Recorded AI-generation association for that copy. It does not identify the AI model or certify an unchanged generation.
<code>generatedAt</code>	No	XML Schema <code>dateTime</code>	Generation timestamp supplied for the associated copy. Omitted when unavailable; never substituted with export time. See section 5.

The pair (`sheet`, `address`) must be unique in a metadata part.

Repeated `copyId` or `generationId` values are allowed: the same copy can appear in multiple exported cells. There is no explicit version attribute; the namespace carries the format version.

4. Labels and provenance semantics

The only supported label is the exact literal `artificially generated or manipulated`, with lowercase letters and single spaces between words. No leading or trailing whitespace is part of the value. This wording follows Article 50(2); the wording alone does not independently establish legal compliance.

The label is backed by the following mechanism: the copy has a positive recorded generation ID, and the exporter associates that copy with a qualifying exported cell. The label is not a calculated percentage of AI authorship. Human edits may be included if the source system retains the generation association after those edits. Readers must preserve the literal label and must not infer an authorship percentage or a generation-versus-editing classification from it.

The exporter collects only explicitly tagged, authorized cells. It requires positive copy and generation IDs, nonempty exported content, and an exact match between the original source text supplied to the collector and the corresponding retrieved copy text. Before writing a record it also checks that the tracked displayed value still matches the value being exported. These checks reduce mismatched attribution; they do not provide a signature or a durable content hash.

Display transformations can occur. For example, a Boolean feature may appear as `X`, and rich text or XML cleanup can change presentation. The record identifies the final exported cell even where that cell is not a literal rendering of the original generated text. Positions reflect the export's final row and column order, including its header row. Repeated article rows receive separate cell records.

An absent record means no provenance assertion is available for that cell. It does not mean human-authored, AI-free, or exempt from any legal requirement. Omission can result from an unsupported export path, missing generation data, empty cells, changed or unmatched source text, or later metadata removal. Historical and aggregate exports must not be assumed to have this coverage.

5. Timestamps

`generatedAt` follows XML Schema date/time syntax. Serialization preserves the source date/time kind:

Example	Interpretation
<code>2026-09-21T16:42:07Z</code>	UTC instant.
<code>2026-09-21T09:42:07-07:00</code>	Date/time with an explicit UTC offset.
<code>2026-09-21T16:42:07</code>	Timezone unspecified. Do not assume UTC or the reader's local timezone.
Attribute omitted	Generation time unavailable.

Fractional seconds may be present. Preserve the original value and precision. Only normalize to a UTC instant when an offset is known or an authoritative source-timezone agreement exists. SQL date/time values can arrive without timezone information. The timestamp is neither an export timestamp nor a signature timestamp and does not establish the time of a later human edit.

6. Validation and reader behavior

Use the accompanying XSD to check element structure, required attributes, the exact permitted label, positive integer ranges, timestamp syntax, and duplicate locations. Additional package-level checks are required: the XSD cannot confirm that a sheet or cell exists, that a coordinate falls within Excel's full limits, or that metadata still describes the displayed content.

The supplied `read_provenance.py` discovers the metadata through OPC relationships, validates it against the XSD, checks referenced sheets and cells, and prints JSON. It reports `not_present` when no matching part exists, `present` for structurally valid mapped records, and `error` with a nonzero exit status for malformed or unsupported input. `present` means a recorded assertion was found, not that the assertion's authenticity was independently verified.

Requirements: Python 3.10 or newer and `lxml`. Keep the reader and XSD in the same directory. Example:

```
python -m pip install lxml
```

```
python read_provenance.py exported-workbook.xlsx
```

A successful record includes the original attributes and `timestampTimezone` equal to `specified`, `unspecified`, or `absent`. Unknown fields or labels within the v1 namespace fail strict validation. A future Talkoot provenance namespace is reported as unsupported rather than silently interpreted as v1. Unrelated custom XML is ignored after parsing.

The reader uses bounded ZIP/XML reads, disables network access and entity expansion, rejects DTDs, and does not follow external links. Its conservative resource limits can reject unusually large legitimate workbooks. It is a reference implementation, not a workbook repair utility, malware scanner, or provenance-authenticity verifier.

7. Integrity, editing, and accessibility limits

This is removable metadata. It contains no digital signature, content digest, trusted timestamp, or tamper-evident binding between a record and cell content. A modified workbook can retain stale records that still pass structural validation.

Excel edits, sorting after export, row/column insertion or deletion, sheet renaming, copying cells between workbooks, and conversions such as XLSX to CSV can invalidate or remove provenance. Consumers must not assume metadata automatically follows those operations. A downstream editor should preserve and correctly update provenance only when it can establish the mapping and meaning; it should surface uncertainty when that cannot be established.

The metadata is not displayed by ordinary Excel worksheet views as a built-in AI label. Reading it requires software that understands this convention. This format does not supply a visible disclosure, prove human review, distinguish every form of assistive editing, or independently demonstrate legal compliance.

8. Versioning and compatibility

This publication candidate defines the initial v1 format with one supported label: `artificially generated or manipulated`. Specification revision 1.2 is the current draft revision, not a separate namespace version. A documentation-only clarification can update the specification revision without changing the namespace. Changes to field meanings, label semantics, or incompatible XML structures should use a new namespace version. Retain each published specification and schema for historical files.

Namespace comparison is exact and case-sensitive. Consumers must not treat an unfamiliar version as v1. Do not reinterpret the recorded generation association as proof of entirely AI-written content.

9. API content properties

Scope and placement

Add the following properties to each API content object whose AI-generation association is available. Put them on the individual content object, alongside that object's content and identifiers, rather than on an enclosing response, model, job, or array. Preserve the exact property names and JSON types.

The local implementation verified for this revision uses `CopyApiJsonConverter` on items in `CopyApiResponse.Copies`. It adds both properties and removes `GenerationId` from that JSON object. This is verified source behavior, not confirmation that all API endpoints or deployed environments expose the fields. Other content response types require explicit adoption and verification; this specification does not claim they are already covered. The XML namespace identifies the XLSX format and is not an extra JSON property.

Property	JSON type	Presence for covered objects	Meaning and source
<code>AIGenerated</code>	Boolean	Always present	<code>true</code> when the content's recorded <code>GenerationId</code> is greater than zero; otherwise <code>false</code> . The field name includes substantively modified content where a generation association is retained. It is not a percentage or an assertion of exclusively AI-written content.
<code>AIGenerationDate</code>	String or <code>null</code>	Always present	Stored generation date/time for the same content object, serialized by the API's configured date serializer. The verified converter reads <code>Copy.AIGenerationDate</code> directly; absent dates become explicit JSON <code>null</code> . Do not use an empty string, a zero date, an update date, or the response/export time as a substitute.

`false` means no positive generation ID was recorded on the returned object. It does not prove human authorship or absence of AI involvement. On older or unsupported endpoints, a missing `AIGenerated` property means unavailable provenance information, not `false`. A missing `AIGenerationDate` property indicates an older/unsupported contract or an incomplete response, while explicit `null` on a covered object means the date is unavailable.

JSON examples

These are illustrative fragments of individual content objects; ordinary content and identifier fields are omitted for brevity.

Recorded generation with a known UTC timestamp:

```
{
```

```

    "AIGenerated": true,

    "AIGenerationDate": "2026-09-21T16:42:07Z"
}

```

Recorded generation with an unavailable timestamp:

```

{
    "AIGenerated": true,

    "AIGenerationDate": null
}

```

No recorded generation association and no stored timestamp:

```

{
    "AIGenerated": false,

    "AIGenerationDate": null
}

```

The current converter computes the Boolean and date independently. A stored date can therefore be returned alongside **AIGenerated: false** if the source data has that combination. Do not document or enforce a rule that **false** automatically clears the date unless the implementation and source-data contract are deliberately changed. Consumers should flag that combination for source-data review and should not infer **true** merely from the presence of a date.

Date handling

For new API adoption, use ISO 8601 date/time strings and preserve source timezone information. The inspected API setup uses Newtonsoft.Json with no explicit date-format or timezone override in **WebApiConfig**; the converter delegates date serialization to the active serializer. Verify the actual wire format on each deployed endpoint, especially if its serializer configuration differs.

A suffix **Z** means UTC; an explicit offset identifies an offset-based time; no suffix means the source timezone is unspecified. Do not add **Z** unless the source is known to be UTC. Fractional seconds may be present. A null or timezone-less value must not be replaced with the client's current time or assumed local timezone. The rules and examples in section 5 also apply to interpreting API generation timestamps.

Adding the fields to additional content objects

1. Retrieve the generation association and date for the exact content/version returned. Do not assign a current revision's provenance to historical content or infer authorship by examining the text.
2. Add an explicitly named JSON Boolean **AIGenerated** and a nullable date/time **AIGenerationDate** to the response projection or serializer. Preserve explicit **false** and **null** values even where general serialization settings omit defaults or nulls.
3. Treat these as server-supplied response provenance. This documentation does not authorize clients to set or override them and does not claim that request-side write protection has been verified. Define and enforce request behavior separately.
4. Preserve the endpoint's existing permissions and field filtering. Review any exposure of generation dates in the same access context as the returned content. Do not expose internal generation IDs merely to support the Boolean.
5. Update the endpoint's API/OpenAPI description and examples. Test generated content with a date, generated content without a date, no recorded generation, timezone-less dates, offset dates, retained associations after human edits, and any historical-content path.
6. Verify both properties on the serialized response, not only on an in-memory object. Announce the additive fields to consumers that use strict response schemas. Confirm behavior across single-object and collection responses before claiming endpoint coverage.

The addition of these fields is a response-contract extension, not a change to the XML namespace or XSD. Extending other endpoints is an implementation task separate from this documentation update.

Mapping API and XLSX provenance

API	XLSX	Relationship
AIGenerated: true	A qualifying cell record with the current label	Both use a positive recorded generation association. An API object does not guarantee an XLSX cell record: the export also applies authorization, source matching, content presence, and cell tracking.
AIGenerated: false	No matching positive-generation assertion	Neither establishes human authorship. Missing XML metadata can also have other causes.
AIGenerationDate string	Optional generatedAt attribute	Both describe a source generation timestamp. API serialization

API	XLSX	Relationship
		reads <code>Copy.AIGenerationDate</code> ; XLSX serialization reads <code>SmallCopy.GenerationDate</code> . Source queries must map them to the same generation before equivalence is claimed.
<code>AIGenerationDate:</code> <code>null</code>	<code>generatedAt</code> omitted on a qualifying record	Both communicate an unavailable generation timestamp.

10. Relationship to Article 50(2)

Article 50(2) of the EU AI Act addresses machine-readable marking and detectability and requires technical solutions to be effective, interoperable, robust, and reliable as far as technically feasible. This specification makes Talkoot’s format interpretable by third parties. Publishing it is one supporting measure; publication and schema validation alone do not establish compliance or eliminate the integrity limitations above.

The Article itself does not prescribe this namespace or explicitly require an XSD. Applicability, label semantics, assistive-editing exceptions, and the adequacy of the complete marking approach require separate assessment. This document does not assert that the custom format is an approved standard or that Talkoot has signed an EU code of practice.

Sources: [European Commission, Article 50](#); [Commission transparency guidance](#). Reviewed 24 September 2026. The Article page notes amendments and should be read with the Commission’s current guidance.