ARCHITECTURE ARTICLE

Goodbye DITA Reuse. Hello DITA!

Agentic Content Reuse

How digital-twin knowledge graphs, inference, and constrained agents can deliver the benefits of content reuse without complex reuse libraries and author-facing indirection

Michael Iantosca
Senior Director of Knowledge Platforms and Engineering
Avalara

Executive premise

DITA reuse mechanisms solve legitimate enterprise content problems: duplication, consistency, controlled variation, update propagation, and maintenance cost. But the mechanisms are not the objectives. In an architecture with a digital twin of the content corpus expressed in RDF/OWL, a real inference engine, graph-derived vectors, and agents constrained by deterministic semantic services, those objectives can be addressed at the knowledge layer instead of through extensive conref libraries, key indirection, and reusable-fragment repositories.

The architectural shift is simple but consequential: DITA reuses content. A semantic digital twin can reuse knowledge.

Why DITA reuse exists

DITA reuse mechanisms emerged to solve a set of recurring problems in large documentation environments. When the same instruction, warning, prerequisite, product name, legal statement, or conceptual explanation appears in many places, independently maintaining each occurrence creates drift. A correction made in one location may not reach another. Product variants multiply the problem, and localization makes duplicated content increasingly expensive.

DITA addresses these pressures with mechanisms such as content references, key-based references, reusable topics, maps, key scopes, and conditional processing. These mechanisms can be highly effective, but they also introduce indirection. The source an author sees may no longer contain the text a reader ultimately sees. Understanding a topic can require traversing keys, scopes, reusable fragments, maps, and processing rules.

That complexity is often accepted because the alternative appears to be uncontrolled duplication. In a modern semantic architecture, however, there is a third option: retain the benefits of reuse while moving identity, dependency, applicability, and change propagation into a deterministic knowledge layer.

The six reasons organizations use reuse

The clearest way to evaluate whether DITA reuse mechanisms are still necessary is to separate the reasons for reuse from the mechanisms used to implement it. Most enterprise reuse programs are ultimately trying to achieve six outcomes.

Objective	Traditional DITA mechanism	Underlying business need
Single source of truth	conref, conkeyref, reusable topics	Avoid independently maintaining equivalent information.
Consistency	Shared source fragments	Keep facts, instructions, warnings, and terminology aligned.
Change propagation	Reference resolution	Ensure a change reaches every affected deliverable.
Contextual variation	Keys, key scopes, profiling, conditions	Resolve the correct information for product, version, audience, market, or situation.
Lower maintenance and localization cost	Reuse instead of duplicate source	Reduce repeated authoring, review, translation, and validation.
Controlled indirection	keyref and conkeyref	Separate semantic intent from a physical file or location.

None of these outcomes intrinsically requires a content reference. They require authoritative identity, dependency knowledge, applicability logic, controlled resolution, and reliable propagation. Those capabilities can be implemented elsewhere if another layer can provide them more completely and more deterministically.

A different architectural center of gravity

Consider an environment in which the entire content corpus has a digital twin represented as an RDF/OWL knowledge graph. Content components have persistent identities. Their relationships to products,

capabilities, APIs, interfaces, audiences, versions, code, requirements, tests, and other content are explicitly modeled. An inference engine can reason over those relationships. Graph-derived vectors provide semantic similarity and discovery without replacing the graph as the source of truth.

In that environment, the knowledge graph can become the enterprise reuse system. DITA remains valuable as a structured authoring and publishing representation, but it no longer has to carry every enterprise reuse relationship inside the documents themselves.

> *Keep semantics in the semantic system. Keep DITA structural.*

From file-level reuse to knowledge-level reuse

Traditional reuse starts with a piece of content. An author identifies a paragraph or element that should be shared, places or designates it as reusable, and creates references to it. The reusable object is therefore a textual or structural artifact.

Knowledge-level reuse starts one layer higher. The reusable object is the knowledge assertion, rule, concept, relationship, or behavior represented by the content. Multiple content instances may realize that same knowledge in different forms. Conversely, two passages that look nearly identical may represent intentionally different assertions because their product, audience, risk, or version contexts differ.

This distinction allows a semantic system to manage reuse without forcing every reuse decision to become author-visible markup. The graph can know that a particular warning, prerequisite, behavioral statement, or UI instruction is authoritative and applicable to a set of content instances even when each published topic contains fully materialized text.

How the semantic architecture replaces the major reuse functions

DITA-oriented function	Semantic and agentic equivalent
conref	Persistent canonical content or knowledge identity
conkeyref	Semantic relationship to a canonical knowledge node
keyref	Entity and reference resolution through the graph
Key definitions	Graph bindings between entities, contexts, and representations
Key scopes	Applicability context for product, version, audience, market, or configuration
Conditional processing	Inference-driven applicability resolution
Reusable-fragment library	Corpus-wide addressable knowledge graph
Find usages	Reverse-edge traversal and dependency queries
Reuse impact report	Graph dependency and change-impact traversal

DITA-oriented function	Semantic and agentic equivalent
Variable text	Property or entity-value resolution
Manual reuse discovery	Vector-assisted candidate discovery plus graph validation

Vectors discover candidates; the graph decides

Graph-derived vectors are particularly useful because they can discover potential reuse that has never been explicitly modeled. They can identify passages that appear semantically equivalent, detect near-duplicates, surface likely terminology conflicts, and locate content that may be affected by a product or conceptual change.

The vector layer should not, however, become the authority for reuse resolution. Similarity is probabilistic. Two passages can be close in vector space and still differ in applicability or meaning in ways that matter operationally. Candidate discovery and authoritative resolution should therefore be separate responsibilities.

- Vectors identify likely semantic matches and change candidates.
- RDF/OWL relationships establish identity, provenance, and explicit dependencies.
- The inference engine determines applicability and valid contextual resolution.
- Deterministic services constrain the exact content nodes or document regions that may be changed.
- Agents execute transformations within those constraints.
- DITA validation and product-specific tests verify the resulting structure and behavior.

This pattern retains the flexibility of AI while preventing probabilistic generation from replacing deterministic reference semantics.

Why this can be safer than replacing conrefs with an LLM

A semantic architecture should not replace deterministic DITA resolution with an instruction such as “find the right reusable paragraph and insert it.” That would exchange a deterministic mechanism for a probabilistic one. The stronger design is to use agents as bounded operators over a deterministic semantic substrate.

The inference engine and graph services should decide what is authoritative, what is applicable, what is affected, and what can be changed. The agent should receive only the relevant content components and the permitted operation. This sharply reduces the possibility that an update intended for one part of a document will alter unrelated content.

The resulting control model is especially valuable in enterprise documentation, where small unintended edits can create legal, safety, support, or product-accuracy consequences. Agentic flexibility belongs inside a deterministic scope, not in place of it.

Eliminating the special category of “reusable content”

One of the more consequential possibilities is that a mature digital twin can eliminate the need for a separate conceptual class called reusable content. In a file-centric system, authors often have to decide whether a fragment belongs in a reuse library, whether an equivalent fragment already exists, which key or source should be referenced, and what other topics could be affected by a change.

In a corpus-level graph, every meaningful component is already addressable. Reuse can be represented as a relationship among knowledge, contexts, and content instances rather than as a special storage location. The

system can discover that several paragraphs realize the same assertion, while also recording that one variant is intentionally divergent for a particular product or audience.

This changes reuse from an authoring task into a semantic governance capability. Authors no longer need to navigate a large reuse repository simply to obtain consistency. The system can surface canonical knowledge, flag unintended divergence, and constrain proposed changes based on corpus-wide context.

Materialized source without losing identity

Aggressive DITA reuse can make source difficult to understand because the author does not always see what the reader will see. A topic may contain references whose visible value depends on map context, key scope, or processing. The more indirection is layered into the source, the further the authored representation can drift from the final document.

A graph-centered approach permits fully materialized DITA while retaining semantic identity externally. The topic can contain the actual warning, instruction, or explanation that will be published. Behind it, the digital twin records which knowledge node the element realizes, its provenance, its applicability, and its dependency relationships.

The result is resolved source for authors and deterministic semantic traceability for automation. This is not uncontrolled copy-and-paste because the content instances remain linked to their canonical knowledge and can be evaluated or updated through graph traversal.

Change propagation becomes richer than “write once, use everywhere”

Traditional content reuse propagates a change because multiple locations point to one shared source fragment. That is useful, but it only captures explicit reference relationships. A semantic graph can calculate a wider class of impact.

A product behavior change, for example, may affect a conceptual explanation, one or more procedures, a warning, API documentation, troubleshooting guidance, release information, test cases, and learning content. These artifacts may not share any text at all. What connects them is the knowledge and product behavior they describe.

Graph traversal can therefore answer a more important question than “where is this paragraph reused?” It can answer “what content is semantically dependent on the thing that changed?”

A more capable change-impact model

- Exact replacement – a canonical statement changed and all equivalent instances require the same update.
- Contextual adaptation – the underlying knowledge changed, but the affected topic requires a context-specific expression.
- Review required – the graph establishes dependency but cannot deterministically authorize an automatic rewrite.
- Intentional divergence – an apparently similar instance is governed by a different applicability rule and should not change.

A reference architecture for agentic knowledge-level reuse

A practical implementation can be organized as a pipeline in which probabilistic capabilities are used for discovery and transformation, while deterministic services control authority and scope.

1. **Detect change** A code, product, requirement, UI, policy, or content change enters the system.
2. **Resolve entities and knowledge** The digital twin maps the change to product entities, capabilities, behaviors, concepts, and canonical assertions.
3. **Traverse dependencies** Graph queries identify content instances and related artifacts that are directly or inferentially affected.
4. **Discover additional candidates** Graph-derived vectors surface semantically related content not yet connected by explicit relationships.
5. **Apply inference** OWL reasoning and deterministic applicability rules determine which candidates truly apply to the current context.
6. **Constrain the operation** The orchestration layer passes the agent only the authorized content nodes or subsections and specifies permitted transformations.
7. **Materialize the update** The agent creates or modifies resolved DITA rather than inserting opaque reuse indirection.
8. **Validate** Schema validation, terminology rules, graph consistency checks, product tests, and other quality gates verify the result.
9. **Update the twin** The graph records the new content state, provenance, relationships, and downstream dependencies.

What should remain in DITA

Moving reuse into the semantic layer does not imply abandoning DITA. In fact, DITA provides two capabilities that are especially important when the content corpus serves as the materialized representation of a semantic digital twin:

1. **1. DITA provides the schema-defined semantic structure needed to create and continuously maintain an accurate digital twin.** Its typed topics, elements, attributes, metadata, and specialization provide explicit, machine-resolvable signals about what a content component *is*, rather than requiring the digital-twin system to infer structure and meaning from prose or presentation conventions. Those semantics can be deterministically mapped to entities, relationships, applicability, and other graph properties, allowing the knowledge graph to be generated, synchronized, validated, and updated automatically as the DITA corpus changes. This makes DITA particularly well suited to serve as the materialized content representation of a digital twin.
2. **2. DITA provides the semantic containerization and addressability required for precise knowledge-layer reuse.** Knowledge-level reuse depends on being able to associate a canonical assertion, concept, behavior, or relationship with the exact content components that realize it. DITA supplies bounded, typed, and independently addressable structures – from topics and sections to steps, warnings, properties, and specialized elements – that give the digital twin stable targets for identity, provenance, applicability, dependency, and change-impact relationships. The knowledge layer can therefore determine what knowledge is authoritative, where it applies, what content depends on it, and what must change, while agents operate only on the specific content components authorized for modification. This makes semantic reuse more precise and governable without requiring the reusable knowledge itself to be represented through content-reference indirection.

While a digital twin graph can be constructed from lightweight formats such as Markdown, AsciiDoc, or reStructuredText, those formats generally do not provide the same degree of explicit semantic typing, structural isolation, and addressability natively. Comparable precision requires additional conventions, enrichment, metadata, or human effort to establish the boundaries and semantics that DITA already

provides. That difference becomes particularly significant for knowledge-layer reuse, where reliable identity, dependency traversal, targeted change, and deterministic validation depend on knowing exactly what a content component represents and where its boundaries lie.

DITA should therefore remain responsible for the structured content capabilities it is particularly well suited to provide:

- Topics and task, concept, reference, or specialized semantic structures where they add value.
- DITA maps for deliverable assembly and navigation where they remain operationally useful.
- Semantic markup, metadata, and specialization needed by publishing and downstream systems.
- Schema validation and transformation pipelines.
- Interchange compatibility with partners, tools, and ecosystems that expect DITA.
- Limited key-based or traditional reuse where external interoperability or a specific publishing requirement makes it advantageous.

What can be questioned is not DITA itself, but the need for DITA to also function as the enterprise knowledge-resolution layer. Pervasive conref libraries, reusable-fragment repositories, deeply nested key scopes, and author-managed reuse indirection become less necessary when a more expressive semantic layer already provides persistent identity, canonical knowledge, context, applicability, inference, dependency management, and change propagation.

In this architecture, the separation of responsibilities is deliberate: **the digital twin governs knowledge; DITA provides its structured, addressable, and validated materialization.** That separation allows reuse to move to the knowledge layer without sacrificing the structural precision required to safely automate content creation, maintenance, and change.

Knowledge-level reuse extends beyond DITA

Moving reuse into the knowledge layer has another important consequence: enterprise reuse no longer has to be limited by the reuse capabilities of the underlying content format.

Traditional DITA reuse is powerful because DITA provides explicit mechanisms such as conref, conkeyref, keys, key scopes, reusable topics, and conditional processing. Formats with less sophisticated native reuse capabilities cannot easily provide the same degree of controlled reuse, applicability, dependency management, and change propagation on their own.

A digital-twin architecture changes that constraint because the reuse system no longer resides primarily in the source format. Persistent knowledge identity, canonical assertions, applicability, semantic dependencies, provenance, inference, and change-impact relationships are maintained in the knowledge layer. The content format becomes a materialization of that knowledge rather than the mechanism through which all reuse relationships must be expressed.

As a result, the same canonical knowledge can govern content across heterogeneous representations. A product behavior, requirement, prerequisite, warning, or conceptual assertion might be realized in a DITA topic, a Markdown README, an AsciiDoc document, HTML, API documentation, learning content, a support article, or another content system. Those instances do not need to share identical wording or participate in the same source-format reuse mechanism. The digital twin can establish that they realize or depend on the same underlying knowledge and determine which instances are affected when that knowledge changes.

This changes the scope of reuse substantially. Traditional content reuse primarily asks, *"Where is this content reused?"* Knowledge-level reuse can instead ask, *"Where is this knowledge expressed or depended upon?"* The second question crosses document boundaries, repository boundaries, and format boundaries.

> *Knowledge-level reuse is format-independent in scope but format-dependent in precision.*

DITA nevertheless remains particularly well suited to participate in this architecture. Its schema-defined semantics, typed structures, metadata, and addressable content boundaries provide the knowledge layer with precise targets for identity, dependency, applicability, validation, and automated change. Less structured formats can participate in the same reuse architecture, but comparable precision may require additional metadata, parsing conventions, enrichment, or semantic annotation.

In this sense, the architecture does more than replace traditional DITA reuse. It extends the benefits that made DITA reuse valuable – consistency, controlled variation, reliable change propagation, and reduced maintenance – across a broader enterprise content ecosystem, including formats that historically lacked DITA's degree of reuse. DITA remains an especially strong materialization layer, while the digital twin becomes the common semantic reuse substrate across heterogeneous content systems.

The larger strategic implication

DITA reuse was designed for an era in which the document and its processing architecture had to carry much of the intelligence about content relationships. A corpus-level digital twin changes that assumption. The graph can represent relationships that are difficult or impossible to express through content-reference mechanisms alone: semantic dependency, product behavior, code lineage, requirement traceability, test coverage, interface relationships, risk, lifecycle status, and inferred applicability.

Once those relationships exist outside the document, content reuse no longer needs to be primarily a file-resolution problem. It can become a knowledge-governance problem solved by graph semantics and executed through constrained automation.

This is also why knowledge-level reuse can go beyond the traditional promise of "write once, use many." The goal is not merely to repeat the same wording. The goal is to preserve the consistency of the underlying knowledge across every place where that knowledge must be expressed, even when the expression differs by context.

> *The future state is not "no reuse." It is reuse at a more fundamental layer – knowledge identity, semantic dependency, and applicability – with content materialized where people and publishing systems need it.*

Conclusion

Organizations should not retain complex DITA reuse architectures simply because reuse has historically been implemented through DITA. They should retain the outcomes that matter: authoritative knowledge, consistency, controlled variation, reliable change propagation, lower maintenance cost, and traceability.

Where a digital twin knowledge graph, formal ontology, inference engine, graph-derived vectors, and deterministic orchestration can deliver those outcomes, much of the traditional reuse machinery can become unnecessary. The semantic system can determine what knowledge is authoritative, where it applies,

what depends on it, and what must change. Agents can then execute narrowly scoped transformations against resolved content, and DITA can remain the structured representation rather than the enterprise knowledge-resolution engine.

That separation of concerns can reduce author-facing complexity while improving semantic control. It replaces a model centered on reusable fragments with one centered on reusable knowledge – and it creates a foundation for agentic content operations that are both more capable and more governable.

Terminology note: "Knowledge-level reuse" in this article refers to reuse governed by persistent semantic identity, graph relationships, inference, and constrained automation rather than by textual indirection alone.