Agentic AI for Technical Documentation

A real-world, experience-based review

PART THREE:

Agentic Documentation – People

Michael Iantosca
Senior Director of Knowledge Platforms and Engineering
Avalara

We have covered the process and technology dimensions of agentic documentation reasonably well – and I could easily spend another hundred pages on the implementation details. But technology is not the most important part of this transformation.

People are.

Agentic documentation succeeds or fails based on the people who understand the content, the product, the workflows, the risks, and the quality standards well enough to design and govern the automation.

The Shifting Ground Under Documentation Professionals

We cannot ignore the changing employment landscape for documentation professionals. Generative AI has made one part of the job – producing fluent text – dramatically easier to automate, and that has encouraged some organizations to assume that technical documentation itself can be automated just as easily.

There is a hard truth inside that assumption. Well-designed and well-governed AI, especially agentic AI, can reduce the amount of manual effort required to create and maintain documentation. Our own production workflows demonstrate that clearly.

But reducing manual effort is not the same thing as eliminating the professional discipline.

As of this writing, the U.S. Bureau of Labor Statistics projects 1% employment growth for technical writers from 2024 through 2034 and explicitly notes that AI tools may slow employment growth by making technical writers more productive. At the same time, BLS continues to describe the role as involving much more than writing – including research, collaboration with technical specialists, managing information flow, and maintaining consistency across functions.

That distinction matters. An LLM can generate prose. A documentation organization must still determine what needs to be documented, obtain and validate the source knowledge, decide how information should be structured and reused, manage terminology and metadata, coordinate with product development, verify technical accuracy, handle exceptions, maintain publishing infrastructure, and govern the final customer experience.

Some organizations have nevertheless reduced documentation staffing before equivalent automation, governance, and operating models were fully in place. The motivations vary by company, and it would be speculation to assign a single cause. But one management temptation is obvious: if AI can generate text, then writers may appear to be an easy place to demonstrate immediate productivity or head-count savings.

The danger is treating documentation as though text generation were the entire system.

A break-first-and-repair-later approach (called the “break-fix” management philosophy) can be especially costly in documentation because the damage accumulates quietly. Two conditions make that risk worse:

- Many documentation estates already carry substantial documentation debt from years of product change, inconsistent maintenance, limited staffing, fragmented ownership, and deferred cleanup.
- When documentation capacity is removed faster than an effective replacement operating model is introduced, that debt can grow rapidly – increasing the likelihood of outdated content, poorer self-service, more support demand, longer onboarding, inconsistent product guidance, and avoidable customer friction.

Those outcomes are not guaranteed, nor can every business impact be attributed to documentation alone. But they are predictable risks when a content supply chain loses the people who understand how it actually works.

And here is the paradox: documentation professionals are not merely potential users of agentic documentation systems. They are essential to designing them correctly.

Our own experience made that unmistakable. The role changes – sometimes substantially – but the expertise does not become less important. In many parts of the workflow, it becomes more important because human judgment is being encoded into automation that may operate across thousands of content objects.

Agentic Documentation Design

Any complex business process that involves multiple people, systems, dependencies, approvals, and exceptions requires domain expertise before it can be automated safely. Technical documentation is no different.

It is frequently trivialized because the visible output is a document or a web page. That creates the illusion that documentation is mostly a writing activity.

It is not.

Owning Microsoft Word does not make someone a technical writer any more than owning a digital camera makes someone Ansel Adams. And access to an LLM does not suddenly give a software engineer decades of expertise in information architecture, content strategy, structured authoring, editorial governance, terminology, reuse, localization, publishing, content operations, or user-centered information design.

High-quality technical documentation is produced by an ecosystem of people, technologies, standards, and processes. Depending on the organization, that ecosystem may include technical writers, information architects, content strategists, technical editors, documentation program managers, operations specialists, publishing and tooling engineers, terminologists, localization professionals, taxonomists, accessibility specialists, and product SMEs.

Many of the processes connecting those roles are only partially documented. Some are embedded in tools. Some exist as local conventions. Some are learned through years of experience. A surprising amount of the operating model resides as tacit knowledge in the heads of experienced practitioners.

That becomes critically important when we agentify the workflow.

An agentic system cannot safely automate a process that its designers do not understand. Every hidden decision, exception, dependency, quality check, escalation path, and handoff eventually becomes an architectural question:

- What should the agent do?
- What evidence does it need?
- What rules can be deterministic?
- Where is probabilistic AI appropriate?
- When must a human intervene?
- Who has authority to approve the result?
- What happens when the workflow fails?
- How is the decision audited and recovered?

Answering those questions requires the collective knowledge of the documentation profession.

If we were designing an agentic system for corporate finance, no serious organization would hand the project exclusively to AI engineers and tell them to infer accounting policy, controls, approvals, exception handling, and regulatory requirements as they went. Finance experts would be central to the design.

The same principle applies here.

Documentation professionals must become co-designers of the agentic content supply chain – defining the workflows, information models, quality criteria, terminology rules, taxonomies, ontologies, human-in-the-loop controls, exception paths, and governance mechanisms that determine how the system behaves. That is not a one-off effort; the expertise remains essential as models, tools, products, and workflows evolve.

That is the real shift. The profession is moving from performing every step of the content-production process manually toward designing, supervising, governing, and continuously improving systems that perform many of those steps automatically.

The question, therefore, is not simply whether AI will change the role of the technical writer. It already has.

The more important question is: what skills and responsibilities become more valuable when documentation itself becomes an agentic system?

What Has Become of Our Technical Documentation Professionals?

Enough conjecture. We designed, developed, and deployed an end-to-end agentic documentation supply chain, so we can now talk about what actually happened to the roles of the documentation professionals who helped build it.

We would not have succeeded to the degree we have without them.

I think back to a year before writing this paper, when I sat at the head of a very long table in a very large conference room for several days while we collectively designed the strategy that eventually became our agentic documentation production system. Around that table sat hundreds of years of combined experience – technical writers, information architects, content strategists, documentation operations specialists, platform engineers, localization professionals, software-development-lifecycle experts, user-experience practitioners, and business leaders.

It was humbling, even for someone approaching fifty years across several of those disciplines. More importantly, it made one lesson unmistakable: industrial-strength agentic documentation is a multidisciplinary design problem.

A word to the wise: skip that expertise at your own peril.

As the system evolved, so did the working relationship between our documentation platform engineering team and the documentation professionals. The engineering team owns implementation – orchestration, integrations, services, persistence, security, deployment, observability, and production reliability. But the documentation professionals are the internal customers and domain owners who define what the system must accomplish, what quality means, where humans must intervene, and what constitutes an acceptable result.

That distinction became one of the foundations of our operating model: engineering builds the machinery; documentation professionals define and govern the content system the machinery must support.

Shifting Knowledge Acquisition Upstream

One of the most consequential changes we made was to shift initial responsibility for supplying product knowledge upstream to the product-development organization.

That decision was not made by the documentation team alone. It was a leadership operating-model decision. If an organization intends to reduce the amount of manual discovery performed by technical writers, then the knowledge required to create accurate documentation has to enter the workflow somewhere else – preferably as part of the development process itself.

Historically, many documentation teams have spent substantial time reconstructing product knowledge from scattered sources: Jira tickets, code changes, design files, demonstrations, Slack or Teams conversations, meetings, prototypes, SME interviews, and partially complete specifications. Iterative development methods did not universally eliminate product specifications, but in many organizations they dispersed knowledge across smaller work artifacts and faster development cycles. The result was familiar to technical writers – become investigators first, then knowledge translators.

In our agentic model, we deliberately changed that relationship. Product teams are responsible for providing the minimum viable knowledge collateral required to automatically start the documentation workflow, and automated validation determines whether that collateral is sufficient.

> ***There is still no MCP connector to an SME's brain.***

The change does not eliminate collaboration between technical writers and SMEs. It changes when and why that collaboration occurs. Instead of beginning every project by hunting for basic facts, documentation professionals can spend more of their time validating, structuring, governing, and improving information that has already entered the system through an explicit process.

Development Ownership

We do not expect documentation professionals to become full-time software engineers. Some will learn Python, workflow scripting, prompt configuration, or low-code automation – and those skills can be extremely useful – but hands-on production engineering is not where most documentation professionals provide the most value.

Our agentic developers own the software implementation. They build and maintain the orchestration framework, deterministic services, APIs, integrations, state management, deployment pipelines, security controls, observability, and recovery mechanisms that make the system production worthy.

Lightweight automation platforms can be valuable parts of an agentic solution, especially for integrations and relatively bounded workflows. But the more the documentation supply chain becomes long-running, stateful, highly governed, and deeply integrated with enterprise systems, the more conventional software-engineering disciplines matter.

The documentation professionals therefore do not need to own the code. They need to own the domain artifacts, rules, models, quality standards, and operating decisions that tell the code what the documentation system is supposed to do.

What Documentation Professionals Now Own

In our model, the following responsibilities become more important, not less. Some existed before agentic AI. Others are new. Nearly all of them become more explicit once professional judgment has to be encoded into repeatable workflows.

Document Models

Documentation professionals define the information models the workflow is expected to create and maintain. That includes topic types, required and optional information, structural patterns, metadata expectations, reuse conventions, and relationships among content objects.

For structured environments such as DITA, these models can be grounded in schemas and specialization. In other environments, they may be expressed through templates, validation rules, content types, or other formal contracts. The key is that an agent should not be asked to invent the shape of good documentation on every run. The organization needs an explicit model of what “good” looks like.

Document Templates

Templates become machine-consumable design assets rather than merely starting points for human authors. They define expected sections, sequencing, labels, metadata, tone, and content patterns for guides, procedures, release notes, references, troubleshooting content, FAQs, API documentation, and many other deliverables and content types.

Documentation professionals own these templates because they understand the user need behind each structure. They also have to evolve them as products, delivery channels, and workflow capabilities change.

Agent Prompt Libraries

Prompts remain important, even in sophisticated agentic systems, but they should be treated as governed production assets rather than clever one-off instructions.

Documentation professionals are well positioned to define the content-specific instructions that guide generation, transformation, classification, review, and explanation. Prompt libraries need versioning, testing, ownership, change control, and clear boundaries between instructions that belong in prompts and rules that should instead be implemented deterministically.

We deliberately separated prompt libraries from compiled agent code and made them accessible through Model Context Protocol (MCP) resources and prompts precisely because these are continuously evolving assets that we do not want to embed and recompile frequently. That also meant providing an operational interface to manage an ever-growing governed library.

Agentic Assessment Criteria

Every validation or review agent needs an explicit definition of acceptable quality. Documentation professionals define the rubrics used to assess source collateral, technical completeness, topic quality,

procedural completeness, metadata, terminology, classification, and readiness to advance to the next workflow stage.

Those criteria should be measurable wherever possible. Vague instructions such as “make this complete” are difficult to test. Criteria such as required prerequisites, expected UI states, error behavior, supported platforms, or mandatory reference fields are much easier to automate and audit.

AI-Enabled Style Guides and Deterministic Rules

Traditional style guides were written primarily for people. Agentic systems need style guidance that can also be operationalized by machines.

Documentation professionals increasingly need to separate guidance into categories: rules that can be enforced deterministically, rules that can be evaluated probabilistically, terminology that should be controlled through approved vocabularies, and contextual guidance that still requires professional judgment. The result is a style system that can drive automated editorial checks without reducing editorial quality to a single LLM prompt.

Agentic Workflow Testing

Testing an agentic documentation workflow is not the same as testing a conventional application. The workflow must be tested for both deterministic correctness and probabilistic quality in addition to functional reliability and usability.

Documentation professionals should help design representative test corpora, golden examples, edge cases, adversarial cases, reuse scenarios, incomplete-input cases, and acceptance thresholds. They are also essential in judging whether generated content is merely fluent or actually useful, safe, technically complete, and appropriate for the intended audience.

Governance

Agentic documentation introduces new governance questions: which agents may modify authoritative source, who can approve publication, what evidence is required, how changes are audited, when human review is mandatory, which models may process which content, which parts of a stateful workflow require a human-in-the-loop (HITL), who those people are, and how failures are escalated. The multidisciplinary design team also determines which functions belong in stateful orchestrated workflows and which can safely operate as stateless autonomous agents.

Documentation governance therefore expands beyond editorial policy. It becomes part of the enterprise AI control framework.

Integration with the SDLC

Documentation work has to be triggered by product work, not discovered after the fact. Documentation professionals help define which development events create documentation obligations, what input collateral is required, which ticket states permit automation to proceed, and what documentation conditions must be satisfied before release.

This is where documentation becomes a first-class part of the development lifecycle rather than a downstream request queue.

Product Team Adoption and Use

A beautifully engineered workflow fails if product teams do not supply the required knowledge, respond to deficiencies, perform assigned reviews, or certify technical accuracy.

Documentation professionals therefore become change-management partners. They help define the operating procedures, train product teams, explain why the process exists, monitor adoption, and identify where the workflow creates unnecessary friction.

Agentic Process Metrics and Analytics

Once the workflow is digital and stateful, nearly every stage can produce operational telemetry. Documentation professionals should help determine which measures actually indicate health and quality – cycle time, input completeness, regeneration rates, human edit rates, exception frequency, editorial scores, approval latency, rework, and publication throughput.

The objective is not to measure everything. It is to measure the signals that help improve the content supply chain.

Agentic User Impact Analysis and Metrics

The ultimate purpose of documentation is not to generate content efficiently. It is to help users succeed.

Documentation teams therefore need to connect agentic production metrics with user outcomes such as search success, task completion, content helpfulness, support deflection, onboarding efficiency, findability, and recurring points of confusion. Correlation should not be mistaken for causation, but user-behavior evidence can reveal whether faster content production is actually producing better experiences.

Agentic Business Impact Analysis and Metrics

Agentic documentation also needs a business case that extends beyond head-count reduction. Documentation professionals and business leaders should measure the broader economics of the system, including speed and velocity, production capacity, time to publish, localization efficiency, support impact, compliance risk, reuse, maintenance cost, and the cost of poor or outdated information.

The important question is not simply “How many hours did AI save?” It is “What business capability and benefit did the new content supply chain create, and at what total cost and risk?”

Content Strategy

Content strategy remains foundational. Agentic generation makes it easier to create more content, which makes strategic discipline even more important.

Documentation professionals must decide what content should exist, which audiences and tasks matter, when content should be consolidated or retired, how it supports business and user goals, and where automation adds value versus merely increasing volume.

Information Architecture

Information architecture defines how content is structured, organized, labeled, related, and made findable. In an agentic environment, IA also determines how effectively machines can retrieve, classify, assemble, reuse, and update information.

Poor information architecture becomes an automation problem at scale. Strong information architecture becomes an enabling layer for reliable agentic behavior.

Content Delivery Design and Experience

The workflow may automate production, but users still experience the result through websites, in-product assistance, search, chat interfaces, developer portals, PDFs, APIs, support systems, or future channels that do not yet exist.

Documentation professionals remain responsible for designing the experience across those channels – including navigation, progressive disclosure, contextual delivery, readability, accessibility, and the relationship between generated answers and authoritative source content.

Knowledge Architecture – The Semantic Stack

Agentic systems need more than documents. They need machine-readable knowledge structures that describe what content means, how concepts relate, where information applies, and which source objects are authoritative.

This creates an expanded knowledge-architecture responsibility that includes terminology, taxonomy, ontology, and knowledge graphs. These layers are related but not interchangeable, and each contributes something different to retrieval, classification, reasoning, governance, and impact analysis.

Terminology

Controlled terminology defines the approved names for products, features, UI elements, concepts, actions, and domain-specific language. It improves consistency for users and provides agents with a governed vocabulary.

Terminology assets can also include preferred terms, deprecated terms, synonyms, abbreviations, definitions, product-specific usage, and relationships to localization requirements.

Taxonomy

Taxonomies provide curated classification schemes for organizing content by product, audience, task, lifecycle state, content type, domain, or other business dimensions.

In an agentic system, taxonomy is not merely navigation metadata. It can constrain retrieval, drive delivery, improve filtering, support analytics, and provide structured context for agents.

Ontology

Ontologies describe concepts and the formal relationships among them. They can represent relationships such as a product containing a component, a feature depending on a service, a command affecting a resource, or a user role performing a task.

When linked to documentation, an ontology provides domain context that cannot be recovered reliably from text similarity alone.

Knowledge Graphs

Knowledge graphs connect the content corpus to those concepts, relationships, metadata, identifiers, and source objects. In our architecture, the graph becomes a digital twin of the documentation estate and a critical layer for impact analysis, retrieval, provenance, reuse awareness, and machine reasoning.

Documentation professionals are essential to defining what the graph should represent because they understand both the content and the information relationships the business actually cares about.

Agentic Documentation Workflow Training

Training shifts from teaching only authoring tools and style rules to teaching people how to operate within a human-and-agent workflow.

Different participants need different skills. Product SMEs need to know how to provide usable collateral and certify accuracy. Documentation professionals need to know how to review agent output, interpret workflow state, resolve exceptions, improve agent assets, and recognize when automation should be overridden. Leaders need to understand what the metrics mean and where governance boundaries lie.

Content Management

Content management becomes more important as automation increases the rate at which content can be created and changed.

Someone must still govern source identity, versions, lifecycle state, branching, reuse, metadata, permissions, archival, deletion, and authoritative publication. Agentic systems amplify both good and bad content-management practices.

Documentation Operations Management

An industrial-strength agentic content supply chain requires ongoing operations management. Someone has to monitor queues, failures, quality trends, exceptions, model changes, service dependencies, throughput, publishing health, and human bottlenecks.

This is where documentation operations begins to resemble other mature production functions – observable, measurable, governed, and continuously improved.

Advanced Agent and Agentic Process Design

Perhaps the most significant new opportunity is for experienced documentation professionals to move directly into agent and workflow design.

They do not have to become the engineers who implement every service. They do need to understand enough about agents, models, tools, state, retrieval, deterministic controls, human-in-the-loop patterns, evaluation, and failure modes to translate documentation expertise into system behavior.

The best agentic documentation designer is not simply the person who knows the most about LLMs. It is the person who can connect AI capabilities with deep knowledge of content, users, workflow, quality, and risk.

Writing

Last but not least is writing. Yes, we still need writers for tasks and document types that have not yet been agentified – and there are many. Each document type requires time to design, build, evaluate, and operationalize the agents and supporting infrastructure. Until that capability exists, a traditional technical writer is still required, whether the work is fully manual or AI-assisted.

Even where product SMEs perform the first review of agent-generated drafts, a documentation professional remains necessary downstream for final QA, editorial refinement, exception handling, and the judgment required when content does not fit the automated path cleanly.

Post-Agentic Production (Pre-Publish)

Even the best agentic documentation workflows do not yet automate every post-production and publishing decision reliably. Localization aside, when a workflow generates a collection of topics for a new feature – or new topics for an existing guide – a human may still need to determine where that material belongs in the existing information architecture, how it affects navigation and reuse, and whether the surrounding collection also requires refactoring. Frontier models can assist with those decisions, but probabilistic recommendations alone are not sufficient when structural correctness and downstream dependencies matter. Greater automation will likely depend on stronger knowledge representations, explicit constraints, deterministic validation, and carefully governed human approval.

For a broader treatment of these evolving roles and responsibilities, see my earlier paper, The Future of Agentic AI Automated: Technical Documentation is Already Here.

The Role Is Changing, Not Disappearing

One of the clearest lessons from our implementation is that successful automation exposes responsibilities that were previously informal, distributed, or invisible. Once those responsibilities have to be encoded into a production system, somebody must define, test, govern, and continuously improve them.

Could software engineers do all of that without documentation professionals? They could certainly build software that generates documentation. That is not the same thing as building a reliable documentation system.

The strongest model we have found is a partnership: engineers bring the software architecture and production discipline; documentation professionals bring the content architecture, user perspective, quality model, governance, and domain workflow. Product teams contribute the authoritative product knowledge. Other specialists – localization, UX, legal, accessibility, support, data, and operations – join where the workflow crosses their domains.

The future documentation professional is therefore not merely an author using AI.

The role increasingly becomes that of a content-system designer, knowledge architect, quality governor, workflow operator, evaluator, and agentic-process designer – someone responsible for making sure automation produces information that remains accurate, usable, governed, traceable, and worth publishing.

Who else is going to manage all of that? The answer is not "Johnny Programmer" working alone – and it is not an LLM working alone either. It is the multidisciplinary team, with experienced documentation professionals at the center of the content decisions.

> **What do we call these new documentation professionals?**
>
> **I prefer Knowledge Architects.**

Selected References and Further Reading

Official standards and sources used to validate selected factual and architectural statements in this paper:

- U.S. Bureau of Labor Statistics – Technical Writers, Occupational Outlook Handbook
- Model Context Protocol – specification and documentation