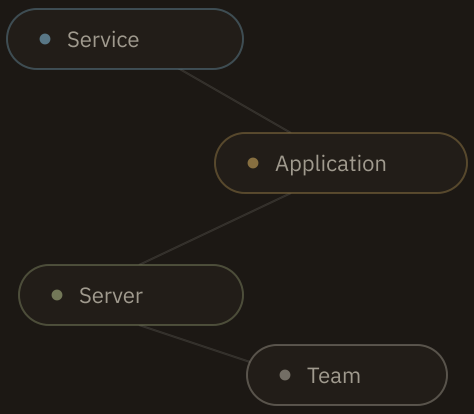
Palmetto Interactive

WHITEPAPER · KNOWLEDGE GRAPHS & ONTOLOGIES

What breaks if this fails?



Retrieval gives an AI text that looks relevant. It doesn't know what your business means by "customer." In regulated work, that meaning is what makes an answer auditable.

Matt Lucas · Founder & Principal, Palmetto Interactive
Twenty years across enterprise risk, finance and compliance technology, and enterprise knowledge-graph delivery.

OCTOBER 2026

01 · THE PROBLEM

Fluency is not truth

An assistant that answers confidently and can’t show its reasoning isn’t a productivity tool. It’s an unreviewed opinion with a good vocabulary.

You’ve probably run this experiment already. Point a good model at your documents, add retrieval, and the demo is genuinely impressive. Then someone from risk or audit asks the question that ends the project: *why did it say that?*

In a retrieval-only system the honest answer is that a few passages scored high on similarity, and the model wrote a good summary of them. That’s a correlation. An examiner can’t check it without re-running the model and taking its word again.

What’s missing here is structural. You need an explicit model of what things are in your business, how they connect, and where every fact came from. Bigger models don’t solve this. Longer context windows don’t either.

In regulated work, that layer is what separates a demo from something anyone will sign off on.

02 · THE COUNTER-ARGUMENT, TAKEN SERIOUSLY

When you don’t need any of this

Any paper arguing for knowledge graphs should start by saying when they’re the wrong answer. Most of the market won’t.

Retrieval is often enough. For narrow question answering over one document set with stable definitions, vector retrieval is cheap, fast, and appropriate. Adding a graph there would be architecture for its own sake.

Long context has genuinely changed the calculus. It’s become dramatically cheaper, and for reasoning inside one coherent document set it’s often the right minimal answer. It won’t resolve identity across systems or reconcile contradictions. The model still does that quietly, at inference time. But that only matters if your problem has those properties.

The strongest objection is the semantic layer. Governed metric layers in the lakehouse are real, well funded, and converging fast. The Open Semantic Interchange effort, started by Snowflake, Salesforce, and dbt Labs, is now incubating at Apache as Ossie. Snowflake’s

Semantic Views became SQL-queryable in March 2026.[1] For well-structured, tabular, metric-shaped questions, a certified semantic layer is the right answer, and the cheaper one.

WHERE THE COUNTER-ARGUMENT WINS

- One corpus, stable definitions, no cross-system identity problem
- Questions answerable as metrics computed from tables
- A known join path already exists in one schema
- No provenance, reuse, or reasoning requirement

WHERE IT DOES NOT

- The same real thing carries different keys in different systems
- Counting, aggregation, or *absence*. "Which systems have no owner?"
- Multi-hop chains across heterogeneous sources
- "As of March 1, what did we know?"
- An examiner needs the reasoning, not the confidence

So the defensible position isn't that every enterprise needs a graph. You need a *governed* semantic layer, and it can be graph-shaped or table-shaped depending on the problem. When the problem is really about identity, relationship, and provenance across systems that disagree with each other, only one of those shapes fits.

03 · WHAT BUSINESS MEANING REQUIRES

The ontology is the agreement

An ontology is a shared, explicit definition of concepts, relationships, and constraints that doesn't belong to any one system's schema. What's a customer. What's a control. What counts as an exposure. Agreed once, referenced everywhere.

Without one, your AI inherits whatever inconsistent vocabulary the underlying systems happen to use. Every downstream answer carries that inconsistency forward, quietly. The formalism isn't the point. The point is that your people and your machines finally work from the same definitions, and that's the whole reason this work is worth doing.

> *Shared definitions, not shared databases, are what let people and AI agree on what a fact means.*

In practice, an ontology engagement is a series of arguments that should have happened years ago. You hold them in one room and write the answers down in a form software can enforce. The modeling is the easy half. Getting three business units to agree which of their three definitions of "active account" survives is the hard half, and that isn't a technical problem at all.

Our companion paper, *Meaning, under version control*, covers how to build and maintain that agreement. Methods, reference ontologies, tests, and release practice.

04 · THE HARD PART NOBODY MARKETS

Entity resolution

Entity resolution means deciding that two records point at the same thing in the world. Across systems with no shared key, attributes that drift, and entities that merge and split as the business changes.

Modern language models made extraction cheap. They didn't make resolution cheap. That's the consistent report from practitioners through 2026, and it's the step most likely to be under-resourced in a plan. Extraction demos in an afternoon. Resolution for a single domain usually runs six to twelve weeks, and enterprise-wide ontology work runs to months.[2]

It's also where unsupervised automation is most dangerous. Ask a model to merge records and it will merge records, including ones it shouldn't. **A false merge is usually worse than a missed one.** A missed match leaves two records where there should be one, and you can see that. A false merge invents a relationship that never existed, and you can't.

WHAT THIS MEANS FOR A PLAN

Any proposal treating entity resolution as a configuration step is either inexperienced or selling you something. Budget it as the primary work of the engagement. Build human review into the resolution logic from the start, rather than bolting it on after a quality incident.

05 · PROVENANCE

Every fact remembers where it came from

A fact in a governed graph carries its source, its timestamp, the method that produced it, and a confidence signal. That metadata isn’t bookkeeping. It turns "the system said X" into "the system said X because this source asserted it on this date by this method, and here’s the chain."

Provenance is becoming a security control too, rather than only an audit convenience. As long-term agent memory moves into production, memory contamination and poisoning become a live attack surface. Knowing where a stored fact came from is how you contain it.[3]

06 · NAMING THE DIFFERENCE

Three things that get conflated

A lot of the disappointment in graph investments traces back to buying one of these and expecting another.

TERM	WHAT IT ACTUALLY IS
Graph database	A storage and query engine for nodes and edges. FalkorDB, Neo4j, Neptune, Stardog, TigerGraph, TypeDB. It carries no business meaning on its own. Buying one gets you a knowledge graph about as much as buying a relational database gets you a data model.
Knowledge graph	A graph database populated according to a governed ontology, with entity resolution performed and provenance attached. Engine plus meaning plus identity.
Semantic layer	A governed mapping from business terms to the underlying data. The lakehouse versions are metrics-first and tabular. A knowledge graph is a semantic layer too, tuned for relationships and identity rather than aggregation.

Standards, chosen pragmatically

RDF, OWL, and SHACL buy you portability across stores, formal inference, and constraint validation. They cost you heavier tooling and a steeper adoption curve. Property graphs buy you faster traversal at scale and a simpler developer model. They cost you formal inference, and the schema is specific to your implementation.

Pick RDF and OWL when you need to work across organizations, do formal reasoning, or match an industry-standard model. That's the ground RDF-native platforms like Stardog are built on. Pick a property graph when it lives inside one enterprise and traversal speed and developer ergonomics matter most. Either way, the two camps keep converging. RDF 1.2 reached Candidate Recommendation at W3C in April 2026. Neo4j made schema enforcement with graph types generally available on Aura in June. ISO GQL now runs in Spanner Graph and Microsoft Fabric, and PostgreSQL 19, now in release candidate, adds SQL/PGQ.[4] Constraint portability now matters more than which camp you joined.

07 · GRAPHRAG IN 2026

The economics changed. The hard parts didn't.

The original objection to graph-grounded retrieval was cost. Index-time community detection and hierarchical summarization gave excellent answers to broad questions, at a price that made finance uncomfortable.

That objection has largely dissolved. Deferred-construction approaches move graph building to query time. Microsoft Research's LazyGraphRAG, plus LightRAG and Fast GraphRAG. Microsoft reports indexing cost down to roughly 0.1% of full GraphRAG, and query cost up to 700× lower than GraphRAG global search at comparable answer quality.[5] That's still the biggest economic shift in this space.

The evidence on when graphs help also got more honest, and that's useful. The GraphRAG-Bench authors start from a plain premise. Graph retrieval often underperforms ordinary retrieval on real tasks. A head-to-head study found each approach has distinct strengths, and combining them helped consistently.[6]

A small October 2026 preprint over SEC filings makes the split concrete. Graph retrieval solved 16 of 20 questions, against 10 for text-to-SQL and 3 for vector search. It lost on structured aggregation, and the authors claim nothing beyond their own corpus.[6] That matches what we see. Graphs win on identity, multi-hop links, and provenance. They don't win everything.

Temporal knowledge graphs have separately become the standard substrate for agent memory. Bi-temporal models track both when a fact was recorded and when it was true in the world. That lets a system answer "what did we know on March 1", and resolve contradictions automatically as facts get superseded.[7]

> *Cost is no longer the reason not to ground AI in a graph. Entity noise and governance still are.*

What hasn't been solved deserves equal billing. Entity extraction accuracy still lands somewhere around 60 to 85%, depending on how domain-specific the material is. That's usable with human review and not usable unsupervised. Graph modularity keeps retrieval inside dense clusters and misses cross-cluster connections, and that's still an open research problem. A structurally complete graph also doesn't guarantee an agent walks to the *right* node. That takes governance on top of the structure rather than more structure.

08 · THE 2026 LANDSCAPE

Engines, memory layers, and the plumbing between them

The tool market moved a lot this year. Here's where the options we get asked about stand as of October 2026, and where each one fits.[8]

OPTION	WHERE IT FITS
Neo4j	The most mature ecosystem. Cypher 25 is the default, schema enforcement went GA on Aura in June, and filtered vector search is built in. Aura Agent adds a hosted agent endpoint with an MCP server. The safe default for a general-purpose operational graph.
FalkorDB	A low-latency graph that runs as a Redis module, known for its sparse-matrix design. Version 6.0 rewrote the engine in Rust and shipped in late September. Strong for agent memory and GraphRAG next to Redis. It's licensed under the SSPL, and its published speed comparisons are vendor-run.
TypeDB	A strongly typed database with inheritance across entities, relations, and attributes, and validation built into the schema. On the 3.x line since a full rewrite in Rust. Fits domains with rich, strict types. Its query language is its own, so you trade away GQL portability.
Cloud graphs	Neptune Analytics backs managed GraphRAG in Amazon Bedrock Knowledge Bases. Spanner Graph and Microsoft Fabric graph both speak ISO GQL. Good when your data already lives in that cloud. Each one ties you to it.
RDF-native stores	Stardog and its peers. The right ground for formal reasoning, industry standards like FIBO, and data shared across organizations.

Memory layers sit on top

A new layer has formed between agents and graph engines. These are open-source libraries that turn documents and conversations into a graph an agent can query.

cognee ingests data into a combined graph and vector store, with Neo4j recommended for production. Version 1.6 added conflict resolution for contradictory facts. We run it ourselves, as shared memory for our own agents. **Graphiti**, the open-source engine under Zep, builds bi-temporal graphs on Neo4j, FalkorDB, or Neptune.[9]

One signal runs the other way, and it's worth noticing. Mem0 removed its external graph-store integrations in April 2026, in favor of entity linking inside its vector store.[10] Not every memory problem needs a graph database. The ones we work on usually do, because identity and provenance are the point. Yours may not.

Choose for the exit

Kùzu was a well-regarded embedded graph database. In October 2025 its repository was archived and the team moved on.[11] Projects built on it had to switch. cognee now defaults to a community fork called LadybugDB, and Graphiti deprecated its Kùzu driver.

That's the practical case for keeping the ontology, mappings, and tests outside any one engine. With a portable model, changing the store is a migration. With the model locked inside the store, it's a rebuild.

How agents reach the graph

Every major engine now ships a Model Context Protocol server, so an agent can query the graph as a tool. Neo4j, FalkorDB, TypeDB, cognee, and Graphiti all have one. Maturity varies a lot. Graphiti's documentation says its reference server has no authentication and isn't meant for production. Neo4j's read-only mode depends on classifying each query, and its documentation warns some custom procedures can slip through.[12]

Treat these servers as a starting point. The scoped, typed tool contract in the next section is still your job.

09 · CONTROL, NOT JUST CONTEXT

How a graph constrains an agent

The usual framing is that a knowledge graph gives an agent better context. For anyone accountable for what that agent does, the more useful framing is that a governed graph *bounds* it.

You can scope tool contracts to typed traversals. An impact-analysis agent gets to follow `DEPENDS_ON`, `RUNS_ON`, and `OWNED_BY` edges and nothing else. That's a structural limit on what the system can reach at all, enforced by the data model. Prompt discipline isn't a control anyone should rely on.

It changes what an explanation is, too. When the traversal path is the answer's derivation, "show your work" stops being a metaphor. The path is inspectable, falsifiable, and reviewable by someone who doesn't trust the model. That's exactly the posture a validator should have.

10 · THE REGULATED CASE

Questions similarity cannot answer

BCBS 239 requires lineage from source systems, through aggregation, to risk reports. Assessed on accuracy, completeness, timeliness, and adaptability. A graph makes that lineage a queryable structure rather than a maintained document that goes stale between examinations.

Model risk management is arriving at the same requirement from a different direction. Validators expect inputs, assumptions, and lineage to be explainable, and that expectation is spreading to AI-driven decisioning. A similarity score won't satisfy it. A traversal will: this recommendation follows from this control, to this risk, to this incident, dated and sourced. And you can review it without re-running the model.

Four classes of question are structurally out of reach for vector similarity, not merely harder:

- **Aggregation and counting.** How many controls map to this risk. Similarity gives you passages, never counts.
- **Absence.** Which systems have no documented owner. A vector search can't return something that isn't there.
- **Multi-hop dependency.** What breaks if this fails, transitively, through three layers of infrastructure.

- **Point-in-time reconstruction.** What the record showed on a specific date, before a later correction.

These are traversal problems. No amount of embedding quality turns them into similarity problems.

11 · WHERE YOU ACTUALLY ARE

A maturity model, and the honest advice about it

Most organizations we talk to are at stage one or two. The useful conclusion isn't that they should be at stage five.



STAGE	WHAT IT LOOKS LIKE FROM THE INSIDE
1 · Fragmented	Definitions live in silos, spreadsheets, and people's heads. "Who is the customer" gets three answers from three systems. Lineage is tribal knowledge held by whoever has been there longest.
2 · Cataloged	A catalog or glossary exists. Definitions are documented but not enforced or connected across systems. Lineage is static documentation. Retrieval is deployed over documents and still cannot answer relationship or aggregation questions.
3 · Connected	An ontology exists for one priority domain. Entity resolution runs for priority types with human review. Provenance is captured for consequential facts. Graph-grounded retrieval is piloted for one workflow.
4 · Governed	Ontology and identity span domains under a versioned change process. Provenance and confidence are standard rather than exceptional. Agents operate through typed, permissioned traversal. Auditors can be shown the path.
5 · Adaptive	The graph evolves through a governed, largely automated change process informed by usage. Cross-domain reasoning and point-in-time reconstruction are routine. The graph is infrastructure other systems build on, not a project. New domains onboard in weeks.

Stage three on one high-value domain is almost always the right first move. You can reach it inside a single engagement, it produces evidence rather than architecture diagrams, and it sets the pattern that makes the next domain faster. Programmes that try enterprise-wide modeling before proving one domain tend to spend their credibility before they spend their budget.

What breaks if this fails

Business continuity at a large asset manager depended on knowing which critical business services rode on which systems, teams, and vendors. That knowledge existed. It was spread across spreadsheets, CMDB extracts, vendor contracts, and the memory of long-tenured staff. Answering a regulator’s dependency question took days of manual reconciliation, and what came out was an assertion rather than evidence.

> *Engagement led by Palmetto's principal in a prior role. Presented as professional experience, not as a Palmetto client result or endorsement.*

The model

The entities were unremarkable. Business services, applications, infrastructure components, teams, vendors, locations. The relationships did the work. A service `DEPENDS_ON` an application, which `RUNS_ON` infrastructure, which is `LOCATED_IN` a region. An application is `OWNED_BY` a team and `SUPPORTED_BY` a vendor. A team `ESCALATES_TO` another team.

Every edge carried provenance. Which system of record asserted it, as of when, and with what confidence. Hand-entered facts got flagged for review rather than quietly trusted alongside system-extracted ones. That distinction matters enormously when the output is going to an examiner.

The actual difficulty

The same application showed up as a system identifier in the CMDB, a business name in the application inventory, and a product name in the vendor contract system. Three identifiers, one real thing, no shared key. That reconciliation was the engagement. Everything else followed from it.

The result

"If this component fails, which business services are affected, and who do we call" became a bounded traversal returning in seconds. It had been a multi-day spreadsheet exercise. And because every edge carried provenance, the answer was reproducible and defensible. Evidence for a regulator rather than an assertion.

In the terms of the last section, that's a stage three moving to stage four. One domain, deliberately scoped, with provenance and versioning built in from day one because the eventual audience was a regulator.

13 · WHERE TO START

The first question is not technical

The most reliable predictor of whether this work succeeds isn't platform choice, model choice, or graph topology. It's whether you can name the decisions the knowledge layer has to support, and find the people empowered to settle what the terms mean.

So we start with a short list of competency questions. The specific things your business needs to ask and get a defensible answer to. If that list is easy to write, the engagement has a spine. If it's hard, that difficulty is the finding, and you want to hit it in week two of a diagnostic rather than month six of a platform build.

How Palmetto works on this

We start with the decisions that matter and the questions your knowledge layer has to answer, then model only as far as those uses justify. Every engagement opens with a bounded diagnostic ending in a decision brief. When the honest answer is that you shouldn't build a graph at all, that's what the brief says.

The ontology, decision log, mappings, quality rules, architecture, and governance playbook all go to your team to keep using. We document the tool and platform choices too, so the model never gets trapped inside an implementation nobody can explain.

Palmetto Interactive palmettointeractive.com · support@palmettointeractive.com

REFERENCES & NOTES

1. Open Semantic Interchange, launched September 2025 by Snowflake, Salesforce, dbt Labs, and partners. Renamed Apache Ossie (Incubating) in July 2026, with no final release as of October 2026. Snowflake Semantic Views, standard SQL querying GA in March 2026. Vendor

announcements.

2. Practitioner reporting on entity-resolution effort, 2026. The timelines cited (6 to 12 weeks per domain, 2 to 3 months for enterprise-wide ontology work) are practitioner estimates from industry commentary rather than audited benchmarks. Read them as directional.
3. Research on memory contamination and poisoning in long-term memory-augmented language models, 2026.
4. W3C, RDF 1.2 Concepts, Candidate Recommendation Snapshot, 7 April 2026. SPARQL 1.2 and SHACL 1.2 remain Working Drafts as of October 2026. Neo4j Aura release notes, June 2026, graph-type schema enforcement GA. ISO/IEC 39075:2024 (GQL), as documented by Google Spanner Graph and Microsoft Fabric. PostgreSQL 19, at release candidate in October 2026, adds SQL/PGQ (ISO/IEC 9075-16).
5. Microsoft Research, *LazyGraphRAG*, November 2024. Cost and query-efficiency figures are as reported by Microsoft Research, for global queries on one news dataset with synthetic questions. Independent replication is limited.
6. GraphRAG-Bench, arXiv:2506.05690, June 2025. "RAG vs. GraphRAG," arXiv:2502.11371, 2025. Knowledge graph over SEC EDGAR filings, arXiv:2610.37728, October 2026, a preprint scored by a language-model judge on 20 questions and not yet peer reviewed.
7. Zep / Graphiti, bi-temporal knowledge graph architecture for agent memory. arXiv:2501.13956. Reports 94.8% on the Deep Memory Retrieval benchmark against 93.4% for the prior state of the art. These are the authors' own evaluation results.
8. Neo4j release notes, 2026, and Aura Agent launch, February 2026. FalkorDB v6.0.0 release notes, 29 September 2026. License per the project README. FalkorDB's published comparison with Neo4j (December 2024) is vendor-run, with tested versions unstated, and hasn't been independently replicated. TypeDB 3.14.1, October 2026. AWS, Bedrock Knowledge Bases GraphRAG GA, March 2025. Google, Spanner Graph GA, January 2025. Microsoft, graph in Fabric GA, June 2026.
9. cognee v1.6.6, 29 October 2026, and its graph-store documentation. Graphiti v0.31, October 2026. Both Apache-2.0.
10. Mem0 v2.0.0 release notes, 16 April 2026.
11. Kùzu repository archived, final release v0.11.3, 10 October 2025. *The Register*, 14 October 2025. LadybugDB fork, v0.22.1, October 2026.
12. Neo4j MCP v2.0.0, October 2026. FalkorDB MCP server v1.3.0. TypeDB MCP v1.0.2. cognee-mcp. Graphiti MCP server README, which describes it as a reference implementation.

Extraction-accuracy ranges and production-failure rates in this text come from industry commentary rather than audited studies. Treat them as directional indicators of a pattern rather than measurements. Where a figure materially affects a decision, we source it specifically during the engagement instead of relying on published estimates.