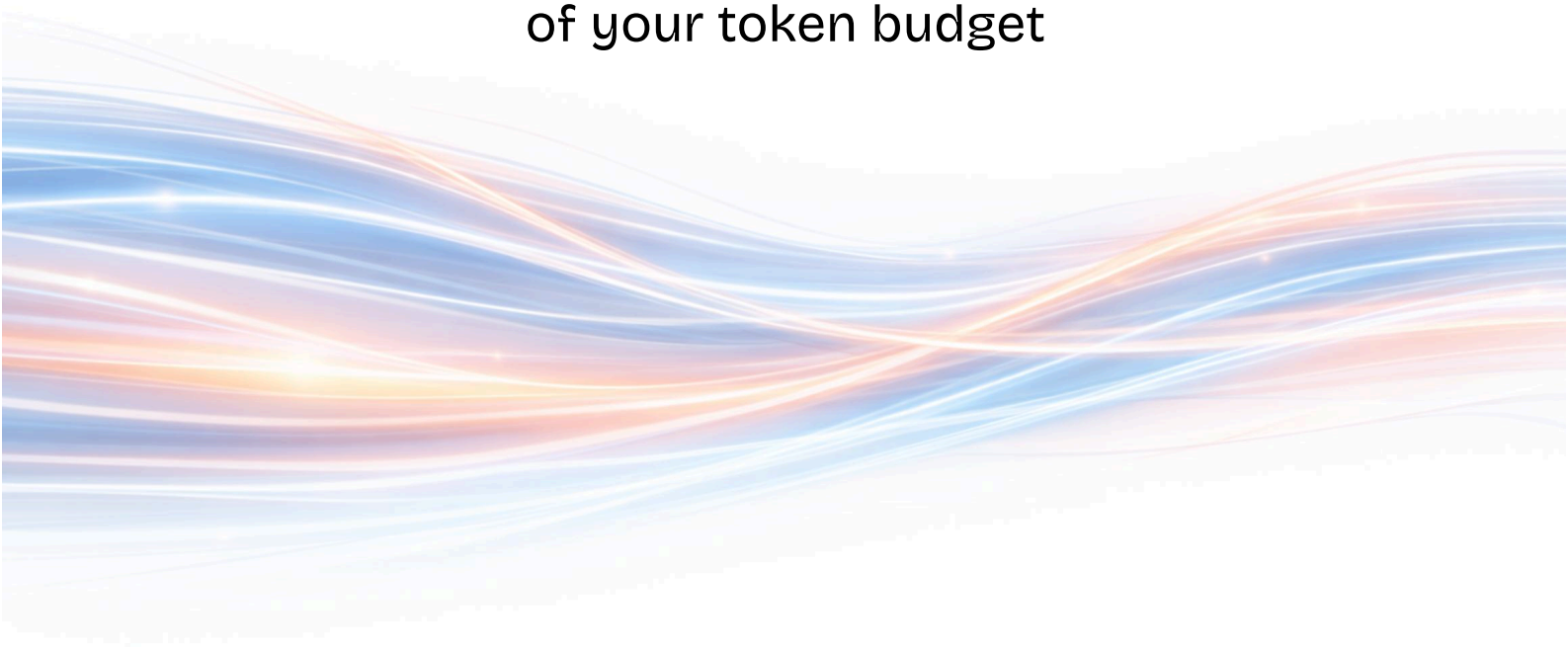
Whitepaper for CIO's

The Vector-Database Tax

How one flaw in AI search returns the wrong answer **and** wastes up to a quarter of your token budget



Aspected

The Vector-Database Tax

*How one flaw in AI search returns the wrong answer **and** wastes up to a quarter of your token budget and how we fixed it with a new retrieval index that fuses metadata as a ranking signal, not a bolted-on filter, so retrieval returns the **correct** answer, not just the similar one.*

Full technical working, every assumption and source shown.

> **The idea in one sentence**
> Every RAG system treats filtering and ranking as two separate, warring steps. Aspected fuses metadata into the vector itself, so they become **one operation,** retrieval ranks on meaning, date, authority and permission together, in a single pass. No filters, no rerankers, no loop. The accuracy and cost gains follow from that one structural change.

Part I — The idea: make filtering and ranking the same operation

Today's retrieval does two things with two different mechanisms that pull against each other.

Ranking is done by vector similarity: embed the query, find the nearest chunks. It understands meaning, but it is blind to metadata. It has no native sense of whether a result is recent, authoritative, permitted, or critical.

Filtering is done by a separate gate: keep only the rows where date, author, tenant, or permission match. It understands metadata, but it is a blunt yes/no that knows nothing about relevance.

Because these are separate steps, they fight. Filter before ranking and you can discard relevant results before they are ever scored. Filter after and the top-N cutoff can exclude the right answer. Teams paper over the conflict with rerankers, hybrid search, graph traversal, and increasingly agentic loops that re-query until something usable comes back. Each layer is a reasonable patch. None removes the root cause and because ranking sees only similarity, the system can return the answer that merely **sounds** right. The superseded policy, the draft, the merely urgent-sounding ticket while the correct one is never surfaced.

The structural fix

The root cause is that **metadata lives outside the vector**. Aspected changes the data structure, not the query logic: for each document, the content vector is f**used with metadata aspect vectors into a single multi-aspect vector**, in a way that preserves each aspect's distinct contribution. Date, authority, criticality and permission stop being gates applied after search and become **dimensions the search ranks on directly**.

Filtering, in other words, becomes ranking. A constraint that used to be a yes/no gate is now a weighted aspect of one similarity computation. There is nothing left to filter, because the filter criteria are already inside the thing being ranked.

> **Why this is a new index structure, not a clever query**
> You cannot get this behaviour by writing a smarter query on a conventional vector store, because in those stores metadata simply is not in the vector. It sits in a side table the similarity math never sees. Aspected changes what is stored and how it is indexed. That is why it is a distinct architecture.

What the change buys

Because every aspect is resolved in one pass, the consequences compound from a single decision:

Typical RAG pipeline	Aspected
Metadata sits outside the vector	Metadata fused into the vector as weighted aspects
Filtering and ranking are separate, conflicting steps	Filtering is ranking, one operation
Multiple stages: search → filter → rerank → trim	Single unified, multi-aspect search
Loops and re-queries to recover relevance	One pass, no loop
Ranks on similarity — can return similar-sounding misses	Ranks on correctness across all aspects
Token cost rises with query difficulty	Token cost stays flatter as queries get harder

Two things that normally trade off, accuracy and cost, improve together, because both were victims of the same structural flaw. The accuracy gain is immediate and architectural; the cost gain, quantified next, follows from it.

Part II — The corollary: what it is worth on a $10M LLM budget

What this estimate is, and is not

The figures below are an **illustrative model**, not a measurement. Every input is a published third-party benchmark, cited inline; none of Aspected's own performance figures are used as inputs (our accuracy and efficiency claims are pending independent validation with TNO). The output is a range, never a single guaranteed number, and it scales linearly with spend.

Aspected changes retrieval, not the generation step, so the reachable saving is the retrieval-driven portion of the bill plus the reasoning loops retrieval triggers. It comes from two levers, applied in sequence so there is no double-counting:

Lever 1 — Fewer LLM calls. With filtering and ranking unified, the complex queries that today escalate to an agentic re-query loop resolve in a single pass, removing the repeated, billed calls.

Lever 2 — Leaner context per call. Precise single-pass retrieval injects the right chunks once instead of padding the prompt with several documents.

Every assumption, with its source

Each input is a Conservative / Base / Aggressive range. The Base column sits at or below the midpoint of every cited benchmark, deliberately, so the estimate cannot be accused of using the high end.

Input	Cons.	Base	Aggr.	Basic & Public Source
Retrieval / RAG share of total LLM spend	45%	55%	65%	Retrieved context adds 2,000–10,000 tokens/query; query-time retrieval cost can exceed index-time within weeks. (MindStudio, Feb 2026; MyEngineeringPath, Mar 2026)
Complex-query share (needs agentic loop)	30%	35%	40%	Single-shot RAG handles ~60–70% of enterprise queries; 30–40% need iterative retrieval; a second source puts classic RAG at 70–80%. (CrackingWalnuts, Mar 2026; Tensoria, Apr 2026)
Agentic cost multiple vs single-shot	1.5x	2.3x	3.0x	Agents add ~2–5× cost; one agentic request may trigger 3–10 LLM calls. Our Base (2.3×) is below these midpoints. (CrackingWalnuts, Mar 2026; Tensoria, Apr 2026)
Input-token reduction from precise retrieval	20%	35%	55%	Tighter retrieval (2–3 chunks vs 4–8 docs) can cut input tokens >50% with no loss of precision. (Silicon Data, 2026)
Input-token share of retrieval-driven cost	55%	60%	65%	Context (input) typically dominates retrieval-heavy calls. (LLM token pricing guides, 2026)

A sense-check from outside the model
Independent LLM-spend reviews routinely find 22–48% of consumption recoverable through retrieval and prompt optimization (Enterprise LLM Cost Comparison, Feb 2026). The Base result below, 25% of total spend, lands inside that independently-cited band.

The arithmetic, step by step (Base case, on $10M)

Let T = total spend, r = retrieval share, c = complex-query share, m = agentic multiple, u = input-token reduction, k = input-token share.

Step 1 — Retrieval-driven spend: T × r = $10M × 55% = $5.5M.

Step 2 — Split by query type, weighted by cost: complex share of spend = c·m ÷ (c·m + (1−c)). Complex = $3.04M; simple = $2.46M.

Step 3 — Lever 1 (unify, remove the loop): complex single-pass cost = $3.04M ÷ 2.3 = $1.32M. Saving = $1.72M.

Step 4 — Lever 2 (leaner context): spend after Lever 1 = $2.46M + $1.32M = $3.78M. Saving = $3.78M × 60% × 35% = $0.79M.

Step 5 — Total: $1.72M + $0.79M = $2.51M saved per year. 46% of retrieval-driven spend, 25% of the $10M budget.

The three scenarios

Result	On $10M spend	On $25M spend	On $100M spend
Conservative	$1.0M / yr	$2.5M / yr	$10.2M / yr
Base	**$2.5M / yr**	**$6.3M / yr**	**$25.1M / yr**
Aggressive	$4.2M / yr	$10.5M / yr	$41.8M / yr
Base as % of total spend	25%	25%	25%

Even the **Conservative case** — every input at its cautious end — returns about **$1M a year per $10M of spend**, scaling linearly. The spread to the Aggressive case reflects genuine uncertainty in the inputs, which is why we publish all three rather than one.

Part III — Why it matters, and what to check before citing a number

Aspected's approach delivers:

- **Higher accuracy.** Ranking results on the right aspect returns the correct result rather than the most similar-sounding one.
- **Lower token cost.** Fewer LLM calls and less re-stuffed context mean a smaller, more predictable bill — and cost that rises far more gently as queries get harder.
- **Less to operate.** One index replaces a multi-system pipeline, reducing the integration points to build, run, secure, and keep in sync.
- **Sovereign by design.** Aspected runs in your own tenant and respects existing access controls, so the efficiency gain does not come at the cost of data governance.

Limitations, read before citing a number

- **Modeled, not measured.** Part II derives from public benchmarks plus arithmetic. Aspected's own results (e.g. the AGFA deployment) are excluded as inputs and remain subject to independent validation with TNO.
- **Retrieval ≠ the whole bill.** Only the retrieval-driven portion and the reasoning loops it triggers are addressed; pure generation spend is untouched.
- **Workload-dependent.** An enterprise with few complex queries or an already-lean pipeline sits toward the Conservative end. A real estimate needs the customer's own query mix and token telemetry.
- **No switching costs modeled.** This is gross token saving; it excludes migration effort and any change to the generation model.

Conclusion

The conclusion a CIO and architect should reach

The idea is sound, unifying filtering and ranking in a new index structure the inputs are public and conservatively chosen, and the arithmetic is shown in full. For an enterprise spending north of $10M a year on LLMs, better answers and a simpler stack, alongside a saving on the order of **$1M (cautious) to $2.5M (base) per $10M of spend**, is a reasonable, defensible expectation, to be confirmed against your own telemetry and the pending TNO benchmark.

About Aspected

Aspected is a new kind of database for enterprise AI developed by Xillio.

Its mission is simple:

Improve the reliability of enterprise AI by improving the knowledge layer underneath it.

Grounded by Aspected

https://aspected.com

Sources

All inputs are drawn from the following public references, used at or below their cited midpoints.

1. Single-shot vs agentic query mix, agentic cost multiple — CrackingWalnuts, “RAG and LLM Platform at Scale” (Mar 2026); Tensoria, “Agentic RAG” (Apr 2026); Redis, “Agentic RAG” (2025).
2. Retrieved-context token volume and retrieval share of spend — MindStudio, “Token-Based Pricing for AI Models” (Feb 2026); MyEngineeringPath, “LLM Token Pricing Comparison 2026” (Mar 2026).
3. Input-token reduction from tighter retrieval — Silicon Data, “Understanding LLM Cost Per Token: 2026” (2026).
4. Independent recoverable-spend sense-check (22–48%) — “Enterprise LLM Cost Comparison 2026” (Feb 2026).
5. Peer-reviewed efficiency results on reducing retrieval operations — L-RAG, arXiv 2601.06551 (2026).
6. Patent — US 12517879, “Multi-aspect vector search index creation and retrieval method and system”; EP application PCT/EP2025/052149.

Note: *Aspected is new kind of database for enterprise AI. The model behind Part II, with live formulas you can change against your own numbers is available on request. Independent benchmarking of our speed and accuracy claims is underway with TNO (Netherlands Organisation for Applied Scientific Research).*

Learn more or schedule a session with our experts? Contact us:

sales@aspected.com

+31 (0)35 622 95 45

www.aspected.com

www.linkedin.com/company/aspected/

Aspected