USE CASE 02 · FOR CDO · CAIO · AI PLATFORM LEADS

HYBRID RETRIEVAL. ONE QUERY. ONE HOP.

Agent performance at enterprise scale is a retrieval problem before it is a model problem. XERJ runs BM25 and HNSW vector search in the same process, against the same segment cache, through a single Elasticsearch-compatible API. No Python orchestrator. No second network hop. No dedicated vector bill.

4× RAG MEMORY · 38 MS P95 ON 1 B VECTORS · ONE QUERY

02 · THE RAG DATA PROBLEM

RAG STALLS AT THE DATA LAYER.

Every RAG pilot works. Most production rollouts don't. The failure is rarely the model. It is almost always the data layer beneath it — fragmented across a vector DB, a full-text store, a session cache, and a retrieval orchestrator that glues the three together in Python.

LATENCY COMPOUNDS	A single retrieval turn crosses three data stores through a Python orchestrator doing RRF. The agent sees the sum of every system's p99. Users see a product that feels slow even when individual queries look fast.
MEMORY EXPLODES	10 M embeddings at 1,536 dim is 92 GB of float32 — before you factor in HNSW graph overhead. Enterprise RAG workloads routinely land in the hundreds of GB of hot memory per tenant.
COSTS MULTIPLY	Per-vector pricing for the embedding store. JVM heap for the keyword index. Redis for session memory. LLM tokens for every orchestrator call. Four bills, four renewals, four procurement cycles.
AGENT MEMORY IS ORPHANED	Conversation turns, tool traces, retrieved context — where do they live? Usually in Redis until the TTL fires. Time-decay, token-budget-aware retrieval ends up as application glue code three teams own.

ONE DATA PLANE · NOT THREE SUBSYSTEMS AND A FRAMEWORK

03 · UNIFIED AI DATA STACK

VECTOR. KEYWORD. ONE ENGINE.

XERJ runs HNSW vector search and BM25 full-text in one process, against one segment cache, through a single ES 8.13 wire-protocol API. SQ8 quantization is default — 4× memory reduction with < 0.5 pp recall loss on MS MARCO. Optional float32 re-rank closes the gap.

EFFICIENCY

4–5×

LESS MEMORY AT AI SCALE

10 M × 1,536-dim fits in 18 GB under SQ8 vs 92 GB for float32 in ES + Pinecone.

LATENCY

38 MS

HYBRID P95 1 B VECTORS

BM25 + vector + aggregation in one query tree, one execution pass, one process.

UNIFIED

1 HOP

AGENT TURN NO RRF GLUE

Reciprocal rank fusion is native — no Python orchestrator, no second network call.

MEMORY

NATIVE

TIME-DECAY TOKEN-BUDGET

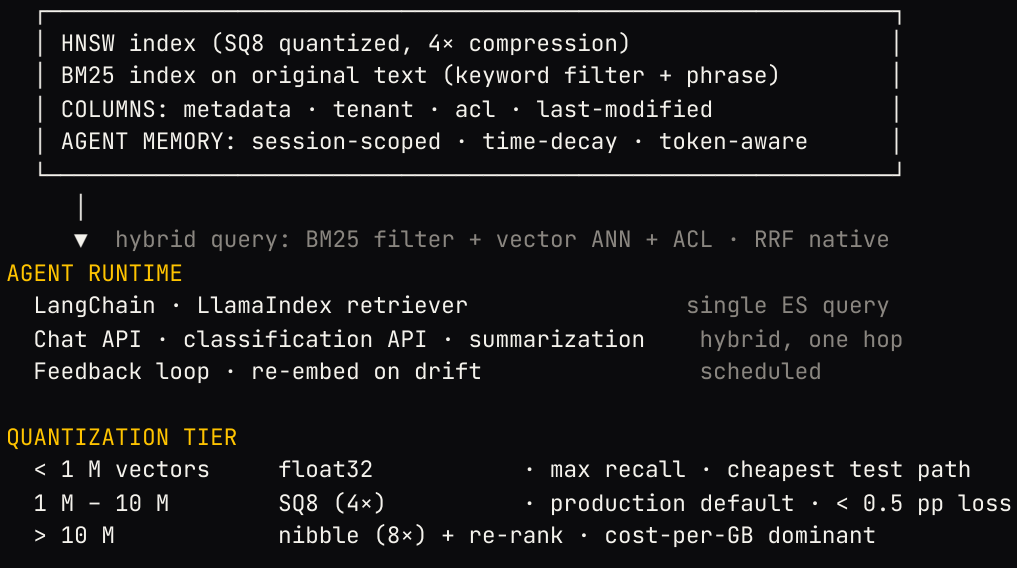
Agent memory (sessions, traces, retrieved context) is a first-class store — not glue code.

ES 8.13 WIRE PROTOCOL · LANGCHAIN · LLAMAINDEX · DSPY WORK UNCHANGED

04 · REFERENCE ARCHITECTURE

RAG BLUEPRINT. ONE PLATFORM.

A three-node XERJ cluster serving an enterprise RAG workload. Same ES-compat API used by LangChain / LlamaIndex retrievers and by the custom chat application.



QUANTIZATION TIER

< 1 M vectors	float32	· max recall · cheapest test path
1 M – 10 M	SQ8 (4×)	· production default · < 0.5 pp loss
> 10 M	nibble (8×) + re-rank	· cost-per-GB dominant

SIZING · ~55 M × 768-DIM / NODE · HYBRID P95 < 40 MS

05 · WHY THIS SCALES FOR RAG

ENGINEERING CHOICES THAT PAY AT SCALE.

Six decisions that matter for an enterprise RAG workload. Each is a measurable property of the data plane — not a framework to configure.

SQ8 BY DEFAULT	Scalar 8-bit quantization runs through the HNSW traversal, not just at rest. Graph RAM shrinks 4× too. Re-rank with float32 on the top-K candidates closes the < 0.5 pp recall gap.
NATIVE HYBRID	Reciprocal Rank Fusion is a planner primitive, not a Python wrapper. BM25 and vector ranks combine in-process; no calibration problem between BM25 float and cosine float.
ACL IN THE QUERY	Tenant / role / document-level ACL is a filter clause on the same hybrid query. No separate permission service, no second round trip, no post-filter leak.
EXPLAIN-PLAN	`POST /v1/indices/rag/explain-plan` returns per-node cost and row-scan estimate for every retrieval. Capacity planning, cost visibility, and outlier debugging share one JSON shape.
AGENT MEMORY NATIVE	Time-decay scoring, token-budget-aware retrieval, session-scoped indices — all native operators. No Redis TTL story, no glue code to maintain across three teams.
COST VISIBILITY	Every query returns realized CPU, I/O, and row counts. Finance audits AI spend at the query level; the CFO sees embedding retrieval cost as a line-item, not as an end-of-quarter surprise.

06 · MEASURED OUTCOMES

VECTOR MEMORY AT ENTERPRISE SCALE.

Memory footprint for hybrid retrieval indices at the three corpus sizes that typically decide hardware plans. XERJ under SQ8 default vs. legacy ES + dedicated vector DB in float32.

CORPUS	XERJ (SQ8)	ES + VECTOR DB (FLOAT32)
1 M × 768-DIM	1.2 GB	4.6 GB — 3.8×
5 M × 768-DIM	5.8 GB	23 GB — 4.0×
10 M × 1,536-DIM · ENTERPRISE	18 GB	92 GB — 5.1×
HYBRID QUERY P95 (1 B VECTORS)	38 ms	orchestrator adds 30 + ms
RECALL @ 10 VS FLOAT32	< 0.5 pp loss	baseline
NETWORK HOPS PER TURN	1	2–3 (RRF orchestrator)

COST IMPACT	10 M × 1,536-dim RAG corpus fits on one node under SQ8. The same corpus on ES + Pinecone floats needs a multi-node cluster plus per-vector cloud billing.
AGENT IMPACT	A 10-hop agent turn completes in ~400 ms including LLM roundtrip. The same chain over orchestrated ES+Pinecone+Redis regularly exceeds 2 seconds at the same corpus size.

07 · YOUR RAG STACK, UNCHANGED

LANGCHAIN. LLAMAINDEX. DROP-IN.

XERJ exposes ES 8.13 semantics. Every RAG framework with an ES retriever works as-is. Embedders plug in through a pluggable provider interface.

CATEGORY	TOOLS	STATUS
RAG FRAMEWORKS	LangChain · LlamaIndex · Haystack · DSPy — ES retriever / backend · zero adapter code	GA
EMBEDDING PROVIDERS	OpenAI · Anthropic · Voyage · Cohere · Bedrock · Azure OpenAI · self-hosted · pluggable	GA
CLIENT LIBRARIES	elasticsearch-py · elasticsearch-js · elasticsearch-java · Go · .NET via official ES clients	GA
DASHBOARDS	Kibana · Grafana · custom · real-time query shape, cost, recall-drift dashboards	GA
AGENT FRAMEWORKS	Anthropic Agent SDK · OpenAI Agents · LangGraph · CrewAI — all via ES-compatible retriever	GA
INGEST	ES _bulk · NDJSON · Vector · custom embedding pipelines via SDK · drift detector (roadmap Q3)	GA
IDENTITY	OIDC / SAML · AWS IAM · Azure AD · Okta · per-tenant index isolation	ROAD Q2
MLOPS	MLflow model registry · W&B traces · recall-monitoring hooks · scheduled re-embed	ROAD Q3

08 · GOVERNANCE

RAG COST. BOARD-READABLE.

AI infrastructure is where finance loses visibility fastest. XERJ makes retrieval-layer cost a queryable metric — not an end-of-month vendor surprise.

PER-QUERY COST	Every retrieval returns CPU, I/O, and row-count estimates before execution and realized counts after. Dashboards query the same field agents query.
CIRCUIT-BREAKERS	Per-request heap · per-request time · per-tenant concurrency · per-index doc-scan ceiling. Runaway agents fail loud, not expensive.
TENANT ISOLATION	Per-tenant indices, quotas, and explain-plans. Noisy-neighbor agents cannot starve another tenant's retrieval budget.
DATA SOVEREIGNTY	Self-hosted. Your cloud, your VPC, your region. Embeddings never leave the perimeter unless an embedder is configured to an external API.
AUDIT TRAIL	Every retrieval logged with principal, query shape, and realized cost. Replayable. Exportable. Queryable as a regular XERJ index.
DRIFT & RECALL	Query-shape dashboard surfaces recall drift, latency outliers, and re-embed candidates. Scheduled monitoring hooks `Q3`.

COST AS DATA · NOT A POST-HOC BILL RECONCILIATION

09 · FROM PILOT TO PRODUCTION

FIVE PHASES. ONE YEAR.

A reversible rollout. Start with one RAG corpus on a single node; end with a consolidated retrieval platform replacing the vector-DB contract, the session cache, and the keyword store.

01	**PILOT** WEEKS 1–2	Single node · embed one corpus (~1 M docs) · run the RAG_BENCH report · compare recall & latency against your current stack on the same corpus. No application change.
02	**SHADOW** WEEKS 3–6	3-node HA cluster · dual-write embeddings to XERJ + incumbent vector DB · LangChain / LlamaIndex retriever switch is one env var. Side-by-side A/B on the chat app.
03	**MIGRATE PRIMARY RAG CORPUS** QUARTER 2	Make XERJ primary for one RAG workload (docs · tickets · KB). Decommission the shadowed vector DB index. First line removed from the monthly vector-DB bill.
04	**ADD AGENT MEMORY** QUARTER 3	Move conversation turns and tool traces into XERJ agent-memory indices. Retire Redis for session cache. Time-decay retrieval now runs natively; glue code is deleted.
05	**CONSOLIDATE** QUARTER 4	All RAG workloads on XERJ. Vector DB contract ends. Single retrieval platform for every agent team. One vendor, one SLA, one audit surface.

REVERSIBLE · ES WIRE PROTOCOL MEANS NO LOCK-IN AT ANY PHASE

10 · NEXT STEPS

RUN HYBRID. ON YOUR CORPUS.

A two-week pilot on a real RAG corpus. Private binary, embed scripts, seeded datasets for the RAG_BENCH report. You bring the corpus and the chat application's current p95.

AI PLATFORM CONTACT

HELLO@XERJ.AI

XERJ.AI · XERJ.COM · XERJ.DEV

WHAT YOU GET	Private binary · RAG_BENCH reproduction scripts · seeded 10 M-embedding corpus · LangChain / LlamaIndex retriever examples · query-shape dashboard JSON.
WHAT YOU BRING	One RAG corpus (docs / tickets / KB) with its embedding model and a baseline recall / p95 number from the current retrieval stack.
COMPANION BRIEFS	`xerj-exec-brief.pdf` (CAIO / CDO) · `xerj-tech-brief.pdf` (architect) · `xerj-industry-finserv.pdf` for regulated RAG