Retrieval Architecture (Modern RAG)

- Modern RAG

Modern RAG

Traditional RAG: retrieve → answer

Modern Agentic RAG: retrieve → compare sources → detect conflicts → score trust → reason over evidence → answer with citations

NEW IN 2026

Agentic RAG is now commonly framed as the evolution of RAG into a "context engine": the retrieve-generate single pass is replaced with planning, reflection, and self-correction, which published comparisons put at roughly a 42% improvement in faithfulness on multi-step enterprise Q&A versus traditional RAG. A parallel trend: GraphRAG (knowledge graphs + community summaries) is increasingly paired with vector retrieval — vector search covers precise local factual queries, GraphRAG covers open-ended questions needing a global view of the corpus.

Best Practices

- Hybrid retrieval (semantic + keyword)
- Metadata filtering
- Freshness weighting
- Source trust ranking
- Contradiction detection
- Multi-hop retrieval (sequential, multi-step search across disconnected sources)
- Graph-augmented retrieval for relationship-heavy or "global" questions (New)

Context Window Growth vs. RAG

Frontier context windows reaching 1M+ tokens by 2026 does not eliminate RAG. For a corpus that fits in-context (e.g., a single product manual), load-and-ask is genuinely simpler, cheaper to debug, and often faster. RAG remains necessary when the knowledge base exceeds context capacity, when questions require multi-hop reasoning across sources not known upfront, when tool use (SQL, code execution, live APIs) is required, or when latency/cost at inference time justifies the added engineering. Benchmark both approaches before committing to the more complex pipeline.