Agentic RAG

Agent Framework Dev Project

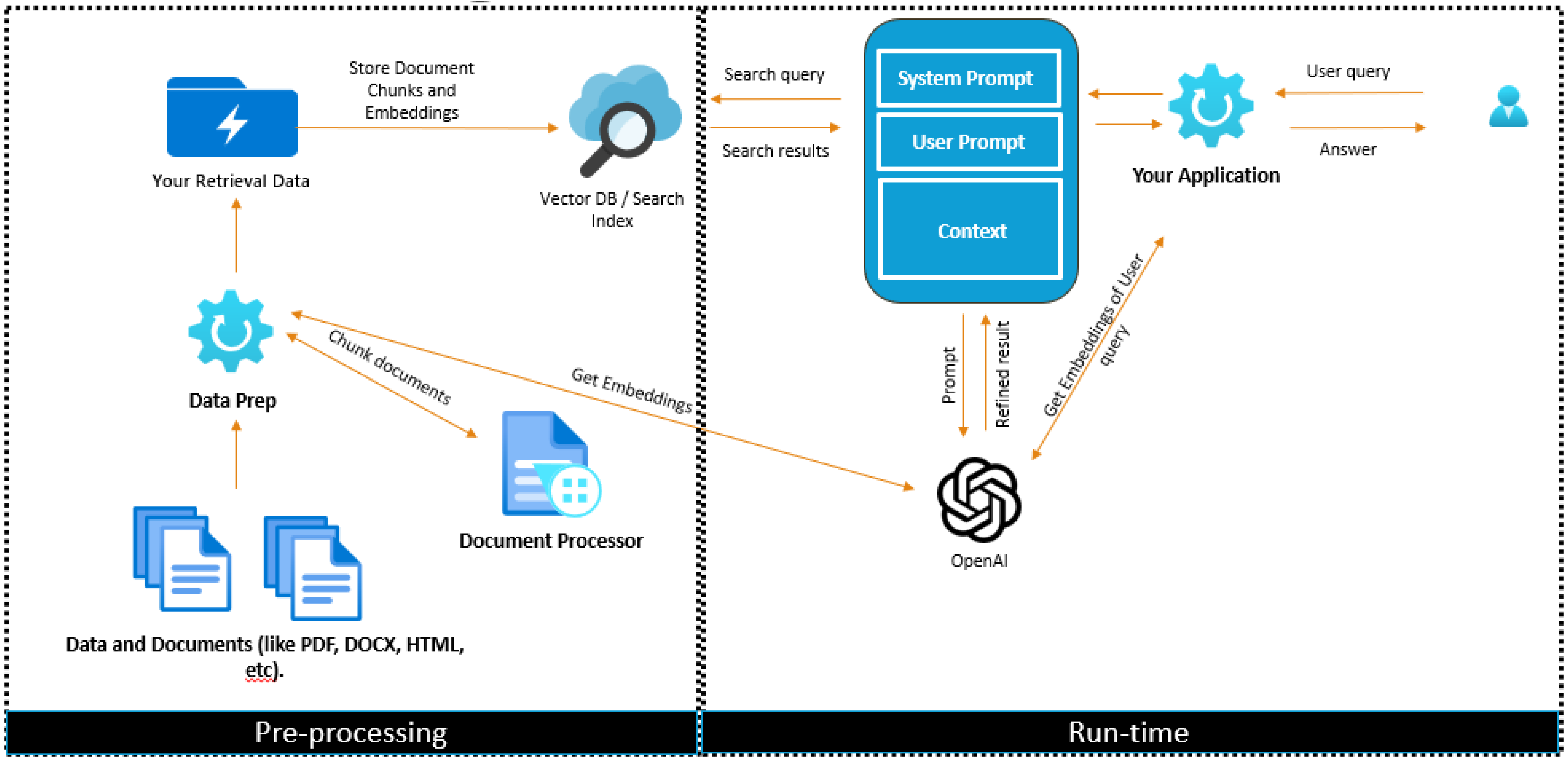
Jan 2026 - Burlington MA edition

Jason Haley

Today’s Overview

- Quick recap of traditional RAG
- Why basic RAG breaks down
- What makes RAG agentic
- Core concepts and architecture
- When agentic RAG does and doesn’t make sense
- Key takeaways

Traditional RAG in 60 Seconds



Traditional RAG: Quick Recap

The Basic Flow

User query → Embed → Similarity search → Retrieve context → Generate answer

The Problem It Solves

Grounds LLM responses in your data, reducing hallucinations and providing source attribution

Simple & Effective

Works great for straightforward queries with clear intent

Why Traditional RAG Breaks Down

Example Query

"What were the key findings from our Q3 reports and how do they compare to industry trends?"

Fixed Retrieval

Single similarity search can't break down complex queries

Single Source

Can't search internal docs and external data

No Iteration

One shot only, can't refine or follow up

The Core Problem

Traditional RAG treats all queries the same way.

What Makes Agentic RAG “Agentic”?

The LLM becomes a reasoning engine

It decides what to do, which tools to use, and when to stop

Planning

Break complex queries into steps

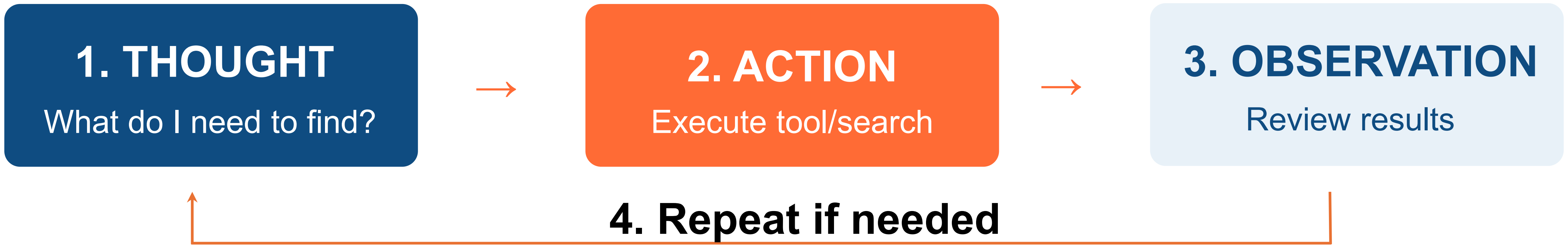
Tool Selection

Choose the right data source

Iteration

Refine until satisfied

The Agents’ Jobs



Example: "What's our pricing vs competitors?"

Thought: Need internal pricing first

Action: search_internal_docs("pricing")

Observation: Found our pricing → $49/month

Thought: Now need competitor data

Action: web_search("competitor pricing")

Observation: Found competitors → Average $55/month

Query Augmentation & Multi-Step Retrieval

Example Query

"What were the key findings from our Q3 reports and how do they compare to industry trends?"

Query Rewriting

Transform original user query to be effective for your retrieval system

Query Expansion

Enhance results by generating multiple queries from the original query

Query Decomposition

Breakdown complex user queries into focused sub-queries

Retrieval Is Just One Tool

Vector Store

Semantic Searches

Web Search

Realtime External Data

SQL/APIs

Structured Data Searches

Code Interpreter

Tools Using Code to Analyze Data

Document Diffing

Tools for Advanced Document Comparison

Calculators

Tools to Aggregate or Perform Complex Math

When agentic RAG does and doesn’t make sense

Traditional RAG

✅ **When to use:**

- Simple, direct queries
- Single data source
- Speed is critical
- Low cost requirement
- Predictable behavior needed

📝 **Example:**

"What is our return policy?"

Agentic RAG

✅ **When to use:**

- Complex, multi-part queries
- Multiple data sources
- Query decomposition needed
- Iterative refinement valuable
- Flexibility over speed

📝 **Example:**

"Compare our Q3 performance to industry trends and suggest areas for improvement"

Key Takeaways

Agentic RAG = RAG + Reasoning

The LLM plans, selects tools, and iterates until it has enough information

🎯 Best for complex, multi-source queries

Use when query complexity justifies the additional cost and latency

⚖️ Trade flexibility for predictability

More intelligent but less deterministic than traditional RAG

🔧 Start small and iterate

2-3 tools, clear descriptions, comprehensive logging