AKKA InfoQ

Design Patterns for Agentic AI:

Building Scalable, Event-Driven Systems

May 1, 2025

Moderated by: Erik Costlow, InfoQ Editor

TYLER JEWELL
CEO @ Akka

RICHARD LI
Founder of Amorphous Data

Today's agenda

AKKA

01 LLMs, agents, agentic systems

02 System engineering challenges

03 System engineering practices

04 Resources and next steps

05 Q&A

Poll question

Agentic is real, but…there is a lot to learn

AKKA

Visit akka.io

The basics:	What is agentic AI?
User stories:	Agentic AI customer stories
Webinar:	A blueprint for agentic AI services
Samples	Production-ready agents
Blogs:	Agentic AI blogs
News:	Akka launches new deployment options for agentic AI at scale
Get Started:	Develop your own agentic app

Agents and agentic systems are distributed systems, powered by AI

...that must deliver reliable outcomes

...while depending upon unreliable LLMs.

AI Agency

AKKA

Capacity to make meaning from your environment

RPA
- low autonomy
- coded decisions
- human control

Agents
- partial autonomy
- LLM advice
- human guidance

Agentic
- high autonomy
- distributed decisions
- group coordination

Humans

Low Agency

- static
- reactive
- tasks
- supervised

"A big gap exists between current LLM-based assistants and full-fledged AI agents, but this gap will close as we learn how to build, govern and trust agentic AI solutions."

–Gartner

High Agency

- adaptive
- proactive
- goals
- autonomous

economic productivity

cost

A paradigm shift to AI-fueled app ecosystems

AI agents and apps become part of a symbiotic existence

By 2028, 33% of enterprise software applications will include agentic AI, up from less than 1% in 2024.
Gartner, *TSP 2025 Trends: Agentic AI — The Evolution of Experience, 24 February 2025*

App Ecosystem

- Cloud-native Applications

+

- Agentic AI Services

Enhanced User Experience	AI agents personalize interactions to increase satisfaction
Operational Efficiency	AI agents automate routine tasks to allow humans to focus on strategic initiatives
Scalability	AI-driven SaaS adapt to business needs without proportional increases in cost

LLM-powered app services are intelligent

AKKA

Models can be prompted to perform a range of user & system tasks

App Ecosystem

Cloud-native Applications

+

Agentic AI Services

Input LLM automation varies by data type		Response SaaS app use cases and behavior
audio / video	LLM	interpret / summarize / analyze
metrics	ML	trend projection
questions	LLM	answers
state / data changes	LLM	recommendations
template	LLM	populate fields
document	LLM	validate against a schema
user behavior	LLM	personalization
parameters	LLM	routing decision (path a or b?)
functions	LLM	tool selection + invocation
algorithm description	LLM	compilable, runnable code

Rethinking *how* your system makes decisions

AKKA

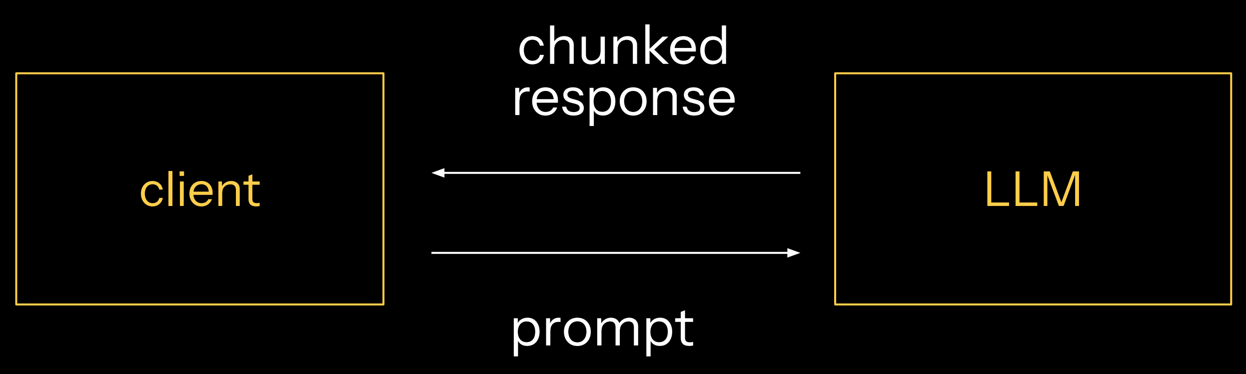
Solve problems where deterministic and rule-based approaches fall short

Multi-faceted decision making	Workflows involving judgement, exceptions or context-sensitive decisions, for example when to escalate a support ticket
Constantly changing rules	Systems whose rulesets frequently change, have extensive conditions, or burdensome to maintain, such as identifying inappropriate language
Reliance on unstructured data	Extracting meaning from content, interpreting language, audio or images, and conversational responses, such as with a support chatbot

From LLMs to Agentic Systems

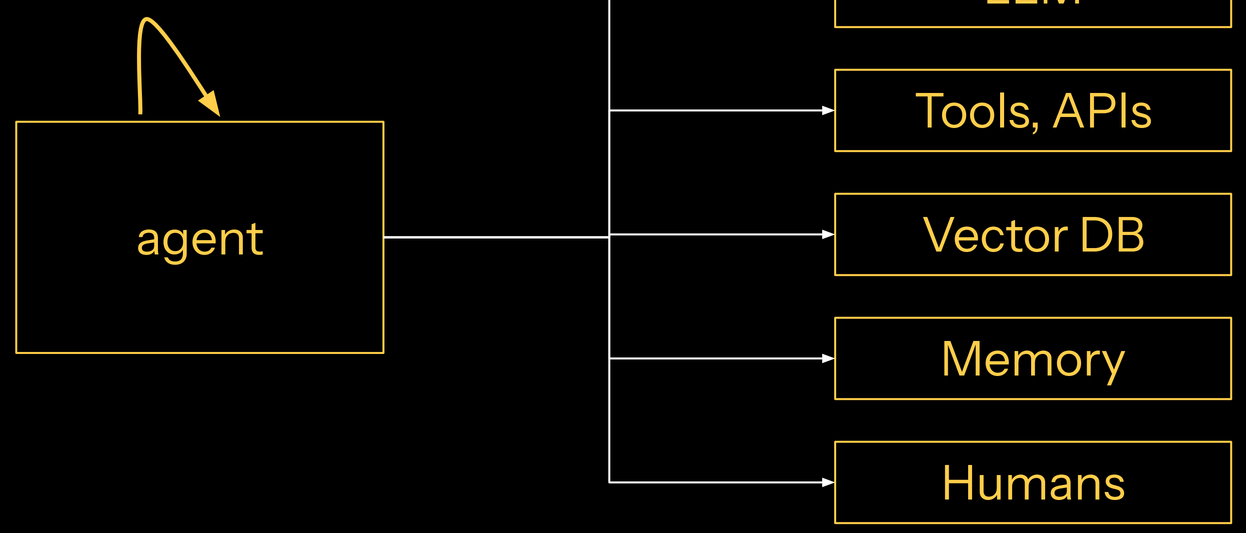
Agents give structure to LLMs; agentic systems give scale to agents

LLM



Stateless, long-running, computationally intensive resources that can analyze, reason, and plan

Agent



Structured enrichment loop that builds context, invokes tools, takes action, and gathers human feedback

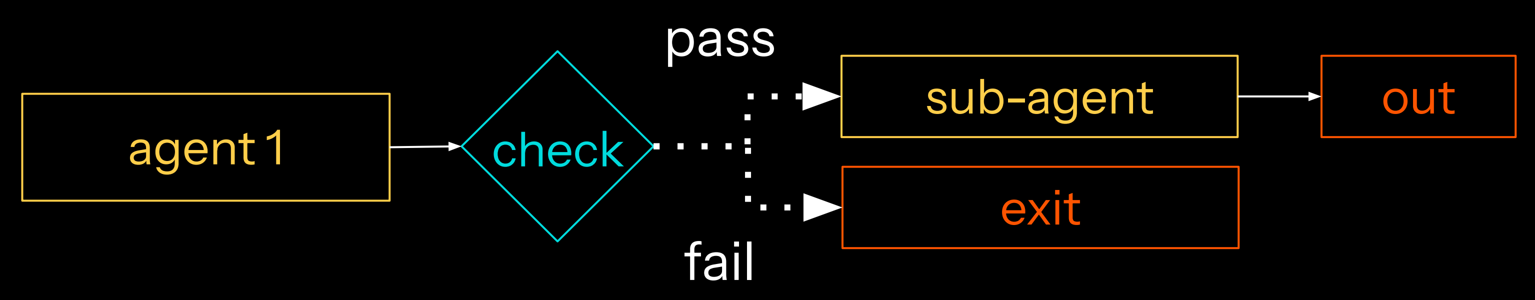
Agentic System

Networks of multiple agents orchestrated to solve complex tasks

Patterns for agentic systems create intelligence

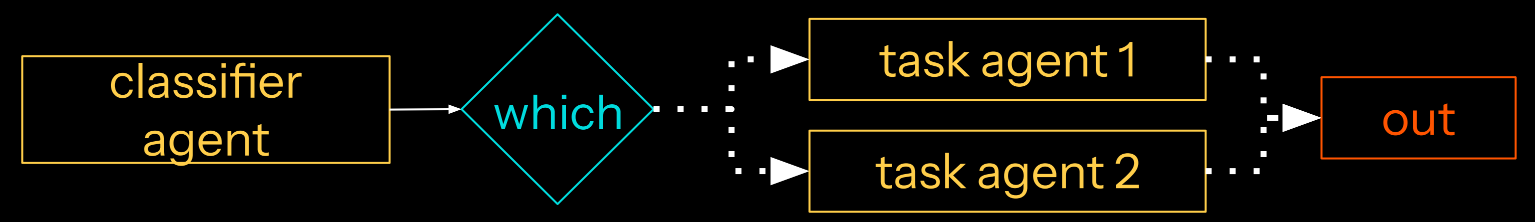
Agent collaboration enables reliable, goal-driven reasoning

prompt chaining



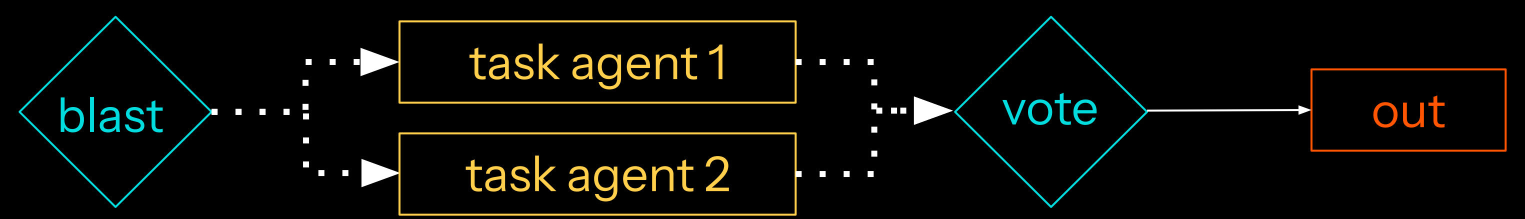
Tasks that can easily be decomposed to subtasks:
e.g. write a blog then translate to French.

routing



An LLM router classifies a task for routing to an LLM specialist:
e.g. classify this support call as either sales or technical

parallelization

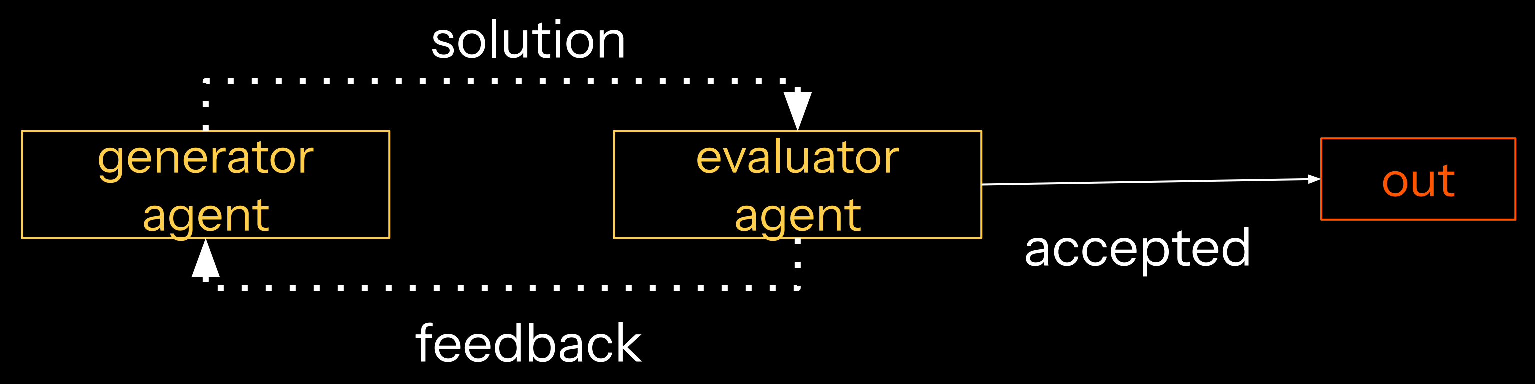


LLM subtasks can be divided for speed or multiple runs:
e.g. execute security tests from different povs, with success voting

synthesizer

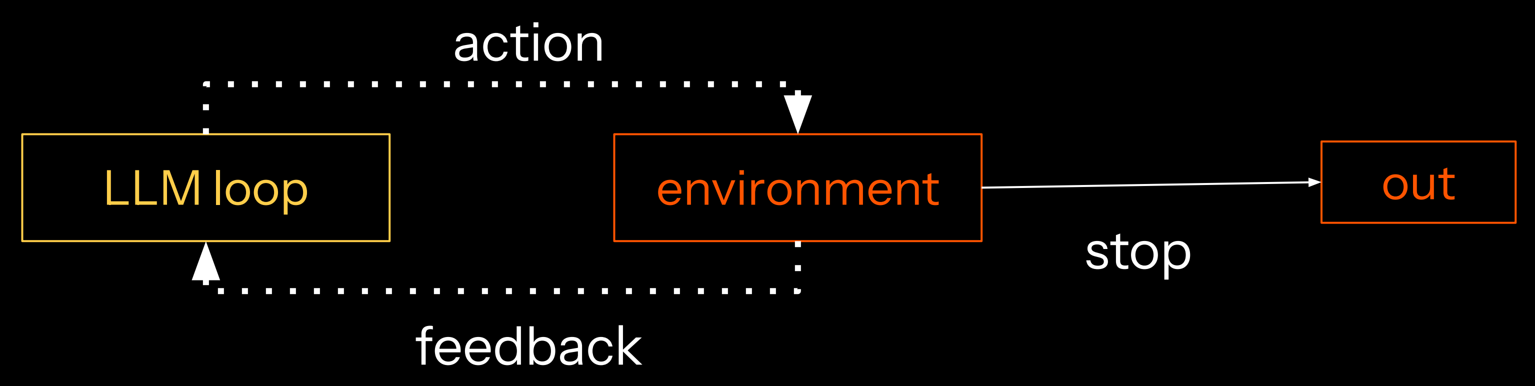
An orchestrator LLM breaks down tasks not known in advance:
e.g. gathering information from targets identified by orchestrator LLM

evaluator-optimizer



One LLM generates a response while another provides feedback:
e.g. a translation LLM that has nuance checking from evaluator LLM

service agent



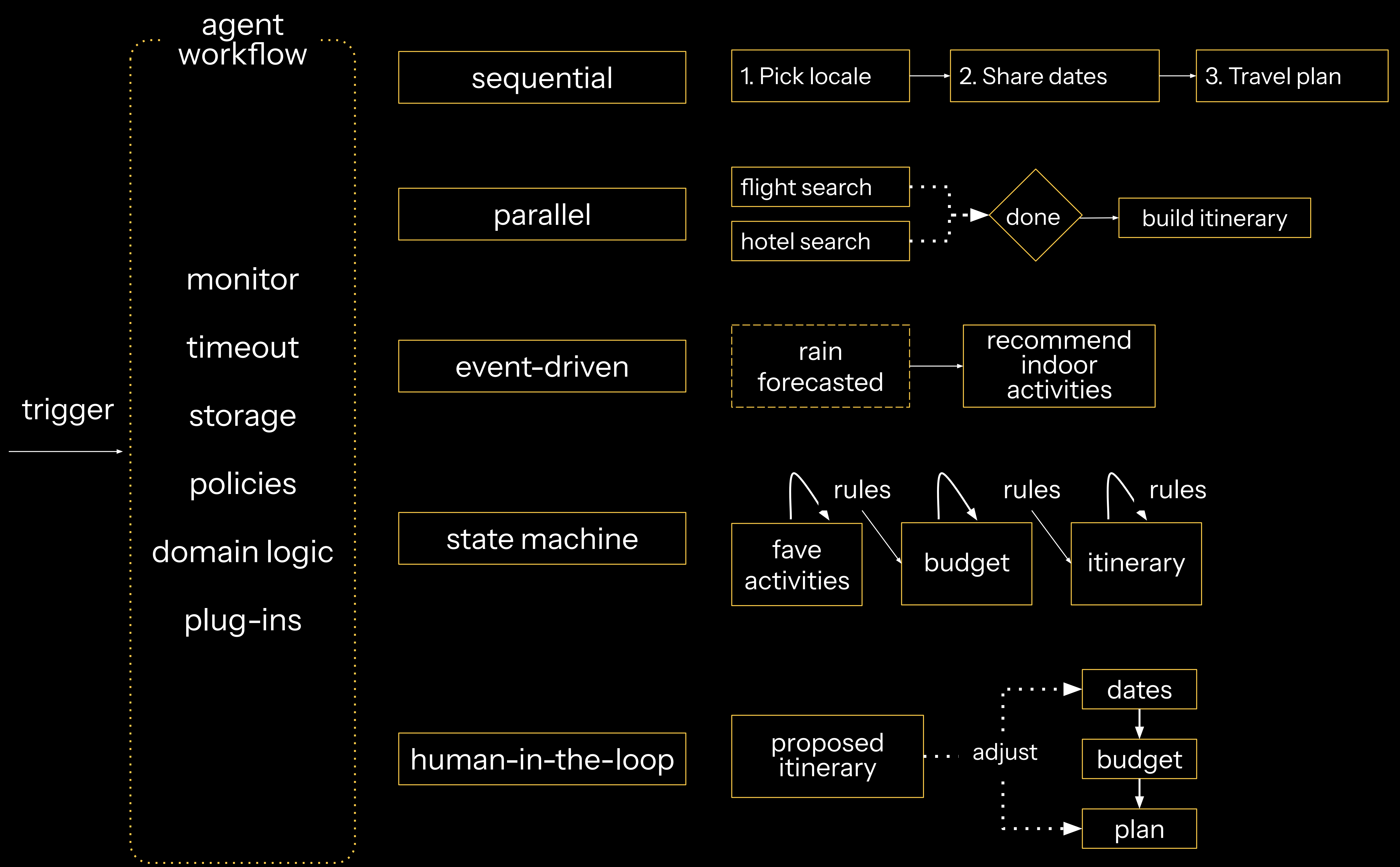
Create and execute a complex plan while staying “grounded” with feedback:
e.g. create a travel itinerary and book all reservations for a vacation

Multi-agent systems are orchestrated

Traceable, auditable, debuggable, with point-in-time recovery

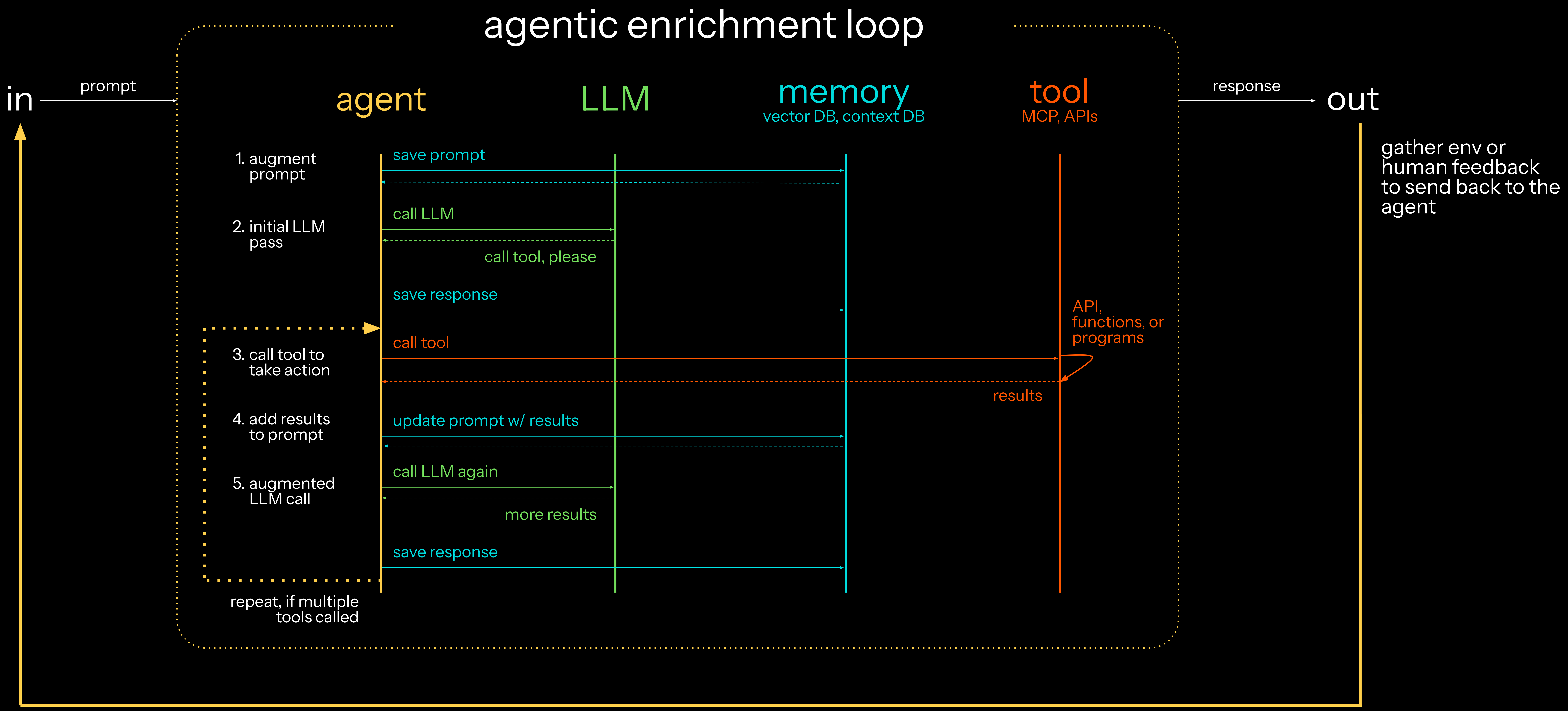
Agentic systems are workflows

reliable execution of AI tasks with visibility into request / response data, built-in retries, and error compensation



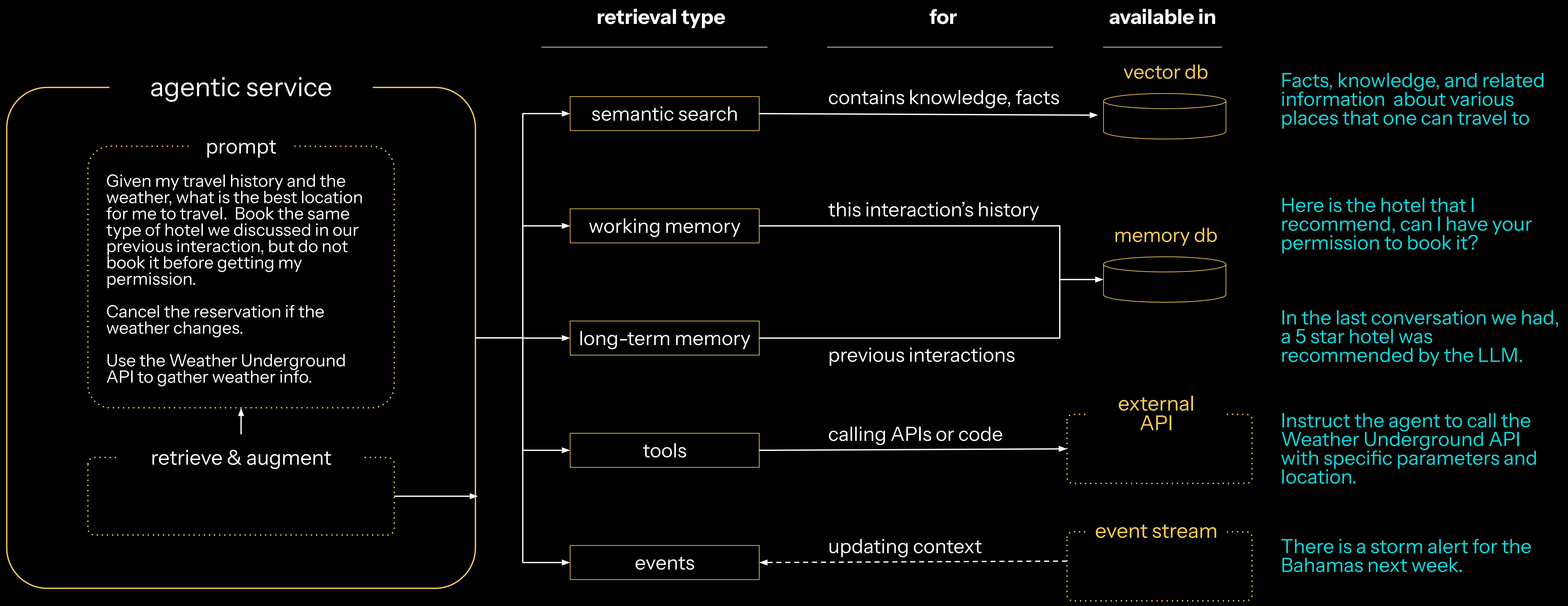
Single-agent enrichment loop

Prompt → retrieve → enrich → repeat is a repetitive cycle & pattern



LLMs are stateless. Context is assembled.

Agentic services augment prompts with data from many sources.

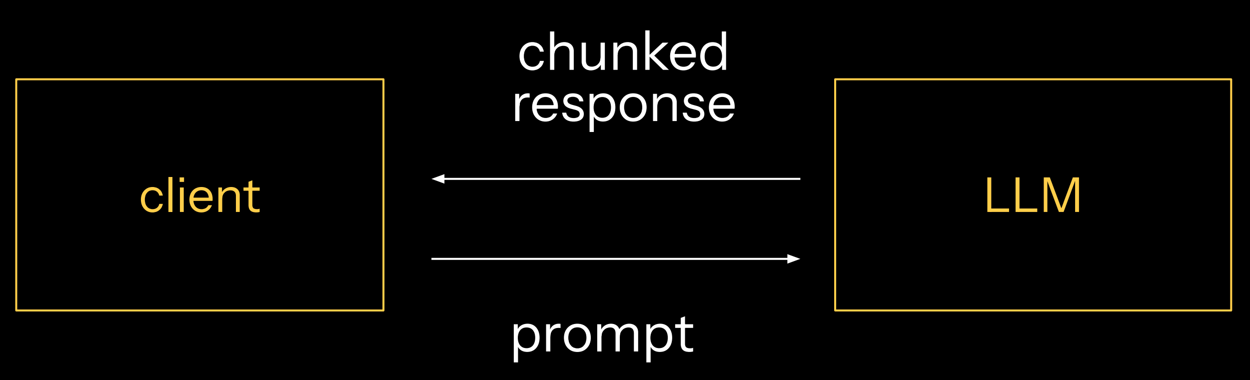


AKKA

Agentic systems are **distributed systems**

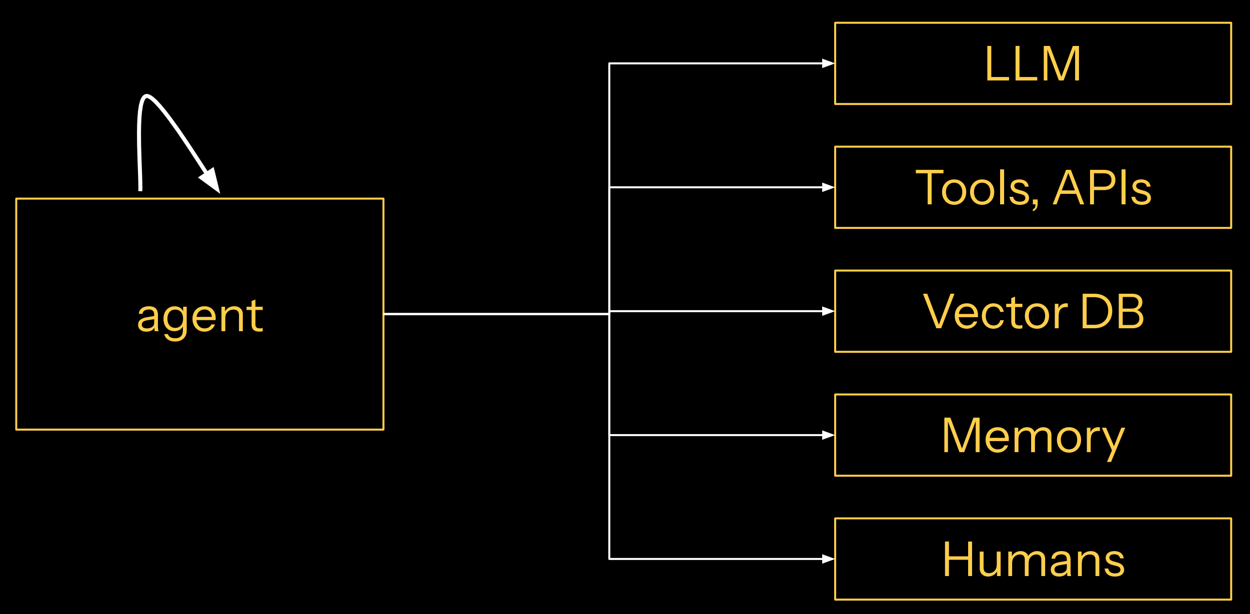
Architectural techniques and practices required for scale and resilience

LLM



- Async, non-blocking invocation
- Event-based, streaming responses
- Backpressure

Agent



- Event-driven architecture
- Human-in-the-loop interaction
- Streaming real-time ingest
- Retries, circuit breakers, timeouts
- Memory & tool integration
- CQRS
- Replication and failover

Agentic System

- Durable workflows
- Distributed tracing
- Discovery & mesh networking
- Multi-agent protocols: A2A, BeeAI

Poll question

Today’s agenda

AKKA

- 01 LLM, agents, agentic systems
- 02 **System engineering challenges**
- 03 System engineering practices
- 04 Resources and next steps
- 05 Q&A

Bumpy path from POC to production

Top three enterprise challenges: uncertainty, privacy, and scale

AKKA

52%

fail to reach production

8+ months

POC to production

"Leaders reported that only 48% of AI POCs (Proof Of Concept) make it into production, and they take an average of 8.2 months to go from POC to production."

Gartner

Uncertainty: From deterministic to stochastic

AKKA

Randomness cannot be eliminated and must be embraced

LLMs are not deterministic components
- Same prompt ≠ same output.
- They predict tokens, not answers.
- You don't pass parameters – you design prompts.
- Hard to predict outputs, validate correctness, or reproduce behavior.

Scaling makes it even harder
- LLMs are slow, expensive, and limited by token windows.
- You need streaming, chunking, caching, windowing, reranking, fallback.
- You're not calling a model – your orchestrating a distributed system.
- Cough, cough - why Akka :)

Prompting isn't programming
- No function signatures. No modular reuse.
- Tiny prompt changes can break results.
- Long prompts increase latency.
- And prompts don't always work the same across workflows or chains.

Debugging is a black box
- No stack traces.
- No explanations.
- Logs give you input/output, not reasons.
- Prompt tweaks can cause side effects far from where you made change.

Retrieval adds more uncertainty
- In RAG, you're combining semantic search, reranking, and formatting.
- Each step adds noise.
- Generating over possibly irrelevant context.
- Now the system is doubly stochastic: retrieval + generation.

Expectations ≠ reality
- People expect memory, perfect instructions, stable outputs, and truth.
- LLMs forget, hallucinate, and drift based on sampling.
- Without scaffolding, they will be (and feel) brittle and inconsistent.

Testing LLMs isn't straightforward
- There's no .assertEqual().
- Heuristic metrics are flawed.
- Human evals are expensive and inconsistent.
- Even stable outputs might still be wrong.

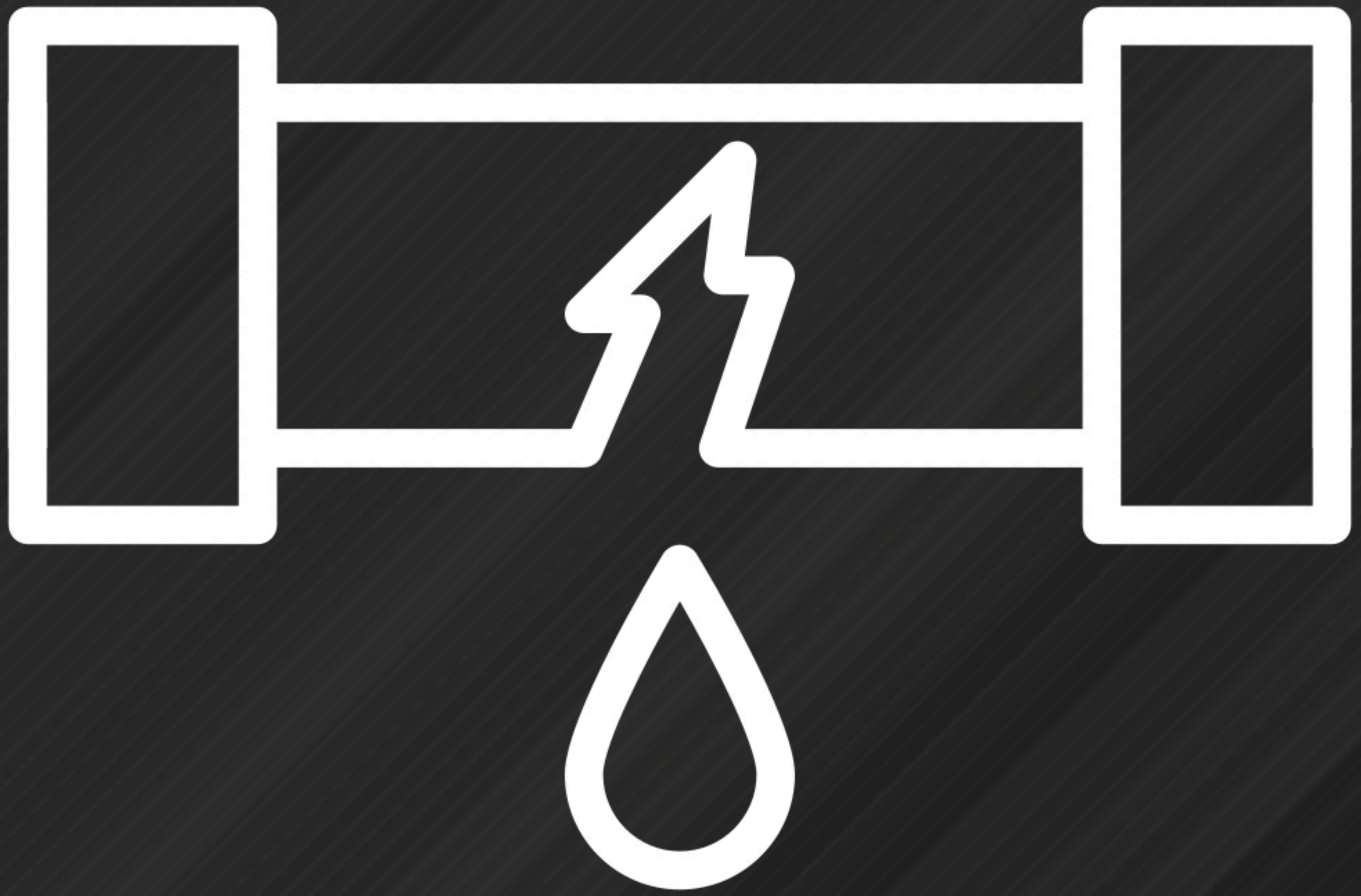
If you're looking for vibes, it will be short lived
- LLMs are probabilistic pattern matchers - not deterministic APIs.
- Building with them means thinking in systems, not functions.
- It means controlling chaos, not eliminating it.

AKKA

Privacy and compliance horror show

LLMs are leaky sieves creating numerous holes for security to plug

- LLMs memorize or reproduce PII from training data
- Private or proprietary data mishandled by agents
- Training data lacking explicit consent from individuals
- Insecure data handling in APIs
- Anonymized data can be reconstructed by LLMs
- Regulations with data deletion requirements are not LLM-enforced
- LLM decisions using personal data can introduce bias and ethics issues
- LLMs may expose PII by integrating with unapproved systems



- Establish clear security & compliance guidelines
- Enable enterprise-grade data controls
- Implement agentic service interaction and logging policies
- Choose agentic platform with tracing and reasoning auditing
- Implement risk mitigation with content filtering
- Implement agent identity with roles and permissions
- Memory hardening with trust controls and min-access-policy

AKKA

Enterprise agentic scale requires efficiency

More txs: each slower, less predictable and more costly

	SaaS	Agentic
Users	billions	20x
TPS	10,000	100x
p(99) Latency	10-80ms	15-400x
Cost / LLM tx	cheap	10-10,000x

Mar 25: the best performing LLM @ 86% MMLU accuracy costs $98 / 1M tokens, or ~850,000x more expensive than the average database transaction. The worst performing LLM @ 36% MMLU accuracy costs $.01 / 1M tokens, or 7x more expensive.

Poll question

Today's agenda

AKKA

01 LLM, agents, agentic systems

02 System engineering challenges

03 **System engineering practices**

04 Resources and next steps

05 Q&A

Agentic systems engineering for reliability

AKKA

1. Execute a DDD and AI-DD process	➔ Produce context map, ubiquitous language, and bounded contexts ➔ Define overall orchestration and flow across bounded contexts ➔ Develop localized workflows for each bounded context
2. Define data sovereignty and scope	➔ Company-specific requirements (e.g., retention policies, audit logging) ➔ Country or regional regulations (e.g., GDPR, HIPAA, financial data rules)
3. Establish evaluation strategy	➔ Make reasoning visible and measurable from the start ➔ Build synthetic evaluation sets to test reasoning steps
4. Select the right AI models	➔ Reasoning models: OpenAI o3, Claude Sonnet, DeepSeek ➔ General models: OpenAI GPT-4o, Gemini Pro, LLaMa ➔ Small language models: Phi-4, Mistral 7B, Claude Haiku, Gemini Flash ➔ Fine-tuned industry models: DeepSeek-Coder, CodeLlama
5. Select agentic platform architecture	➔ Choose platform that enables services that transact and reason ➔ Rqmts: Durable execution, event-driven, memory, streaming, and tools support ➔ Rqmts: Elastic, <20ms p99 latencies, resilient, multi-region failover
6. Build developer workflow and agents	➔ Refine developer workflow ➔ Build initial versions of your agent(s)
7. Deploy and observe	➔ Release, monitor, and refine agent based on real-world behavior

Techniques for reducing uncertainty

AKKA

Design to anticipate randomness while embracing failure as expected

Leverage strategies that create layers of certainty	Create reasoning layers that break complex plans into stages, steps, or sub-tasks that can be validated or checked by downstream agents or humans.
Incorporate eval-driven development	Continuous testing and experimentation of different inputs (real-world, synthetic, adversarial) to track and validate accuracy.
Choose an Agentic AI Platform proven to operate services scalably, safely	Leverage a framework and platform based upon proven runtime that supports distributed orchestration, event-driven behaviors, backpressure, streaming, and embedded memory.

Create layers of certainty

Incorporate multi-agent and human verification strategies

AKKA

Human in the loop	Delegate decisions to humans with workflow
Agentic awareness	Take more LLM thinking time when observing uncertainty
Check and balance	Get second opinions from other agents
Specialization	Limit LLMs to making decisions in one area of expertise
Restricted decisioning	Limit LLMs to a finite set of outcomes

Incorporate **evaluation-driven development**

A system is only as reliable and accurate as its evaluation framework

vary inputs (adversarial, synthetic, real-world) and measure output validity

Given a set of inputs ...

Context(s)

Prompt(s)

Model(s)

Post-Inference

Guardrails

... is the output accurate and reliable?

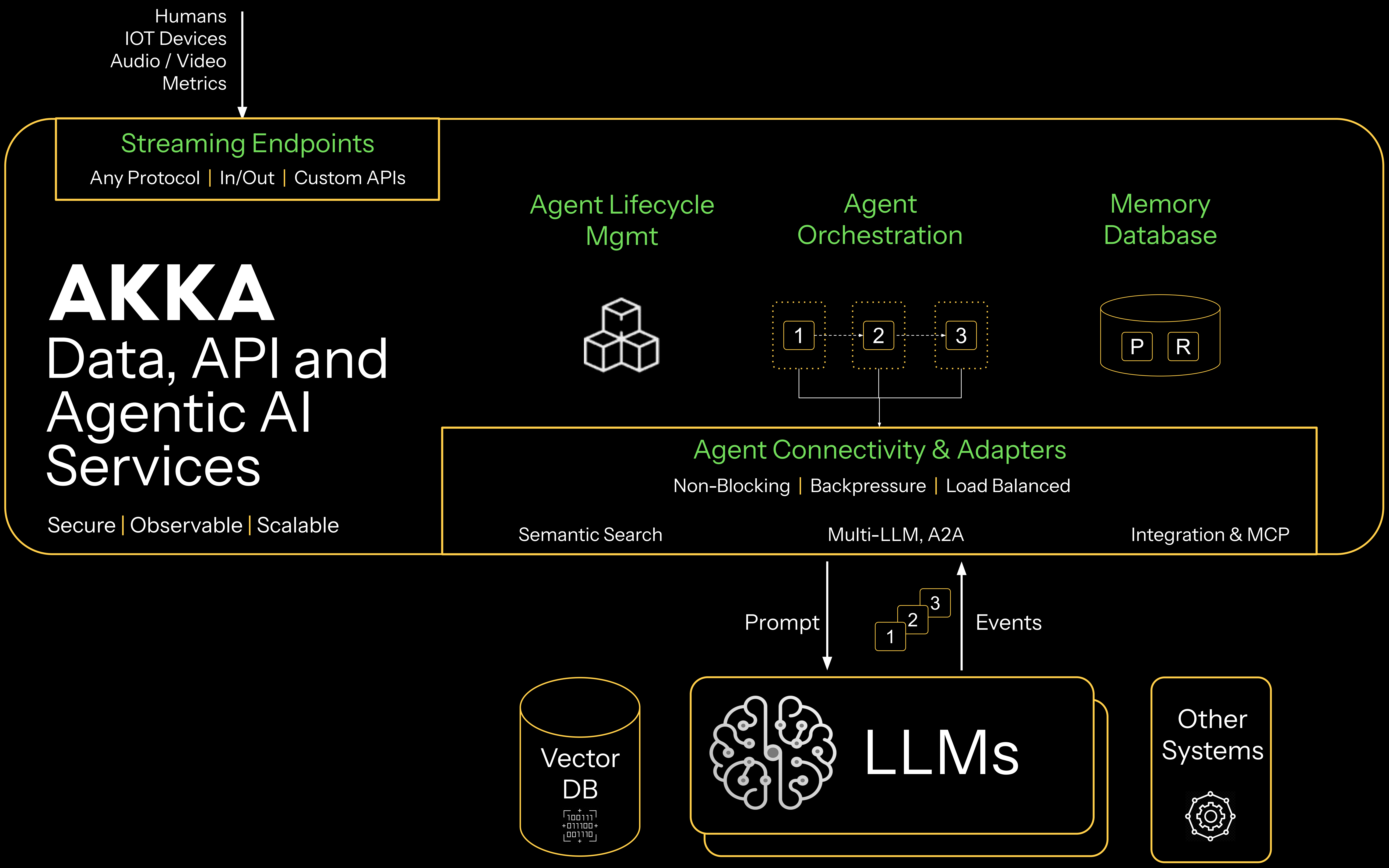
Choose a proven Agentic AI Platform

AKKA

LLMs unlock reasoning – but there is no free lunch

LLMs are stateless No recall of prior interactions	Need a memory system
LLMs need context Must be told everything upfront	Tools integration Knowledge integration (e.g., vector databases)
LLMs are stochastic Same input, different outputs	Rely on deterministic workflows as much as possible Design for uncertainty
LLMs are unreliable May fail to respond or timeout under load	Adopt a distributed systems mindset Use a durable execution framework
LLMs are slow High latency, limited concurrency	Use streaming to improve responsiveness

AKKA



Efficient
70% less compute
API + agentic combo

Elastic
5M TPS
akka clustering

Agile
Prod in days
SDK + ops envs

Resilient
0-0 RTO/RPO
multi-region, multi-master data replication

The Akka agentic advantage

AKKA

- ✓ Agentic, AI, apps & data
- ✓ Hardened runtime
- ✓ Simple, expressive SDK
- ✓ Multi-region
- ✓ Automated ops

Streaming endpoints

- → Shared compute: agentic co-execution with API services
- → HTTP and gRPC custom API endpoints
- → Custom protocols, media types, and edge deployments
- → Real-time streaming ingest, benchmarked to over 1TB

Memory database

- → Agentic sessions with infinite context
- → Context snapshot pruning to avoid LLM token caps
- → In-memory context sharding, load balancing, and traffic routing
- → Multi-region context replication
- → Memory filters for region-pinning and cross-session context creation
- → Embedded context persistence with Postgres event store

Agent connectivity & adapters

- → Non-blocking, streaming LLM inference adapters with back pressure
- → Multi-LLM selection
- → LLM adapters & 100s of ML algos
- → Agent-to-agent brokerless messaging
- → 100s of 3rd party integrations

Agent orchestration

- → Event-driven workflow benchmarked to 10M TPS
- → SDK with AI workflow component
- → Serial, parallel, state machine, & human-in-the-loop flows
- → Sub-tasking agents and multi-agent coordination

Agent lifecycle management

- → Agent versioning
- → Agent replay
- → Event, workflow, and agent debugger
- → No downtime agent upgrades

2B people experience Akka daily

SMILE

A fast ML engine with 100s of ML & LLM inference, powering Google Earth

400K downloads / mo
6K GitHub stars

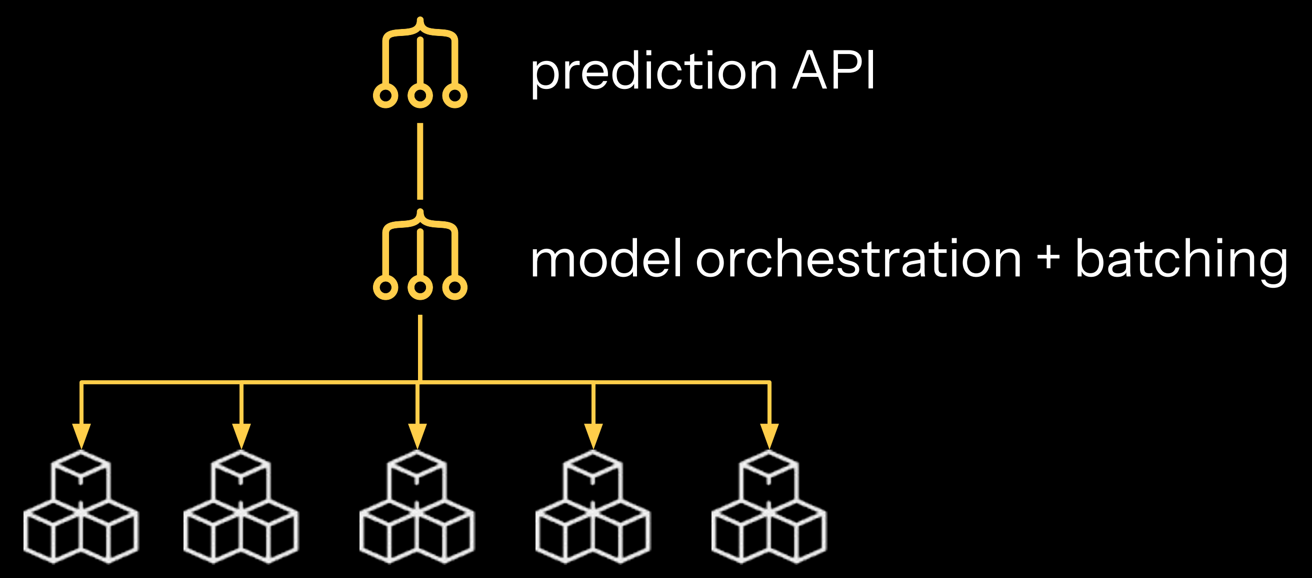
"Akka is used for streaming and back pressure - critical for hosted AI API inference. Akka enables event-driven inference exposed as HTTP efficiently, with low latency."

Haifeng Li – maintainer

Swiggy

API-driven predictions with multi-model fan-out and ultra-low latency

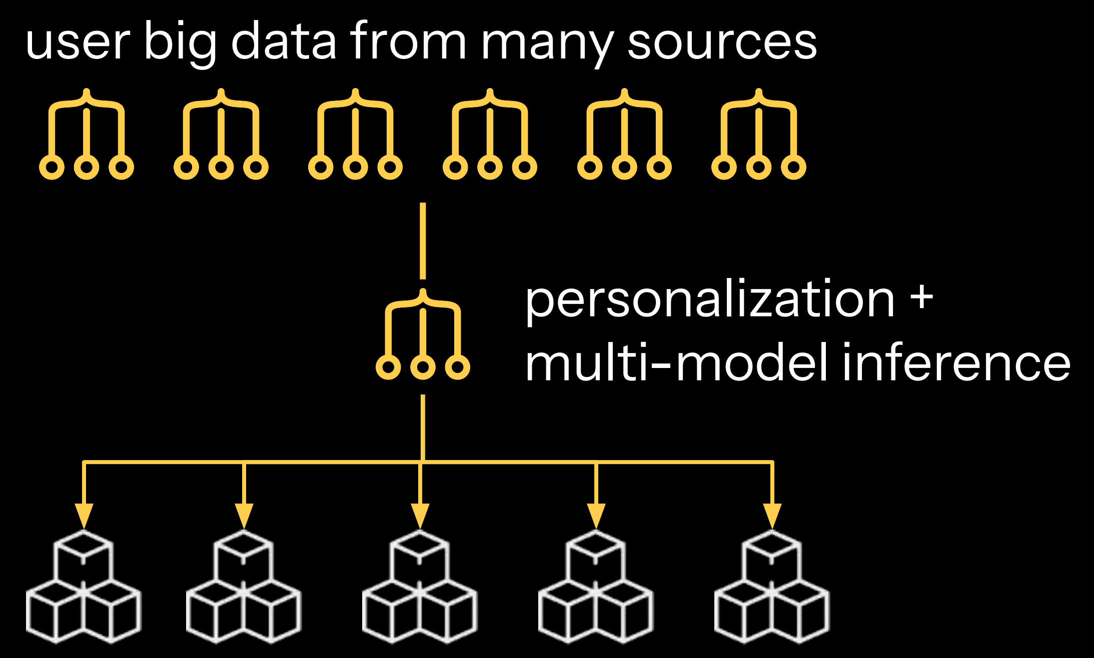
3+ million TPS
71ms p(99) latency



Tubi [Fox]

Tubi applies ML models to real-time streams of data with in-memory, durable journals

2 month time to delivery



Horn

"Zero problems" augmenting high-performance audio and video streams on demand

Tomasz Wujec - Lead Developer

Coho AI

"With Akka, we got to market 75% faster compared to other agentic solutions we had considered."

Michael Ehrlich – CTO

Agentic is real
Let's make it real for you

Additional resources

Webpage: What is agentic AI?
Case Studies: Agentic AI customer stories
Webinar: A blueprint for agentic AI services
Samples Production-ready agents
Blogs: Agentic AI blogs
News: Akka launches new deployment options for agentic AI at scale
Get Started: Develop your own agentic app

Have a project?

concept

proof

48 hours

AKKA InfoQ

Q&A

Contact Us:

Tyler Jewell: tyler.jewell@akka.io

Richard Li: richard@thelis.org

InfoQ: webinars@infoq.com