OLD WORLD NEW WORLD

From Data Pipelines to Decision Pipelines

The next generation of pipelines will not just process data. They will decide what happens next.

Data → Transform → Store

Event → Agent → Reason → Action

Building Agentic Data Pipelines: Orchestrating AI Workflows with Apache Beam

Siddharth Gargava | AWS | Beam Summit 2026

Building Agents Is Easy. Operating Them Is Hard.

The gap between agent demos and production agents is widening rapidly.

AGENT DEMO

Works in a notebook.

PRODUCTION AGENT

100K Events

Planner

Tool Calls

State

Retries

Validation

Actions

⚠ TOOL TIMEOUT

⚠ API RATE LIMIT

⚠ LOST STATE

⚠ INVALID OUTPUT

⚠ DUPLICATE ACTION

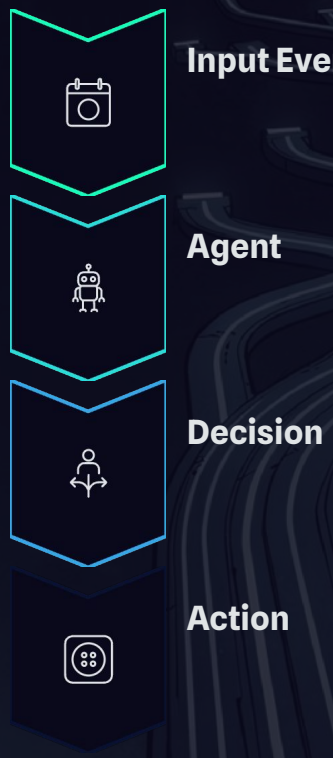
Traditional Pipelines Know The Answer. Agents Must Discover It.

The fundamental shift is from predetermined workflows to dynamic decision-making.

TRADITIONAL PIPELINE

Fixed path. Deterministic & predictable.

AGENTIC WORKFLOW



Runtime path. Dynamic & adaptive.

Key Distinction

Traditional pipelines are deterministic - path fixed at design time. Agentic pipelines are dynamic - DAG constructed at runtime based on context, tool outputs, and model reasoning.

Deeper Technical Note

In ETL, you know the shape of the DAG at design time. In agent workflows, the DAG may be partially constructed and modified at runtime - creating significant challenges for observability, idempotency, state recovery, and cost control.

Meet Our Incident Response Agent

A simple example of a decision pipeline in action.

High CPU Alert

Signal from Kafka, CloudWatch, or Datadog

Plan Investigation

Agent maps out required evidence

Retrieve Logs

Pulls from CloudWatch, Splunk, or OpenSearch

Query Metrics

CPU, memory, latency, error rate

Analyze Deployments

Recent deploys, config & feature flag changes

Generate Diagnosis

LLM correlates signals to root cause

Validate Recommendation

Confidence, safety & format checks

Create Ticket / Page Engineer

Jira, PagerDuty, Slack, or rollback

This workflow is not just processing data. It is deciding what should happen next.

Everything Works. Until It Doesn't.

Every stage introduces a new failure mode.

High CPU Alert
✓ This part works fine

Plan Investigation
⚠ Incomplete context

Retrieve Logs
⚠ Timeout

Query Metrics
⚠ API rate limit 429

Analyze Deployments
⚠ Missing record

Generate Diagnosis
⚠ Hallucination

Validate Recommendation
⚠ Malformed JSON

Create Ticket
⚠ Duplicate on retry

TOOL FAILURES | RATE LIMITS | LOST STATE | DUPLICATE EXECUTION

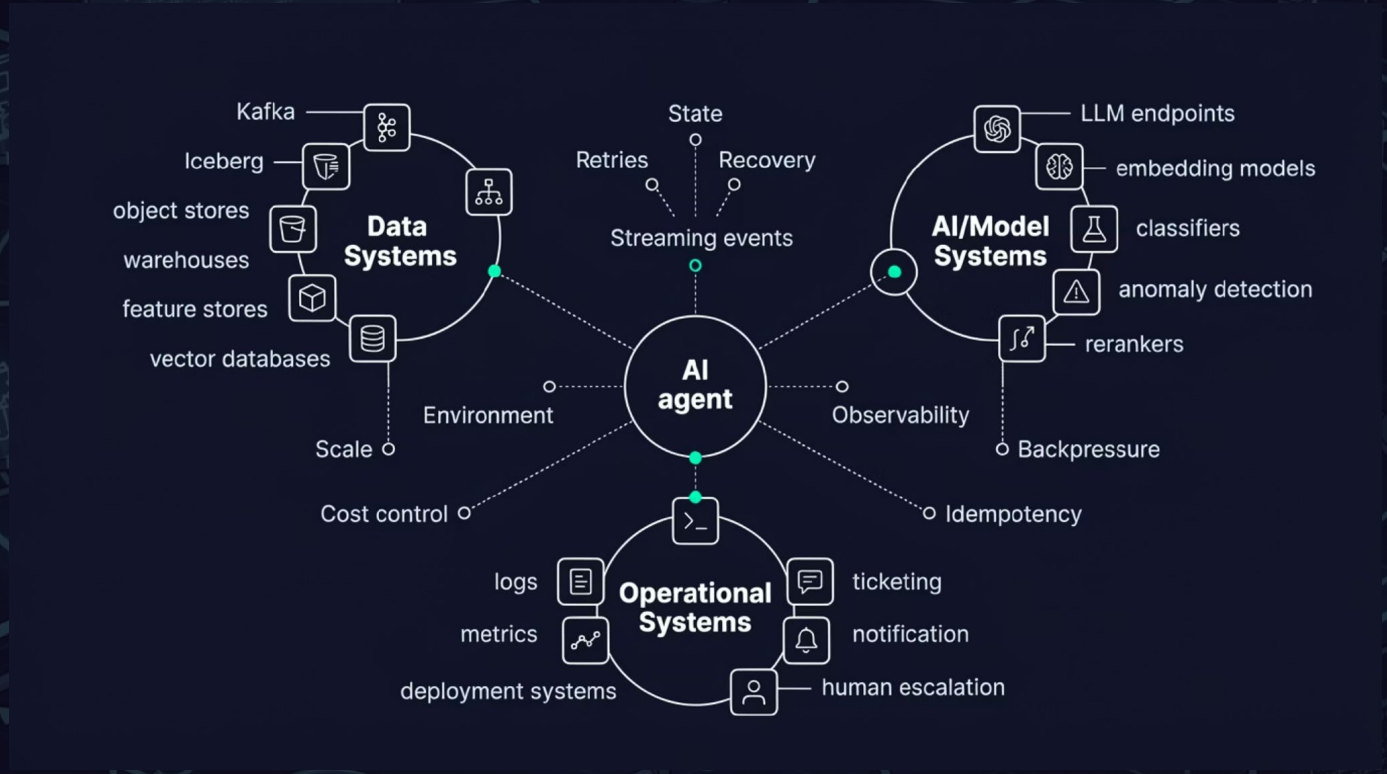
⚠ Once the agent calls external tools, stores state, retries, and acts on the world - it inherits all classic distributed systems problems: partial failure, timeout ambiguity, retry storms, idempotency, consistency, and observability.

Every stage introduces a new failure mode.

OPERATIONAL CHALLENGE | AGENTIC PIPELINES

The Problem Isn't AI. It's Coordination.

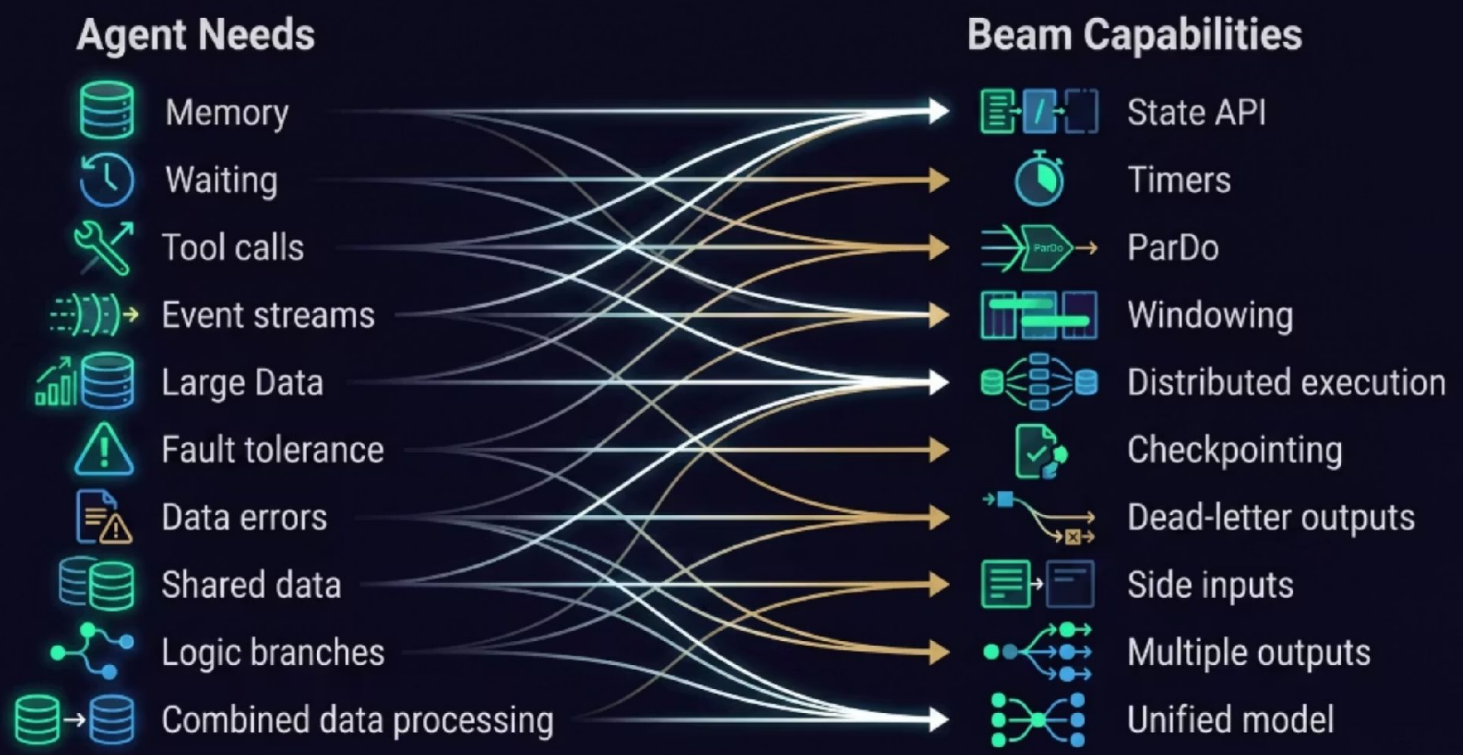
Agent workflows are distributed systems, bringing distributed systems problems.



Agents do not fail only because they are unintelligent. They fail because coordinating everything around them is hard.

What We Need Isn't Another Model.
We Need An Execution Layer.

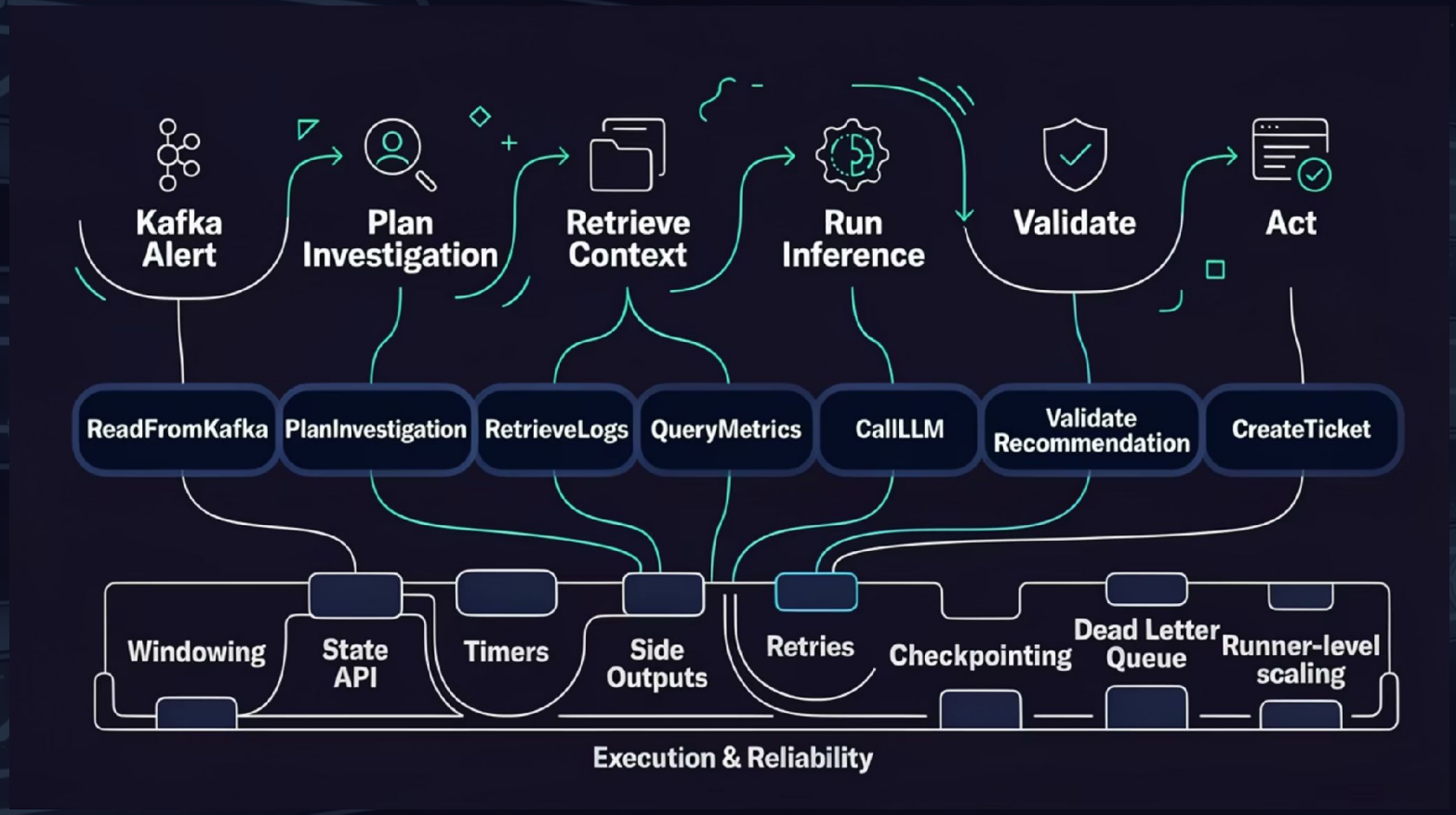
Capability Mapping



Apache Beam was not designed for agents, but many of the primitives agents need already exist.

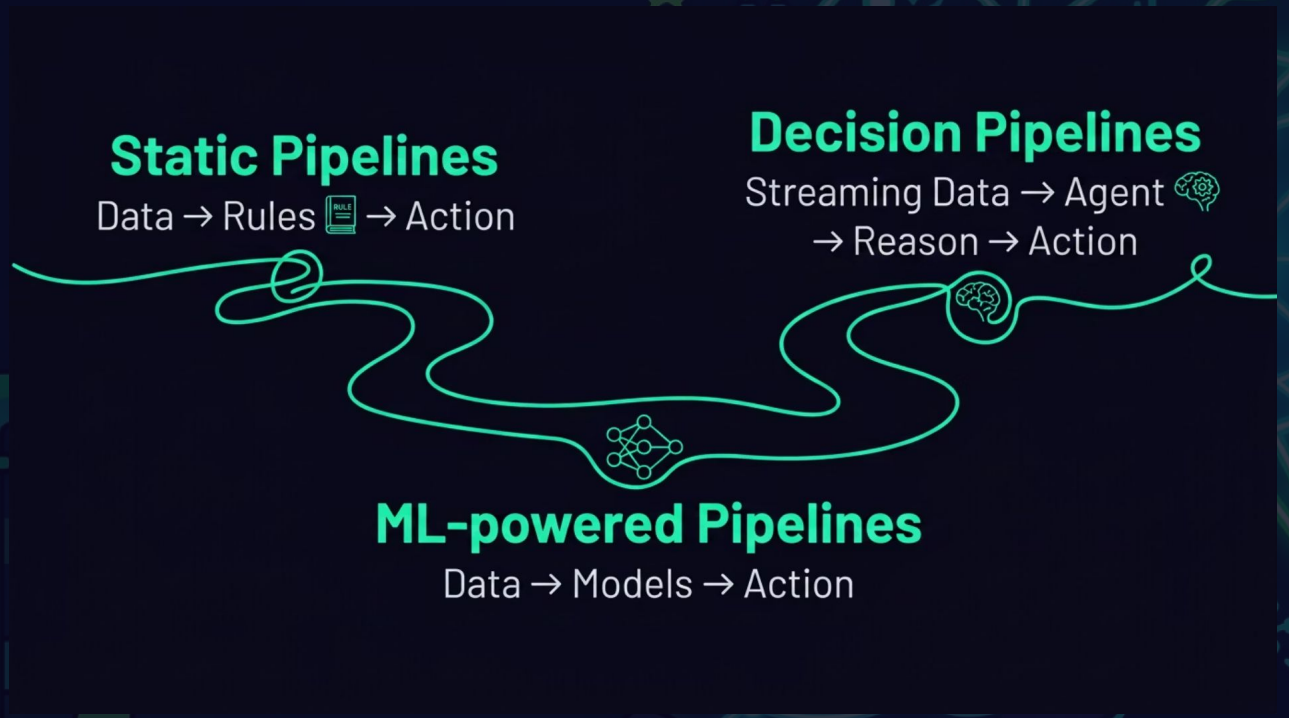
Building A Decision Pipeline

A three-layered view of an agentic workflow in Apache Beam.



Agentic workflows become decision pipelines when they are executed reliably.

The Next Generation Of Pipelines Won't Just Process Data. They'll Make Decisions.



We are moving from static workflows to autonomous workflows that reason over data, call tools, and act in real time.

Important Nuance: Decision pipelines **build on** data pipelines. They add reasoning and action on top of reliable data orchestration, rather than replacing them.

The future is not AI bolted onto pipelines. It is pipelines designed to reason, decide, and recover.

Three Things I'd Like You To Remember

1

Agents are fundamentally workflow problems.

The LLM is only one step. Real agents coordinate tools, data, state, decisions, and actions.

2

Reliability matters more than model choice.

In production, failure handling, state recovery, retries, idempotency, and observability determine whether the system works.

3

Apache Beam provides primitives for decision pipelines.

State, timers, ParDo, windowing, checkpointing, side outputs, and distributed execution map naturally to the needs of agentic workflows.

Data Pipelines → Decision Pipelines

The next frontier is not just smarter agents. It is reliable systems that can operate them at scale.