Powering the Future of Event-Driven Agents at Scale

Reza Rokni

Agenda

Apache Beam - Intro

Event-Driven Agents - Intro

Apache Beam for Agents

Scale

Context

Agent Development Kit (ADK) integration with Apache Beam 2.73.0

Session Overview

In talking about Agents and Apache Beam we occasionally touch upon some more advanced topics and techniques.

Your Learning Path: If a specific topic piques your interest, you can find exhaustive, dedicated deep-dives in:

The 2026 College: Current, specialized sessions for this year's cohort.

Course Archives: Comprehensive material and technical talks from our previous colleges.

Wednesday, April 22, 2026 (New Features)

15:00 - 15:25. Introducing Managed IO, the New Era of Beam Connectors
By Ahmed Abualsaud

15:30 - 15:55. Scaling Iceberg Ingestion with Apache Beam
By Tom Stepp

16:00 - 16:25. Getting Started with Remote ML Inference in Beam Java
By Ganesh Sivakumar

16:30 - 16:55. Real-Time Anomaly Detection with Apache Beam
By Shunping Huang

17:00 - 17:25. Building Scalable Semantic Search and RAG Pipelines
By Claude Van der Merwe

Thursday, April 23, 2026 (Advanced Tips & Tricks)

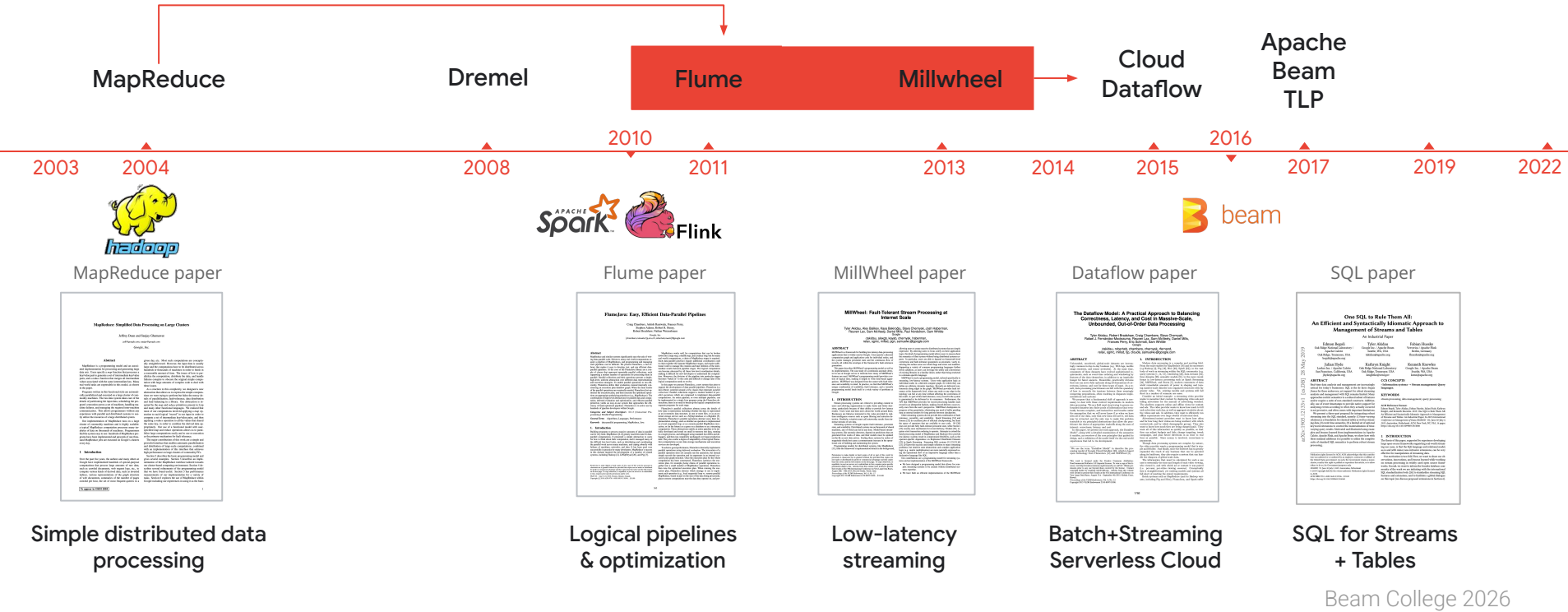
15:00 - 15:25. Video Data Processing with Apache Beam
By Adesh Abhang

15:30 - 15:55. Real-Time Stateful Processing of Video Data
By Darshan Kanade & Aditya Shukla

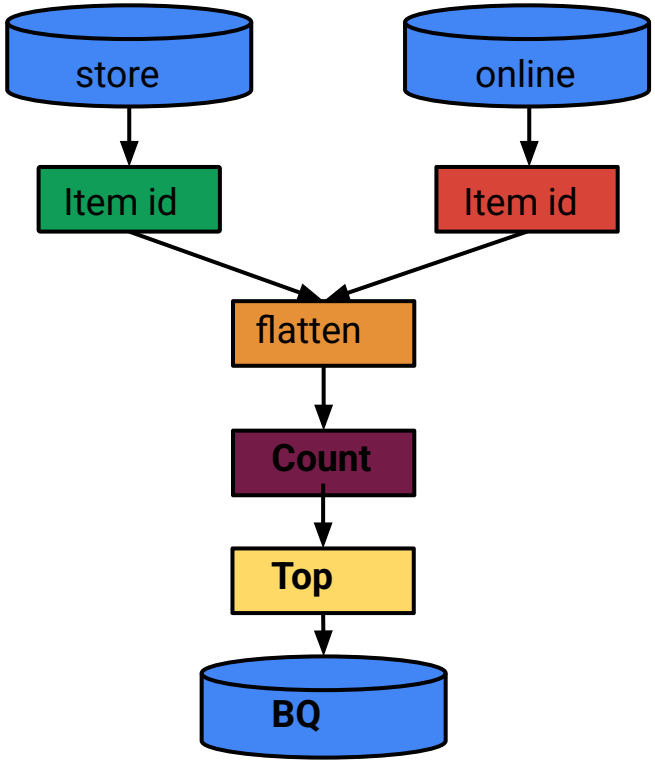
16:00 - 16:25. Assembling the Puzzle: High-Performance Entity Building streaming Beam pipeline using a Two-Tiered State Architecture
By Israel Herraiz & Canburak Tumer

16:30 - 16:55. Beyond Vectors: Building Scalable GraphRAG with Apache Beam and Cloud Spanner
By Oleksandr Saienko

Apache Beam : It started with BigData



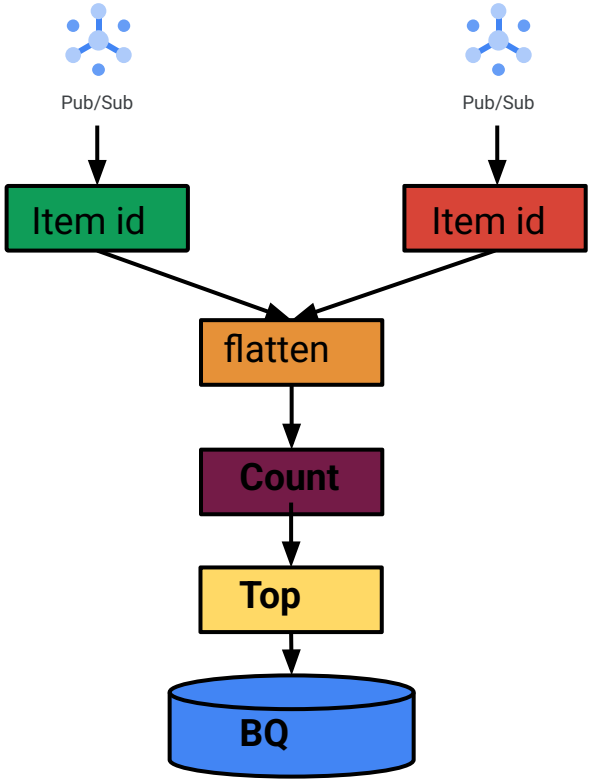
Writing code: Apache Beam Python SDK



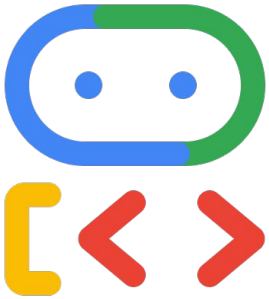

```
storeSales = p  | beam.io.ReadFromText("purchases-store")
                | ...
                | beam.Map(lambda s: (s.itemId, s.value))
onlineSales = p | beam.io.ReadFromText("purchases-online")
                | ...
                | beam.Map(lambda s: (s.itemId, s.value))
topSales = (storeSales, onlineSales)
                | beam.Flatten()
                | beam.Combiners.Count.perKey()
                | beam.Combiners.Top.of(10, key = lambda x: x[1])


topSales | beam.io.WriteToBigQuery(topSales)
```

Writing code: Apache Beam Python SDK




```
storeSales = p  | beam.io.ReadFromPubSub("purchases-store")
                | beam.Map(lambda s: ...)
onlineSales = p | beam.io.ReadFromPubSub("purchases-online")
                | beam.Map(lambda s: ...)
topSales = (storeSales, onlineSales)
                | beam.Flatten()
                | beam.WindowInto(
                        beam.window.FixedWindows(10))
                | beam.Combiners.Count.perKey()
                | beam.Combiners.Top.of(10, key = lambda x: x[1])
topSales | beam.io.WriteToBigQuery(topSales)
```



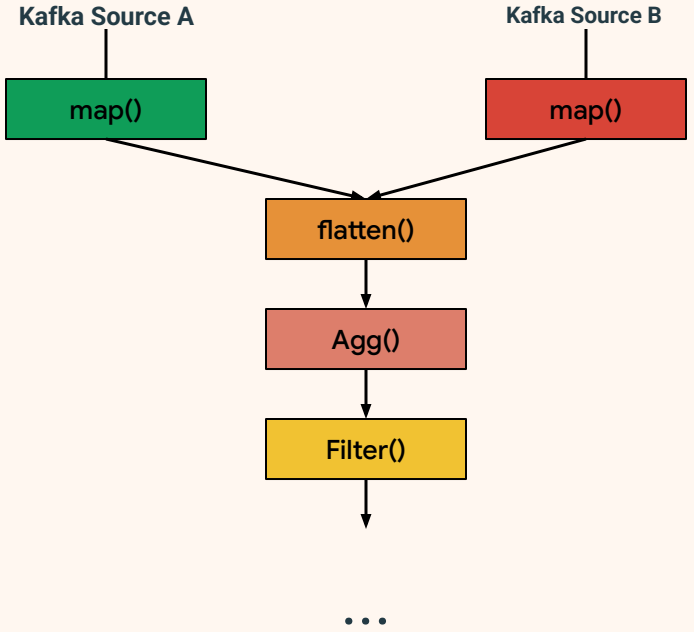
From rigid pipelines to actionable intelligence

Typical Stream Processing

Core Characteristics:

- High volume data streams
- Asynchronous processing
- Fixed pipeline shape (Directed Acyclic Graph DAG)

Pipeline Structure



From rigid pipelines to actionable intelligence

Agent

Dynamic Capabilities

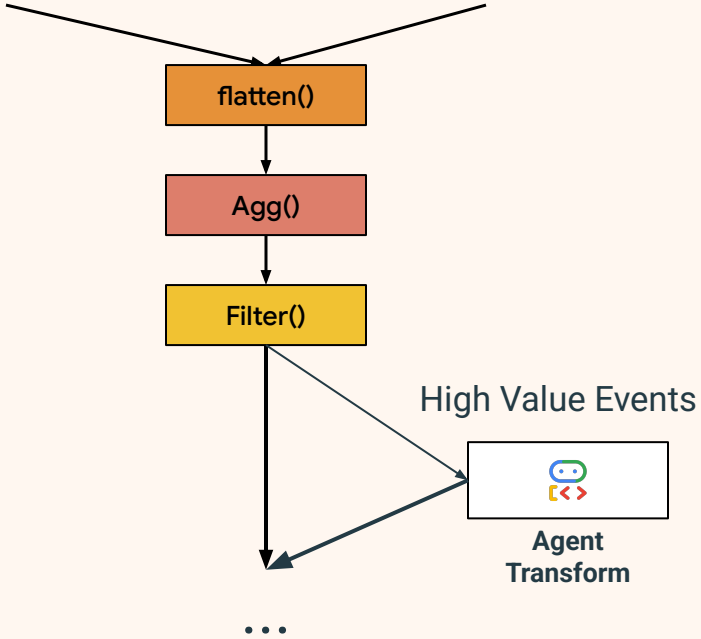
Orchestrates its own plan

Explores solution space with

- Prompts
- Skills
- Tools

Can loop to find optimal solution

Pipeline Structure



Adaptive Real-Time Intelligent Pipelines

Intelligent Recommendations

Upstream Filter == Count

- Explore stock levels
- Explore past purchases
- Explore abandon carts

Offer a personalized price bundle

Intelligent Support Resolution

Upstream Filter == Sentiment

- Gather more system data
- Summarise ticket
- Suggest remediation plan

Escalate to correct department

Intelligent Financial Analysis

Upstream Filter == Anomaly

- Gather transaction history
- Analyze patterns
- Reassess score

Flag fraudulent issue

Apache Beam & Agents

Apache Beam: The Beam ML beginning

Apache Beam: Turn Key transforms

Building blocks

Goal

Apache Beam: Turn-key Transforms

```
with beam.Pipeline(options=pipeline_options) as p:
(p
| beam.io.fileio.MatchFiles("gs://my_bucket/*")
| beam.io.fileio.ReadMatches()
| beam.Map(preprocess_image)
| beam.xxx_pattern_xxx(Configuration)
... )
```

Simplified declarative pattern

Agents at Scale - Challenges

01. Real Time Context

Integrating diverse data streams for agent intelligence:

- **Historic:** Accessing deep archival information.
- **In-Stream:** Processing real-time data flow.
- **External:** Ability to call vector databases for RAG enrichment

02. Scalability & Management

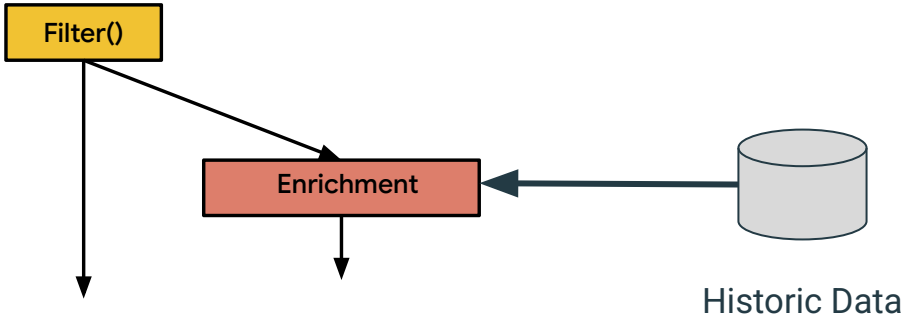
Operationalizing agents across massive infrastructures:

- Processing and delivering the right context from high velocity stream directly to the agent in real time.
- Orchestrating and running many agents at scale, managing their lifecycle and **sessions**

beam

1 - Context : Historic information

```
with
beam.Pipeline(options=pipeline_options)  as
p:
...
low,high | beam.map(high_value_filter)
...
high | Enrichment(bigtable_handler)
...
)
```



1 - Context : In Stream

Extraction of features

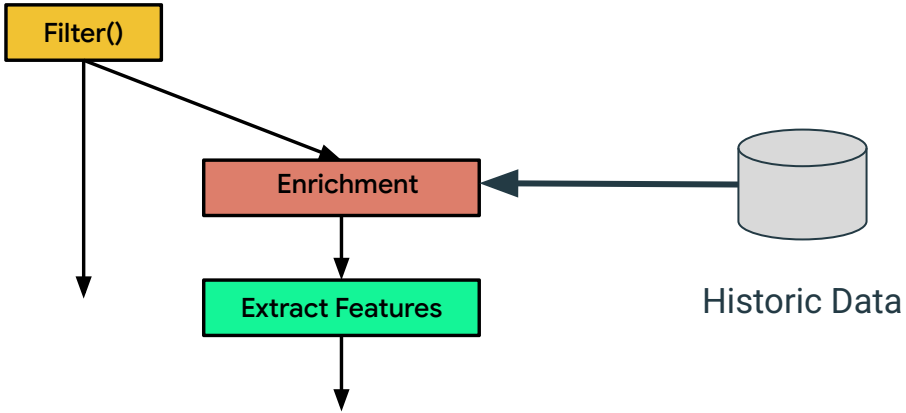
Using core primitives

DoFn, GBK, Combine, Windowing etc ...

Using advanced features

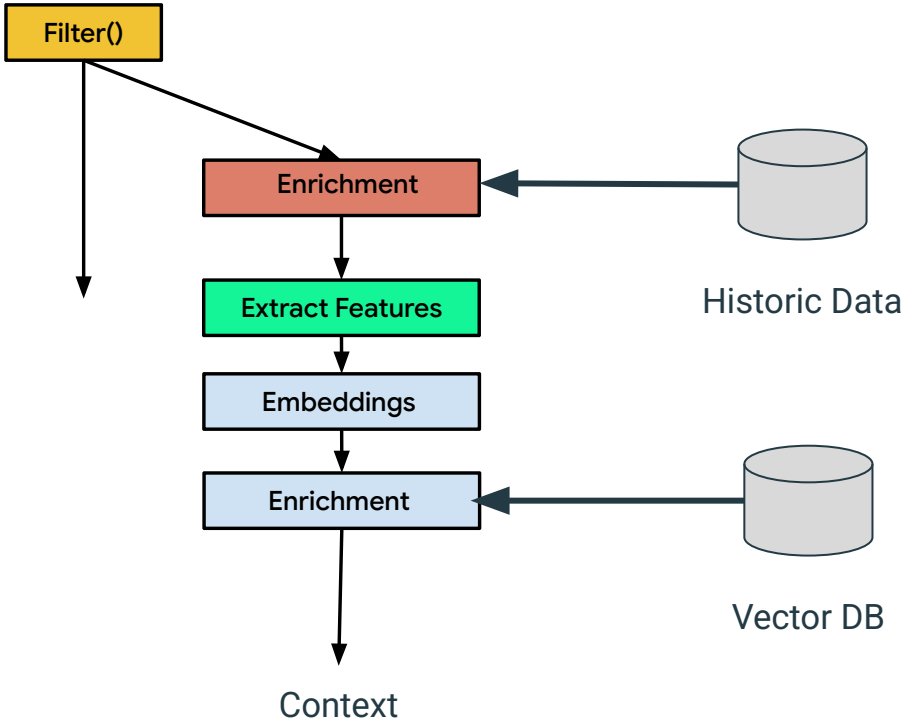
State and timers

OrderedListState



1 - Context : Retrieval Augmented Generation (RAG)

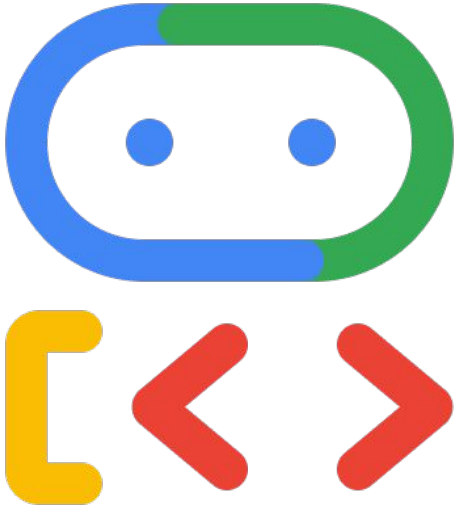
```
...
|
'Generate Embeddings' >>
MLTransform(write_artifact_location=tempfile
.mkdtemp())
       .with_transform(huggingface_embedder)
| 'Vector Search' >>
Enrichment(search_handler)
```



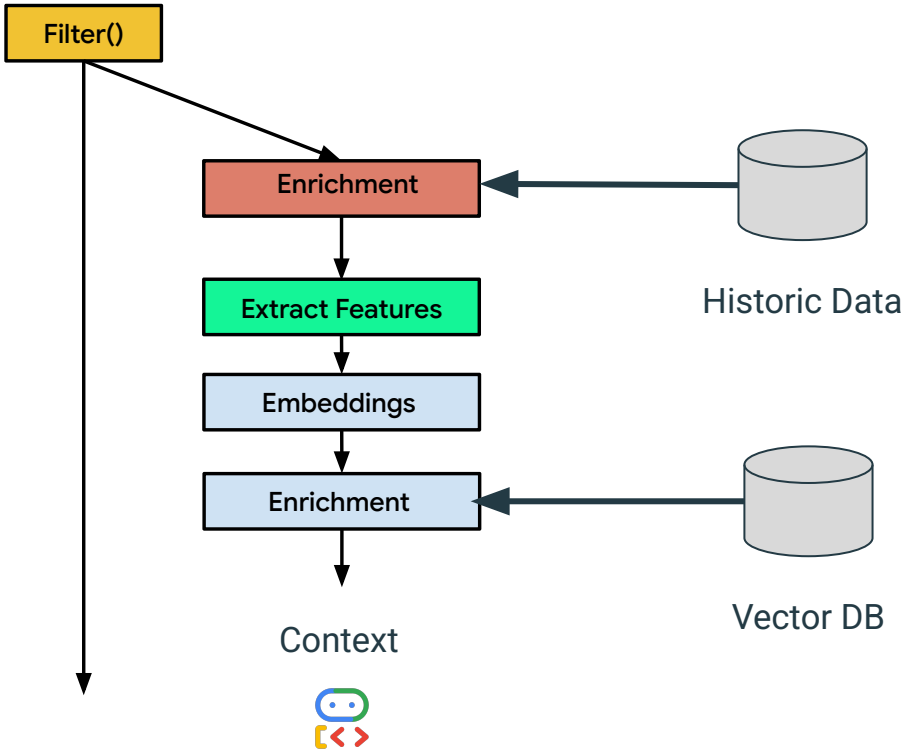
2 - Apache Beam Agents

Agent Development Kit (ADK) is designed to empower developers to quickly build, manage, evaluate and deploy AI-powered agents. https://adk.dev/

```
agent = Agent (
        name="recommender",
        model="...",
        instruction="You help match products to customer needs.",
        tools=[foo_tool, bar_tool, ...]
)
```



Apache Beam: ADK Integration 2.73.0

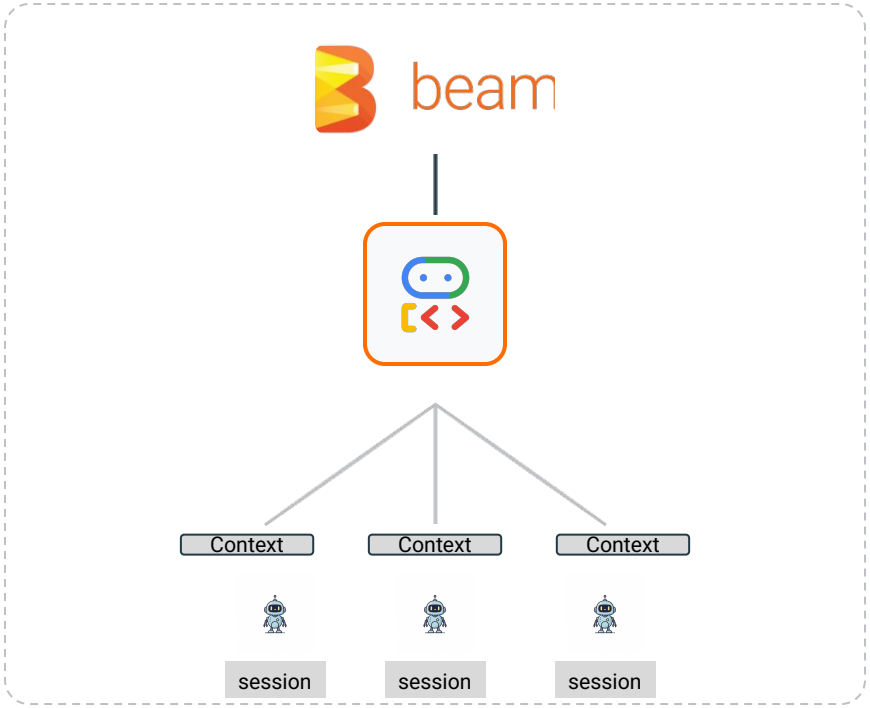


2 - Apache Beam Agents

Session memory service options

- **InMemorySessionService**
  (Recommended only for Stateless)
- DatabaseSessionService
- VertexAiSessionService
- Custom

Passed in with elements as **session_id**



Summary

- Agents add a powerful new tool to the Apache Beam tool kit
  - Create dynamic and adaptive pipelines
- Real time Context can be created from
  - Historic data sources, Enrichment
  - Feature extraction from the In-Stream data
  - RAG with data sources in Vector Databases
- ADK integration, provides easy declarative way to incorporate your ADK agents into your pipeline

Whats next?

- Models running locally , like Gemma made easy with ADK