

Databricks' Data Pipelines: Journey and Lessons Learned

Yu Peng, Burak Yavuz

07/06/2016



Who Are We

Yu Peng



Data Engineer at Databricks

Building Databricks' next-generation data pipeline
on top of Apache Spark

BS in Xiamen University

Ph.D in The University of Hong Kong

Burak Yavuz



Software Engineer at Databricks

Contributor to Spark since Spark 1.1
Maintainer of [Spark Packages](#)

BS in Mechanical Engineering at Bogazici University

MS in Management Science & Engineering at Stanford
University

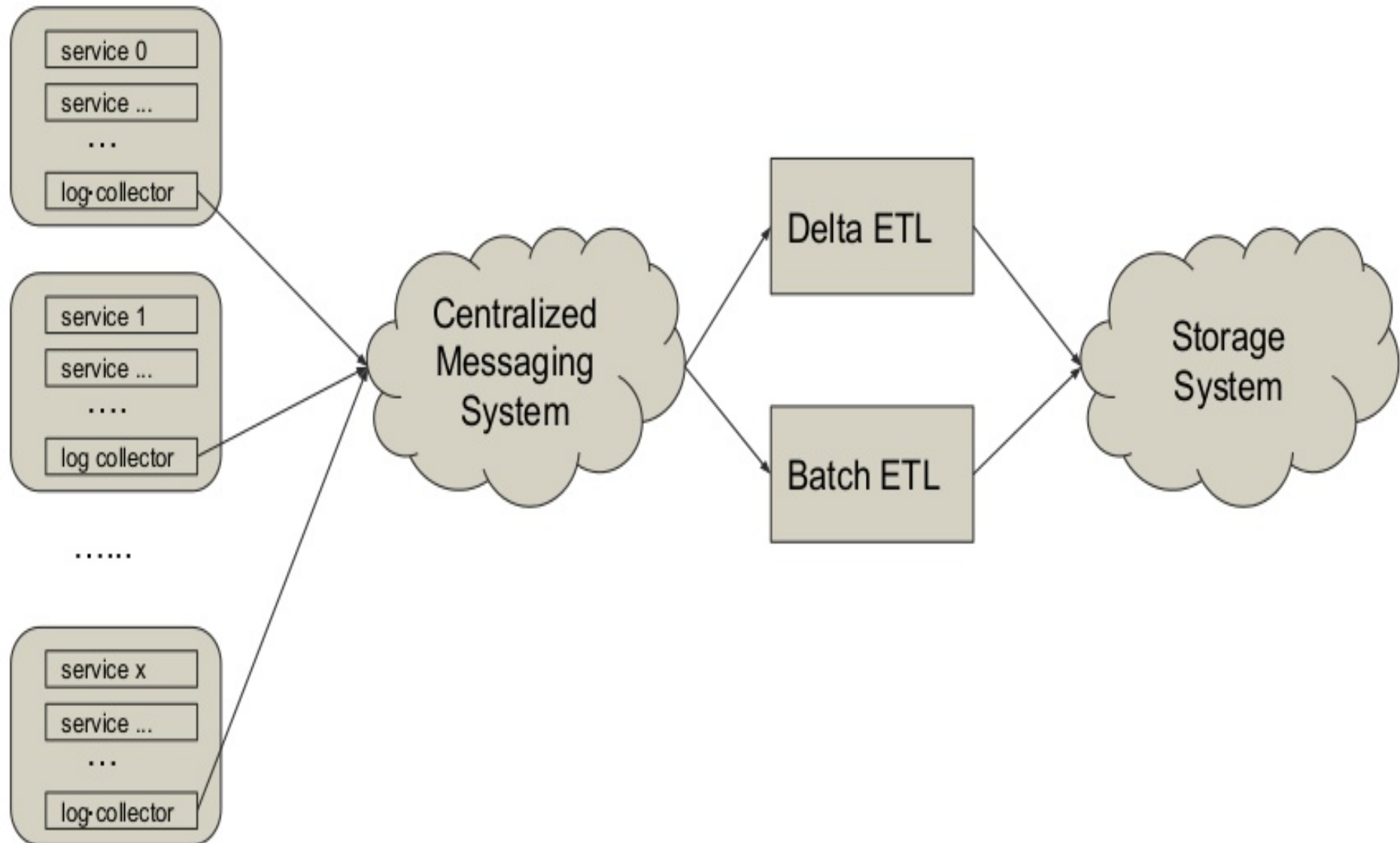
Building a data pipeline is hard

- At least once or exactly once semantics
- Fault tolerance
- Resource management
- Scalability
- Maintainability

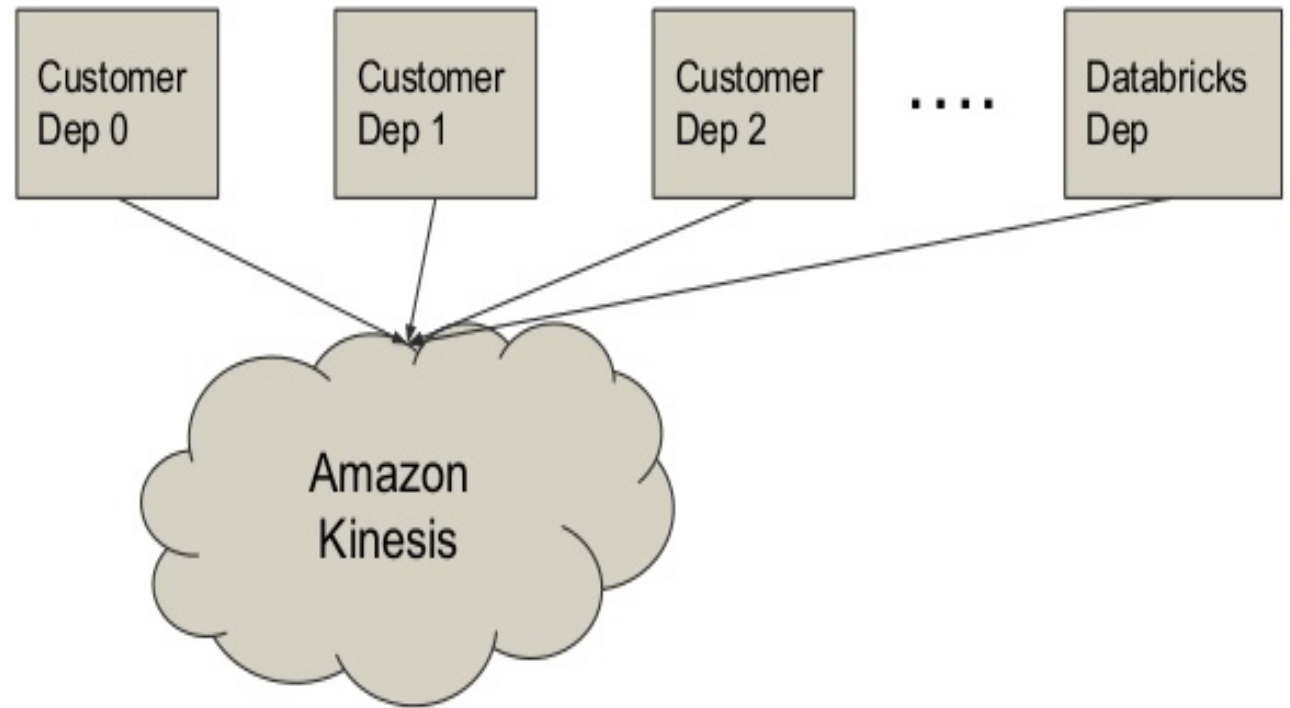
Apache[®] Spark[™] + Databricks = Our Solution

- All ETL jobs are built on top of Apache Spark
 - Unified solution, everything in the same place
- All ETL jobs are run on Databricks platform
 - Platform for Data Engineers and Scientists
- Test out Spark and Databricks new features

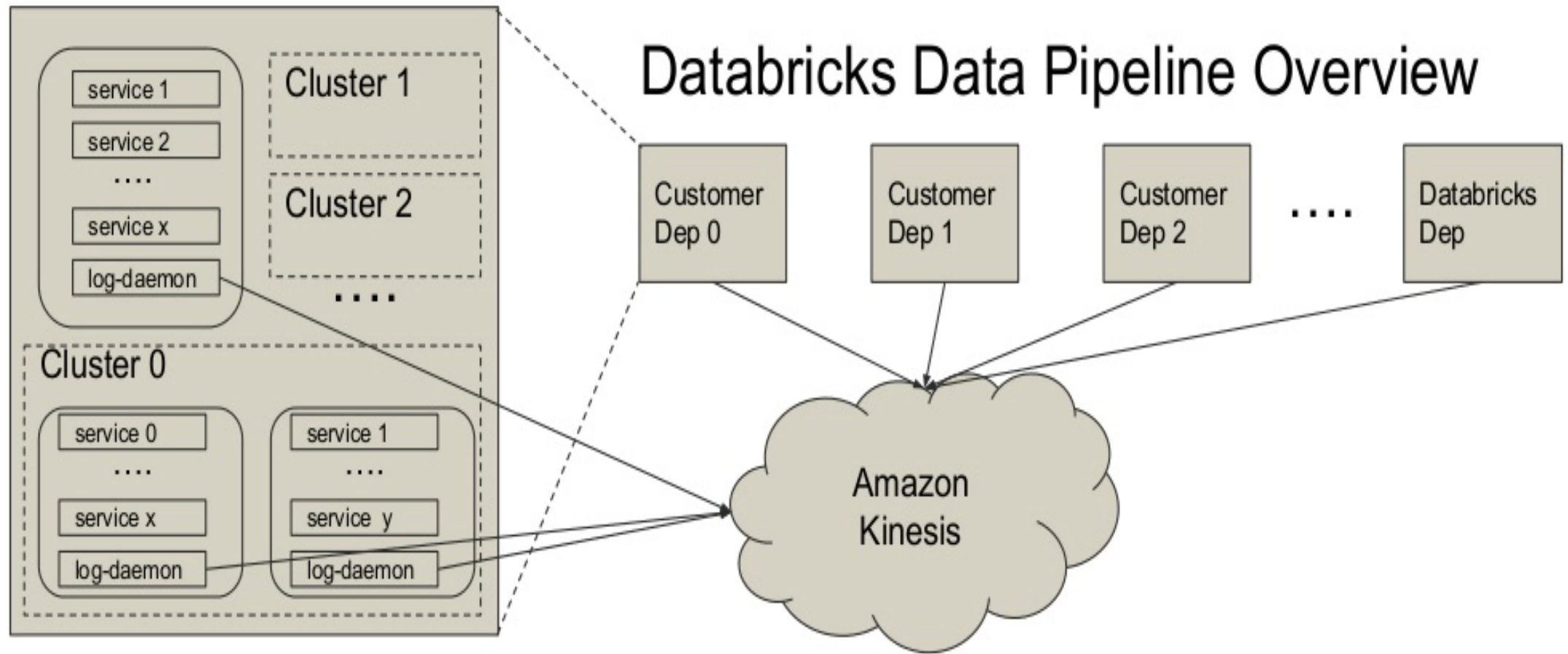
Classic Lambda Data Pipeline



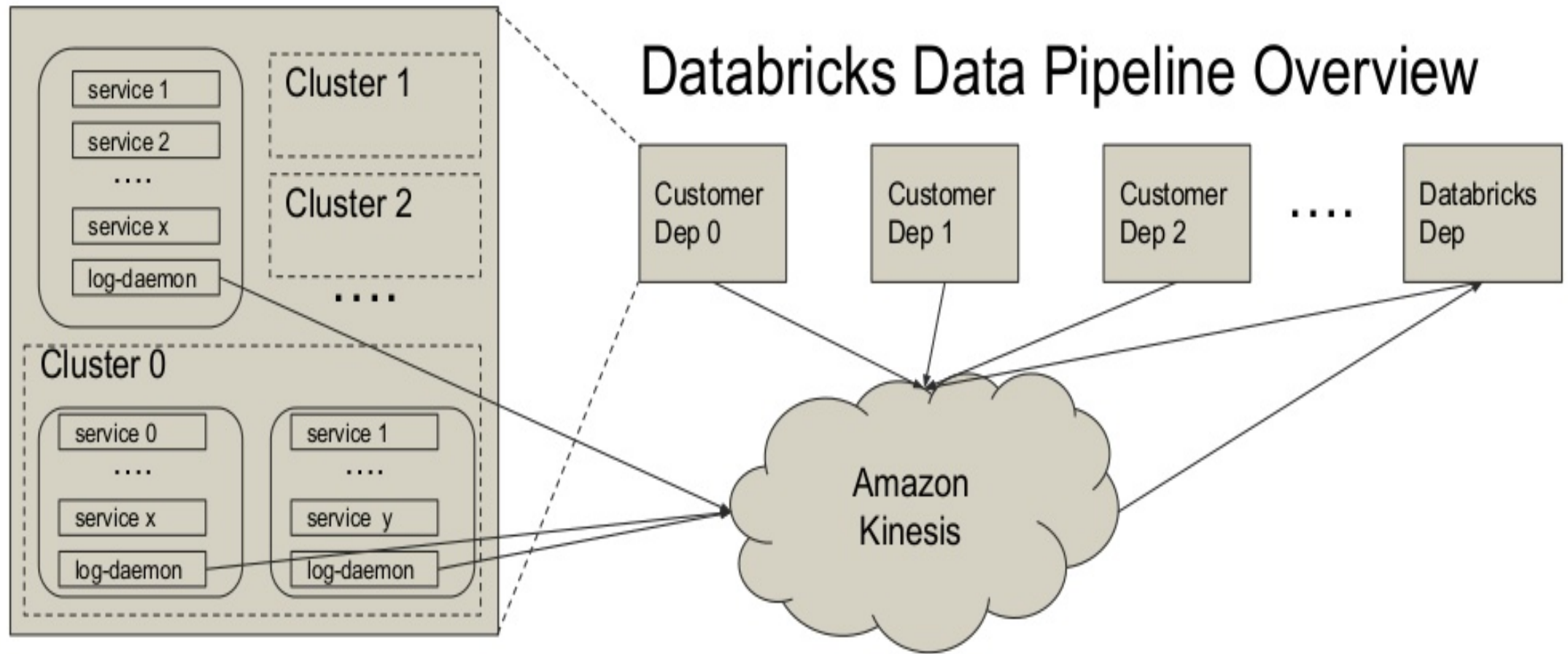
Databricks Data Pipeline Overview



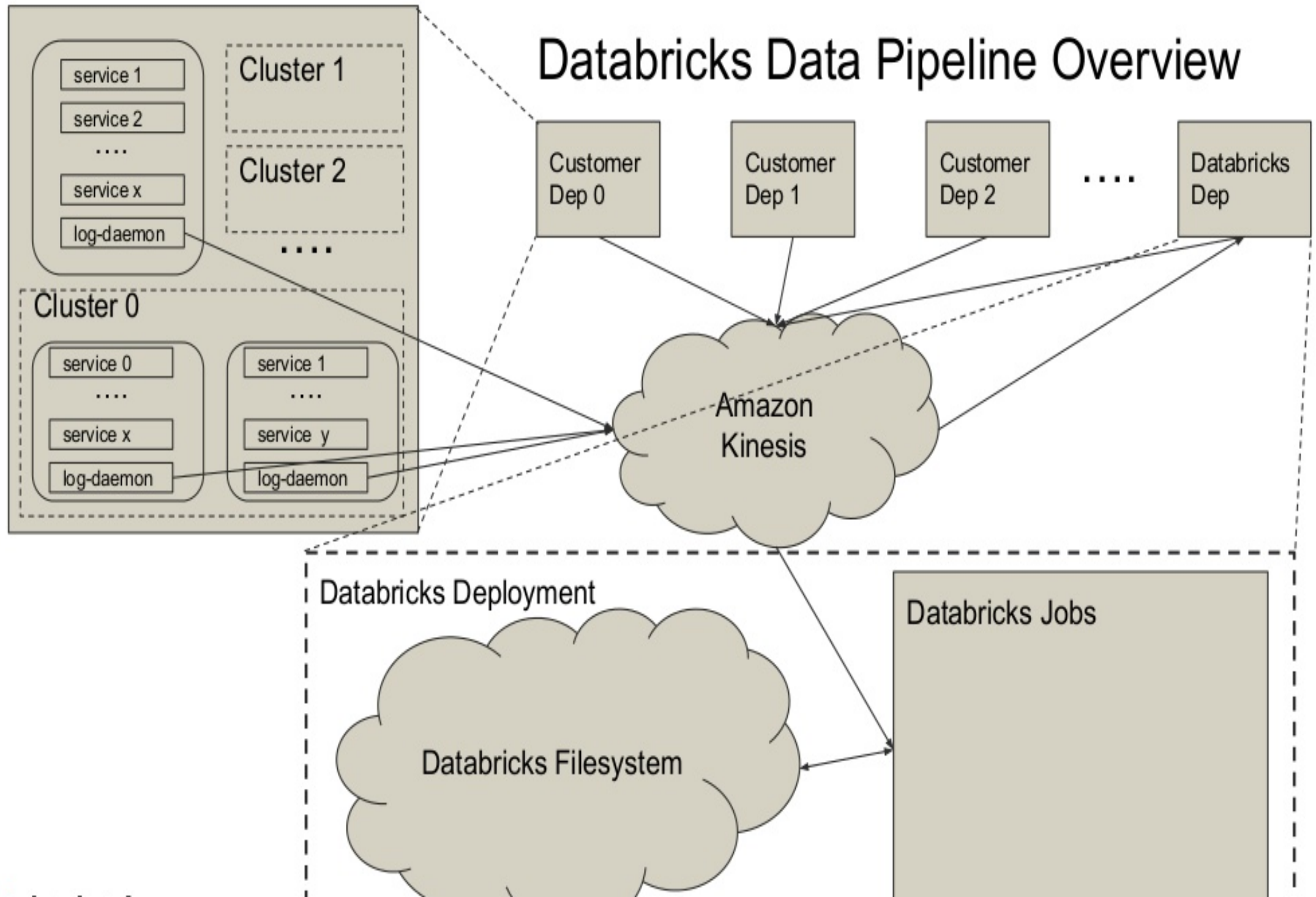
Databricks Data Pipeline Overview



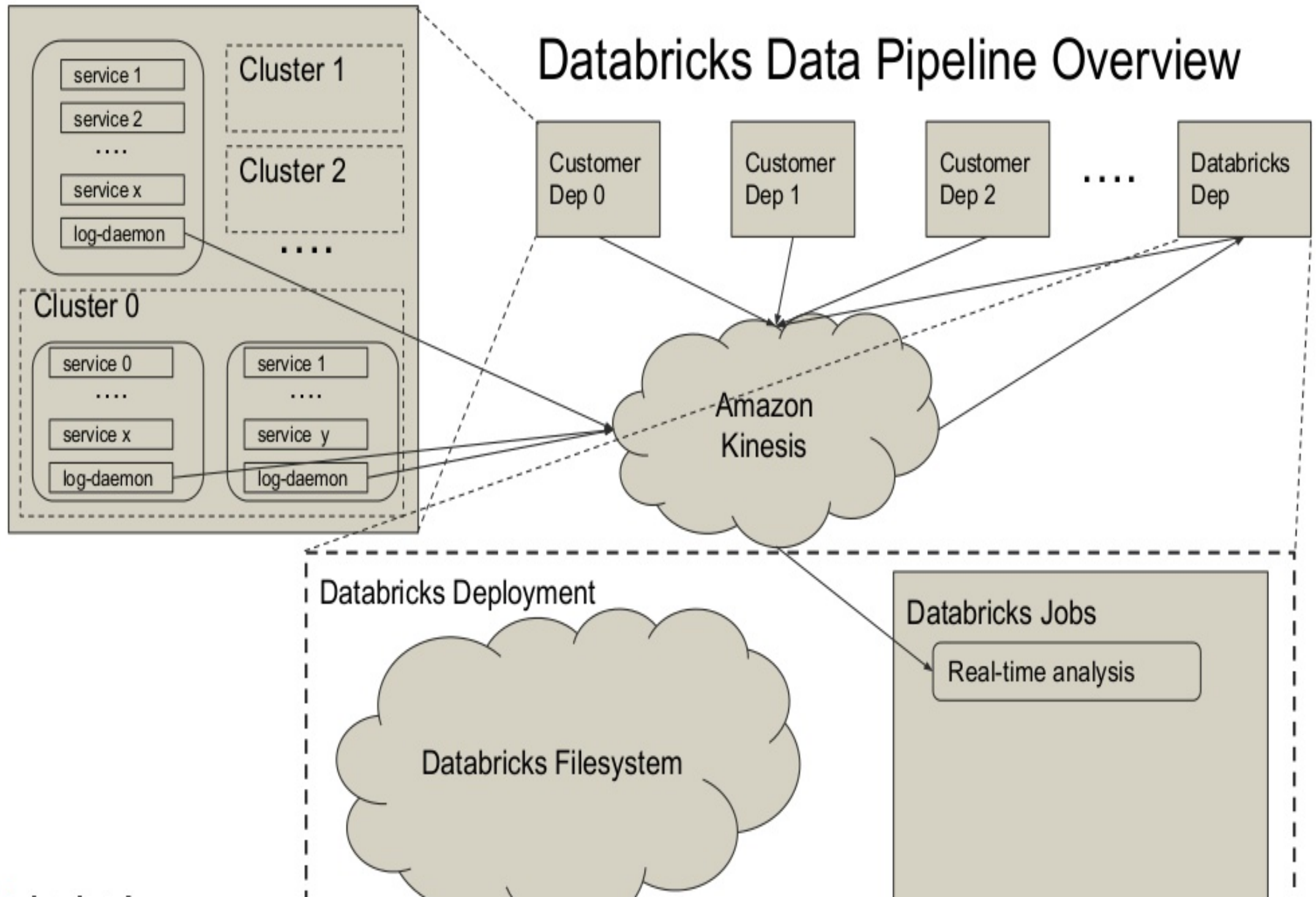
Databricks Data Pipeline Overview



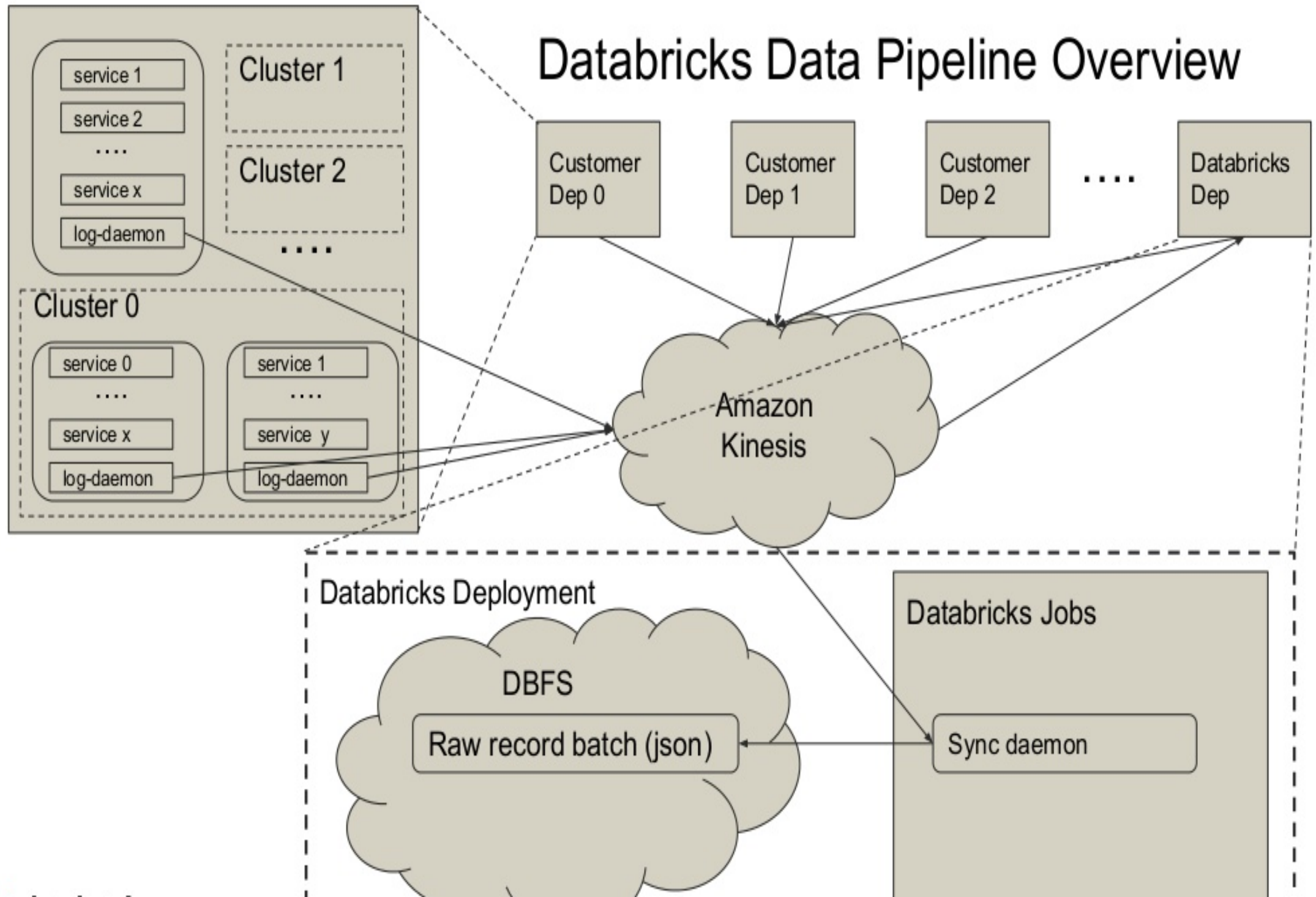
Databricks Data Pipeline Overview



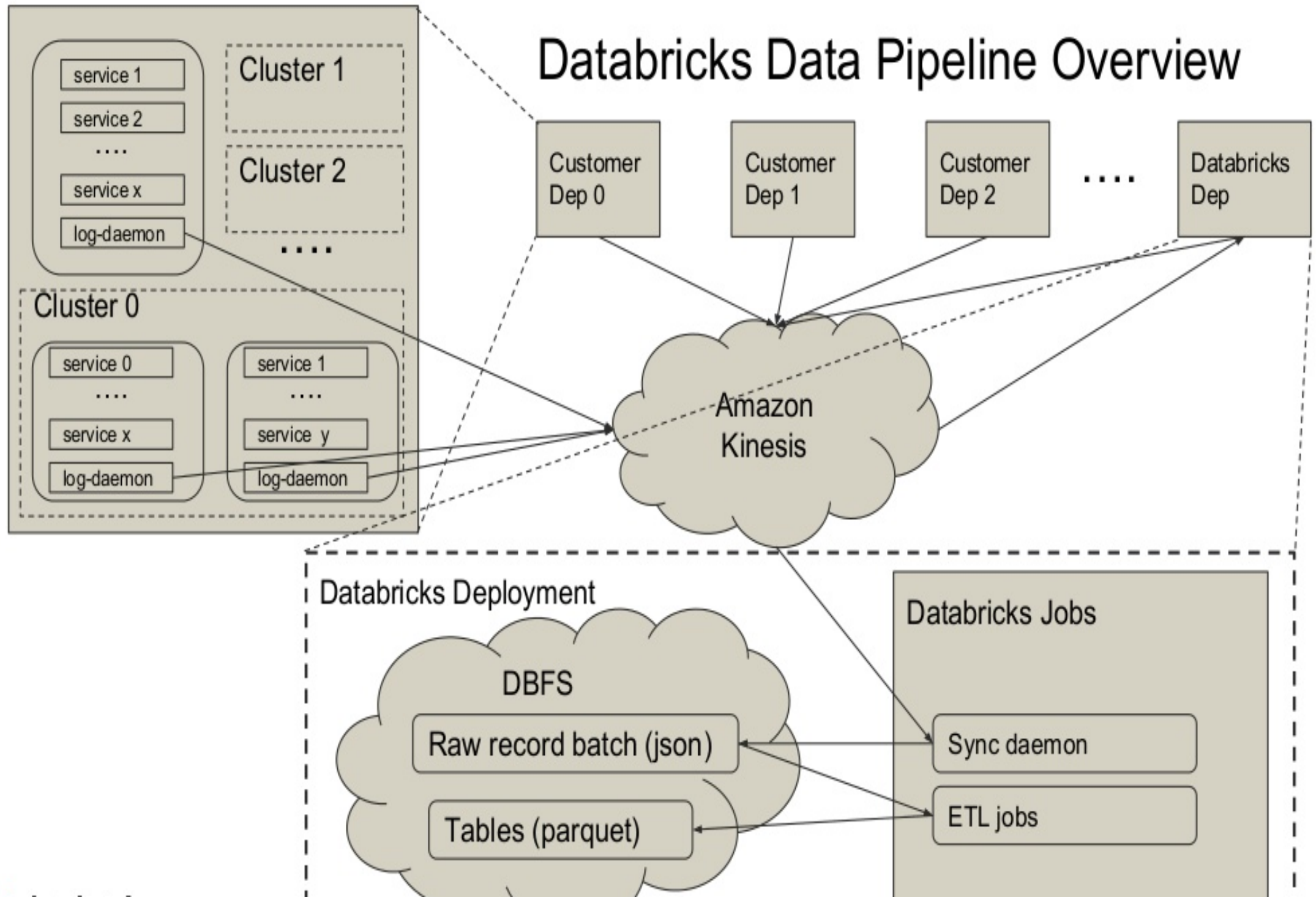
Databricks Data Pipeline Overview



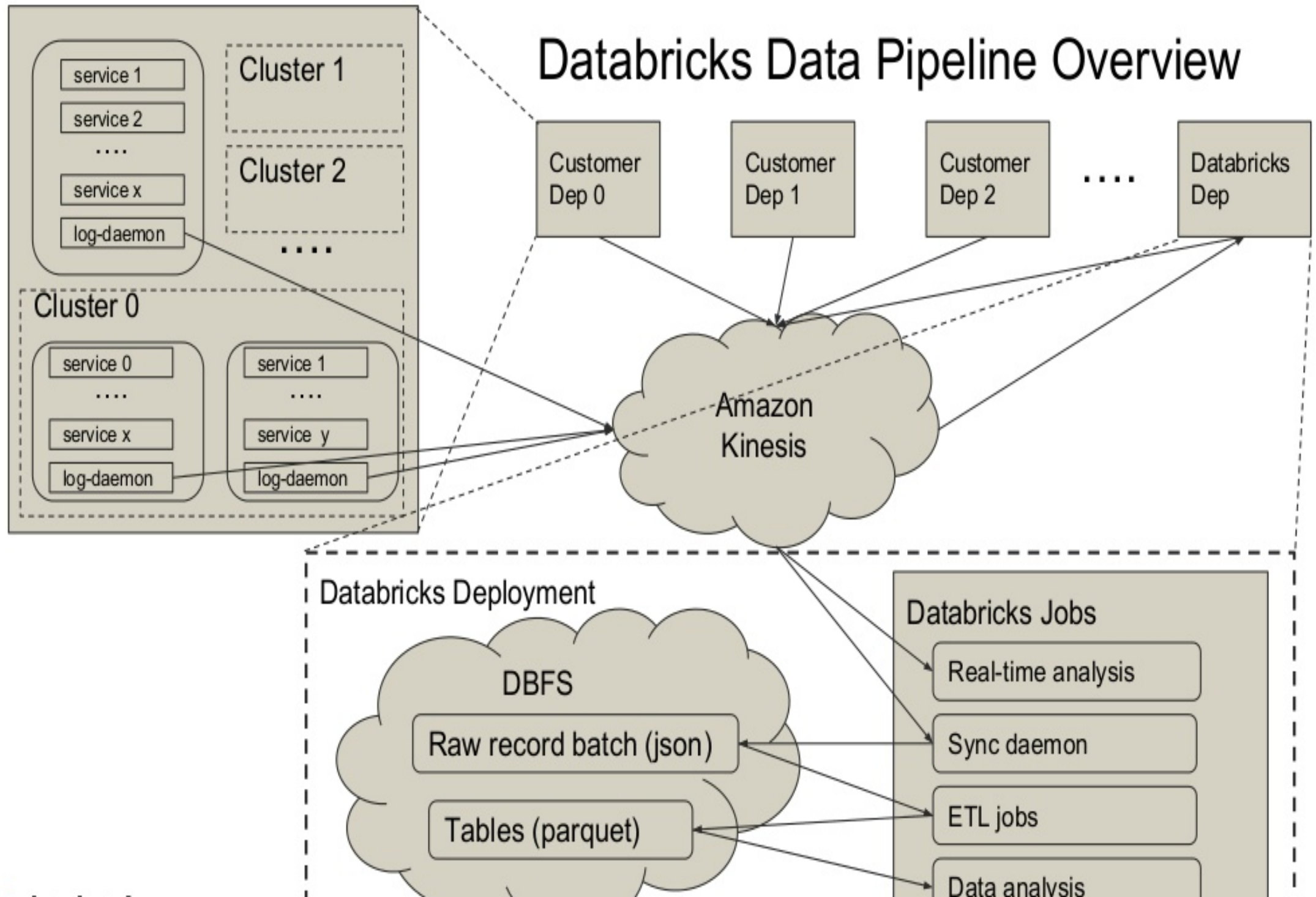
Databricks Data Pipeline Overview



Databricks Data Pipeline Overview



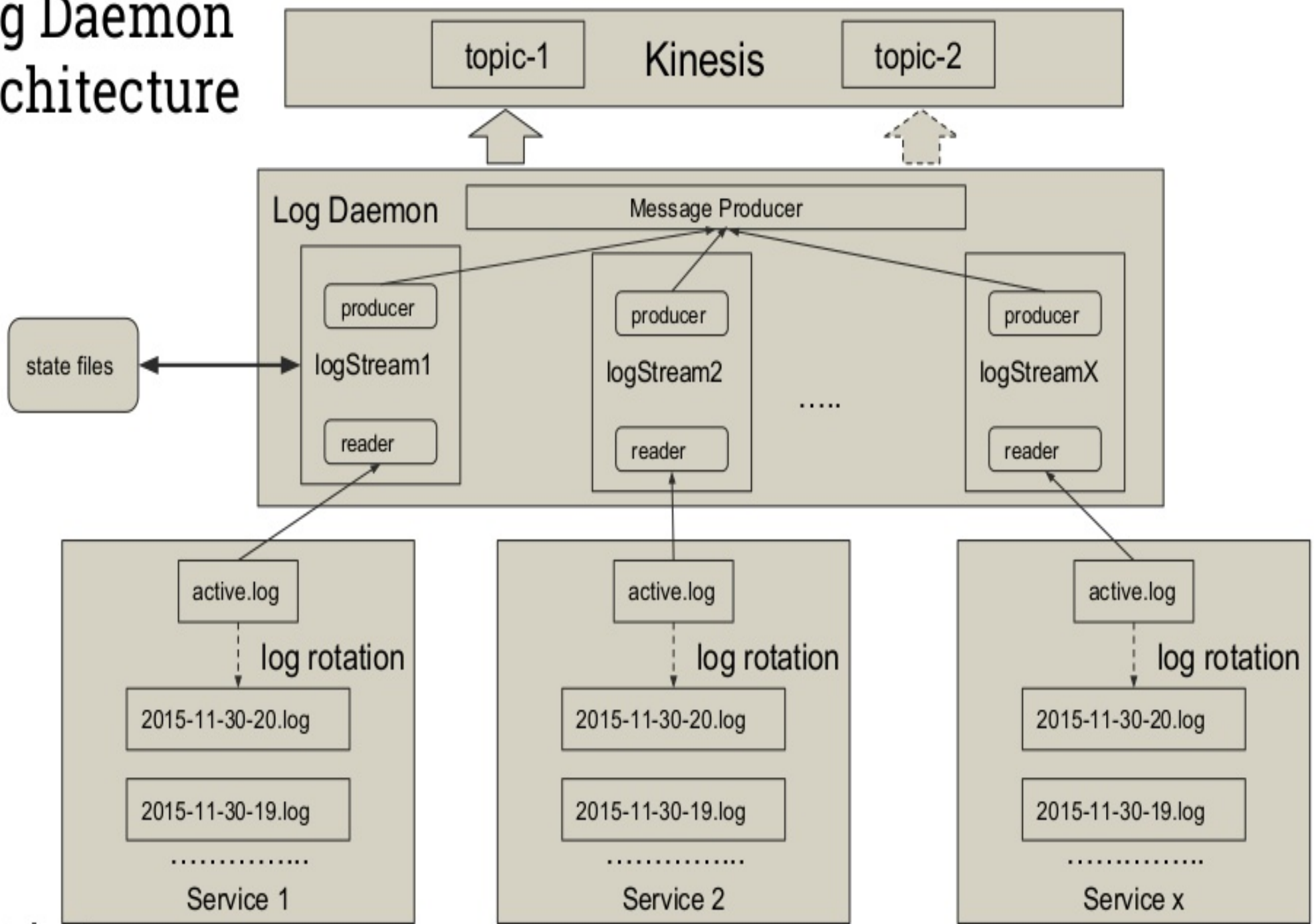
Databricks Data Pipeline Overview



Log collection (Log-daemon)

- Fault tolerance and at least once semantics
- Streaming
- Batch
 - Spark History Server
- Multi-tenant and config driven
 - Spark container

Log Daemon Architecture

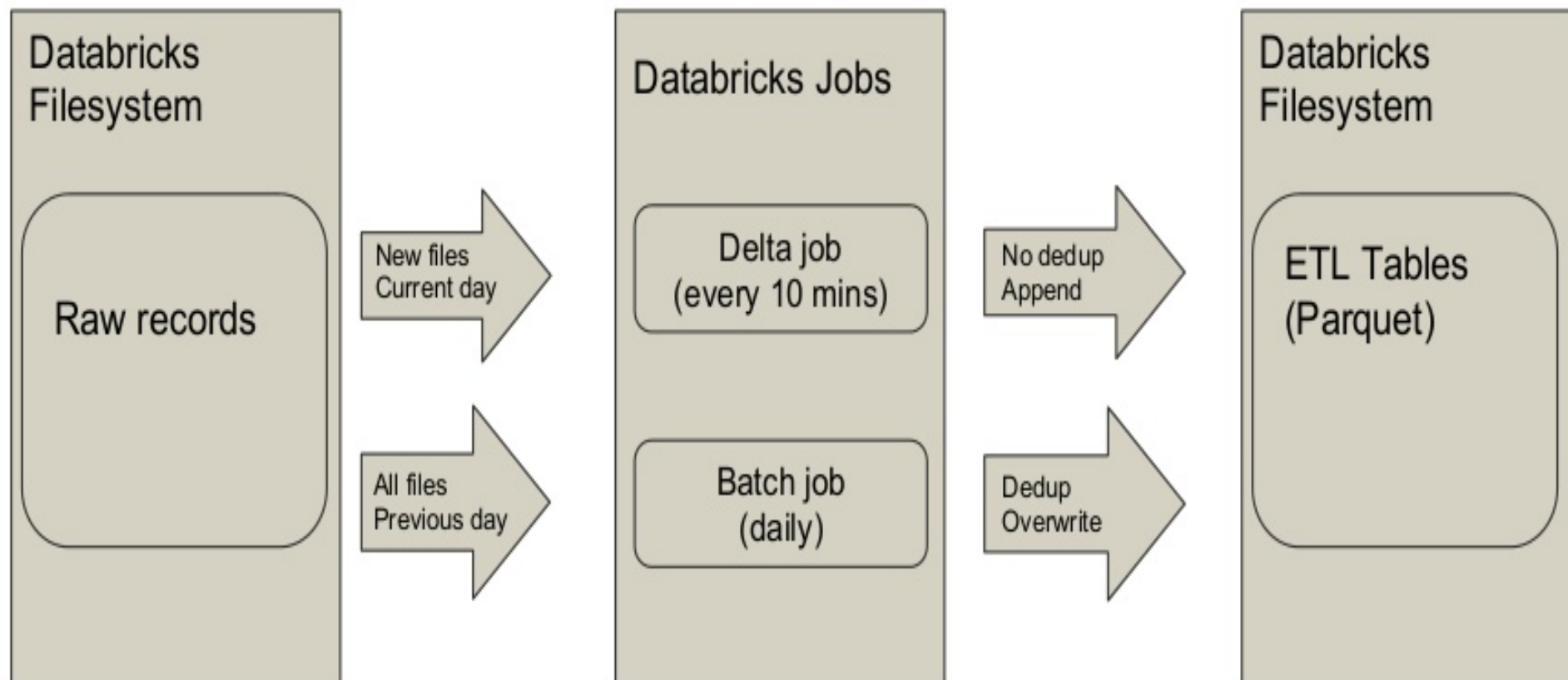


Sync Daemon

- Read from Kinesis and Write to DBFS
 - Buffer and write in batches (128 MB or 5 Mins)
 - Partitioned by date
- A long running Apache Spark job
 - Easy to scale up and down

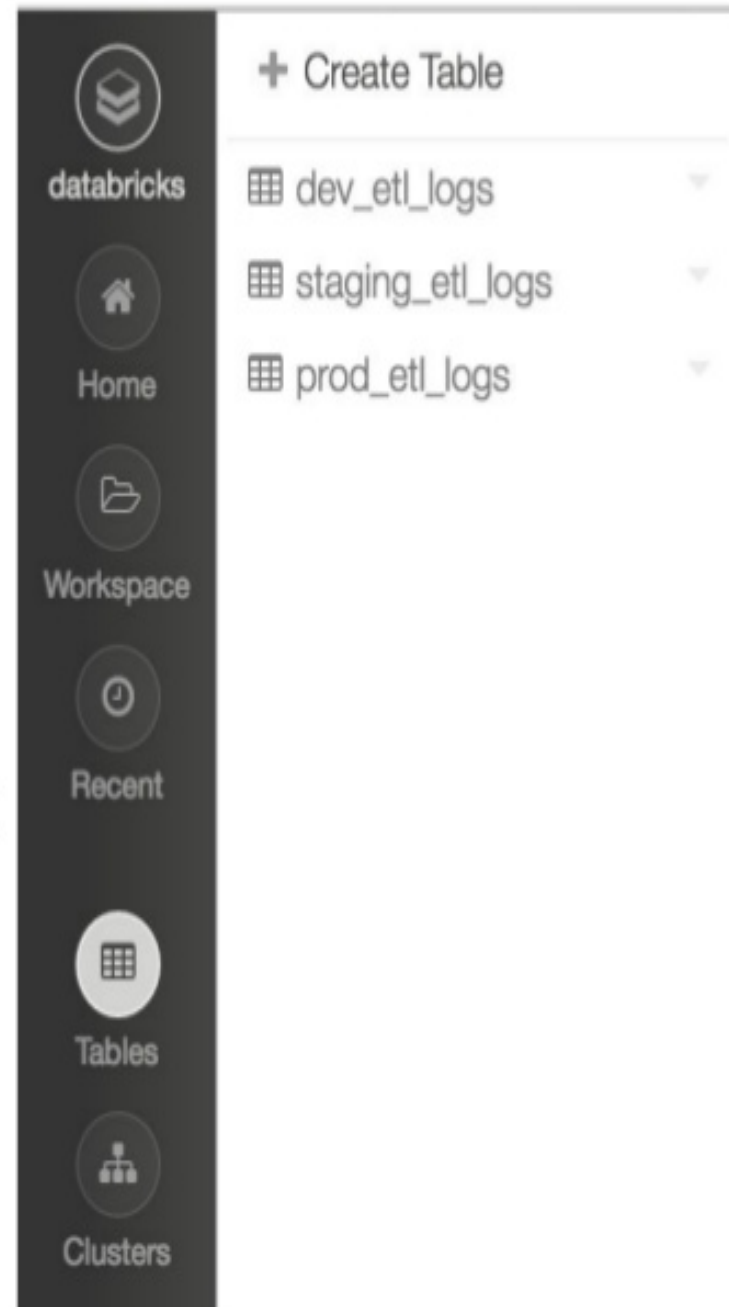
ETL Jobs

Databricks Deployment



ETL Jobs

- Use the same code for Delta and Batch jobs
- Run as scheduled Databricks jobs
- Use spot instances and fallback to on-demand
- Deliver to Databricks as parquet tables



Lessons Learned

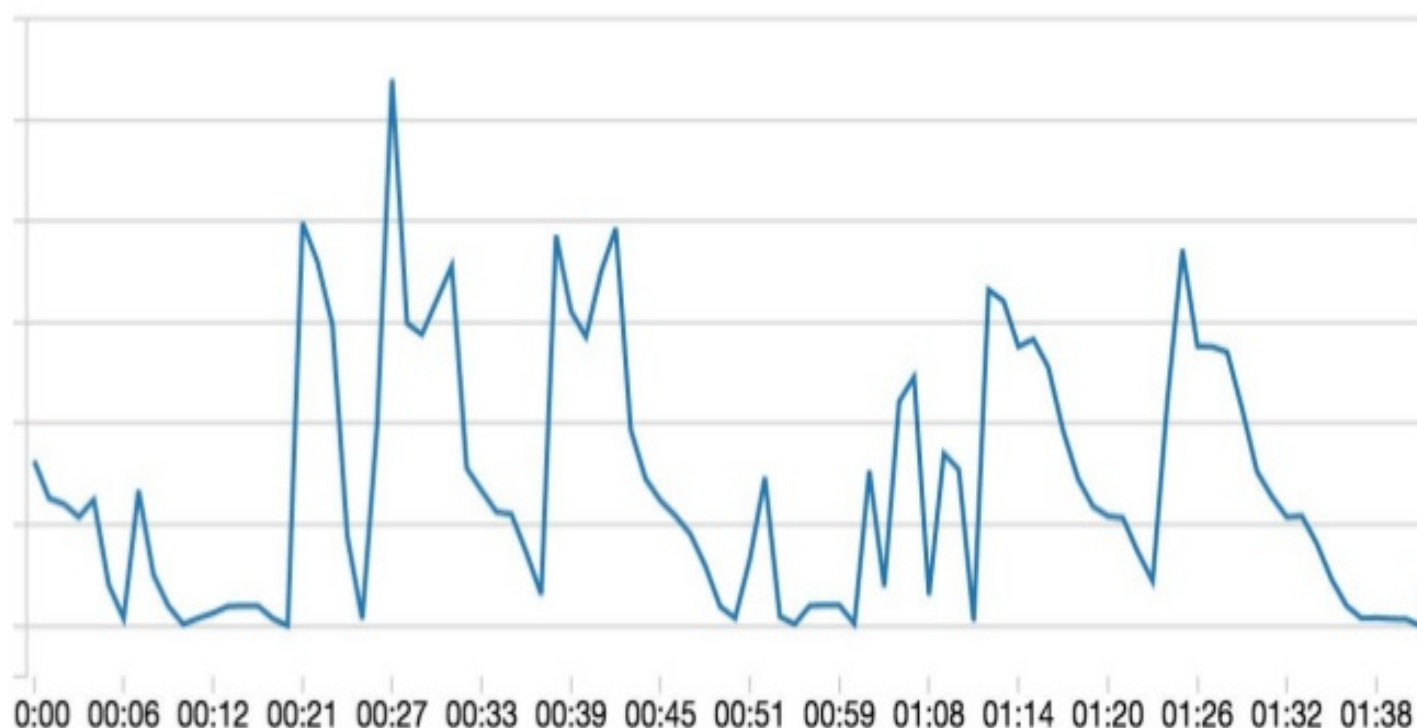
- Partition Pruning can save a lot of time and money

Reduced query time from 2800 seconds to just 15 seconds.

Don't partition too many levels as it leads to worse metadata discovery performance and cost.

Lessons Learned

- High S3 costs: Lots of LIST Requests



Metadata discovery on S3 is expensive. Spark SQL tries to refresh it's metadata cache even after write operations.

Running It All in Databricks - Jobs

databricks

Home

Workspace

Recent

Tables

Clusters

Jobs

Search

Log ETL: Prod - Service Logs / Batch

[?](#)

[< All Jobs](#)

Log ETL: Prod - Service Logs / Batch

Task: Notebook at [/Users/etl@databricks.com/Log_ETL/Log_ETL_Prod-Service_Logs_Batch](#) - [Edit](#) / [Remove](#)

◦ Dependent Libraries: [Add](#)

▪ [log_processing.jar](#) - (Jar) [Remove](#)

Cluster: Memory Optimized, 930 GB, Spot, fall back to On-Demand, Spark 1.6.1 (Hadoop 1) [Edit](#)

Schedule: Every day at 10:00pm (US/Pacific) [Edit](#) / [Remove](#)

Advanced ▾

Alerts: [Edit](#)

◦ On error: *****@databricks.com

Timeout: 360 minutes [Edit](#) / [Remove](#)

Retries: Limit 5x , 5 min delay [Edit](#) / [Remove](#)

Active runs

Run	Start Time	Launched	Duration	Status
No active runs. Run Now				

Completed runs

[Latest successful run \(refreshes automatically\)](#)

[< Previous 20](#)

[Next 20 >](#)

Run	Start Time	Launched	Duration	Status
Run 296	2016-06-05 22:00:00	By scheduler	1h 49m 46s	Succeeded
Run 295	2016-06-04 22:00:00	By scheduler	1h 17m 14s	Succeeded
Run 294	2016-06-03 22:00:00	By scheduler	54m 57s	Succeeded
Run 293	2016-06-02 22:00:00	By scheduler	49m 39s	Succeeded
Run 292	2016-06-01 22:00:00	By scheduler	1h 12m 54s	Succeeded

Running It All in Databricks - Spark

The screenshot displays the Databricks web interface. On the left is a sidebar with navigation icons for Home, Workspace, Recent, Tables, Clusters, Jobs, and Search. The main content area shows the configuration for a job named 'Log ETL: Prod - Service Logs / Batch'. A 'Configure Cluster' modal dialog is open in the foreground.

Configure Cluster [Cancel] [Create Cluster]

30 Workers, 900 GB Memory, 120 Cores and 1 Driver, 30 GB Memory, 4 Cores

Cluster Type: New Cluster

Spark Version:

- ✓ Spark 1.6.1 (Hadoop 1)
- Spark 1.3.0 (Hadoop 1)
- Spark 1.4.1 (Hadoop 1)
- Spark 1.5.2 (Hadoop 1)
- Spark 1.6.0 (Hadoop 1)
- Spark 1.6.1 (Hadoop 2)
- Spark 2.0 (apache/branch-2.0 preview)

Size: 30

Workers (900 GB Memory, 120 Cores)
Spot Driver (30 GB Memory, 4 Cores)

[Show advanced settings]

Background Job Configuration:

Log ETL: Prod - Service Logs / Batch

< All Jobs

Task: Notebook at /Users/etl@dat

- Dependent Libraries: Add
 - log_processing.jar - (Jar)

Cluster: Memory Optimized, 930

Schedule: Every day at 10:00pm

Advanced ▾

Alerts: Edit

- On error: *****@databrick

Timeout: 360 minutes Edit / Re

Retries: Limit 5x , 5 min delay E

Active runs

Run

No active runs. Run Now

Completed runs

Latest successful run (refreshes a

< Previous 20

Run

Data Analysis & Tools

We get the data in. What's next?

- Monitoring
- Debugging
- Usage Analysis
- Product Design (A/B testing)

Debugging

Access to logs in a matter of seconds thanks to Apache Spark.

[Exit](#)

Dump Service Logs

Update

1. Customer name : 2. Service name :
3. Start time (yyyy-MM-dd HH:mm:ss) : 4. End time : 5. Time Zone :

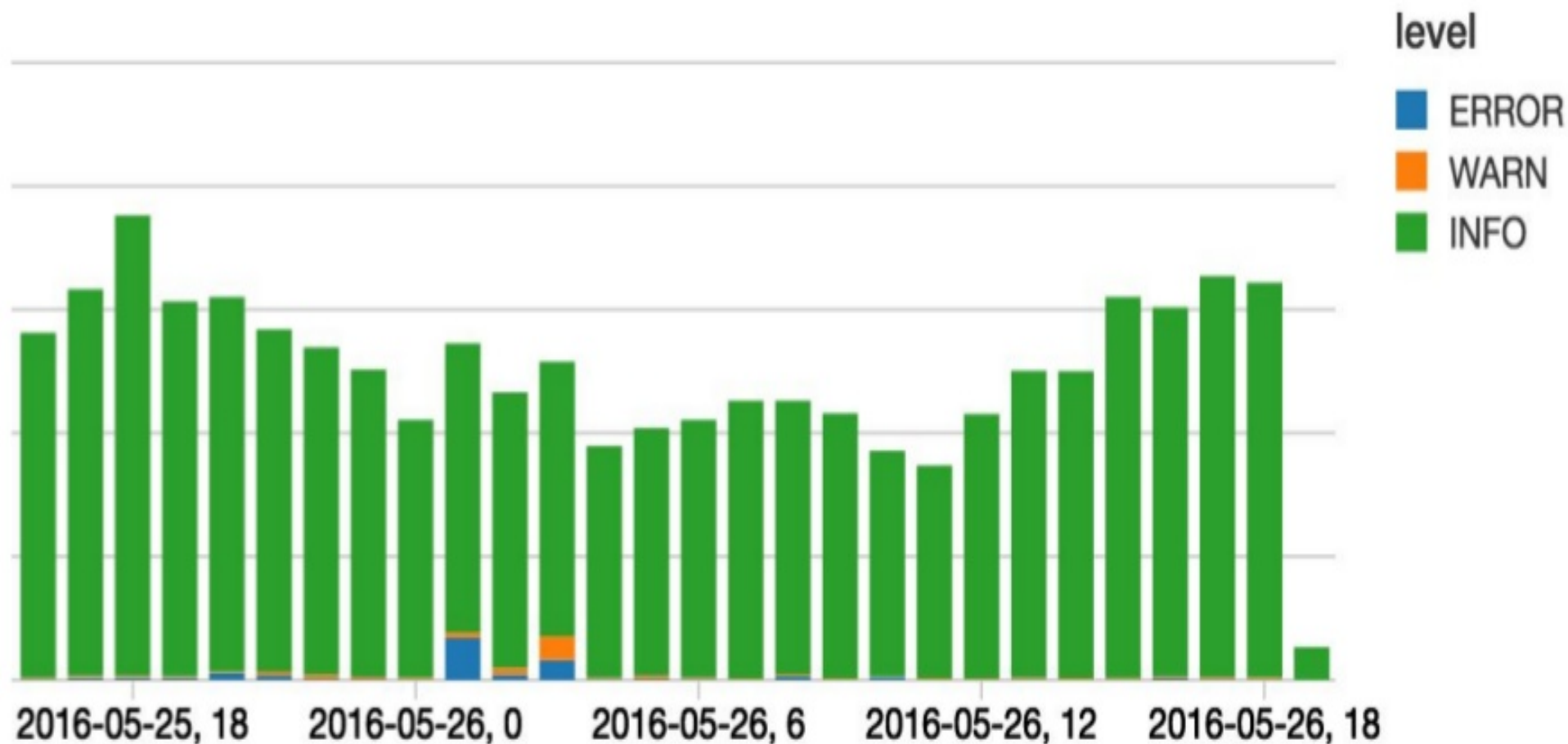
Please provide shard/service info and start/end times above and then click "Update" on the top right.

If you prefer calling `dumpServiceLogs` method directly, please visit [this notebook](#).

Download service logs: [dbc-*****_webapp_20160602-020000_20160602-040000.tar.gz](#)

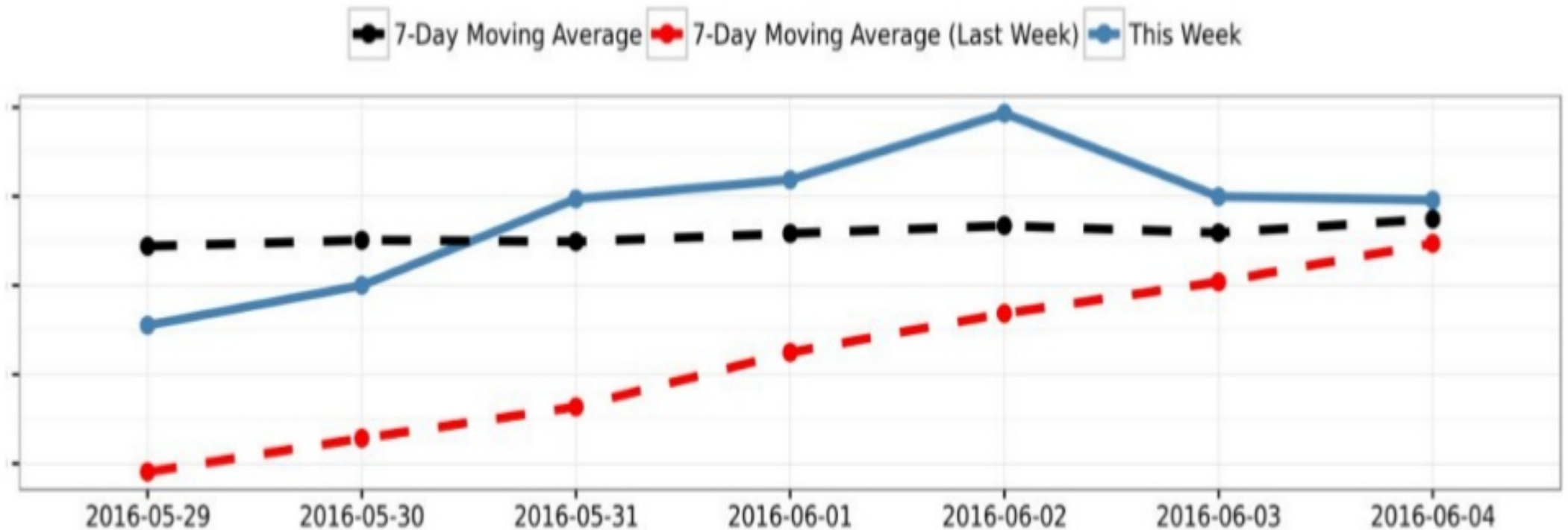
Monitoring

Monitor logs by log level. Bug introduced on 2016-05-26 01:00:00 UTC. Fix deployed in 2 hours.



Usage Analysis + Product Design

SparkR + ggplot2 = Match made in heaven



Summary

Databricks + Apache Spark create a unified platform for:

- ETL
- Data Warehousing
- Data Analysis
- Real time analytics

Issues with DevOps out of the question:

- No need to manage a huge cluster
- Jobs are isolated, they don't cannibalize each other's resources
- Can launch any Spark version

Ongoing & Future Work

Structured Streaming

- Reduce Complexity of pipeline:
Sync Daemon + Delta + Batch Jobs => Single Streaming Job
- Reduce Latency
Availability of data in seconds instead of minutes
- Event Time Dashboards

Try Apache Spark with Databricks

<http://databricks.com/try>

Thank you.

Have questions about ETL with Spark?
Join us at the Databricks Booth 3.45-6.00pm!

