Dive into Streams with Brooklin

Celia Kung
LinkedIn
<table>
<thead>
<tr>
<th>Outline</th>
</tr>
</thead>
<tbody>
<tr>
<td>Background</td>
</tr>
<tr>
<td>Scenarios</td>
</tr>
<tr>
<td>Application Use Cases</td>
</tr>
<tr>
<td>Architecture</td>
</tr>
<tr>
<td>Current and Future</td>
</tr>
</tbody>
</table>
Background
Nearline Applications

- Require near real-time response
- Thousands of applications at LinkedIn
  - E.g. Live search indices, Notifications
Nearline Applications

• Require continuous, low-latency access to data
  ○ Data could be spread across multiple database systems
• Need an easy way to move data to applications
  ○ App devs should focus on event processing and not on data access
Heterogeneous Data Systems

- Oracle
- kafka
- MySQL
- Microsoft EventHubs
- Espresso
  (LinkedIn’s document store)
- Amazon Kinesis
- DynamoDB Streams
Building the Right Infrastructure

- Build separate, specialized solutions to stream data from and to each different system?
  - Slows down development
  - Hard to manage!
Need a centralized, managed, and extensible service to continuously deliver data in near real-time
Brooklin
Brooklin

- **Streaming** data pipeline service
- Propagates data from **many** source types to **many** destination types
- **Multitenant:** Can run several thousand streams simultaneously
- Streams are **dynamically provisioned** and **individually configured**
- **Extensible:** Plug-in support for additional sources/destinations
Scenarios
Scenario 1:

Change Data Capture
Capturing Live Updates

1. Member updates her profile to reflect her recent job change
Capturing Live Updates

2. LinkedIn wants to inform her colleagues of this change.
Capturing Live Updates

Updates

Member DB

query

News Feed Service

Mochi K's job update

Congratulate Mochi for starting a new position as Sales Clerk at Zzz Catnip Dispensary

Like

Comment

Congrats Mochi
Congratulations!
Capturing Live Updates

Updates

Member DB

News Feed Service

Search Indices Service

Mochi K’S job update

Congrats Mochi Congratulations! Con

zzz catnip dispensary Filters

People Jobs Content Companies Schools

1 result

Mochi K • You
Sales Clerk at Zzz Catnip Dispensary
San Francisco Bay Area
Capturing Live Updates

- Member DB
- Updates
- News Feed Service
- Search Indices Service
- Notifications Service
- Standardization Service

Diagram showing flow from Member DB to various services.
Capturing Live Updates

Member DB

Updates

News Feed Service

Search Indices Service

Notifications Service

Standardization Service
Capturing Live Updates

Updates

Member DB

News Feed Service

Search Indices Service

Notifications Service

Standardization Service

query

query

query

query
Change Data Capture (CDC)

- Brooklin can stream database updates to a change stream
- Data processing applications consume from **change streams**

**Isolation:** Applications are decoupled from the sources and don’t compete for resources with online queries
- Applications can be at **different points** in change timelines
Change Data Capture (CDC)
Scenario 2:

Streaming Bridge
Stream Data from X to Y

- Across...
  - cloud services
  - clusters
  - data centers
Streaming Bridge

- **Data pipe** to move data between different environments
- Enforce **policy**: Encryption, Obfuscation, Data formats

Diagram:

- AWS
  - Kinesis
  - Kinesis

- Azure
  - EventHubs
  - EventHubs

- LinkedIn
  - Kafka
  - Kafka
  - Kafka
  - Kafka

Diagram shows the integration of AWS Kinesis with Azure EventHubs through LinkedIn Kafka, with brooklin as the streaming bridge.
Mirroring Kafka Data

- Aggregating data from all data centers into a centralized place
- Moving data between LinkedIn and external cloud services (e.g. Azure)
- Brooklin has replaced Kafka MirrorMaker (KMM) at LinkedIn
  - Issues with KMM: didn’t scale well, difficult to operate and manage, poor failure isolation
Use Brooklin to Mirror Kafka Data

Sources
- Databases
  - Oracle
  - MySQL
- Messaging systems
  - Microsoft EventHubs
  - Amazon Kinesis

Destinations
- Databases
  - Couchbase
- Messaging systems
  - Microsoft EventHubs
  - Amazon Kinesis

Brooklin
Kafka MirrorMaker Topology
Brooklin Kafka Mirroring

- Optimized for stability and operability
- Manually pause and resume mirroring at every level
  - Entire pipeline, topic, topic-partition
- Can auto-pause partitions facing mirroring issues
  - Auto-resumes the partitions after a configurable duration
- Flow of messages from other partitions is unaffected
Application Use Cases
Application Use Cases

Cache
Application Use Cases

- Cache
- Search Indices
Application Use Cases

Cache
Search Indices
ETL or Data warehouse
Application Use Cases

- Cache
- Search Indices
- ETL or Data warehouse
- Materialized Views or Replication
Application Use Cases

- Cache
- Search Indices
- ETL or Data warehouse
- Materialized Views or Replication
- Repartitioning
Application Use Cases

Adjunct Data
Application Use Cases

Adjunct Data

Bridge
Application Use Cases

Adjunct Data
Bridge
Serde, Encryption, Policy
Standardization, Notifications …
Architecture
Example:
Stream updates made to Member Profile
Capturing Live Updates

Updates

Member DB \(\rightarrow\) \text{brooklin} \(\rightarrow\) kafka \(\rightarrow\) News Feed Service
● **Scenario**: Stream Espresso Member Profile updates into Kafka
  ○ **Source Database**: Espresso (Member DB, Profile table)
  ○ **Destination**: Kafka
  ○ **Application**: News Feed service
Datastream

Name: MemberProfileChangeStream

Source: MemberDB/ProfileTable
   Type: Espresso
   Partitions: 8

Destination: ProfileTopic
   Type: Kafka
   Partitions: 8

Metadata:
   Application: News Feed service
   Owner: newsfeed@linkedin.com

- **Describes** the data pipeline
- **Mapping** between source and destination
- Holds the **configuration** for the pipeline
1. Client makes REST call to create datastream

```
create POST /datastream
```

**Diagram:**
- Client (Brooklin Client)
- Load Balancer
- Datastream Management Service (DMS)
- Coordinator
- Espresso Consumer
- Kafka Producer
- ZooKeeper
- Coordinator (Leader)
- Espresso Consumer
- Kafka Producer
- Brooklin Instance
- Member DB
- News Feed service
2. Create request goes to any Brooklin instance
3. Datastream is written to ZooKeeper

ZooKeeper

Datastream Management Service (DMS)
- Coordinator
- Espresso Consumer
- Kafka Producer

Espresso Consumer
Kafka Producer

Coordinator (Leader)

Datastream Management Service (DMS)
- Coordinator
- Espresso Consumer
- Kafka Producer

Coordinator

Datastream Management Service (DMS)
- Coordinator
- Espresso Consumer
- Kafka Producer

Member DB

News Feed service

Brooklin Instance

Load Balancer

Brooklin Client

kafka
4. Leader coordinator is notified of new datastream
5. Leader coordinator calculates work distribution
6. Leader coordinator writes the assignments to ZK
7. ZooKeeper is used to communicate the assignments.
8. Coordinators hand task assignments to consumers
9. Consumers start streaming data from the source
10. Consumers propagate data to producers
11. Producers write data to the destination
12. App consumes from Kafka
13. Destinations can be shared by apps
Current & Future
### Current

#### Sources & Destinations

- **Consumers:**
  - Espresso
  - Oracle
  - Kafka
  - EventHubs
  - Kinesis

- **Producers:**
  - Kafka
  - EventHubs

- APIs are standardized to support additional sources and destinations

#### Features

- **Multitenant:** Can power thousands of datastreams across several source and destination types

- **Guarantees:** At-least-once delivery, order is maintained at partition level

- **Kafka mirroring improvements:** finer control of pipelines (pause/auto-pause partitions), improved latency with flushless-produce mode
Brooklin in Production

Brooklin streams with *Espresso*, *Oracle*, or *EventHubs* as the source

| 38B messages/day | 2K+ datastreams | 1K+ unique sources | 200+ applications |
Brooklin in Production

Brooklin streams mirroring Kafka data

2T+ messages/day
200+ datastreams
10K+ topics
### Future

#### Sources & Destinations
- **Consumers:**
  - MySQL
  - Cosmos DB
  - Azure SQL
- **Producers:**
  - Azure Blob storage
  - Kinesis
  - Cosmos DB
  - Azure SQL
  - Couchbase

#### Optimizations
- Brooklin auto-scaling
- Passthrough compression
- Read optimizations: Read once, Write multiple

#### Open Source
- Plan to open source Brooklin in 2019 (soon!)
Thank you
Questions?

Celia Kung

✉️: ckung@linkedin.com

พา: /in/ceiliakkung/