Taking the Pain Out of Developing and Deploying Streaming Applications

Craig Blitz, Senior Product Director
Gerard Maas, Senior SW Engineer
Disclaimer

- © Lightbend Inc 2018. All Rights Reserved

- The information contained in this presentation is provided for informational purposes only. This information is based on Lightbend’s current product plans and strategy, which are subject to change by Lightbend without notice. Lightbend shall not be responsible for any damages arising out of the use of, or otherwise related to, this presentation.

- References in this presentation to Lightbend products, programs, or services do not imply that they will be available in all countries in which Lightbend operates.

- Product release dates and/or capabilities referenced in this presentation may change at any time at Lightbend’s sole discretion based on market opportunities or other factors, and are not intended to be a commitment to future product or feature availability in any way.
Disclaimer

• © Lightbend Inc 2018. All Rights Reserved

• The information contained in this presentation is provided for informational purposes only. This information is based on Lightbend’s current product plans and strategy, which are subject to change by Lightbend without notice. Lightbend shall not be responsible for any damages arising out of the use of, or otherwise related to, this presentation.

• References in this presentation to Lightbend products, programs, or services do not imply that they will be available in all countries in which Lightbend operates.

• Product release dates and/or capabilities referenced in this presentation may change at any time at Lightbend’s sole discretion based on market opportunities or other factors, and are not intended to be a commitment to future product or feature availability in any way.
First Things First… What is a Streaming Data Application?

It’s an application! Customer-facing, internal-facing, or IOT-facing

With all the requirements we associate with applications

- Scalable, resilient, available… and agile

Infuses “AI” or smarter real-time analytics to existing apps

- loan approval, device maintenance, next best offer, recommendation engine

Mixes domain logic with streaming analytics and model serving
Lightbend Fast Data Platform 2.0

Streaming Engines
- akka streams
- Spark
- Flink
- Kafka (Kafka Streams)

Microservices
- akka
- play
- lagom

Machine Learning
- Spark
- ML
- ...

Data Backplane
- Kafka

Storage Options
- HDFS
- SQL, NoSQL
- Cloud Storage (S3 etc)
- Elasticsearch

Container Orchestration
- kubernetes

Intelligent Management & Monitoring and Security
- Fast Data Platform Manager

- Lightbend Enterprise Suite
Age’s Dream

Imagine a world in which you can quickly create an HTTP endpoint to ingest streaming data and keep it safe (durable, available, and scalable). And then, just as easily, post-process that data using Spark and then expose a projection of the results as a microservice by providing a simple flattening function.

You have now written an app that most enterprises spend months on; You’ve written it in one day and made it easily deployable and scalable out of the box.
An app developer, data engineer, and data scientist walk into a bar...

**App Dev**
- Domain Logic, Agile, Scalable, Robust
- New challenge: infusing app with intelligence

**Data Scientist**
- Big Data, Assisting business in decision making
- New challenge: exporting models to assist applications in real-time decision making

**Data Engineer**
- Data Preparation, Cleansing, Streaming, Data Pipelines
- New challenge: real-time processing, meeting same expectations (scalability, robustness) as App Dev
Did I forget someone?
Why Fast Data Pipelines? (Part 1)

Developing Fast Data applications involves a lot of complexity

- Managing and composing applications
- Evolving applications
- Serving Machine Learning models
- Exposing streaming pipelines as (micro)service
Sample use case - hand-coded and operated

- Lots of Moving Parts
- Manually managed Kafka topics, serialization
- Schema?
- Hand-crafted and deployed Akka-based REST microservice
- Manual deployment
Why, Fast Data Pipelines! (Part 2)

Fast Data Pipelines increases productivity, decreases time to production, and improves the devops experience with tooling that takes care of the boilerplate, resiliency, serialization & schemas, deployment, monitoring, and other such drudgery.
**Concepts**

*Streamlets*: implement the actual stream processing logic.

Streamlets have different shapes, including *ingress, egress, fan-in, fan-out, processors, and views.*

Each Streamlet is strongly typed with *inlet* and *outlet* schemas.

A *Blueprint* comprises a set of composed streamlets. An Application Blueprint can then be deployed.
Simplifying Development

Developer Tasks

- Developer publishes processing code without boilerplate (Streamlets)
- Developer wires streamlets together in a Blueprint
- Developer issues command to deploy a Blueprint
Without Pipelines...
import pipelines.spark.SparkProcessor
import pipelines.spark.sql.SQLImplicits._

class MovingAverageSparklet extends SparkProcessor[Data, Agg] {
  override def process(session: SparkSession): Dataset[Data] ⇒ Dataset[Agg] = {
    inDataset ⇒
    val query = inDataset
      .withColumn("ts", ".timestamp".cast(TimestampType))
      .withWatermark("ts", "2 minutes")
      .groupBy(window("ts", "2 minute", "1 minute"), "src", "gauge").agg(avg("value") as "avg")

    query.select("src", "gauge", "avg" as "value").as[Agg]  
  }
}
Simplifying deployment and operations

Pipelines orchestrates/manages the deployment
- Orchestrates deployment of distributed applications
- Provides centralized management (Lightbend Console)
- Automates scaling and SLA management (future)
UX - Workflow

1. Code, Compose, Build (local)
   - Source Code (Streamlets)
   - Blueprint

2. Push Images
   - Docker Image

3. Deploy
   - Kubernetes
   - Runtime
Integrating Stateful Microservices into Streaming Pipelines

We would like to:

▪ Automatically expose data as microservices
▪ Embed domain or business logic as part of a data pipeline
▪ Eliminate boundaries between microservices and stream processing
▪ Allow for a flexible deployment model
  • Support stateless & stateful services
  • Support stateless & stateful stream processing
Streaming Stateful Microservices: Stream Processing + Event Sourcing

- Deploy durable stateful microservice as part of a streaming pipeline.
  - E.g. Akka-Persistence Typed
- Event Stream Partitioning for easy scale out
- Streaming Stateful Microservice ingest exposed as microservice
- Resulting events exposed as egress
  - Server-Sent Events HTTP endpoint
  - Projections and Queries
Remember this use case?
Demo Takeaways

- Schema Driven
- Focus on Business Logic
- Compile-time Validation
- Cloud-Native
Lightbend Fast Data Pipelines

Orchestrate and Operate Multi-Stage, Multi-Component Fast Data Pipelines and Microservices

Features
1. Develop, compose, and operate streaming apps
2. Choose streaming engine that works for you
3. Automated provisioning and management
4. Expose streaming data pipelines as microservices
5. Support collaborative multi-team development

Benefits
1. Ingest, transform, analyze, serve data in real time
2. Focus on business logic - avoid boilerplate code
3. Transform legacy systems into fast data apps
For More Information

https://www.lightbend.com/fast-data-platform

Email: craig.blitz@lightbend.com
Backup
The Dream Expanded, or.... Your Nightmare

Remove Boilerplate
Integrate Stream Processing from Different Teams
Deploy and Manage
Operationalize Machine Learning as Part of Stream Processing
Integrate with Microservice-based Applications
Why Do You Keep Talking About Microservices?

Big Data Never Talks About Microservices

Other Stream Processing Technologies Don’t Talk About Microservices

- Stream Processing driven by real-time requirements
- Enriching customer experience
- IOT
- It’s all about Applications, not Business Intelligence!

Applications drive scale, performance, and availability requirements
But, Also, Remember What Kicked Off the Microservice Revolution?

Waterfall, monolith leading to long release cycles
Business frustrated by inability to roll out new features
Project risk because of massive rollouts
Scalability difficult
Fragile

All of this applies to the streaming pipelines as well. We’ve learned our lessons.
FDP Pipelines
From components to application pipelines

- Increased developer and operational productivity
- Develop and operate cross-component streaming pipelines & microservices
- CLI (pipectl) and GUI-based composition and operational tooling
- Processing in Akka Streams, Spark,...
- Durable, scalable data between processing stages
- Automated serialization between processing stages
- Standardized configuration and data formats/schemas between processing engines
- Integrated with monitoring functionality from Enterprise Suite 2
What is Fast Data Pipelines?

- Help users create, orchestrate, and manage fast data applications using a pipeline-first approach
- Allow choice of technology for each Pipeline processing stage (or streamlet)
- Streamlets are composed into Pipelines

*Pipelines will be the preferred mechanism for developing applications on the Fast Data Platform.*
Pipelines comprise an *ingress*, an *egress*, one or more processing *streamlets*, and possibly a read-only microservice serving applications.
Connections are persistent buffers, allowing for streamlets to be stopped, updated, and restarted.
Streamlets can be stopped/updated/restarted

The original Spark analysis has been replaced by a second version written in Flink.