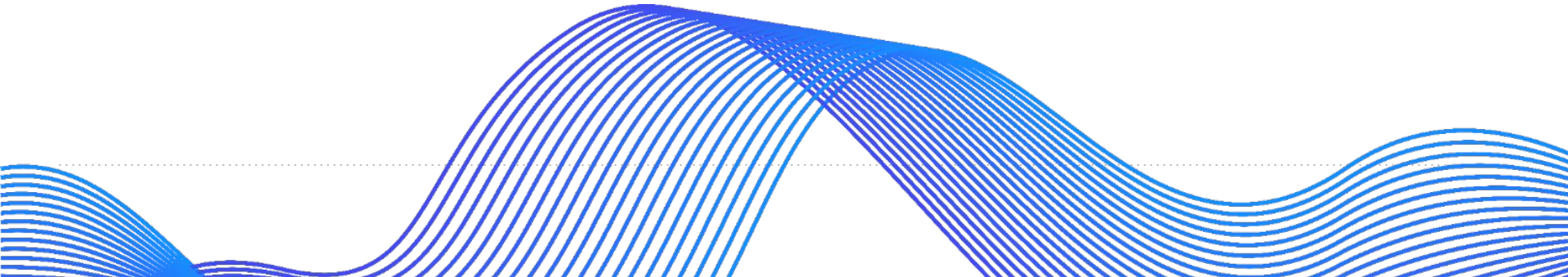
StreamNative

DEVNEXUS™

Designing Event-Driven Applications with Apache Flink, Apache NiFi, Apache Spark and Apache Pulsar

Tim Spann | Developer Advocate



Tim Spann, Developer Advocate at StreamNative

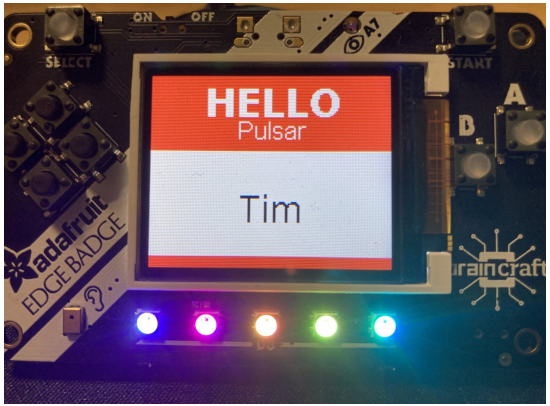
Tim Spann
Developer Advocate

StreamNative

- FLiP(N) Stack = Flink, Pulsar and NiFI Stack
- Streaming Systems & Data Architecture Expert
- Experience:
  - 15+ years of experience with streaming technologies including Pulsar, Flink, Spark, NiFi, Kafka, Big Data, Cloud, MXNet, IoT and more.
  - Today, he helps to grow the Pulsar community sharing rich technical knowledge and experience at both global conferences and through individual conversations.

CLOUDERA · Pivotal · BARNES & NOBLE · FLIP STACK · EY · HORTONWORKS · airis.DATA · Hewlett Packard Enterprise

WHAT?



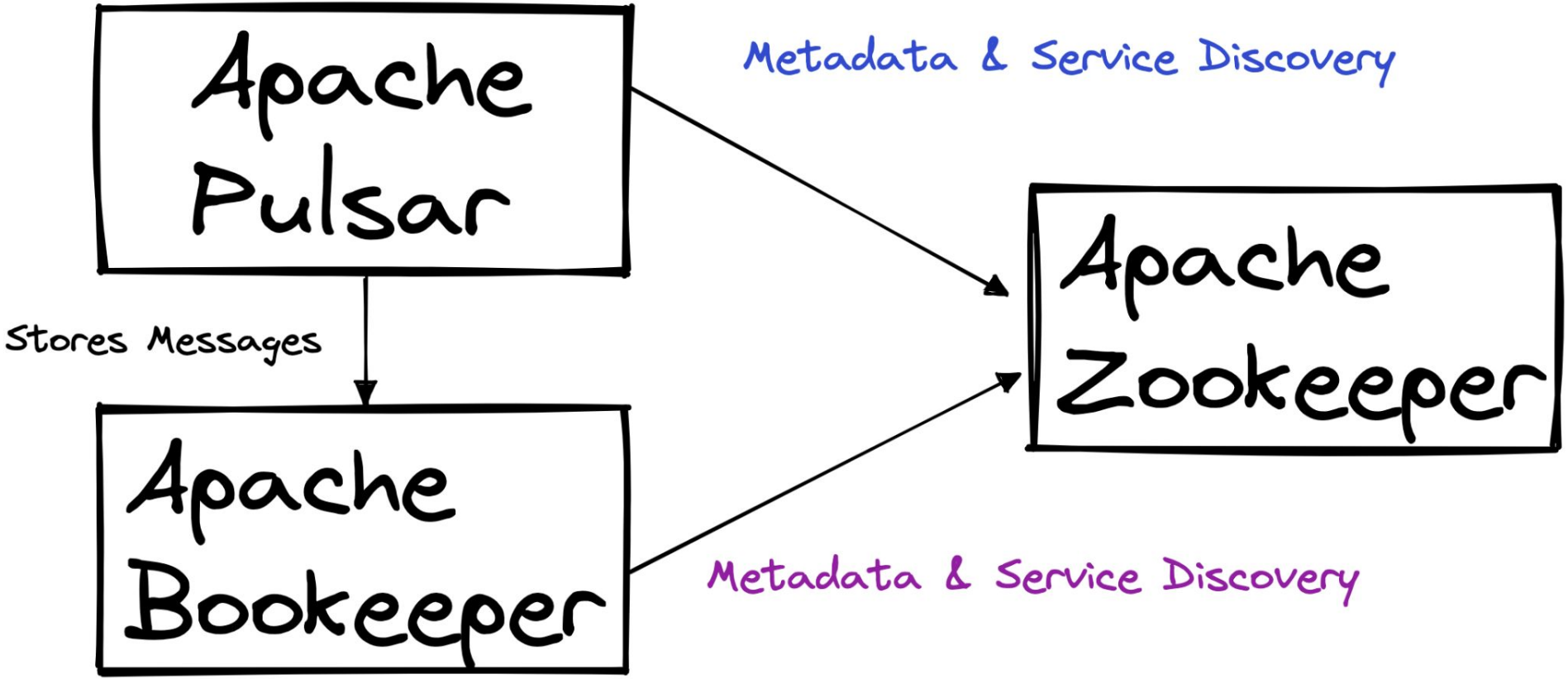
Design Event-Driven Applications

This talk is a quick overview of the HOW, WHAT and WHY of Apache Pulsar, Apache Spark, Apache Flink and Apache NiFi. I will show you how to design event-driven applications that scale the cloud native way.

Use Cases

- Unified Messaging Platform
- AdTech
- Fraud Detection
- Connected Car
- IoT Analytics
- Microservices Development

Apache Pulsar is a Cloud-Native Messaging and Event-Streaming Platform.


Apache Pulsar
Metadata & Service Discovery
Apache Zookeeper
Stores Messages
Apache Bookeeper
Metadata & Service Discovery

Pulsar: Unified Messaging + Data Streaming

Messaging

Ideal for work queues that do not require tasks to be performed in a particular order—for example, sending one email message to many recipients.

RabbitMQ and **Amazon SQS** are examples of popular queue-based message systems.

Pulsar: Unified Messaging + Data Streaming

Messaging

Ideal for work queues that do not require tasks to be performed in a particular order—for example, sending one email message to many recipients.

RabbitMQ and Amazon SQS are examples of popular queue-based message systems.

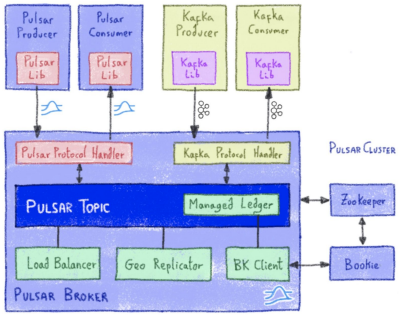
.. and Streaming

Works best in situations where the order of messages is important—for example, data ingestion.

Kafka and **Amazon Kinesis** are examples of messaging systems that use streaming semantics for consuming messages.

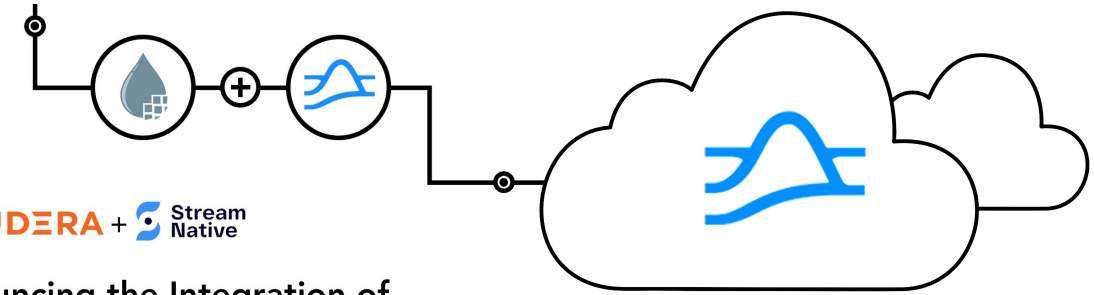
Connectivity

hub.streamnative.io



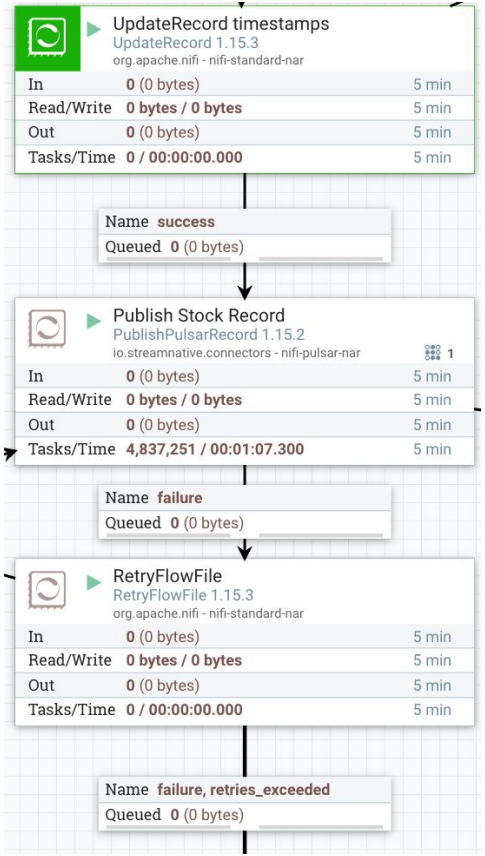
- **Functions** - Lightweight Stream Processing (**Java**, Python, Go)
- **Connectors** - Sources & Sinks (InfluxDB, Kafka, S3, Kinesis, Lambda, …)
- **Protocol Handlers** - AoP (AMQP), KoP (Kafka), MoP (MQTT), RoP (RocketMQ)
- **Processing Engines** - Apache Flink, Apache Spark, Presto/Trino via Pulsar SQL
- **Data Offloaders** - Tiered Storage - (S3)

Apache NiFi Pulsar Connector



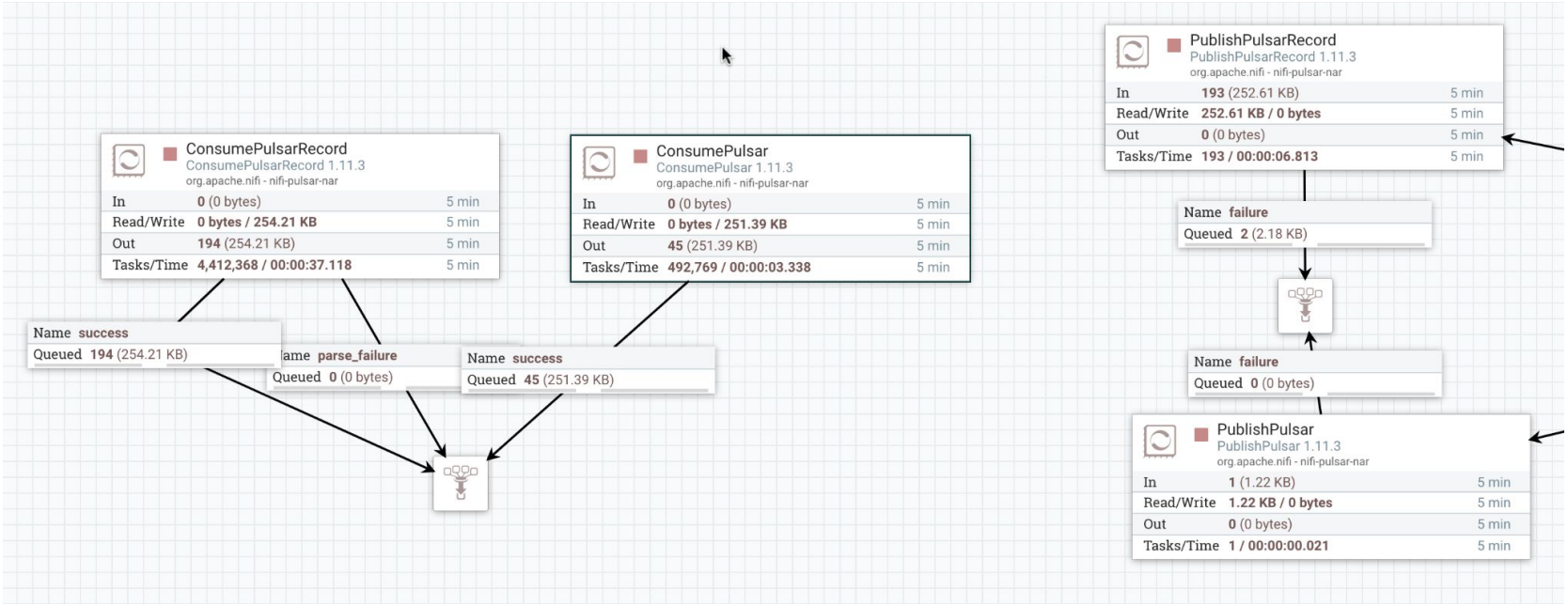
CLOUDERA + Stream Native

Announcing the Integration of
Apache NiFi and Apache Pulsar



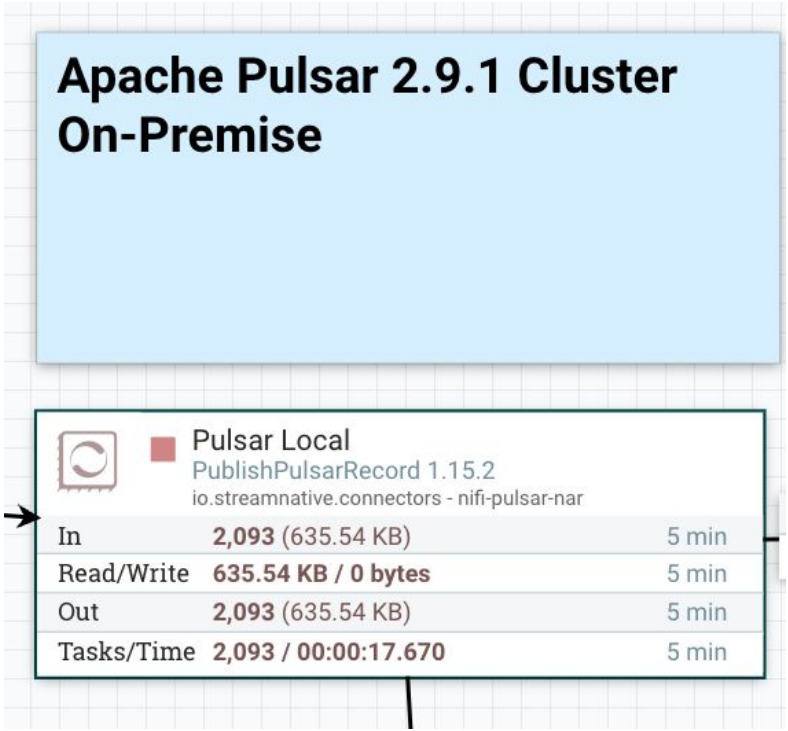
https://streamnative.io/apache-nifi-connector/

Apache NiFi Pulsar Connector

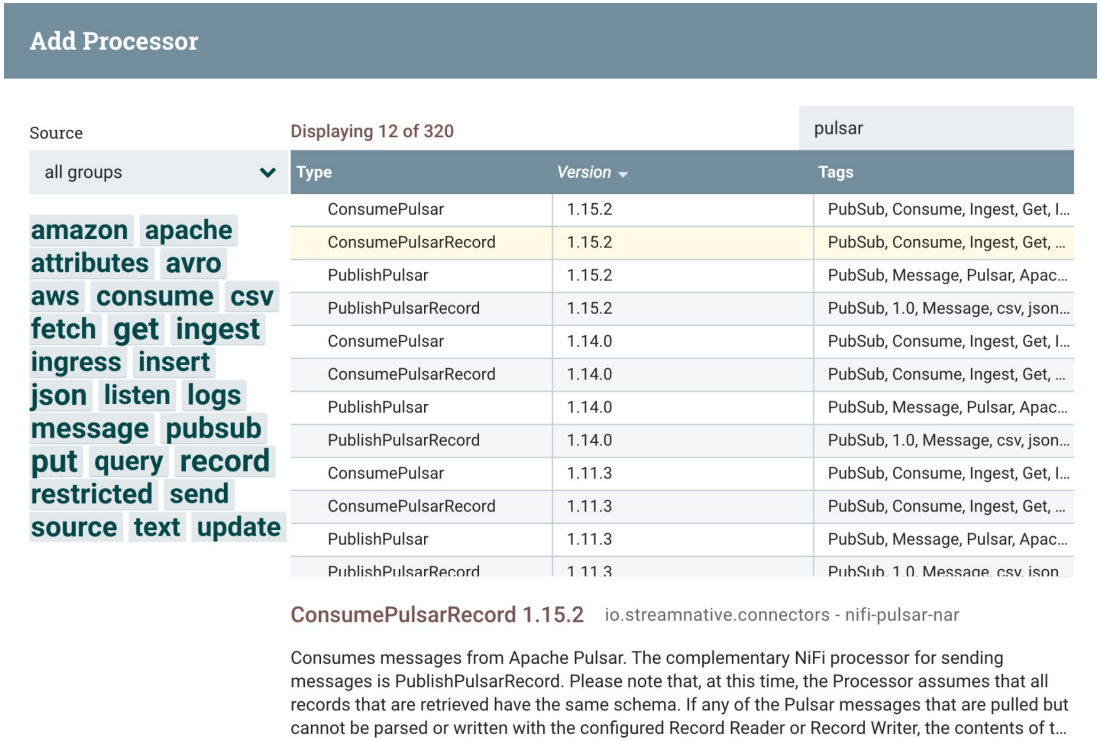


https://github.com/streamnative/pulsar-nifi-bundle

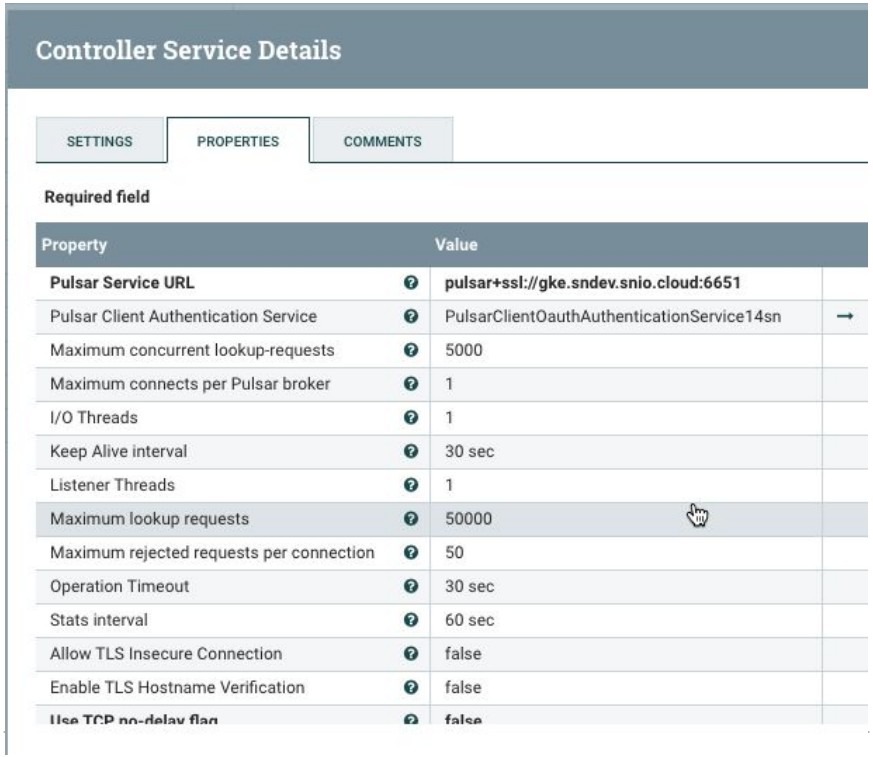
Apache NiFi Pulsar Connector



Apache NiFi Pulsar Connectors

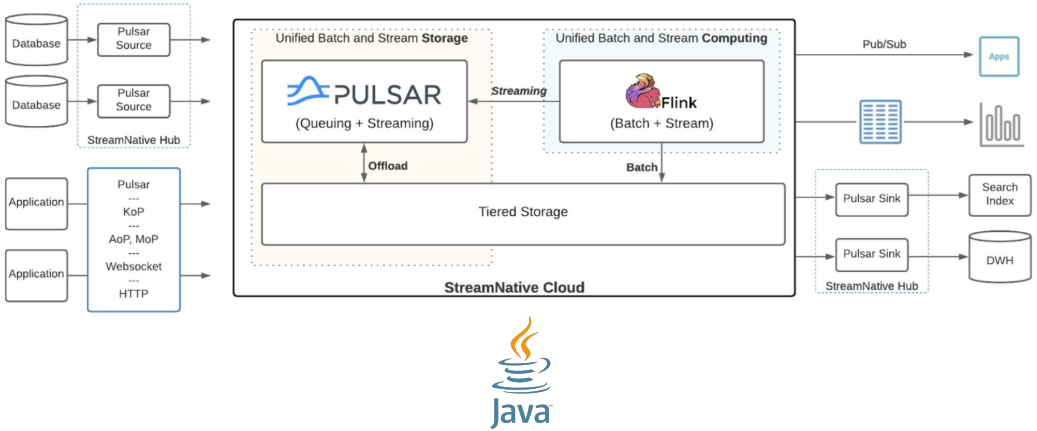


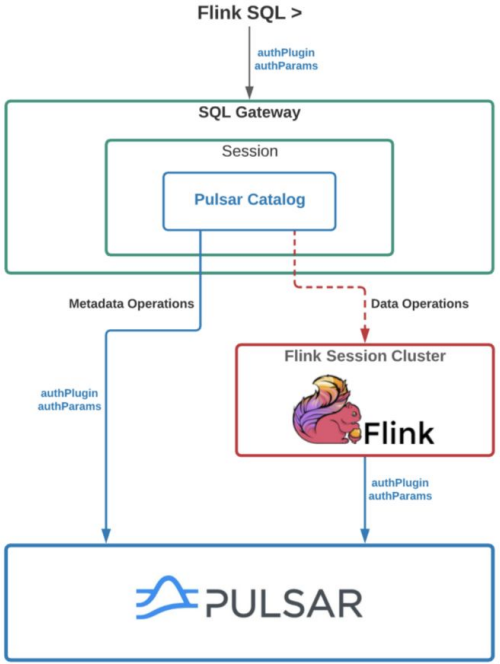
Apache NiFi Pulsar Connectors



Apache NiFi Pulsar Connector

Flink + Pulsar





https://nightlies.apache.org/flink/flink-docs-master/docs/connectors/datastream/pulsar/
https://streamnative.io/en/blog/release/2021-04-20-flink-sql-on-streamnative-cloud

Flink SQL


BETA
Welcome! Enter 'HELP;' to list all available commands. 'QUIT;' to exit.
Flink SQL> CREATE CATALOG pulsar WITH (
>   'type' = 'pulsar',
>   'service-url' = 'pulsar://pulsar1:6650',
>   'admin-url' = 'http://pulsar1:8080',
>   'format' = 'json'
>);
[INFO] Execute statement succeed.
Flink SQL>


SQL Query Result (Table)
Refresh: 1 s
Page: Last of 1
uuid
ipaddress
cputempf
snr_20220323195238
192.168.1.229
99
snr_20220323195243
192.168.1.229
100
Q Quit
R Refresh
+ Inc Refresh
- Dec Refresh
G Goto Page
L Last Page
N Next Page
P Prev Page
O Open Row

Spark + Pulsar


```
val dfPulsar = spark.readStream.format("pulsar")
               .option("service.url", "pulsar://pulsar1:6650")
               .option("admin.url", "http://pulsar1:8080")
               .option("topic", "persistent://public/default/airquality").load()


val pQuery = dfPulsar.selectExpr("*")
.writeStream.format("console")
.option("truncate", false).start()
```

https://pulsar.apache.org/docs/en/adaptors-spark/

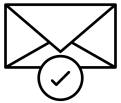

```
      ____              __
     / __/__  ___ _____/ /__
    _\ \/ _ \/ _ `/ __/  '_/
   /___/ .__/\_,_/_/ /_/\_\   version 3.2.0
      /_/

Using Scala version 2.12.15
(OpenJDK 64-Bit Server VM, Java 11.0.11)
```

WHY?

Why Apache Pulsar?

Unified Messaging Platform



Guaranteed Message Delivery

Resiliency

Infinite Scalability

Pulsar across the organization

Ideal for app and data tiers

Less sprawl and better utilization

Cloud-native scalability

Cost effective long-term storage

Build globally without the complexity

Pulsar Benefits

Unified Messaging Model

Multi-tenancy

Scalability

Geo-replication

Tiered storage

Pulsar Benefits

Unified Messaging Model

Simplify your data infrastructure and enable new use cases with queuing and streaming capabilities in one platform.

Multi-tenancy

Enable multiple user groups to share the same cluster, either via access control, or in entirely different namespaces.

Scalability

Decoupled data computing and storage enable horizontal scaling to handle data scale and management complexity.

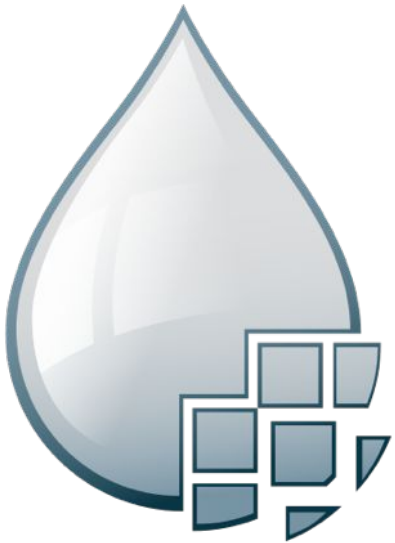
Geo-replication

Support for multi-datacenter replication with both asynchronous and synchronous replication for built-in disaster recovery.

Tiered storage

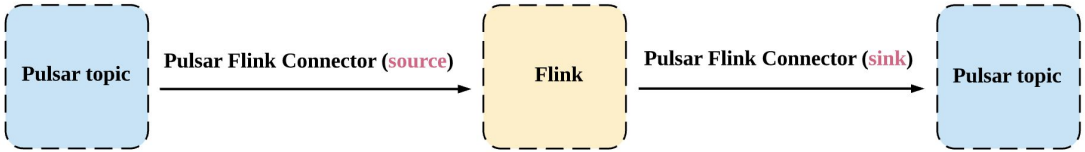
Enable historical data to be offloaded to cloud-native storage and store event streams for indefinite periods of time.

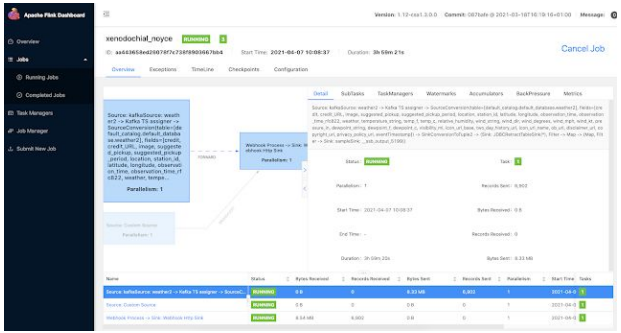
Why Apache NiFi?



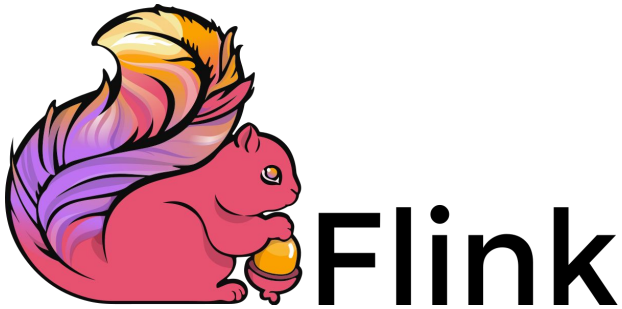
- Guaranteed delivery
- Data buffering
  - Backpressure
  - Pressure release
- Prioritized queuing
- Flow specific QoS
  - Latency vs. throughput
  - Loss tolerance
- Data provenance
- Supports push and pull models
- Hundreds of processors
- Visual command and control
- Over a 300 components
- Flow templates
- Pluggable/multi-role security
- Designed for extension
- Clustering
- Version Control

Why Apache Flink?





- Unified computing engine
- Batch processing is a special case of stream processing
- Stateful processing
- Massive Scalability
- Flink SQL for queries, inserts against Pulsar Topics
- Streaming Analytics
- Continuous SQL
- Continuous ETL
- Complex Event Processing
- Standard SQL Powered by Apache Calcite



Why Apache Spark?

- Java, Scala, Python Support
- Strong ETL/ELT
- Diverse ML support
- Scalable Distributed compute
- Apache Zeppelin and Jupyter Notebooks
- Fast connector for Apache Pulsar

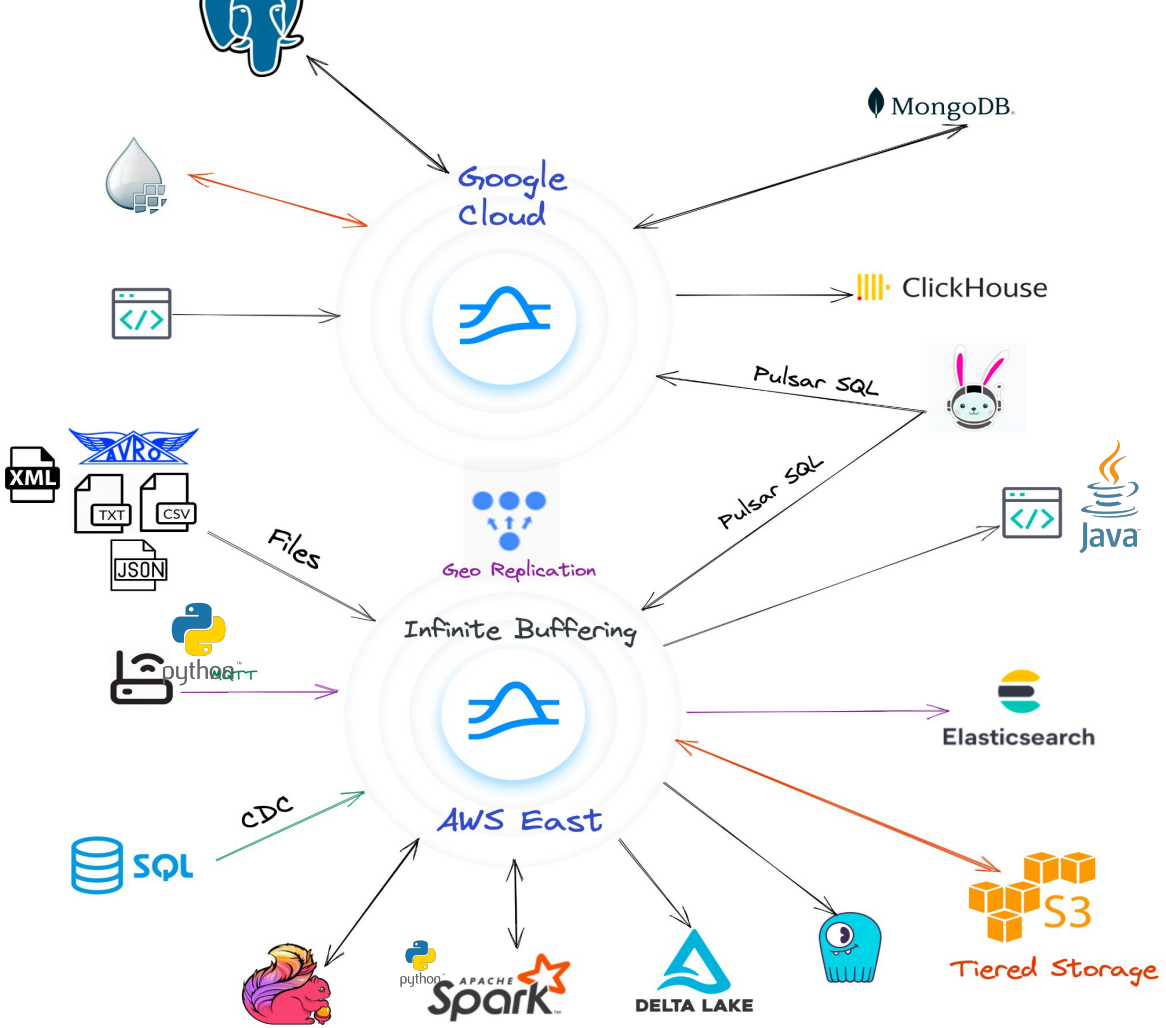
HOW!

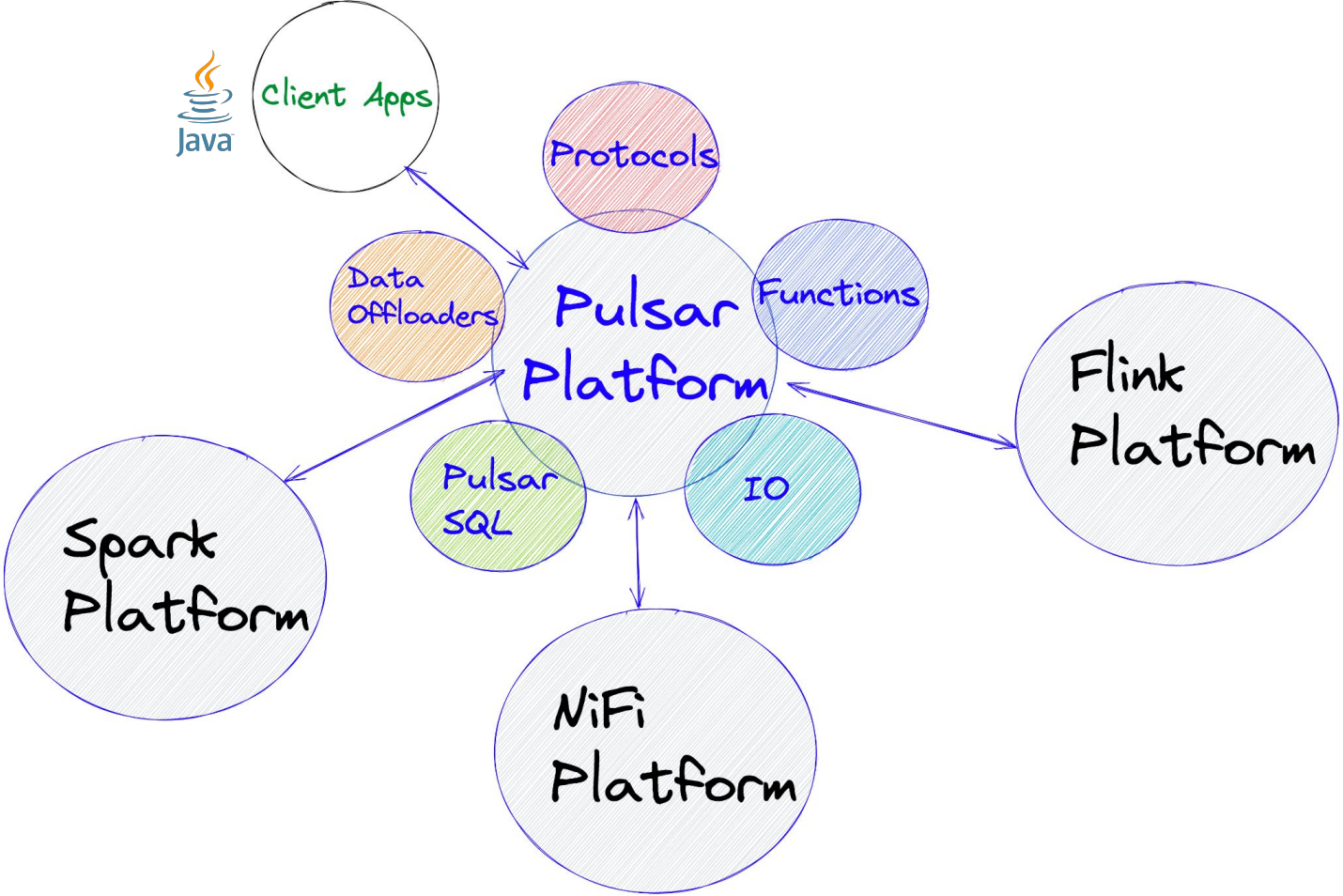
FLiP(N)S Stack

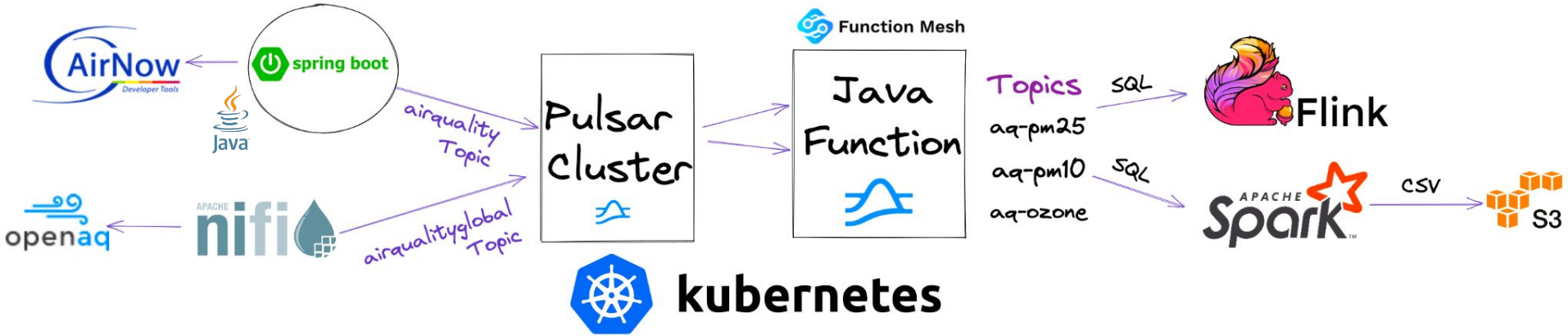
- Apache **Fl**ink
- Apache **P**ulsar
- Apache **N**iFi
- Apache **S**park

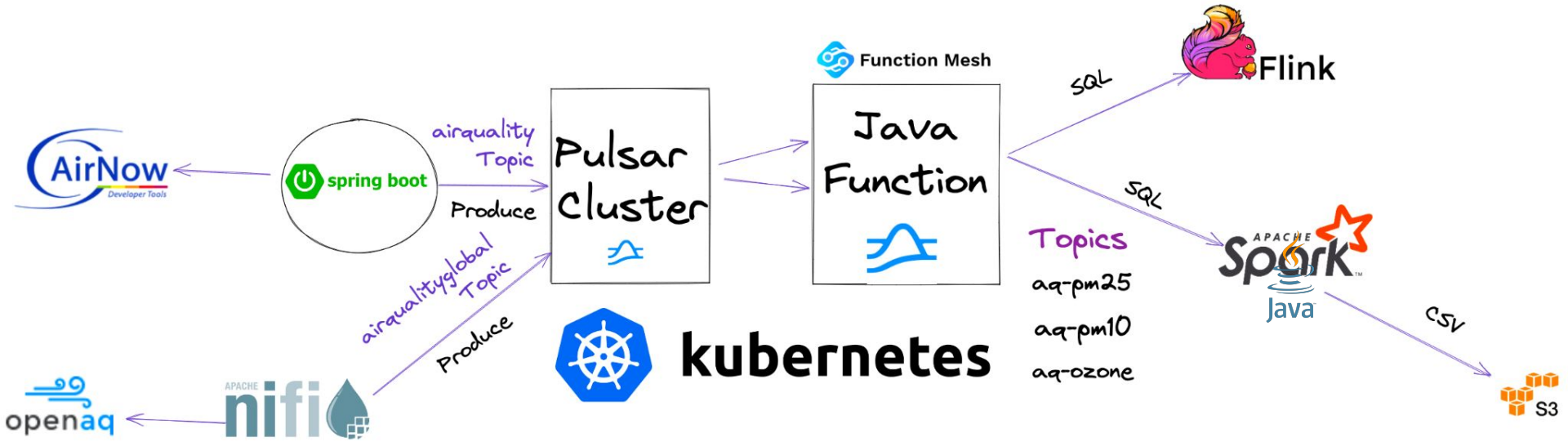
- Pulsar Functions
- Python, **Java**, Golang

- Buffer
- Batch
- Route
- Filter
- Aggregate
- Enrich
- Replicate
- Dedupe
- Decouple
- Distribute

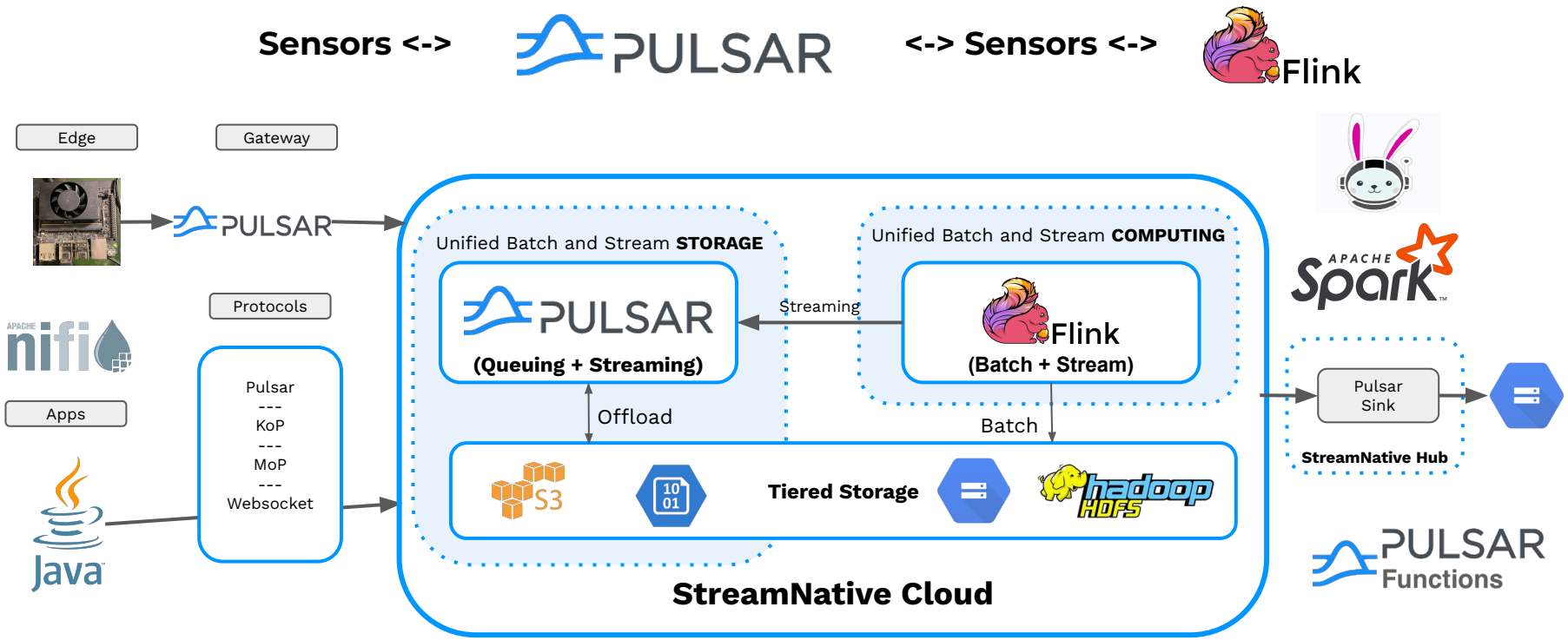



Java
Client Apps
Protocols
Data Offloaders
Pulsar Platform
Functions
Flink Platform
Spark Platform
Pulsar SQL
IO
NiFi Platform


AirNow
Developer Tools
spring boot
Java
airquality
Topic
Pulsar
Cluster
Function Mesh
Java
Function
Topics
aq-pm25
aq-pm10
aq-ozone
SQL
SQL
Flink
APACHE
Spark
CSV
S3
openaq
APACHE
nifi
airqualityglobal
Topic
kubernetes

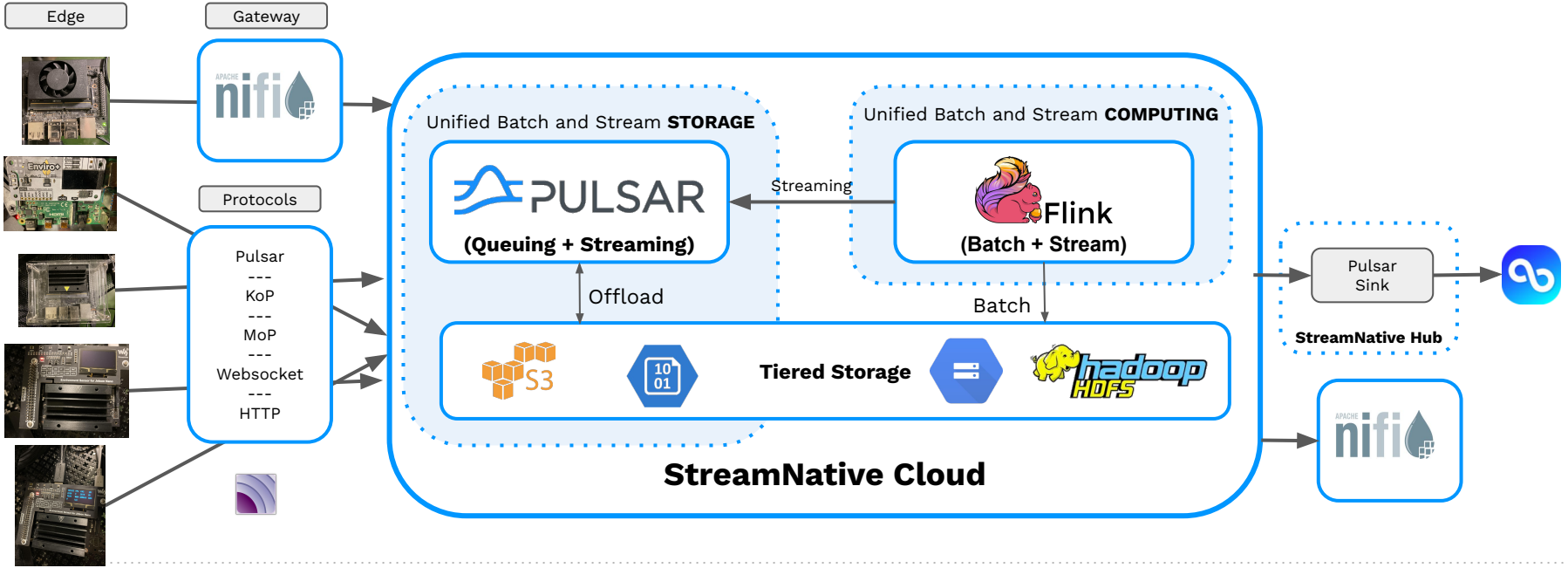

AirNow
Developer Tools
spring boot
airquality
Topic
Produce
Pulsar
Cluster
Function Mesh
Java
Function
SQL
Flink
SQL
Topics
aq-pm25
aq-pm10
aq-ozone
APACHE
Spark
Java
CSV
S3
airqualityglobal
Topic
Produce
kubernetes
APACHE
nifi
openaq

Streaming Java Apps



End-to-End Streaming Edge App

Apache Flink - Apache Pulsar - Apache NiFi <-> Devices

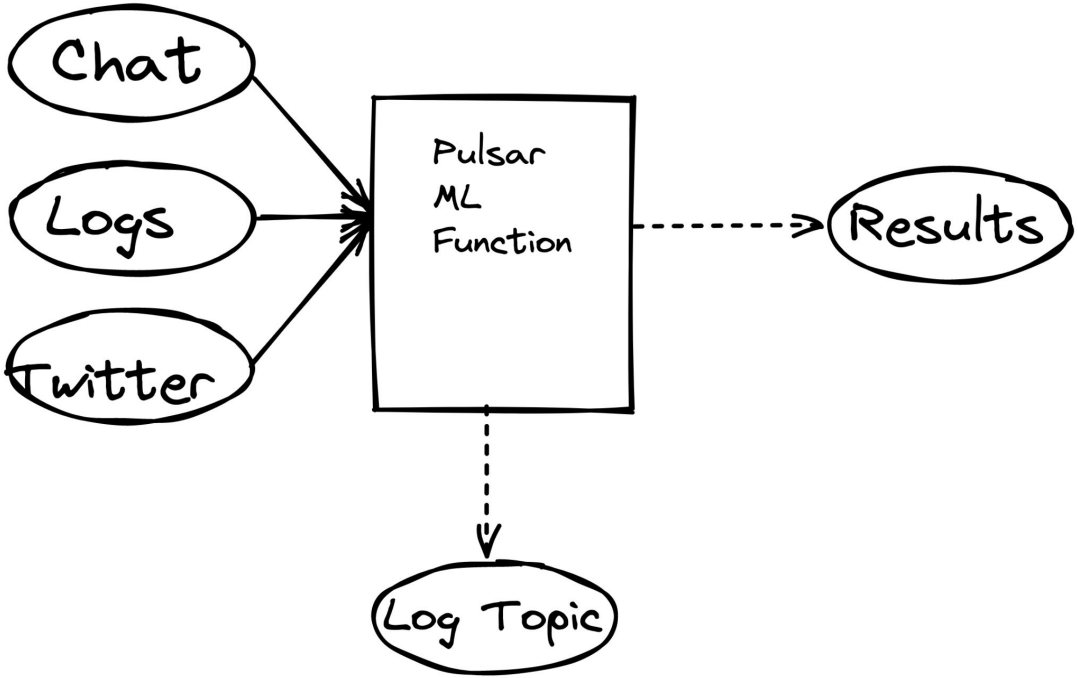


StreamNative

streamnative.io

ML Java Coding

(DJL)



CODE!

Source Code

https://github.com/tspannhw/airquality

https://github.com/tspannhw/FLiPN-AirQuality-REST

https://github.com/tspannhw/pulsar-airquality-function

https://github.com/tspannhw/FLiP-Pi-BreakoutGarden

https://github.com/tspannhw/FLiPN-DEVNEXUS-2022

Pulsar Function Java

Your Code Here

```
import java.util.function.Function;

public class MyFunction implements Function<String, String> {
    public String apply(String input) {
      return doBusinessLogic(input);
    }
}
```

The incoming messages are passed into the function one-by-one

The returned value is automatically published to the output topic

Pulsar Function

SDK

Your Code Here

```
import org.apache.pulsar.client.impl.schema.JSONSchema;
import org.apache.pulsar.functions.api.*;

public class AirQualityFunction implements Function<byte[], Void> {
   @Override
    public Void process(byte[] input, Context context) {
        context.getLogger().debug("File:” + new String(input));
        context.newOutputMessage(“topicname”,
                        JSONSchema.of(Observation.class))
                         .key(UUID.randomUUID().toString())
                         .property(“prop1”, “value1”)
                         .value(observation)
                         .send();
    }
}
```

Setting Subscription Type Java

```
Consumer<byte[]> consumer = pulsarClient.newConsumer()
                .topic(topic)
                .subscriptionName(“subscriptionName")
                .subscriptionType(SubscriptionType.Shared)
                .subscribe();
```

Subscribing to a Topic and setting Subscription Name Java

```
Consumer<byte[]> consumer = pulsarClient.newConsumer()
                .topic(topic)
                .subscriptionName(“subscriptionName")
                .subscribe();
```

Producing Object Events From Java

```
ProducerBuilder<Observation> producerBuilder =
pulsarClient.newProducer(JSONSchema.of(Observation.class))
              .topic(topicName)
              .producerName(producerName).sendTimeout(60,
                                                 TimeUnit.SECONDS);
Producer<Observation> producer = producerBuilder.create();

msgID = producer.newMessage()
        .key(someUniqueKey)
        .value(observation)
        .send();
```

Starting a Function - Standalone

To start a Pulsar Function in a standalone environment:

```
pulsar-admin functions localrun \
  --jar myjar.jar \
  --classname path.to.PulsarFunction \
  --inputs inputtopic \
  --output outputtopic \
  --name functionname
```

Note: The `--jar` path must be fully qualified.

Starting a Function - Distributed Cluster

Once compiled into a JAR, start a Pulsar Function in a distributed cluster:

```
pulsar-admin functions create \
  --jar myjar.jar \
  --classname path.to.PulsarFunction \
  --inputs inputtopic \
  --output outputtopic \
  --name functionname
```

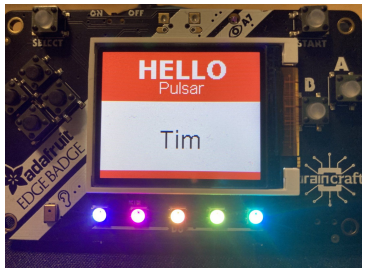
Simple Function (Java Native)

```
import java.util.function.Function;

public class SimpleFunctionJava implements Function<String, String> {
    @Override
    public String apply(String input) {
        return input.toUpperCase();
    }
}
```

WHO?

SCAN ME



FLiP Stack Weekly

This week in Apache Flink, Apache Pulsar, Apache NiFi, Apache Spark and open source friends.

https://bit.ly/32dAJft

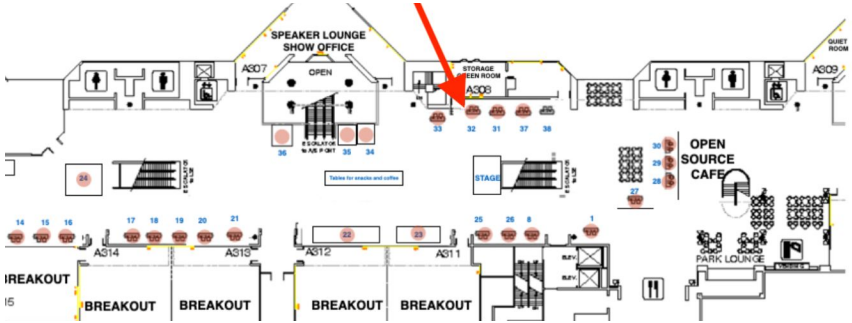
StreamNative Academy

LEARN MORE ABOUT APACHE PULSAR WITH:

- ➔ Pulsar expert instructor-led courses
- ➔ On-demand learning with labs
- ➔ 300+ engineers, admins and architects trained!

Academy.StreamNative.io

Join me
Booth 32



@PaasDev

https://www.linkedin.com/in/timothyspann

https://github.com/tspannhw

Stream Native

Founded by the original developers of Apache Pulsar.

Passionate and dedicated team.

StreamNative helps teams to **capture**, **manage**, and **leverage data** using Pulsar's unified messaging and streaming platform.

streamnative.io

The **Pulsar-as-a-service offering** built by the original creators of Pulsar, with f**lexible support for running in your cloud account or ours.**

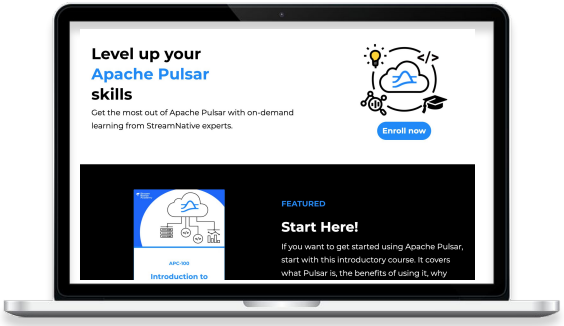
Teams leverage StreamNative Cloud to provide a **worry-free Pulsar cluster** backed up by the **reliability of the Pulsar experts**.

StreamNative Cloud

Production grade clusters in minutes via our UI or API

Deploy in GCP and AWS. Azure coming soon!

Pay-as-you-go pricing ideal to get started, with cost effective clusters at scale



APACHE PULSAR TRAINING

- ➔ Instructor-led courses
- ➔ On-demand learning with labs
- ➔ 300+ engineers, admins and architects trained!

StreamNative Academy

Academy.StreamNative.io

Thanks for stopping by!

Scan the QR Code to learn more about StreamNative and our exclusive booth offerings.

Questions?