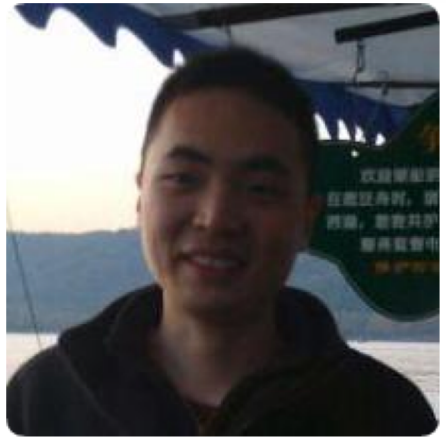
Stream Processing with Apache RocketMQ and Apache Flink

王鑫 · The Apache Software Foundation

Nov.4, 2018, Shanghai, Apache Flink China Meetup

Xin Wang

- Apache RocketMQ Committer
- Apache Storm Committer & PMC member
- Six years distributed system experience
- Love open source & community
- Focus on distributed technologies, especially stream processing
- https://github.com/vesense



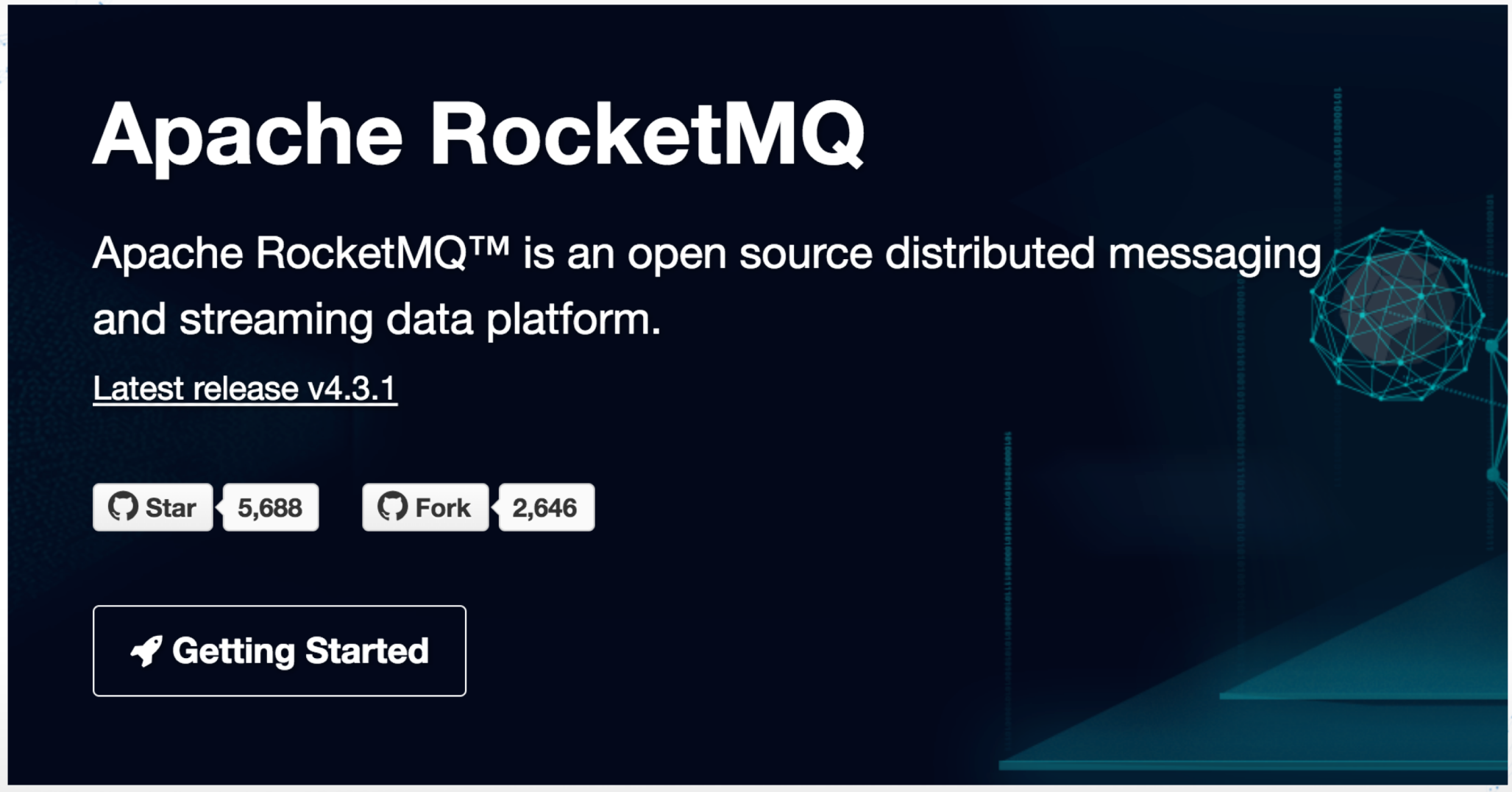
Flink China

CONTENT

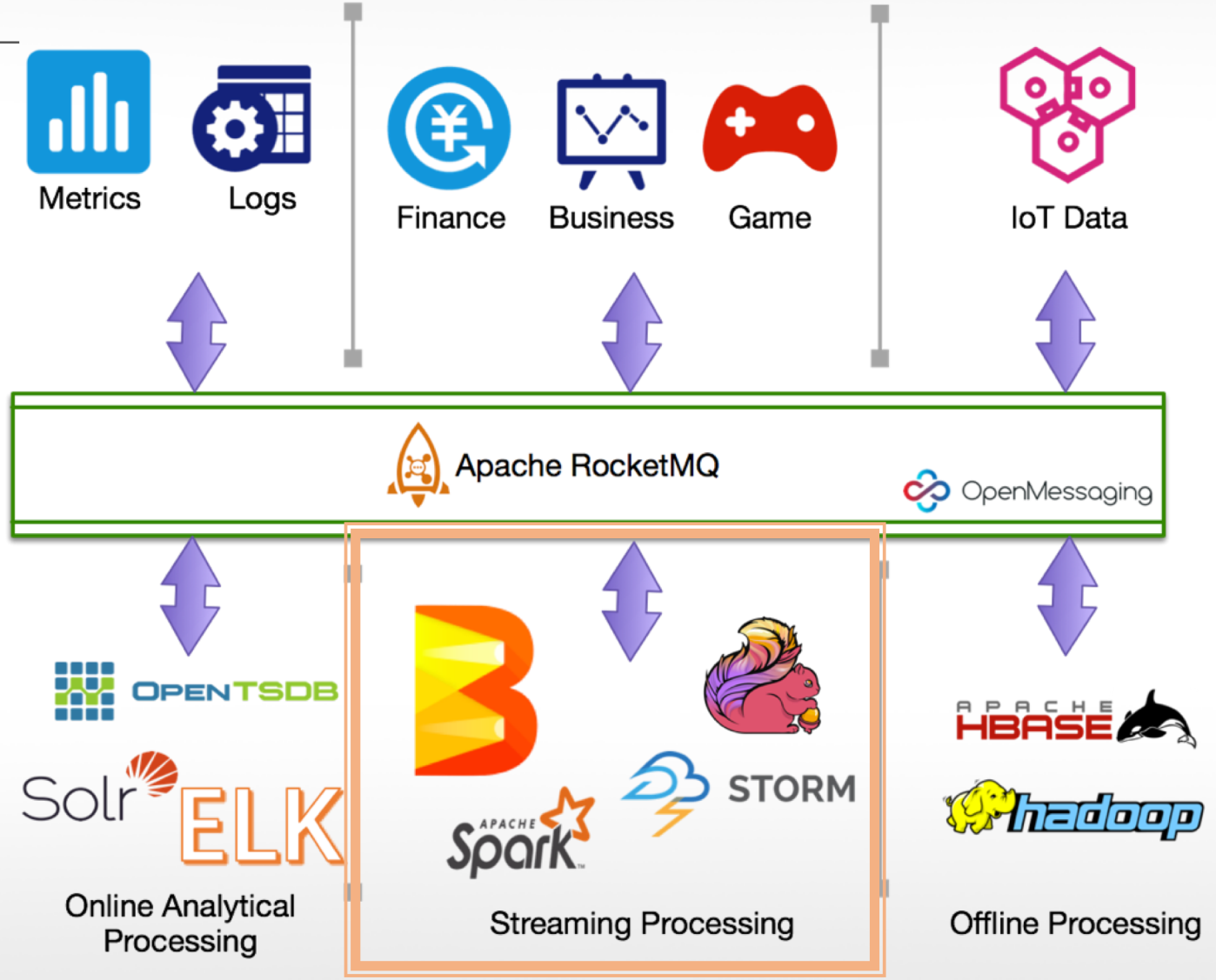
01

RocketMQ streaming ecosystem

Apache RocketMQ – A distributed streaming platform. Not only messaging.



Apache RocketMQ – A distributed streaming platform. Not only messaging.

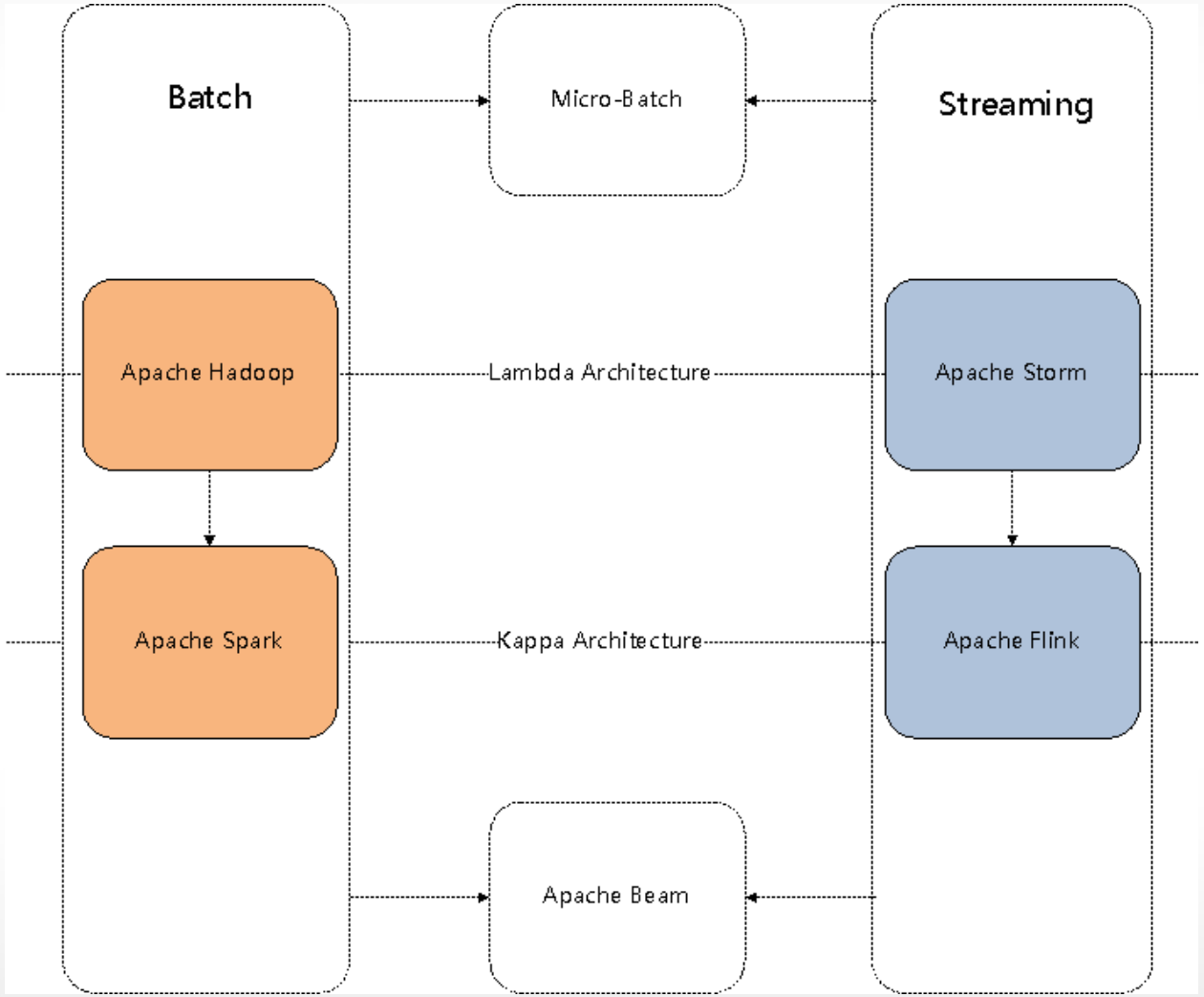


Flink China

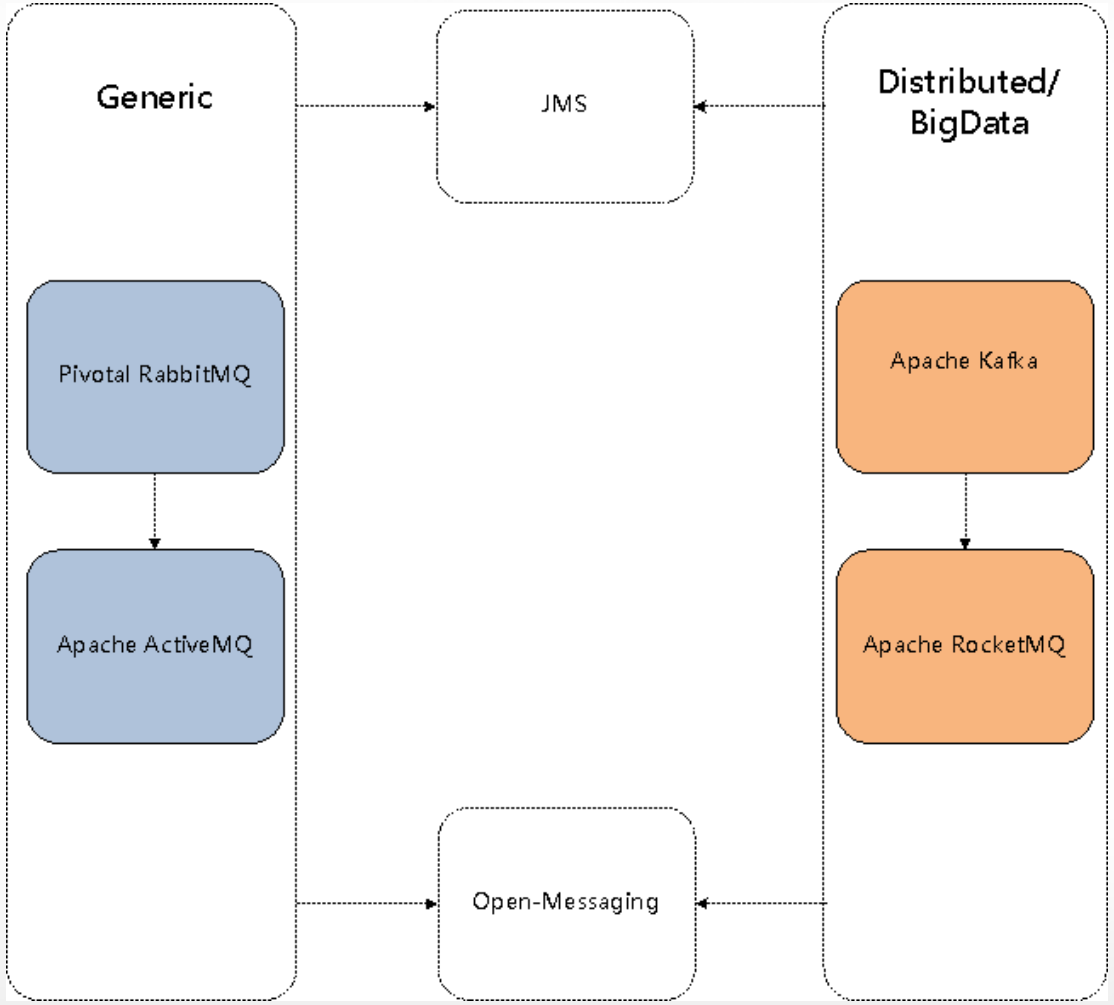
Apache RocketMQ streaming ecosystem projects

- RocketMQ-Flink: https://github.com/apache/rocketmq-externals/tree/master/rocketmq-flink
- RocketMQ-Spark: https://github.com/apache/rocketmq-externals/tree/master/rocketmq-spark
- RocketMQ-Storm: https://github.com/apache/storm/tree/master/external/storm-rocketmq
- RocketMQ-Avro: https://github.com/apache/rocketmq-externals/tree/master/rocketmq-serializer
- RocketMQ-Beam: Work in progress
- OpenMessaging-SQL: Work in progress
- ……

Apache RocketMQ streaming ecosystem



Apache RocketMQ streaming ecosystem



How to process streaming data with Apache RocketMQ?

Flink China

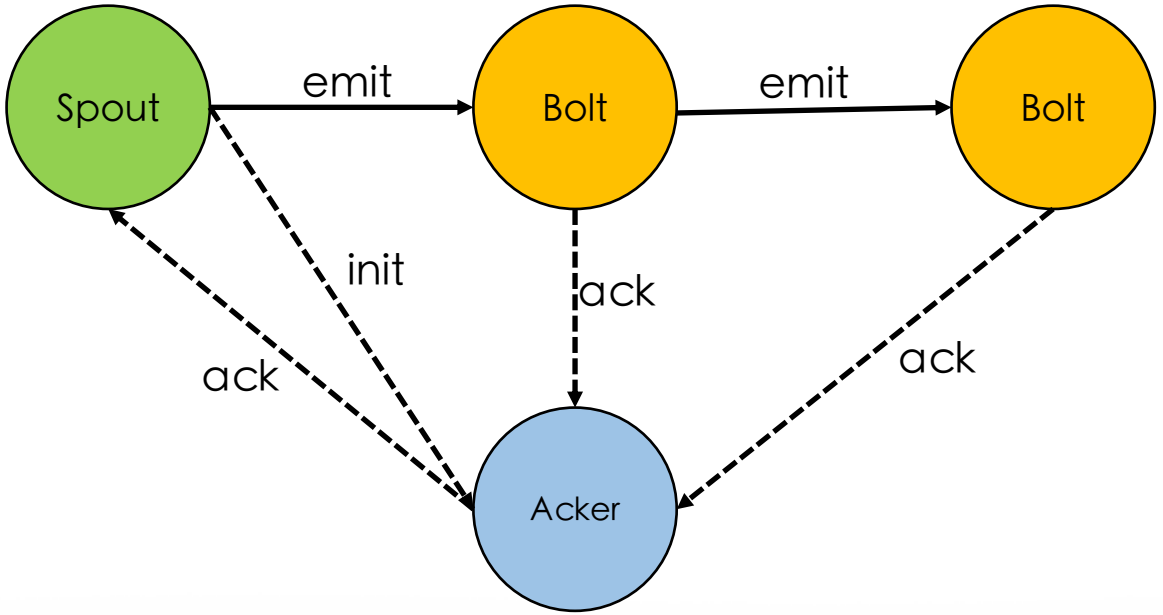
How to process streaming data with Apache RocketMQ?

- Option A: Using a streaming engine, Apache Storm, Flink, Spark-Streaming…
- Option B: Using RocketMQ-SQL(A lightweight streaming library)
- Option C: Writing the logic by yourself(PULL/PUSH API)

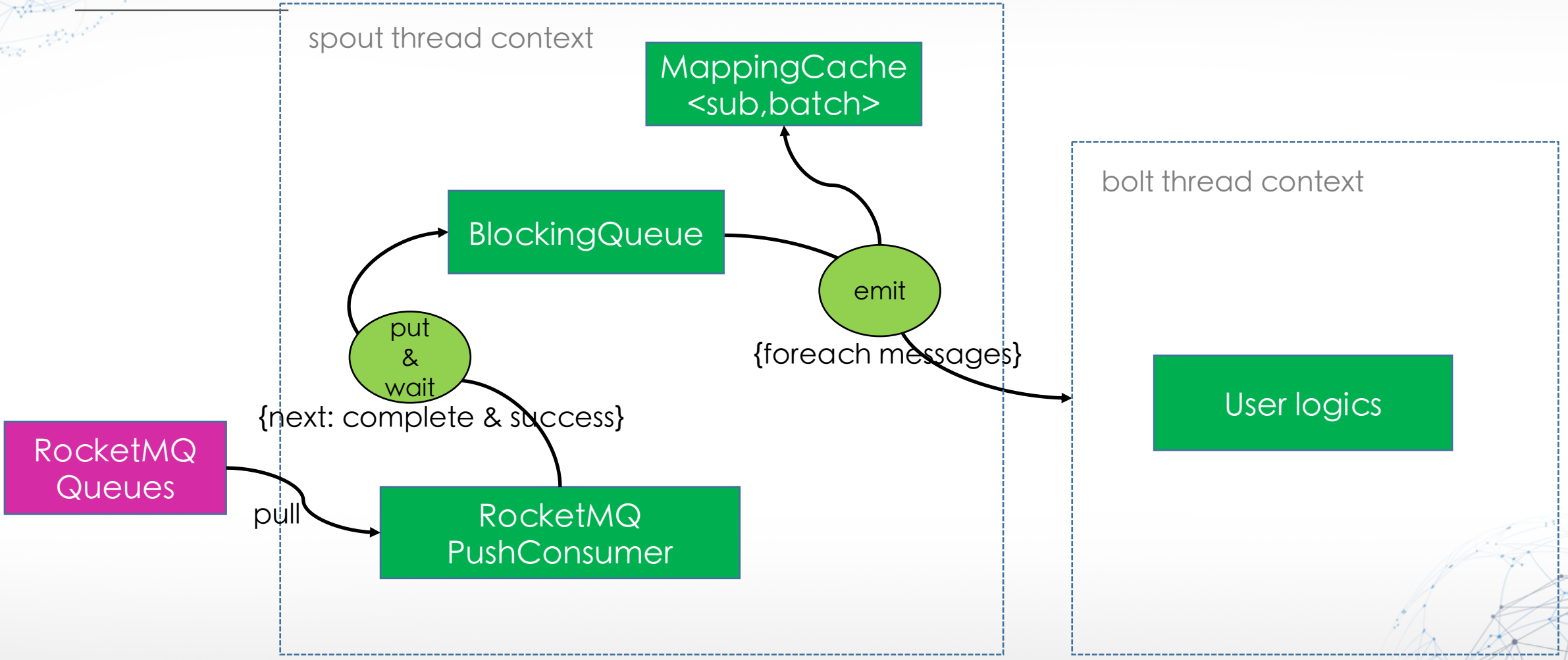
02

Practices of integrating RocketMQ with Flink

Apache Storm data fault-tolerant: Record Acknowledgement



Integrating RocketMQ with Apache Storm

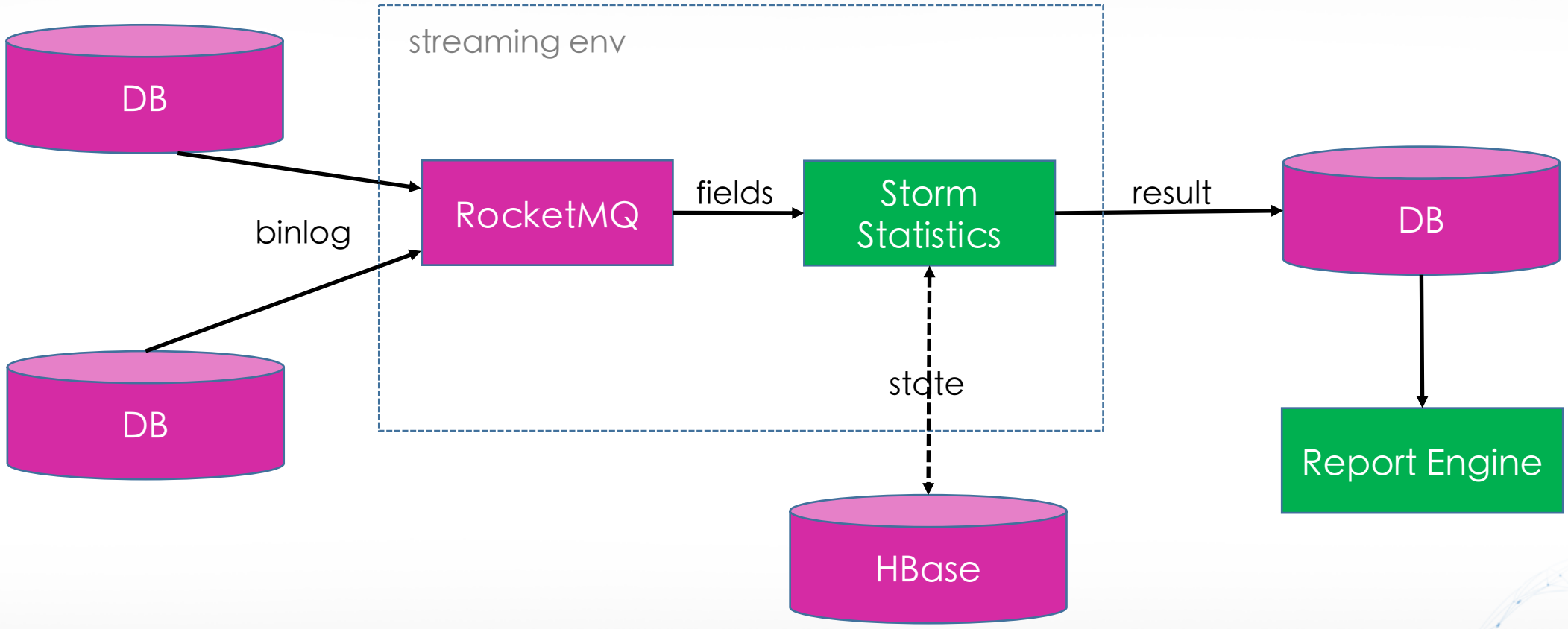


Flink China

RocketMQ-Storm Code Example

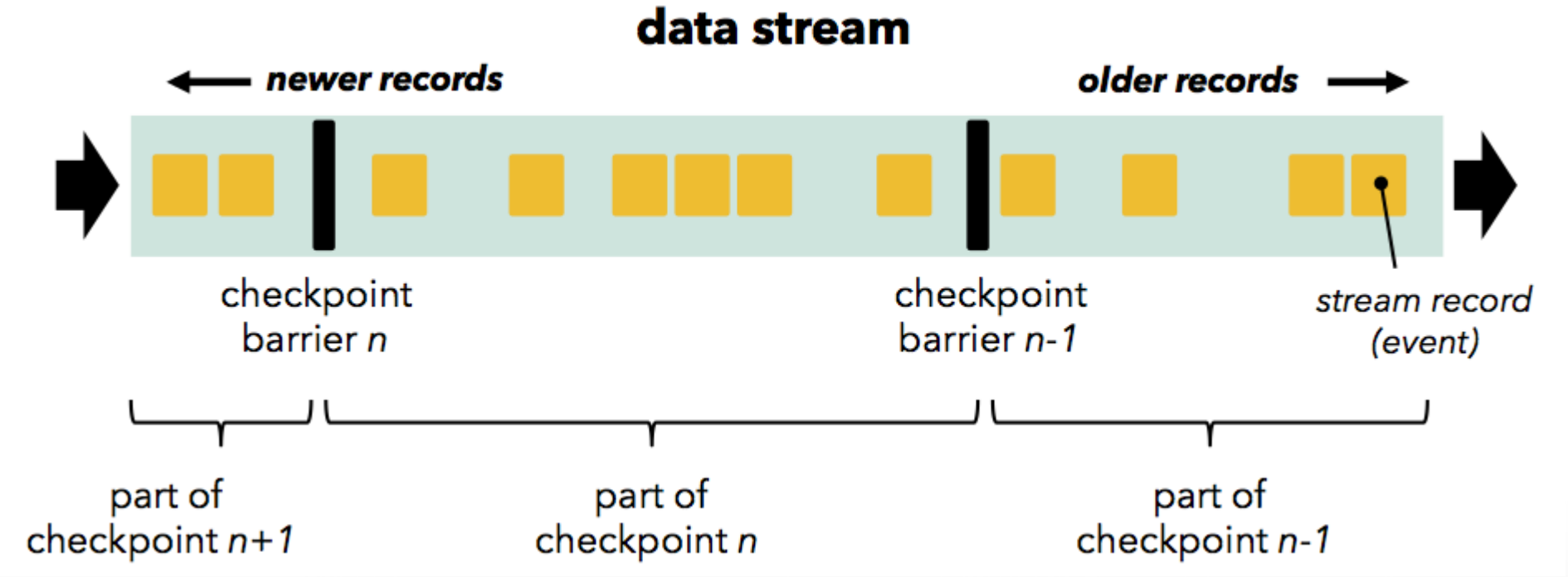
```
Properties properties = new Properties();
properties.setProperty(SpoutConfig.NAME_SERVER_ADDR, nameserverAddr);
properties.setProperty(SpoutConfig.CONSUMER_GROUP, group);
properties.setProperty(SpoutConfig.CONSUMER_TOPIC, topic);

RocketMqSpout spout = new RocketMqSpout(properties);

TupleToMessageMapper mapper = new FieldNameBasedTupleToMessageMapper("word", "count");
TopicSelector selector = new DefaultTopicSelector(topic);

RocketMqBolt insertBolt = new RocketMqBolt()
                .withMapper(mapper)
                .withSelector(selector)
                .withProperties(properties);
```

RocketMQ-Storm Case: BI



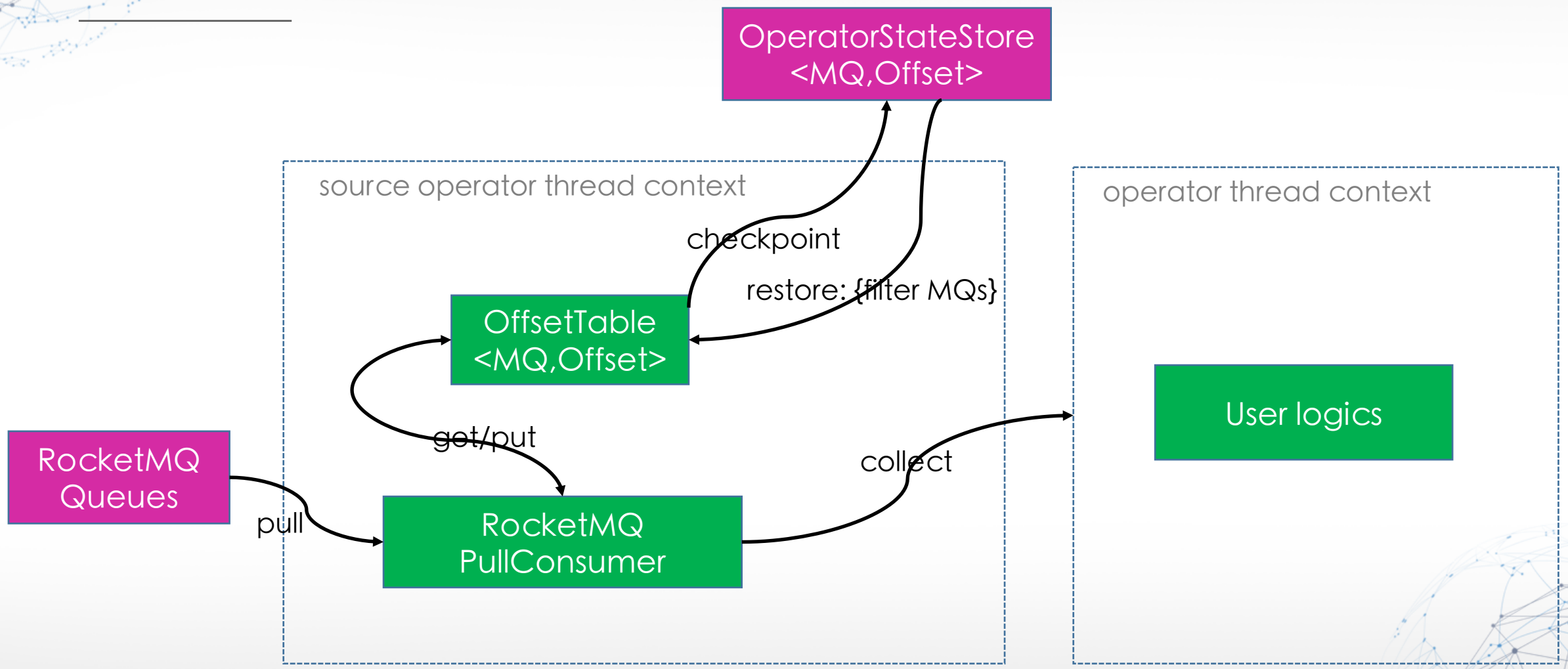
Apache Flink data fault-tolerant: Distributed Snapshot

ABS[1] inspired by Chandy-Lamport[2]



[1] http://arxiv.org/abs/1506.08603
[2] http://research.microsoft.com/en-us/um/people/lamport/pubs/chandy.pdf
[3] picture from the apache flink site

Integrating RocketMQ with Apache Flink

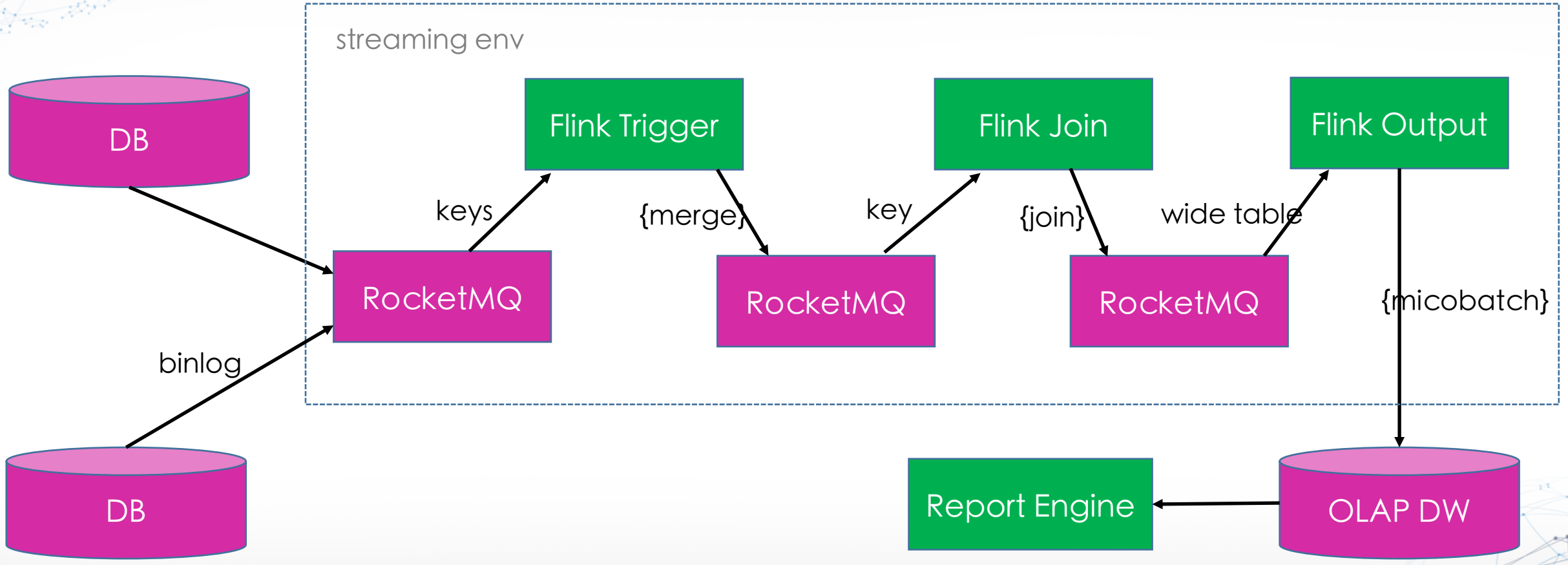


Flink China

RocketMQ-Flink Code Example

```
Properties properties = new Properties();
properties.setProperty(RocketMQConfig.NAME_SERVER_ADDR, "localhost:9876");
properties.setProperty(RocketMQConfig.CONSUMER_GROUP, "c002");
properties.setProperty(RocketMQConfig.CONSUMER_TOPIC, "flink-source");

env.addSource(new RocketMQSource(new SimpleKeyValueDeserializationSchema("id", "address"), properties ))
.name("rocketmq-source")
.setParallelism(2)
.process(new ProcessFunction<Map, Map>() {
            @Override   public void processElement(Map in, Context ctx, Collector<Map> out) throws Exception {
               // do something
               out.collect(result);
            }
        })
.name("upper-processor")
.setParallelism(2)
.addSink(new RocketMQSink(new SimpleKeyValueSerializationSchema( “id” , “province” ), new
DefaultTopicSelector( “flink-sink” ) , properties ).withBatchFlushOnCheckpoint(true))
.name("rocketmq-sink")
.setParallelism(2);
```

RocketMQ-Flink Case: BI



RocketMQ-Streaming Issues & Practices

- Centralized vs local data store
  - state loss or performance loss
- Design vs performance
  - put the lightweight logics into the same operator
- Repairability of messages
  - reliable data source
- End-to-end data auditing
  - find where the data is lost
- Data skew
  - choose the right hash key, two-phase aggregation, or micro-batch
- Big objects serialization:
  - reduce objects size, and enable kryo registry
- Heavy GCs:
  - take care of your heap memory usage
- Too many tasks
  - tuning the number of threads

Flink China

Apache Flink Advantages and Challenges

Advantages

- SQL API
- Stateful computing
- Low latency & High throughput
- Natural Backpressure

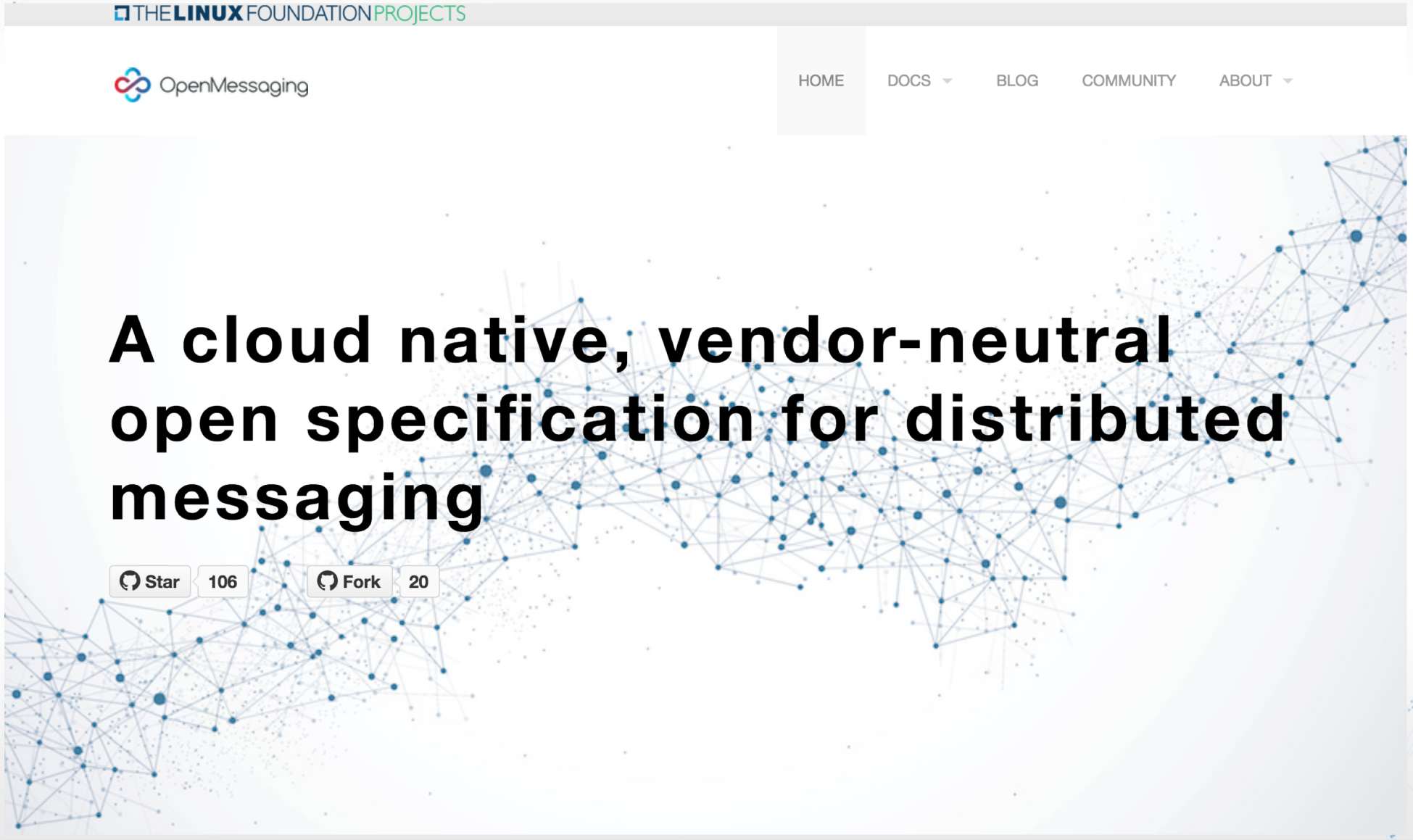
Challenges

- State loss
- Gray Upgrade
- Dynamic Task Profiling

03

The trend of RocketMQ Streaming

The trend of RocketMQ Streaming



SQL!! but, Why?

- Declarative
- Stable
- Optimized
- Unify
- Understandable
- ETL/BI

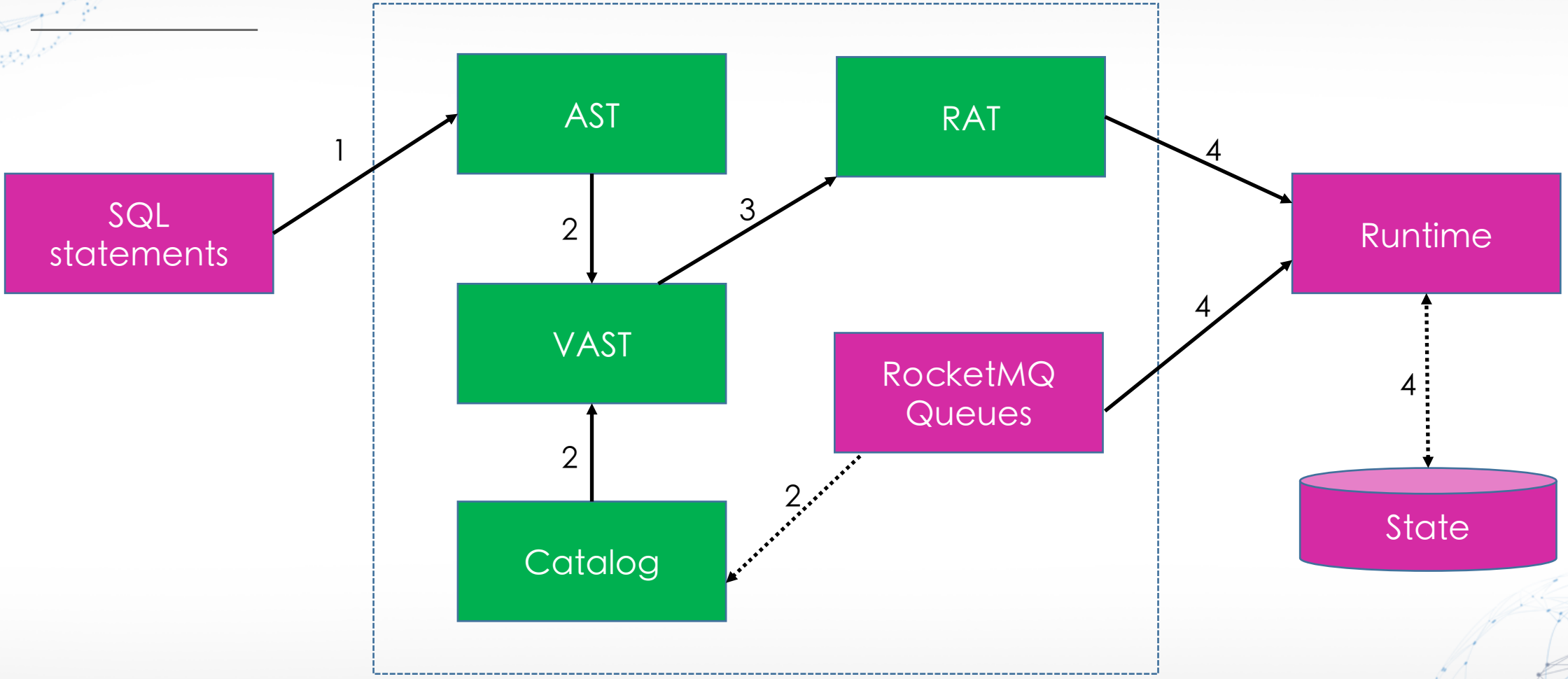
The trend of RocketMQ Streaming

Dose Apache RocketMQ have any plan to support the feature?

The trend of RocketMQ Streaming

Dose Apache RocketMQ have any plan to support the feature?

RocketMQ-SQL Design



1.sql parse 2.ast validate 3. rat optimize 4. rat run

Flink China

RocketMQ-SQL Case: IOT

```
create table {传感器} (
{ID} varchar,
{类型} varchar,
{数据} varchar,
{时间} varchar,
) with (
topic = '{传感器MQ}',
consumerGroup = 'C01'
);
```

```
create table {设备} (
{设备ID} varchar,
{设备区域} varchar,
{传感器ID} varchar,
{传感器类型} varchar
) with (
topic = '{设备MQ}',
consumerGroup = 'C02'
);
```

```
select
  d.{设备ID},
  d.{设备区域},
  s.{ID}
from {传感器} as s
inner join {设备} as d
on s.{ID} = d.{传感器ID}
where cast(s.{传感器数据} as bigint) > 90;
```

THANKS

Flink China社区大群

扫一扫群二维码，立刻加入该群。