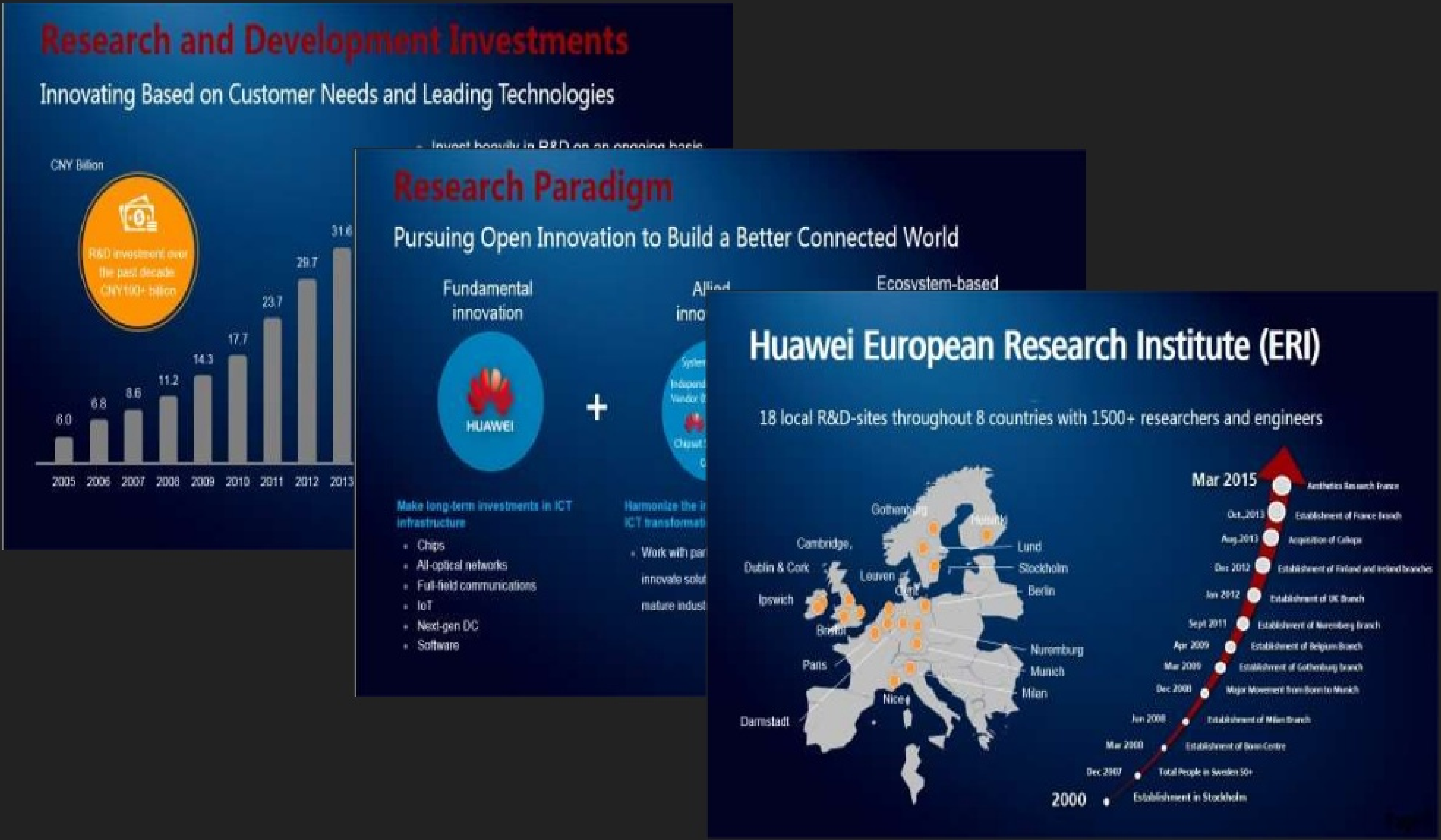
FLINK FORWARD - BERLIN 2017

HUAWEI CLOUD STREAM SERVICE IN PRACTICE

Radu Tudoran
Jinkui Shi

HUGE R&D INVESTMENTS FOR OPEN INNOVATION



Research and Development Investments

Innovating Based on Customer Needs and Leading Technologies

- Invest heavily in R&D on an ongoing basis

R&D investment over the past decade CNY100+ billion

Research Paradigm

Pursuing Open Innovation to Build a Better Connected World

Fundamental innovation

+

Allied inno

Ecosystem-based

Make long-term investments in ICT infrastructure

- Chips
- All-optical networks
- Full-field communications
- IoT
- Next-gen DC
- Software

Harmonize the i
ICT transformati

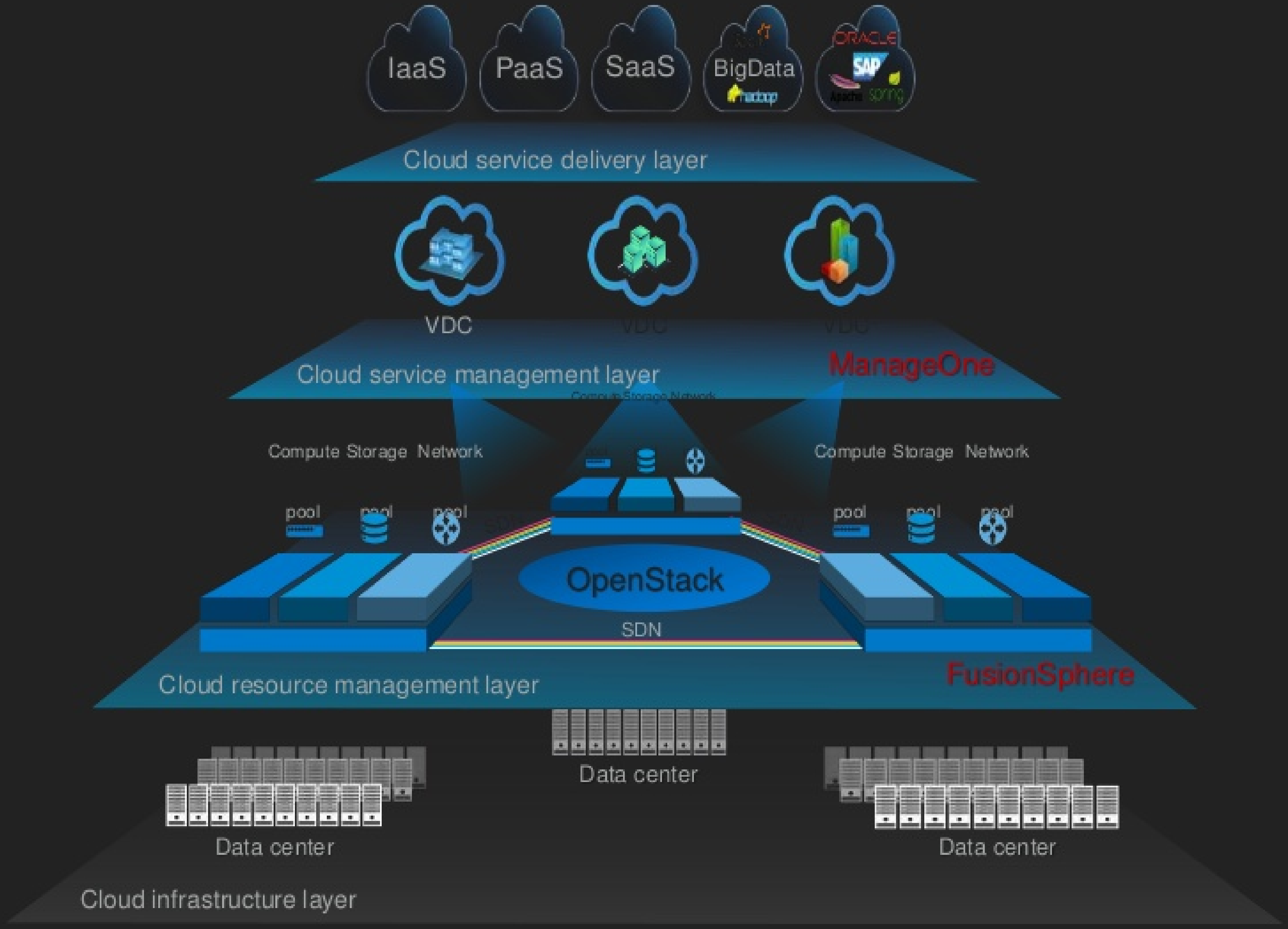
- Work with par
innovate solut
mature indust

Huawei European Research Institute (ERI)

18 local R&D-sites throughout 8 countries with 1500+ researchers and engineers

Mar 2015 Aesthetics Research France
Oct_2013 Establishment of France Branch
Aug.2013 Acquisition of Caliopa
Dec 2012 Establishment of Finland and Ireland branches
Jan 2012 Establishment of UK Branch
Sept 2011 Establishment of Nuremberg Branch
Apr 2009 Establishment of Belgium Branch
Mar 2009 Establishment of Gothenburg branch
Dec 2008 Major Movement from Bonn to Munich
Jun 2008 Establishment of Milan Branch
Mar 2008 Establishment of Bonn Centre
Dec 2007 Total People in Sweden 50+
2000 Establishment in Stockholm

HUAWEI CLOUD COMPUTING SOLUTION



Service-driven distributed cloud data center (SD-DC^2) architecture

AGENDA

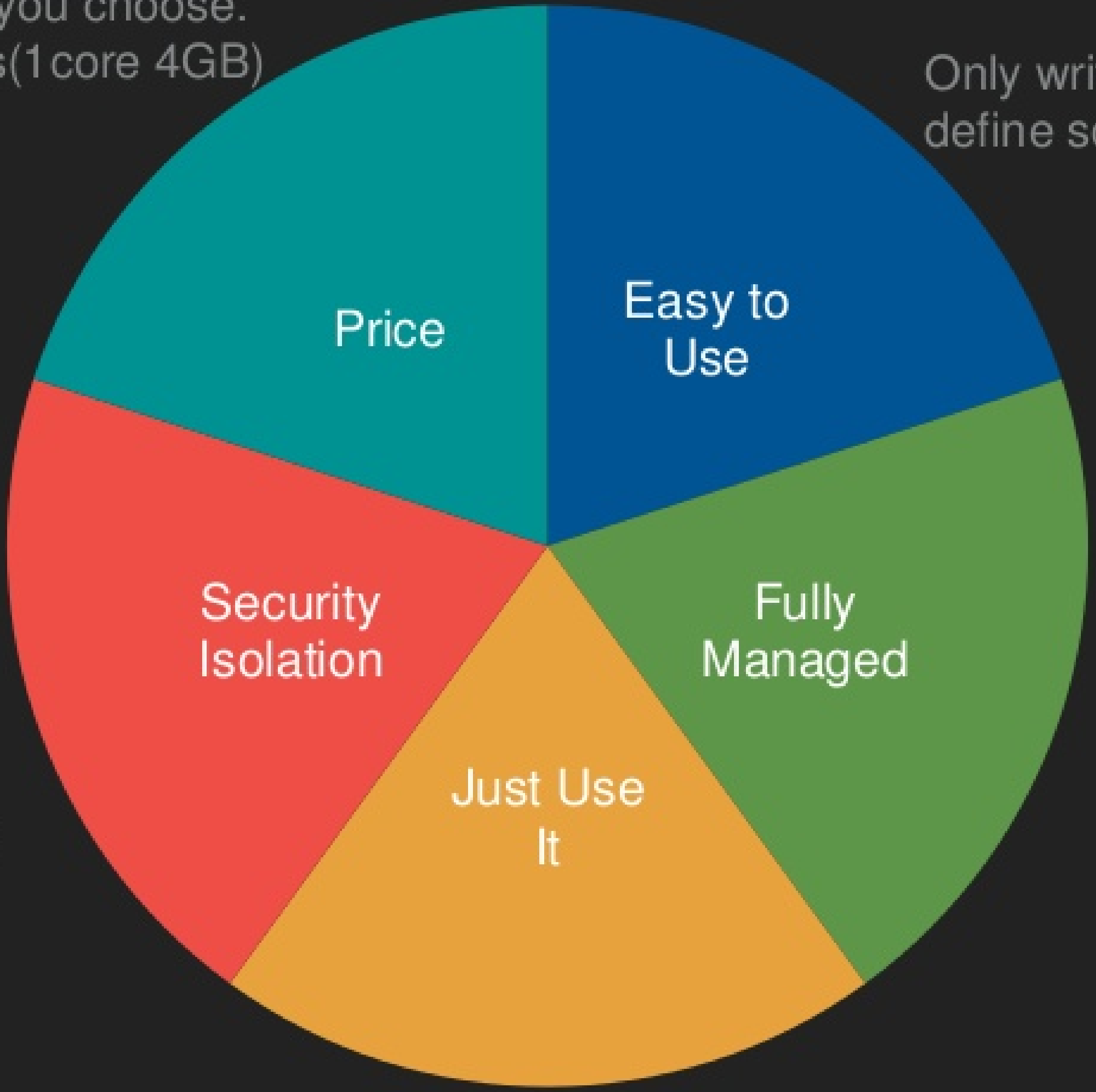
- About Cloud Stream Service
- How to build a cloud native service
- How to wrap Flink as a Service
- How to DevOps Cloud Stream Service
- Flink features focus on cloud platform

ABOUT HUAWEI CLOUD STREAM SERVICE

Cloud Stream Service (CS): It is a cloud native stream analytics service on Huawei Cloud. Fully managed cluster that saves you from the need to touch the running cluster. Only write Stream SQL, submit it to run

Only pay for how many SPUs you choose.
SPU: Stream Processing Units(1core 4GB)

Only write Stream SQL in editor, SQL define source/processing logic/sink.



The running cluster is fully isolated from others. Also Flink internal components are enhanced for communicating with each other.

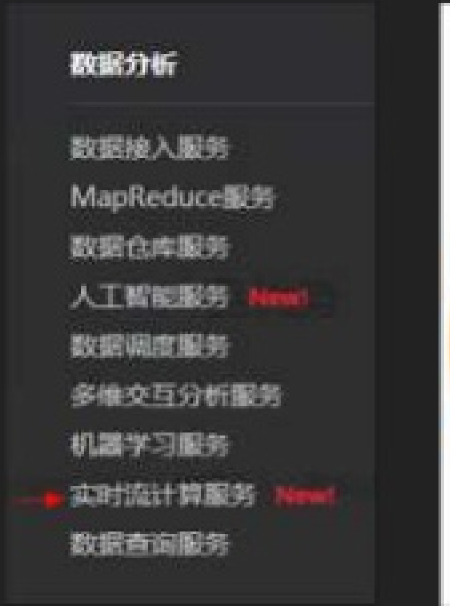
VPC/Container, Sandbox later

User does not manage the running clusters and the Flink job. Only submit SQL in editor, then monitor the job status and the output of sink.

No need to focus on the big data framework such Hadoop, Flink, Zookeeper. Open the SQL editor, just write SQL for testing and running

OVERVIEW OF CLOUD STREAM SERVICE

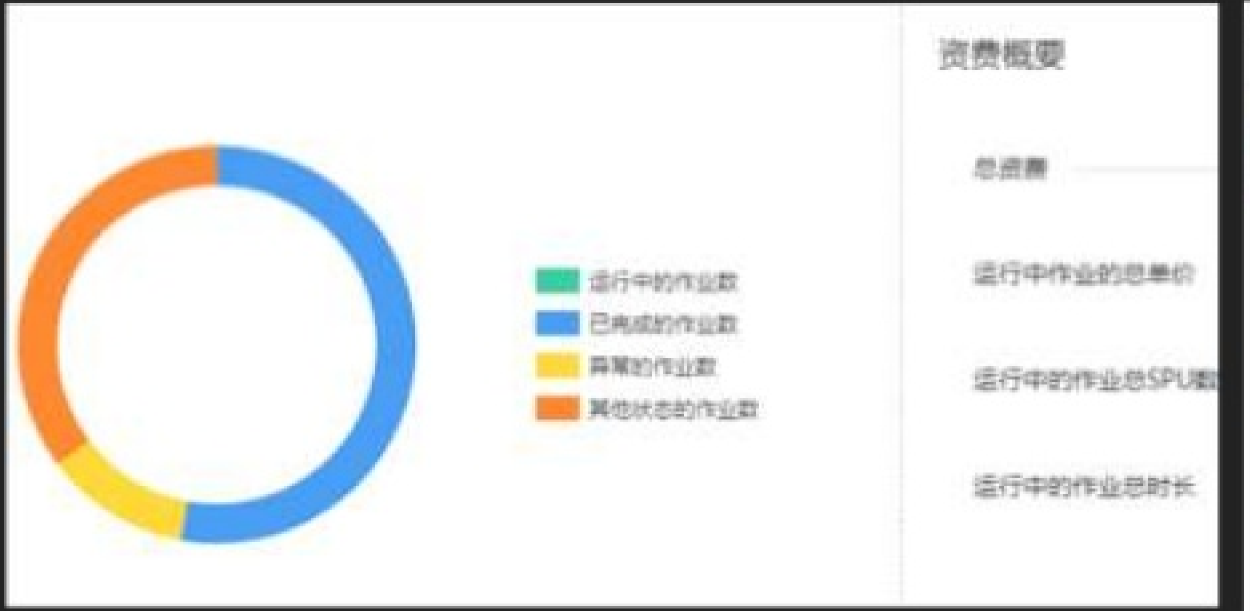
1. menu entrance



数据分析

- 数据接入服务
- MapReduce服务
- 数据仓库服务
- 人工智能服务 New!
- 数据调度服务
- 多维交互分析服务
- 机器学习服务
- 实时流计算服务 New!
- 数据查询服务

2. overview for billing cost



资费概要

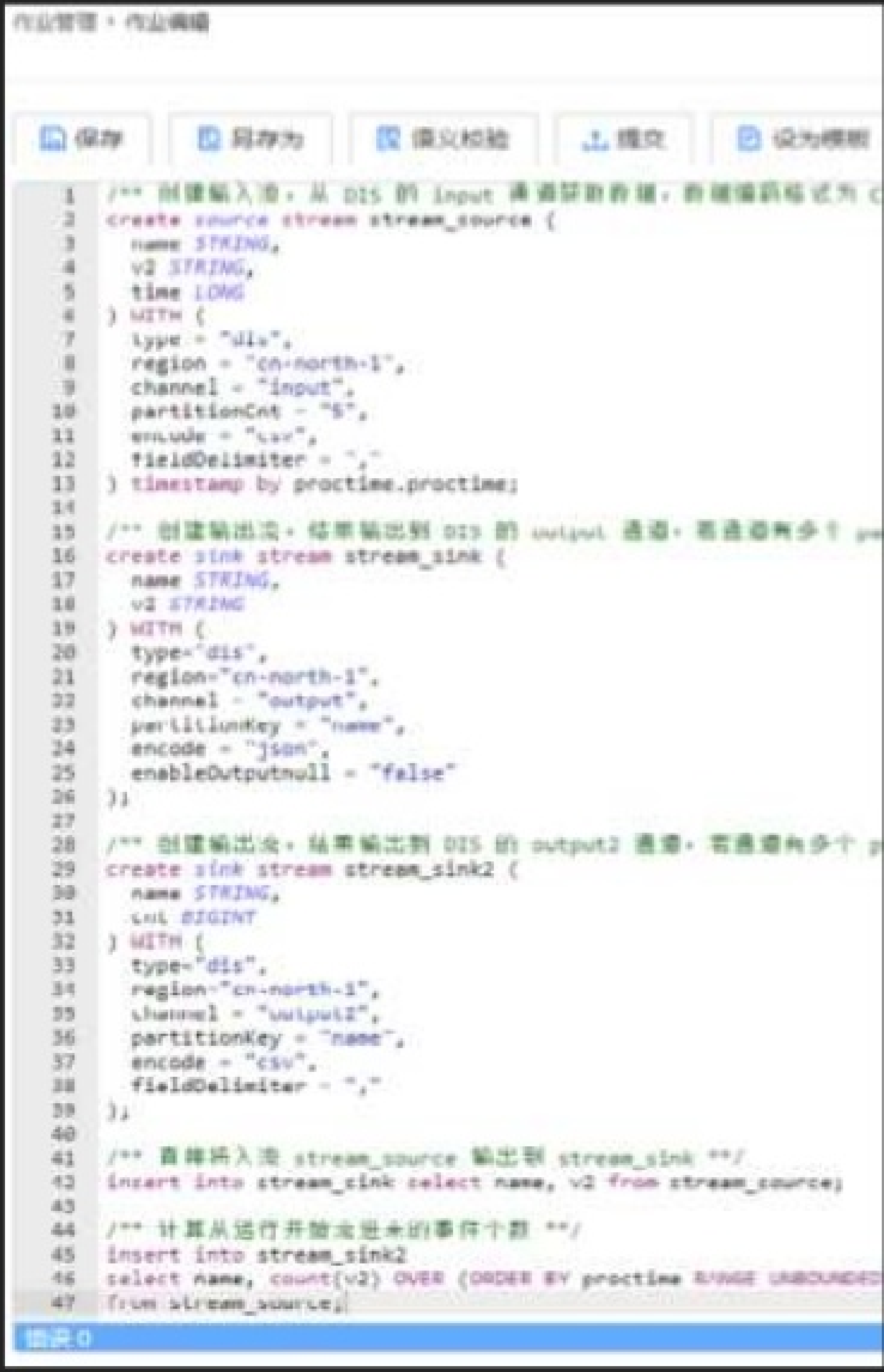
总资费

运行中作业的总单价

运行中的作业总SPU数

运行中的作业总时长

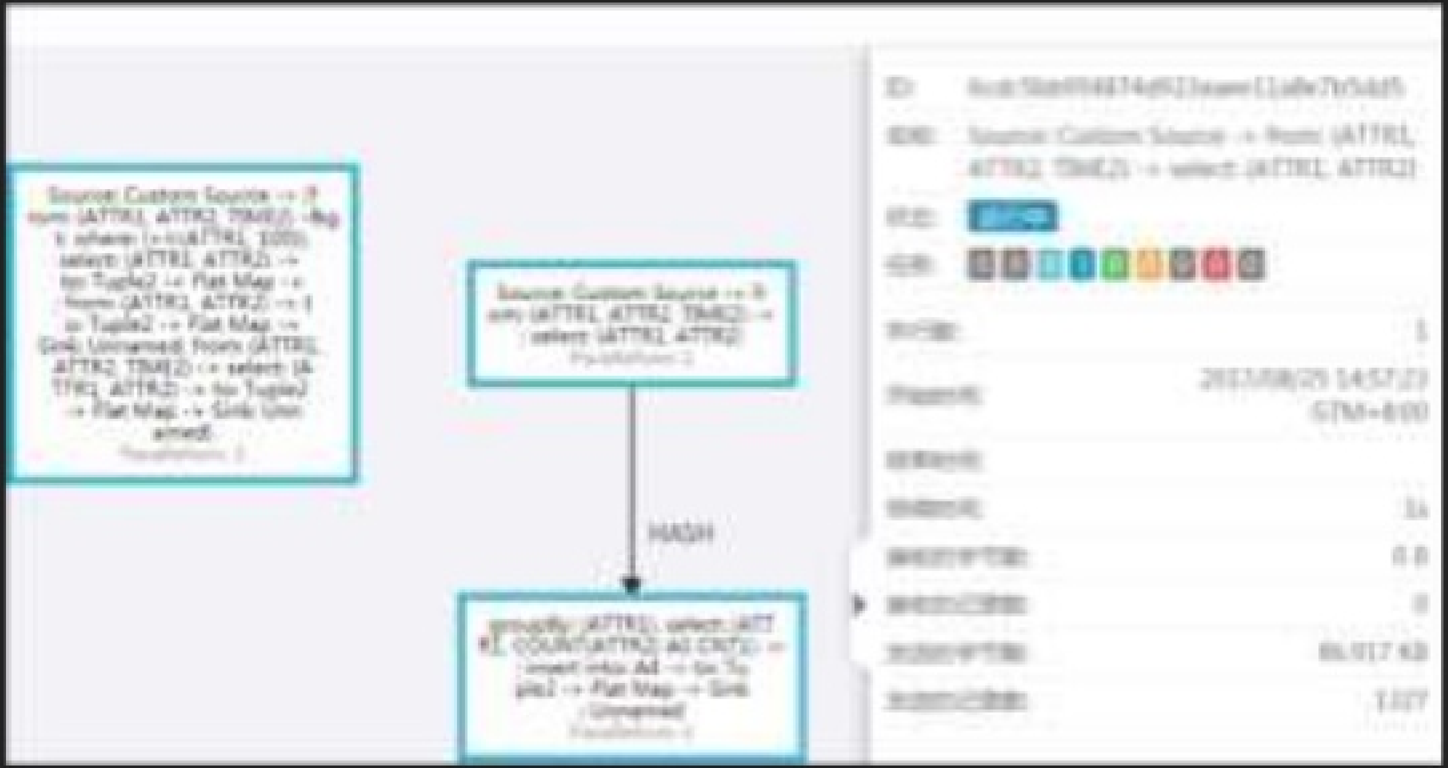
3. Stream SQL Editor



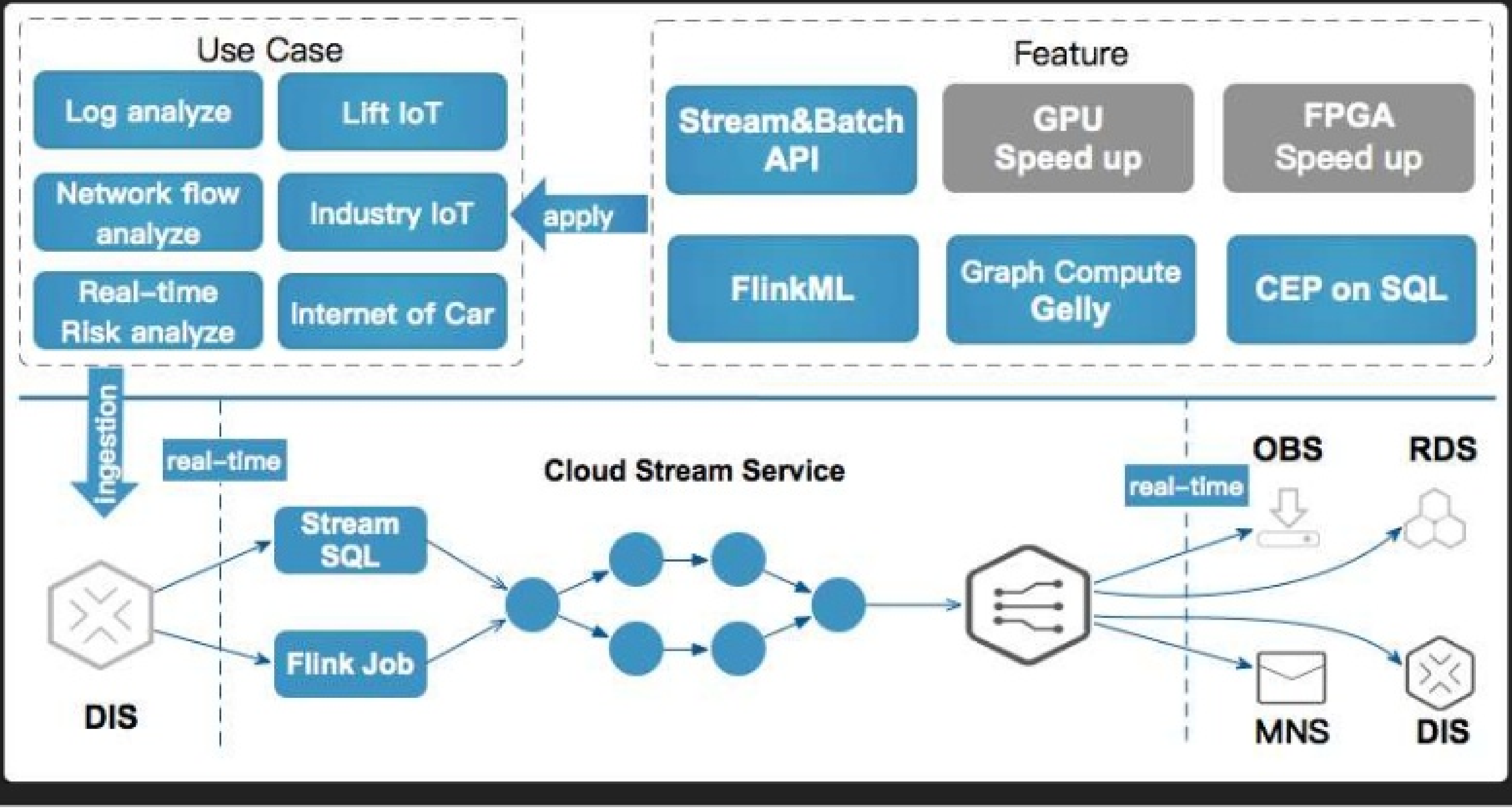
```
/** 创建输入流，从 DIS 的 input 通道获取数据，数据编码格式为 c
create source stream stream_source (
  name STRING,
  v2 STRING,
  time LONG
) WITH (
  type = "dis",
  region = "cn-north-1",
  channel = "input",
  partitionCnt = "5",
  encode = "csv",
  fieldDelimiter = ","
) timestamp by proctime.proctime;

/** 创建输出流，结果输出到 DIS 的 output 通道，若通道有多个 p
create sink stream stream_sink (
  name STRING,
  v2 STRING
) WITH (
  type="dis",
  region="cn-north-1",
  channel = "output",
  partitionKey = "name",
  encode = "json",
  enableOutputnull = "false"
);

/** 创建输出流，结果输出到 DIS 的 output2 通道，若通道有多个 p
create sink stream stream_sink2 (
  name STRING,
  cnt BIGINT
) WITH (
  type="dis",
  region="cn-north-1",
  channel = "output2",
  partitionKey = "name",
  encode = "csv",
  fieldDelimiter = ","
);

/** 直接将入流 stream_source 输出到 stream_sink **/
insert into stream_sink select name, v2 from stream_source;

/** 计算从运行开始至今来的事件个数 **/
insert into stream_sink2
select name, count(v2) OVER (ORDER BY proctime RANGE UNBOUNDED
from stream_source;
```

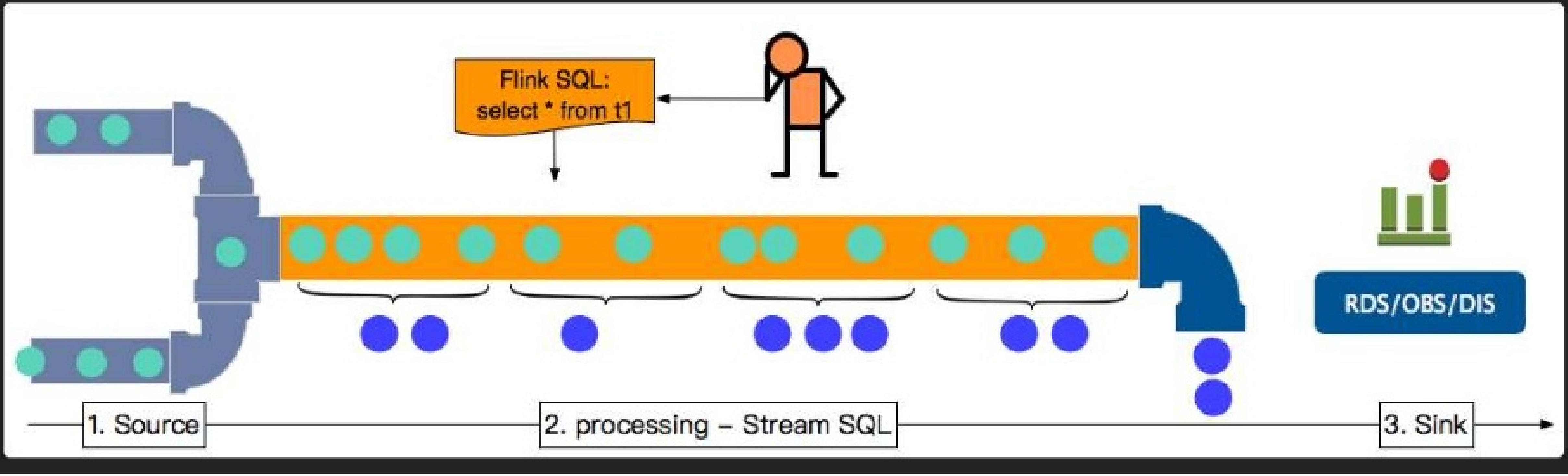
错误 0

4. Running Flink job monitor



DATA FLOW FROM SOURCE TO SINK





STREAM SQL DEFINES THE WHOLE STREAM ANALYSIS PROCESS

```
// step 1:
// Create source， fetch streaming data from DIS topic, the data line format default is CSV,
// default field separator is comma “,”
create source stream stream_source(attr1 int, attr2 string, time2 long) with (
  type= "dis",
  region = "cn-north-1",
  channel = "csinput",
  partitionCnt = "1",
  encode = "csv",
  fieldDelimiter = ","
);

// step 2:
// Create sink, output the result streaming data to DIS topic
create sink stream result_sink(attr1 int, attr2 INT) with (
  type = "dis",
  region = "cn-north-1",
  channel = "csoutput",
  partitionKey = "attr1",
  encode = "csv",
  fieldDelimiter = ","
);

// step 3：
// analyze streaming data in real-time, write the result data to DIS topic
// 计算从运行开始流进来的事件个数
insert into result_sink
select name, count(v2) OVER (ORDER BY proctime RANGE UNBOUNDED preceding) as cnt1 from stream_source;
```

HOW WAS CLOUD STREAM SERVICE BUILT?

Someone

WHAT IS CLOUD NATIVE?

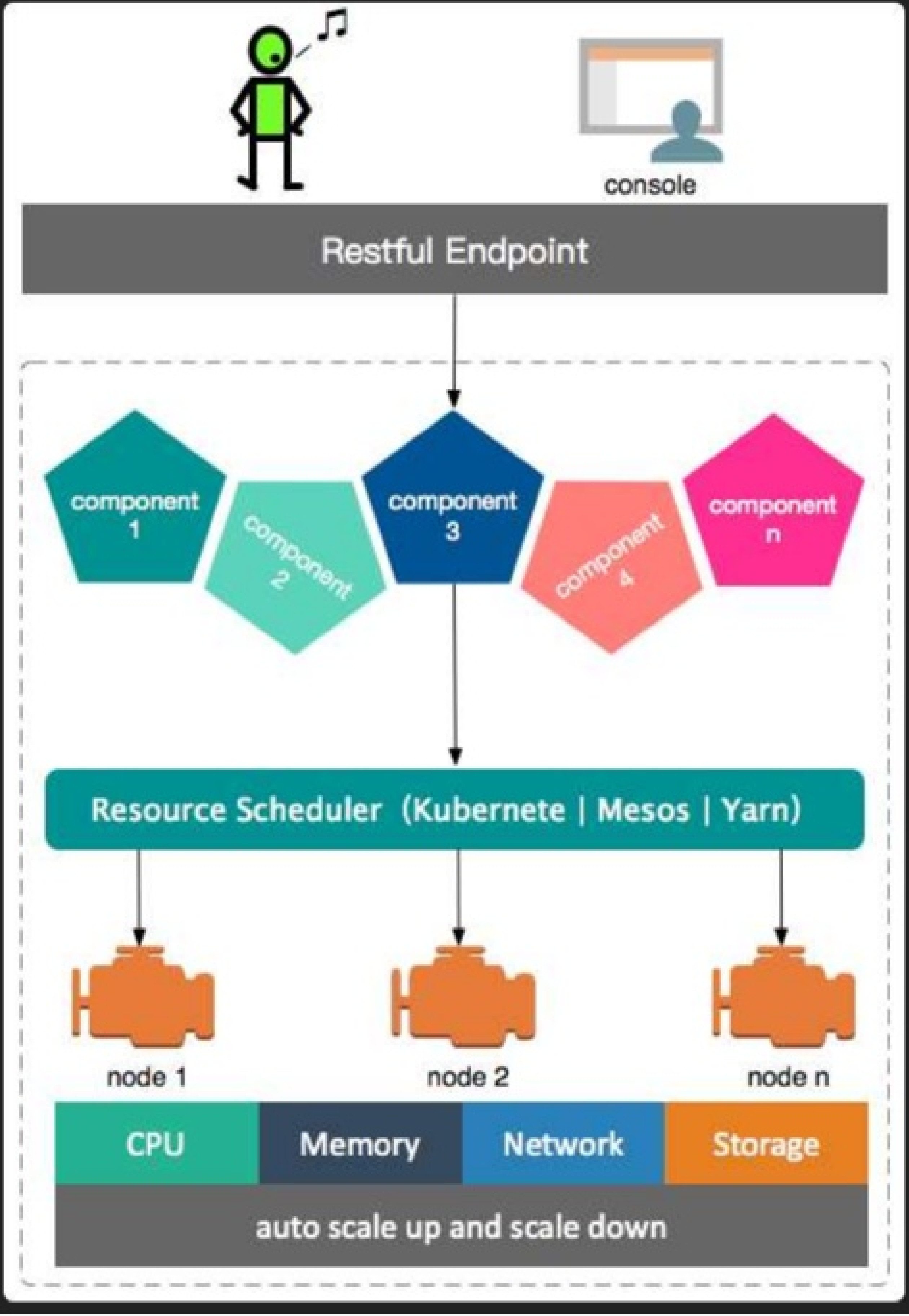
- Devops: Continuous Integration and Continuous Delivery
- Microservice：Independent process with Restful API
- Container：isolation and quota, OS-less
- Reactive: Responsive, Resilient, Elastic, Message Driven

Reference：

1. Developing Cloud Native Applications
2. What are Cloud-Native Applications?
3. The Reactive Manifesto

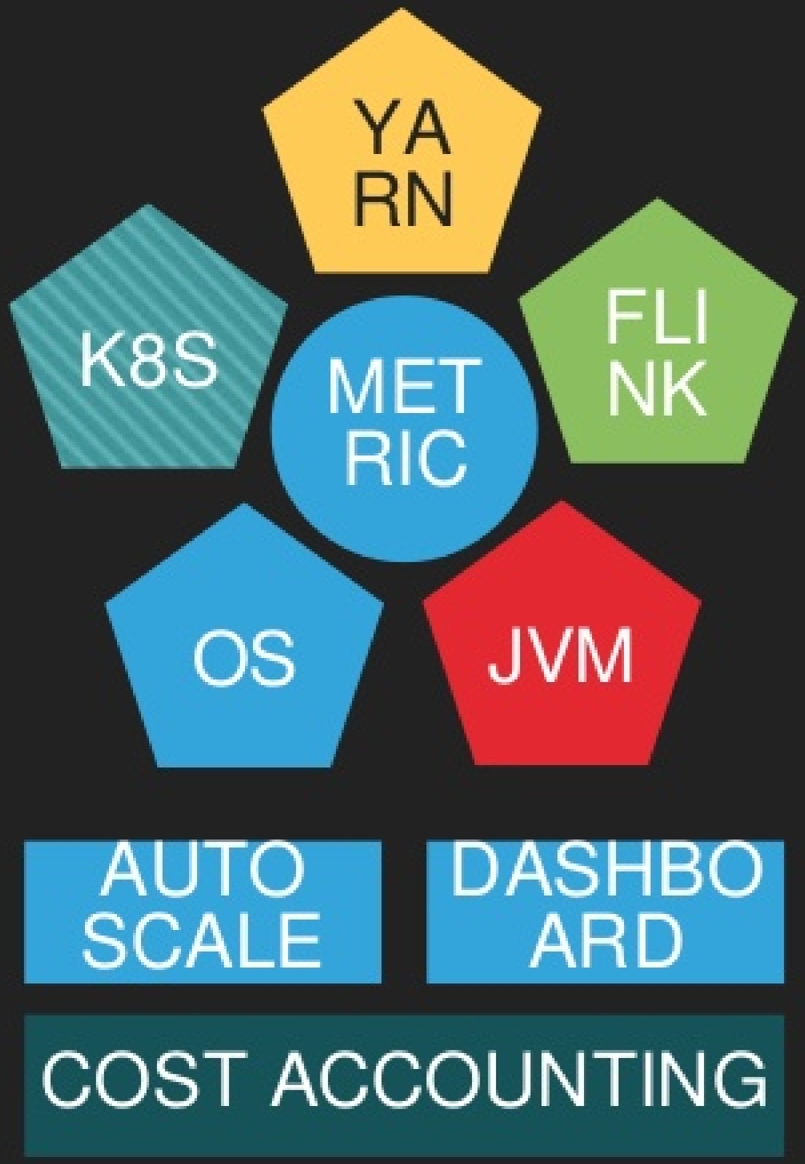
ARCHITECTURE

- Play framework for Restful API and business processing
- Akka
  Message driven, make modules clear
- Netty
  Communication between components
- sbt-native-packager
  build rpm package to run as OS service
- Jenkins: CI and CD

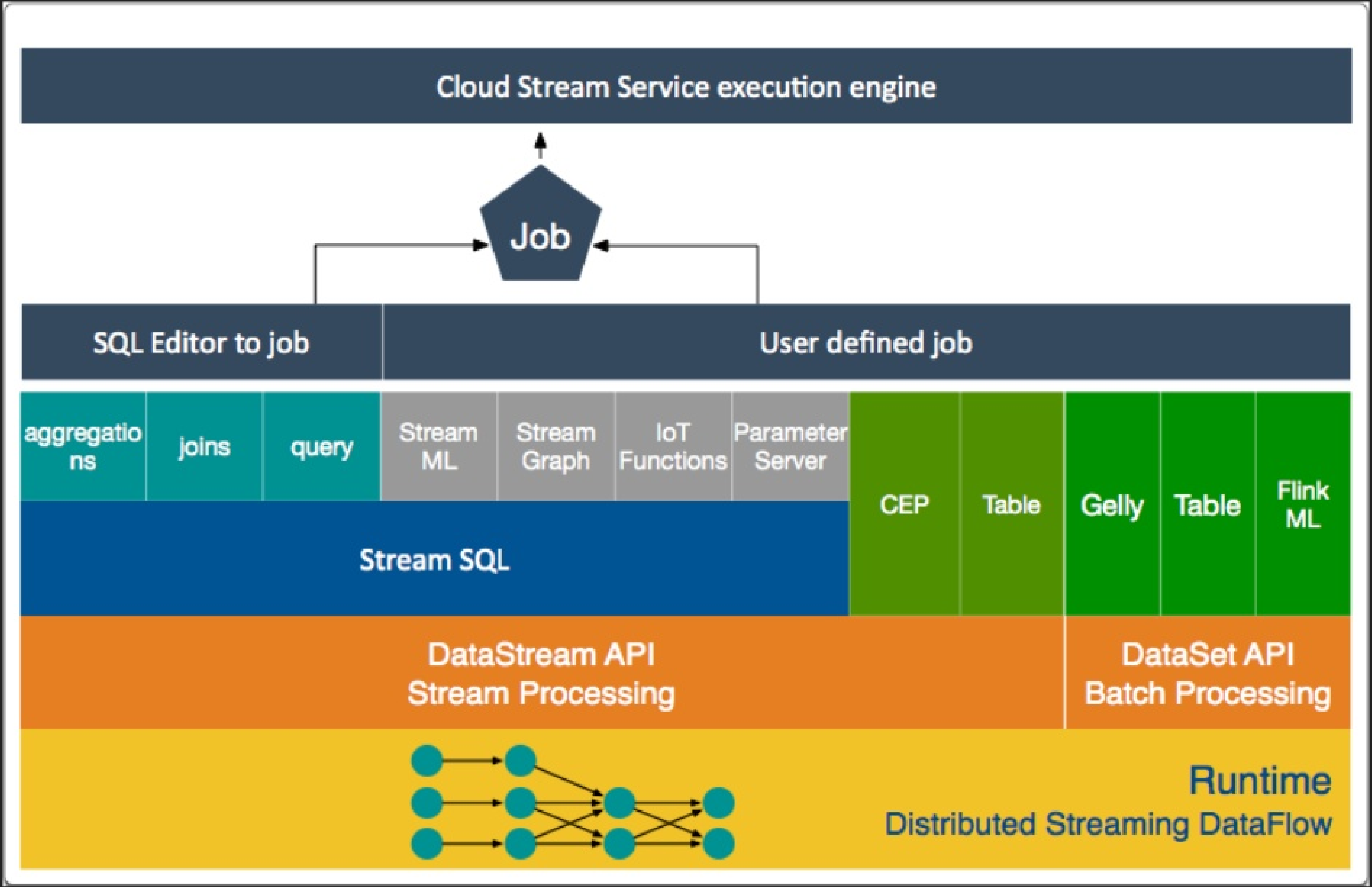


AUTO OPS BY METRICS

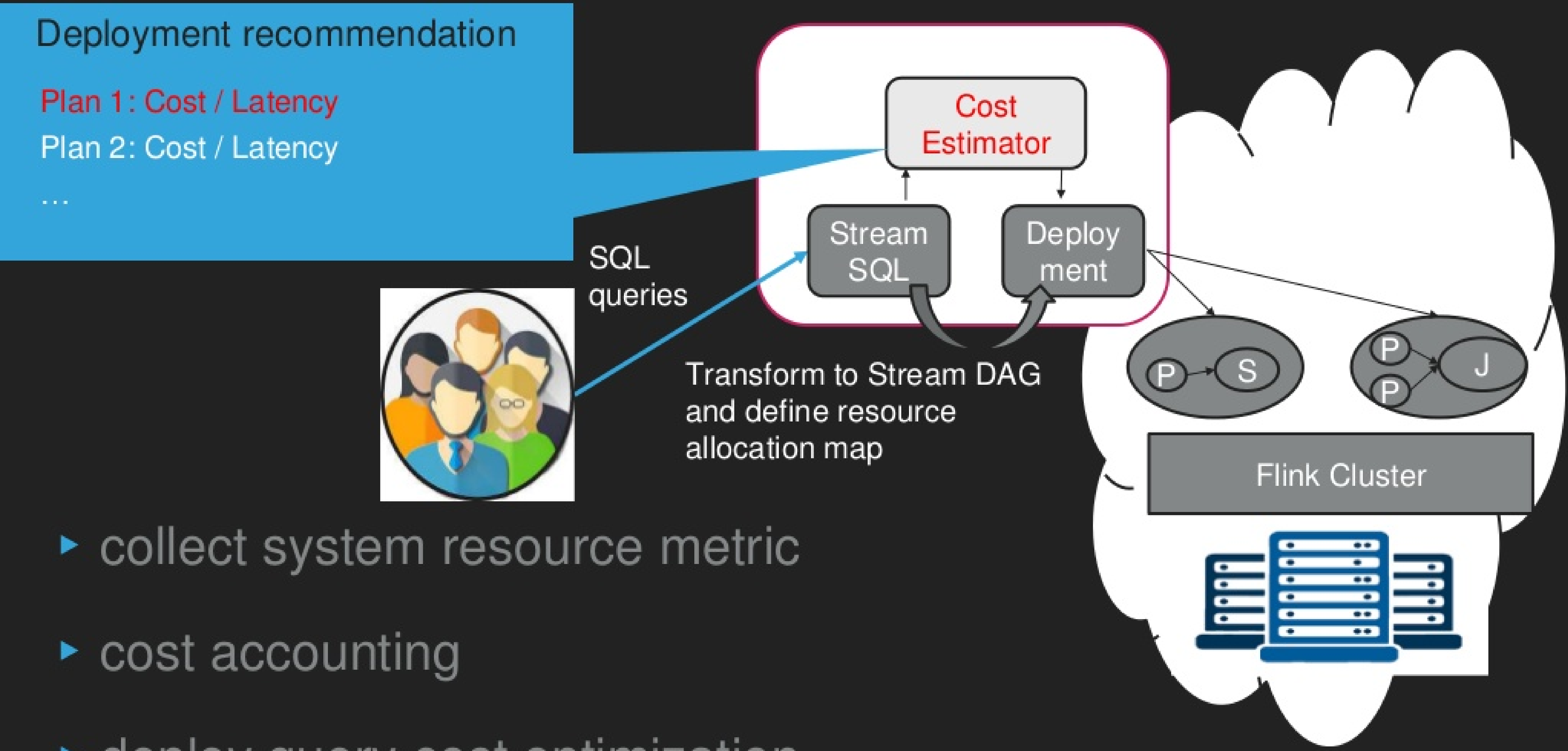
- collect system resource metric
- trigger auto scale
- cost accounting
- trace every request for tunning service
- tenant quote setting
- …



HOW TO WRAP FLINK AS A SERVICE



NEXT STEPS: OPTIMIZED SQL RESOURCE ALLOCATOR



Deployment recommendation

Plan 1: Cost / Latency

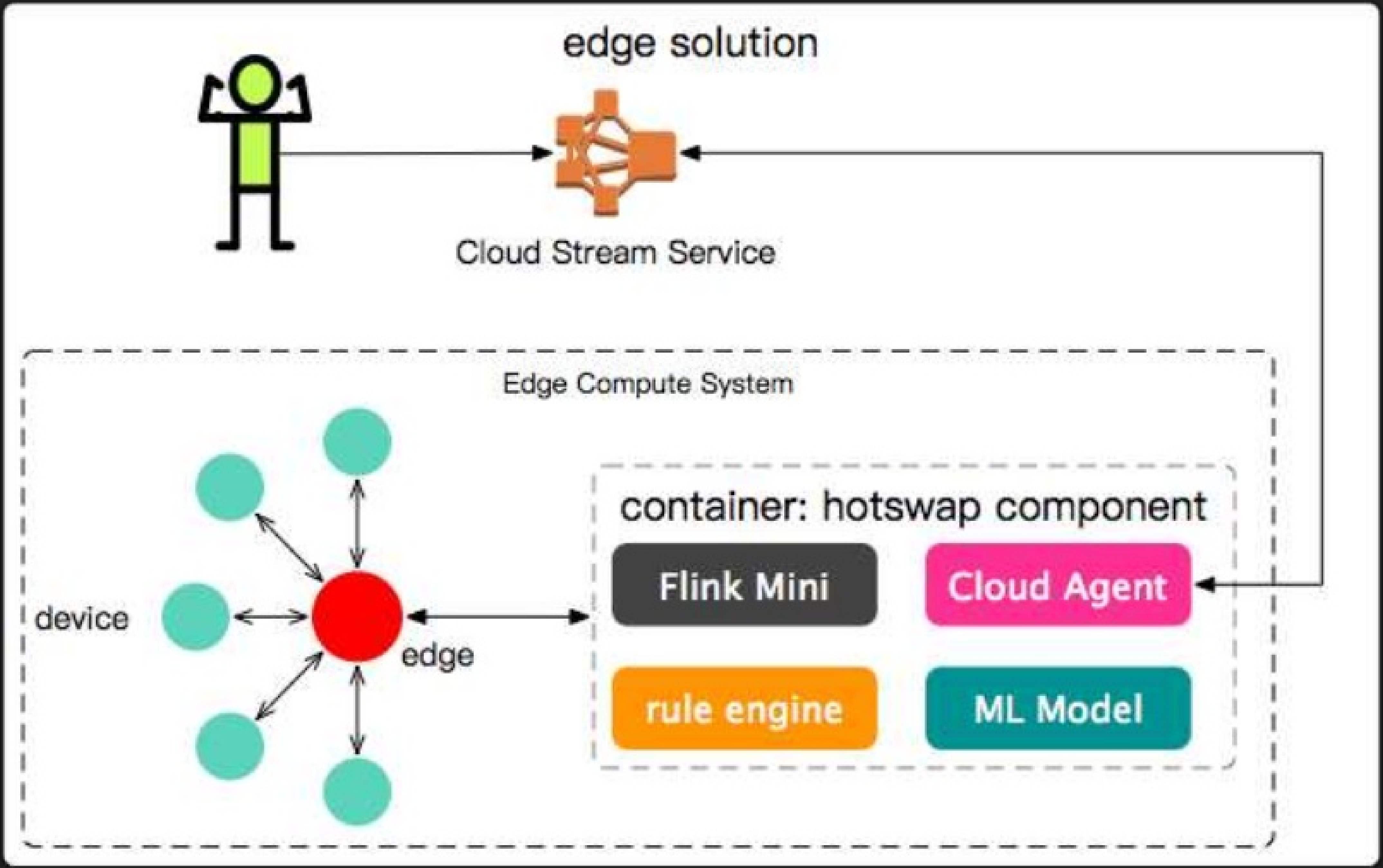
Plan 2: Cost / Latency

…

OTC; Huawei Public Cloud

- collect system resource metric
- cost accounting
- deploy query cost optimization
- adapt the SPU resources to the actual needs
- provide recommendation execution plans

NEXT STEPS: EDGE COMPUTING



Flink core must be minimized and extend IoT language support. Flink can run on an edge device, but it is still not small nor "*edge-smart*" enough .

Other choices:

1. http://edgent.apache.org
2. http://gearpump.apache.org

Suggestions?

We are hiring…

Thanks