58 DataFun.

58大数据系列直播

58技术

DataFunTalk

58大数据平台系列直播——

《实时计算平台架构与实践》

分享人：冯海涛

目录

CONTENT

01
简介

实时计算平台

定位

为集团海量数据提供高效、稳定实时计算一站式服务

- 实时数据存储
- 实时数据计算
- 实时数据转发

基础能力建设

- Kafka
- Storm
- Flink

平台化建设

- DDS数据分发平台
- Wstream一站式实时计算平台

实时计算平台

规模

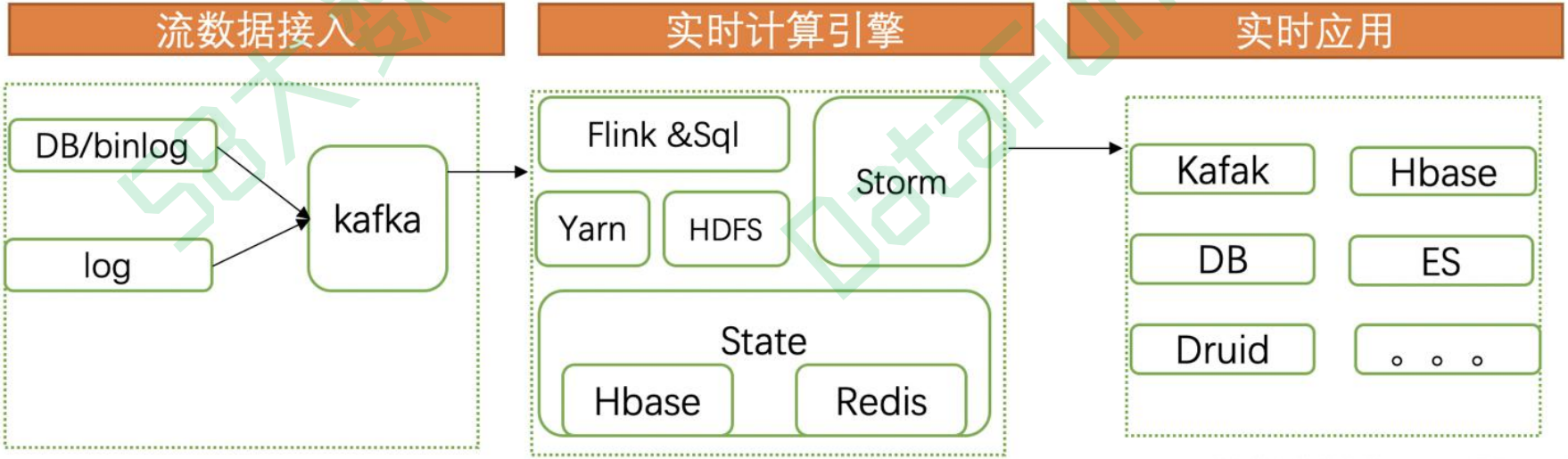
500+
集群规模

600+
实时任务数

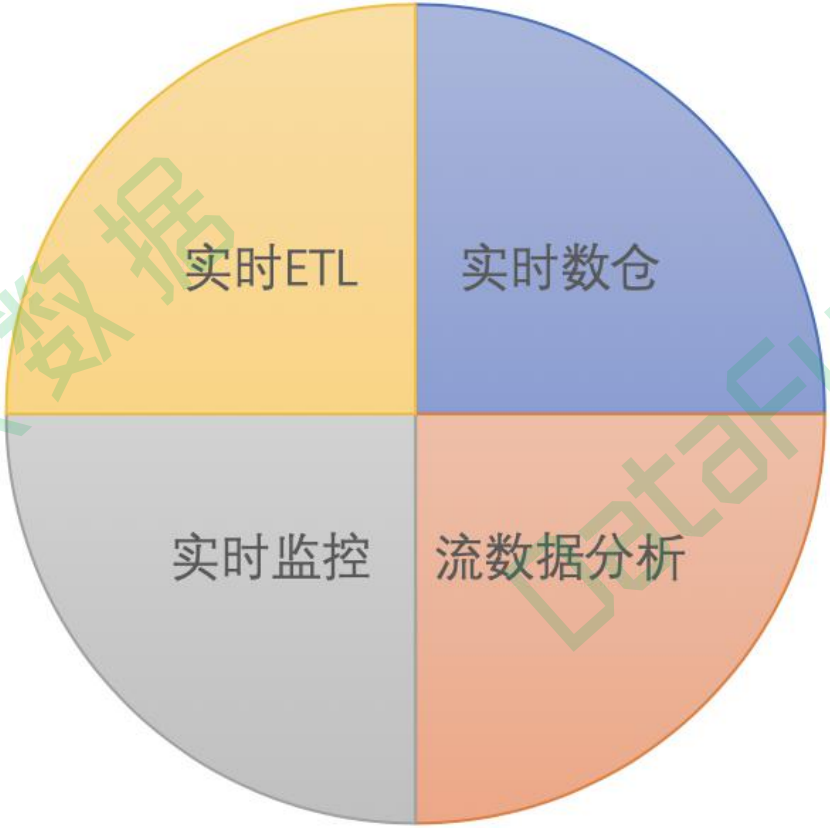
6000亿
日实时计算数据量

1500万
每秒峰值

架构



实时计算场景



02
Flink

基础能力建设

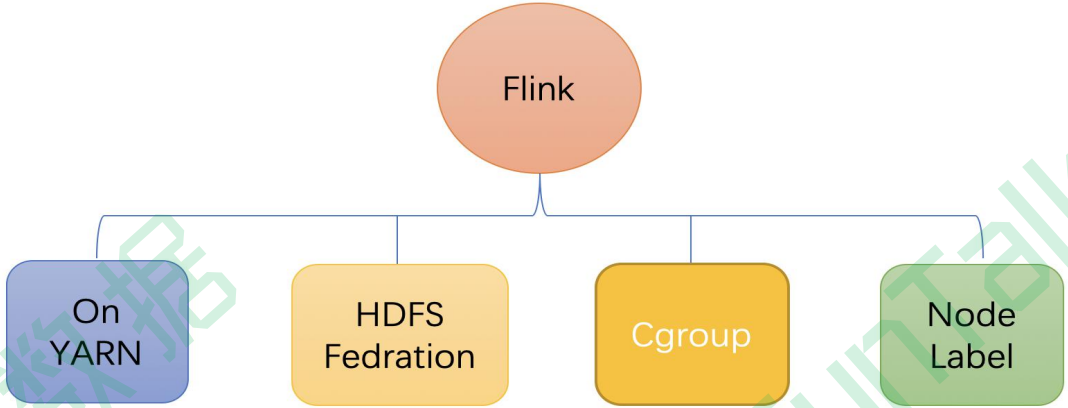
背景

- 高吞吐/低延迟
- 灵活窗口支持
- 数据乱序
- Exactly once语义
- 状态管理

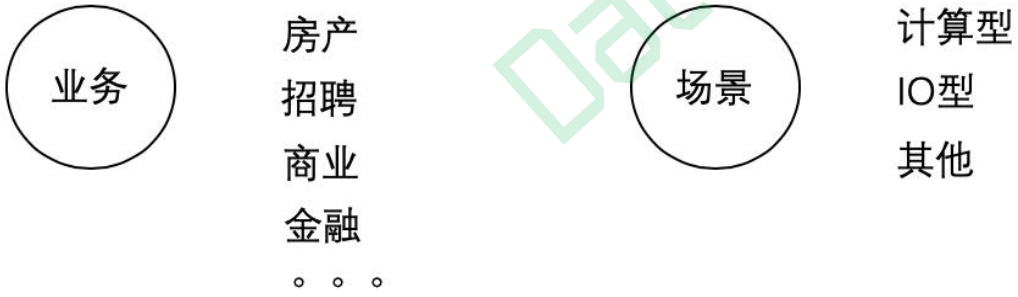
	Storm	Spark streaming	Flink
计算模型	streaming	Mico batching	streaming
数据保障	At least once	Exactly once	Exactly once
延迟	ms	s	ms
吞吐量	low	high	high
容错	ack	dstream checkpoint	Distributed Snapshots
状态管理	N/A	简单	丰富

集群稳定性

- 高可用架构



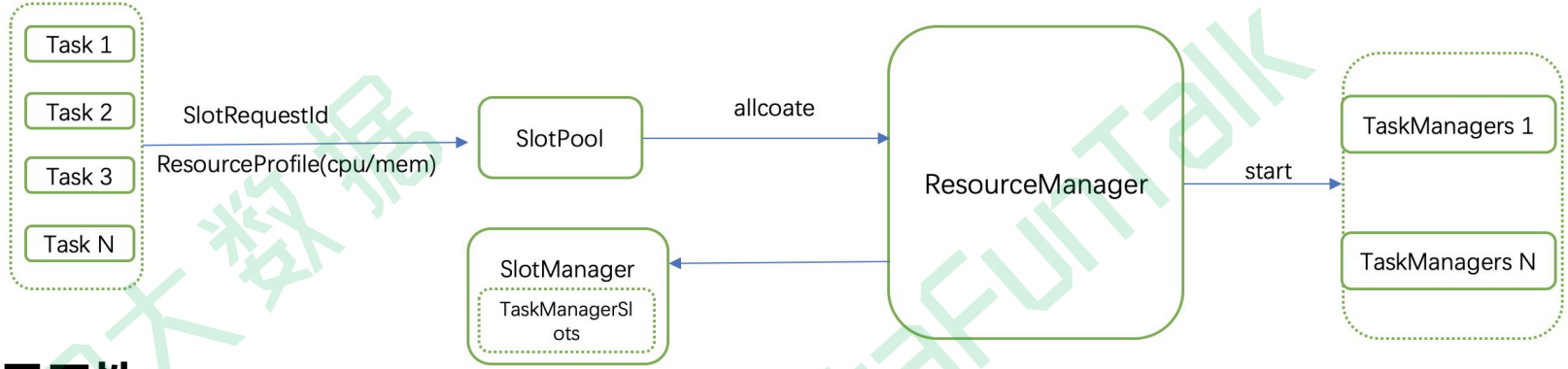
- 任务隔离



Flink 改进

➢ 资源管理

细粒度资源



➢ 易用性

写hdfs支持lzo压缩

数据源自动处理换行

支持第三方依赖jar

支持配置文件自动分发

流式sql背景

- **优势**

开发门槛低　语法稳定 易于理解　自动优化　批流统一

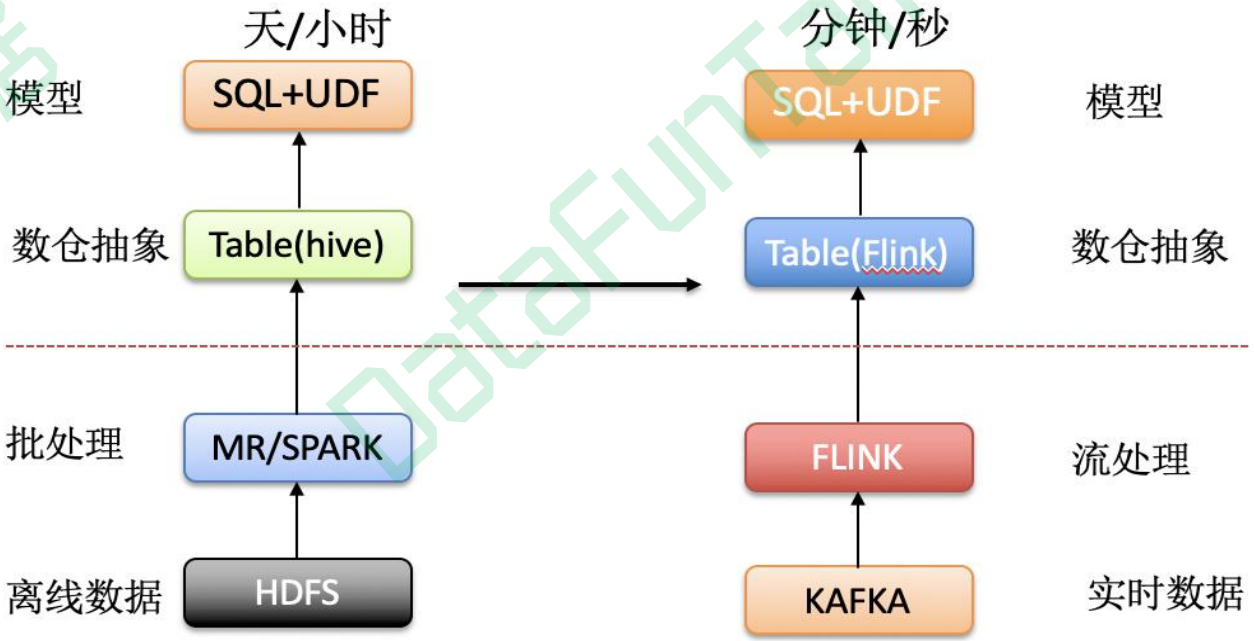
- **数仓实时化诉求**

业务

- 实时报表
- 实时标签
- 实时接口

平台

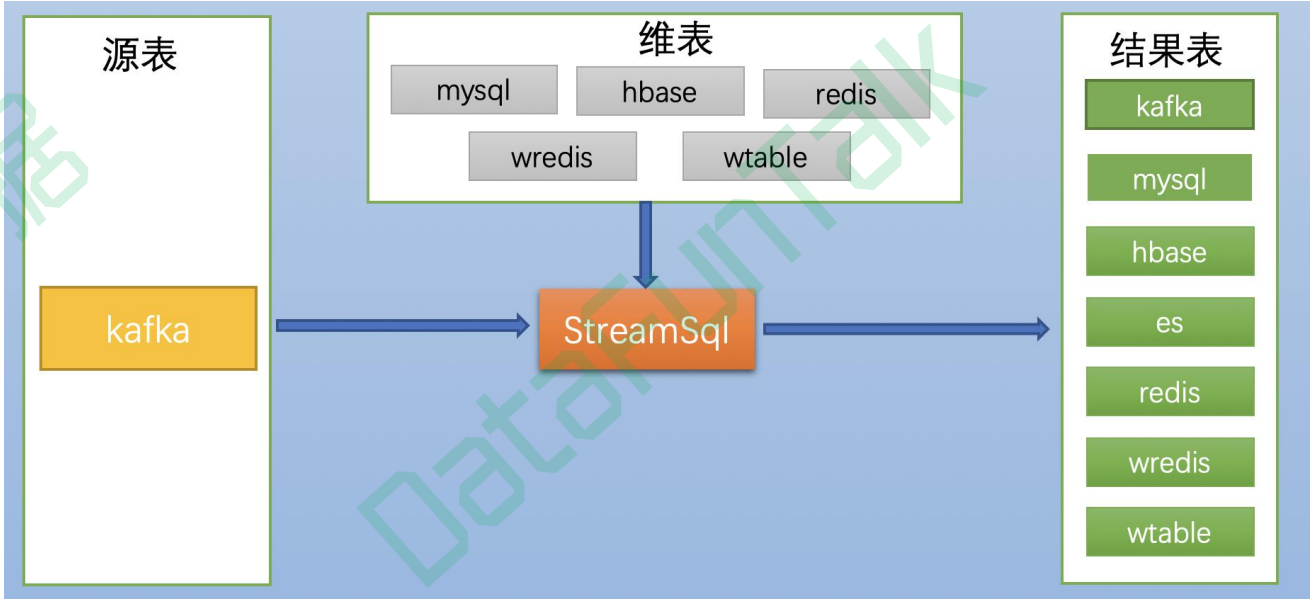
- 任务调度：计算集中在凌晨，集群压力大
- 数据导入：批量导入耗费时间长



核心功能

基于Flink Sql扩展

- 支持自定义DDL语法
- 支持自定义UDF语法
- 实现了流与维表的join
- 打通主流存储以及公司内部实时存储



DDL原理

```
CREATE TABLE MyTable(
 channel STRING,
    name STRING
 )WITH(
    type ='kafka10',
    kafka.bootstrap.servers ='127.0.0.1
    kafka.auto.offset.reset ='latest',
    kafka.topic ='kafkatopictest',
    parallelism ='1'
 );

CREATE TABLE MyResult(
    channel VARCHAR,
    name VARCHAR
 )WITH(
    type ='mysql',
    url ='jdbc:mysql://127.0.0.1:33
    userName ='root',
    password ='1234',
    tableName ='MyResult',
    parallelism ='1'
 );

insert
into
    MyResult
select
    channel,
    name
from
    MyTable;
```

声明source、sink，
由insert语句决定是源还是结果表

表参数，声明配置信息和运行时并发度

业务逻辑

```
    FlinkKafkaConsumer011<Row> kafkaSrc = new FlinkKafkaConsumer011(
            topic: "kafkatopictest",
            new CustomerJsonDeserialization(typeInformation),
            pro);
    kafkaSrc.setStartFromLatest();

    DataStreamSource sourceStream = env.addSource(kafkaSrc, sourceName: "MyTable",typeInformation);
    //sourceStream.setParallelism(2);
    Table kafkaTable1 = tableEnv.fromDataStream(sourceStream, fields: "channel,name");
    tableEnv.registerTable(sourceStream.getName(), kafkaTable1);

    TableSink sink = new MySQLTableSink();
    tableEnv.registerTableSink( name: "MyResult",new String[]{"channel", "name"},types,sink);

    tableEnv.sqlUpdate( stmt: "insert into MyResult select channel,name from MyTable");

    System.out.println(env.getExecutionPlan());
    env.execute();
```

注册源表

注册结果表

业务逻辑

维表join

- 语法解析

  Apache Calcite

- 标准语法

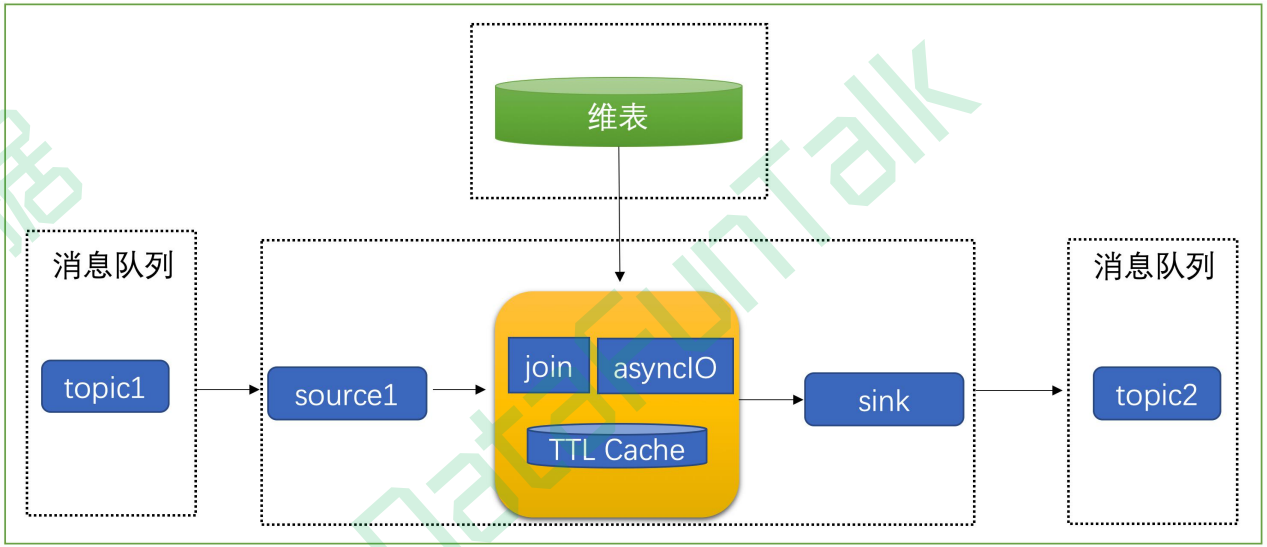
  SQL:2011 Temporal Table

- 性能

  Flink Aysnc I/O

- 缓存

  LRU/ALL



拥抱社区

- topN
- Minibatch
- ROW_NUMBER 高效去重
- Local-Global 数据热点
- MATCH_RECOGNIZE　cep支持

Storm迁移Flink

Storm集群问题

- 编程模型简单，开发成本高
- 依赖zookeeper，性能瓶颈
- 消息保障机制导致吞吐不高
- 重要业务独立集群，集群数量多，运维成本高

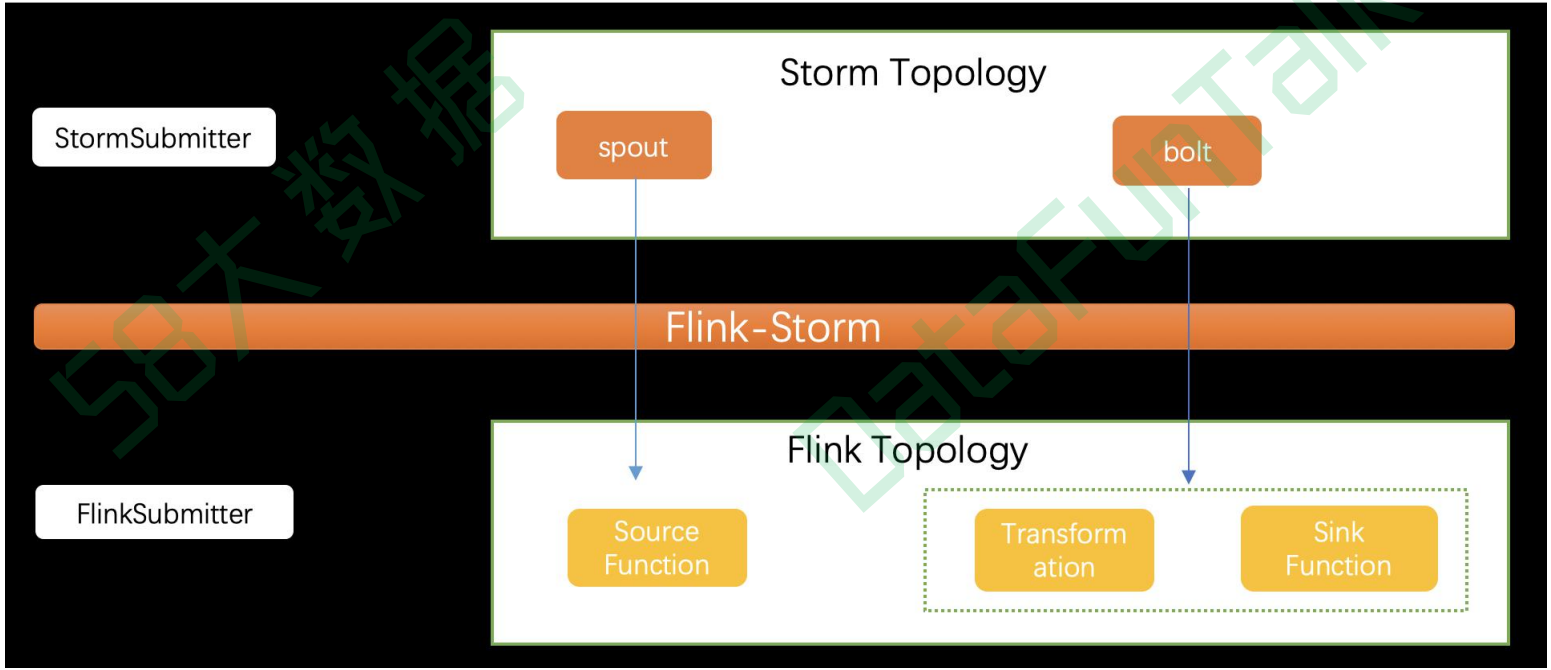
Flink的优势

- flink on yarn解决集群管理资源调度
- 采用nodelabel和cgroup机制，无需多集群
- 状态管理，支持sql，多样化窗口
- Exactly once ，高吞吐，低延迟

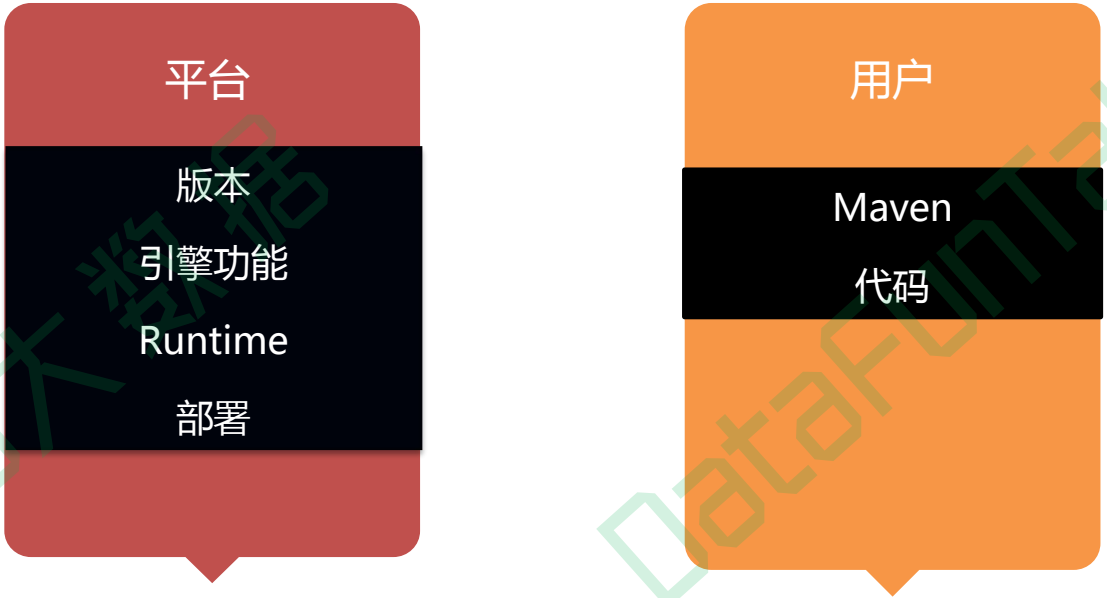
Storm迁移Flink

难点

- 用户任务多，重新改造工作量大

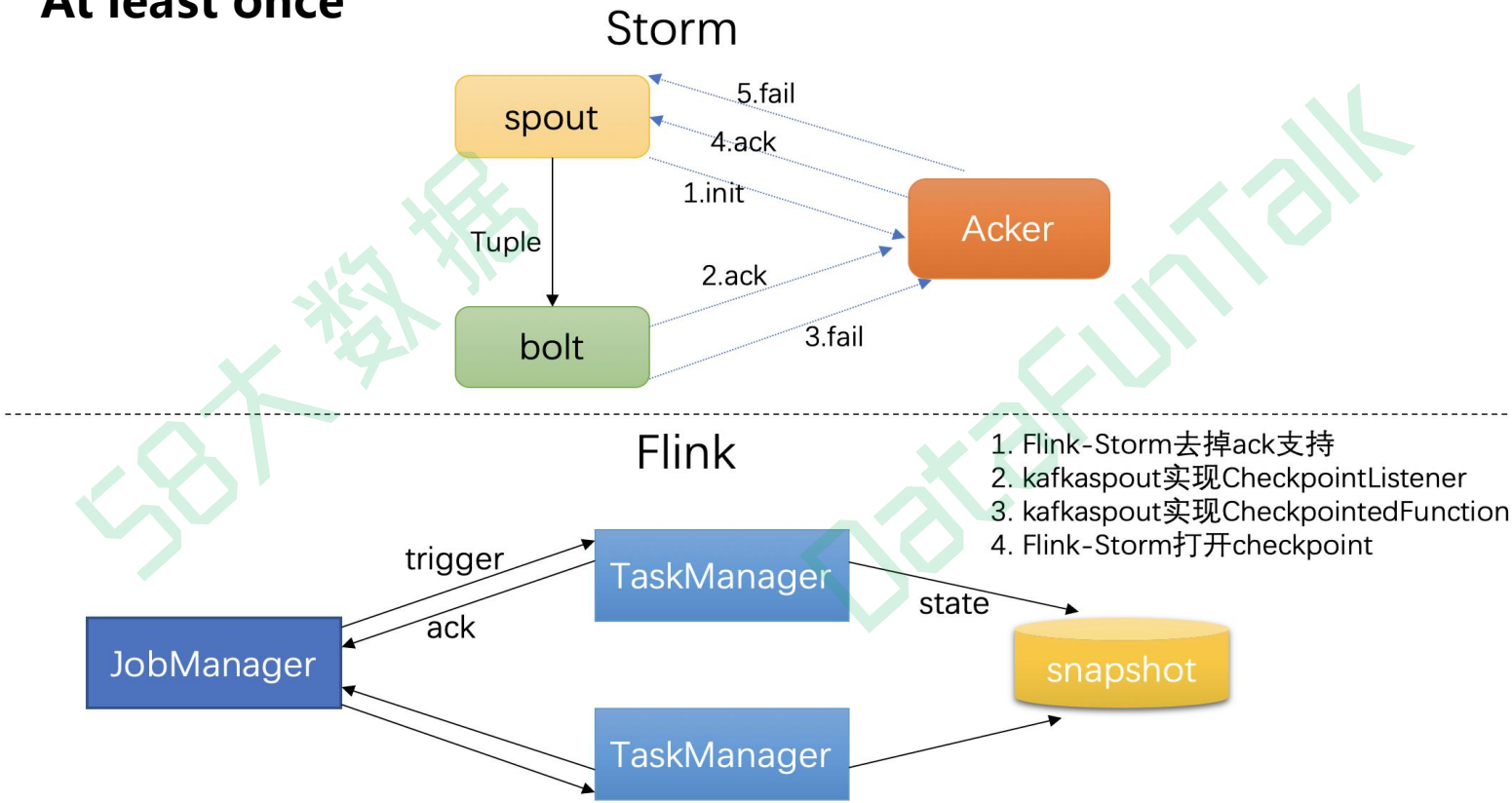


兼容性



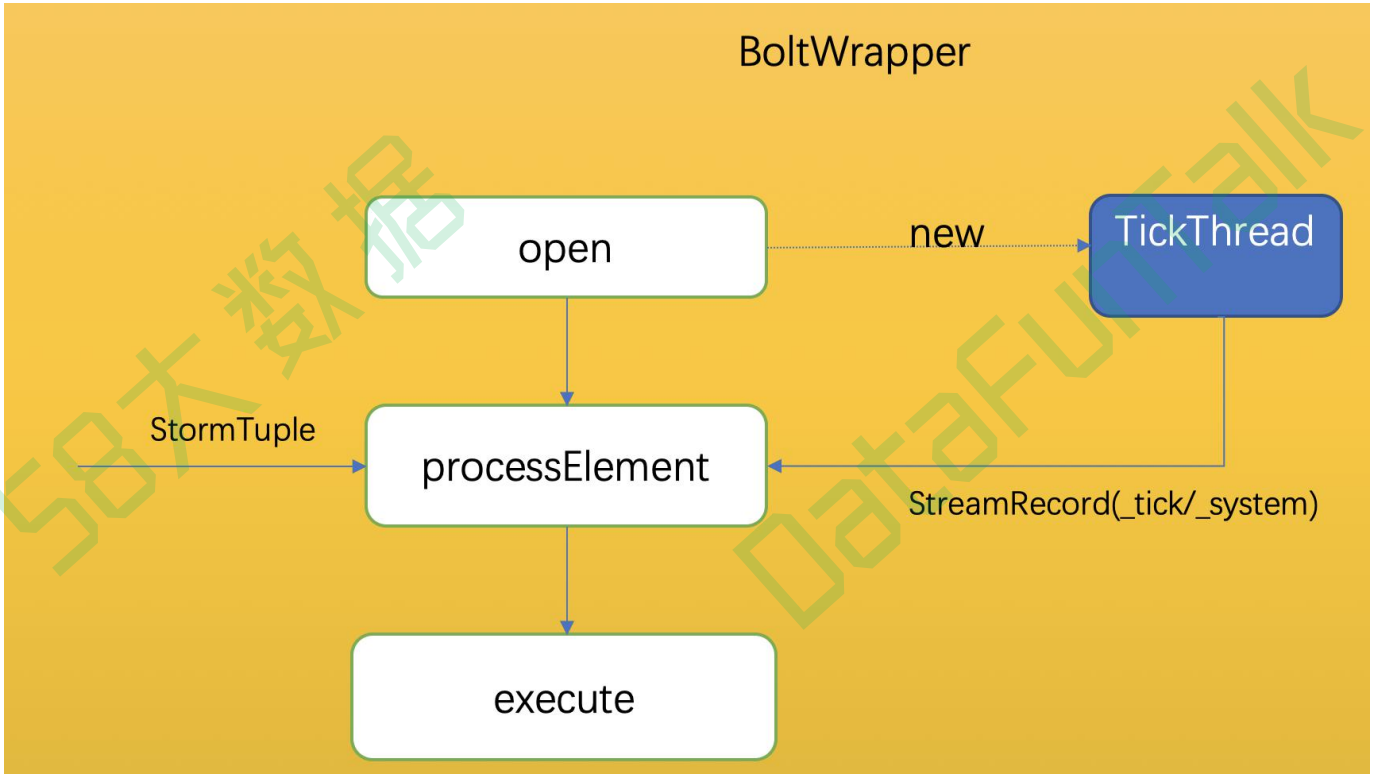
消息保障机制

At least once

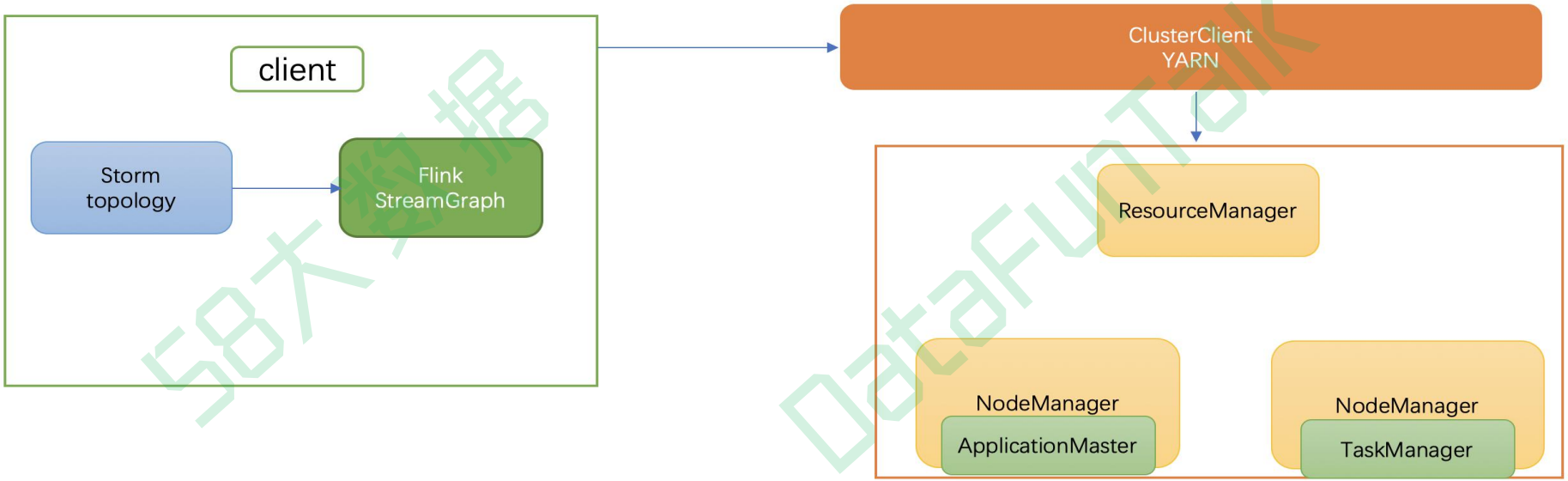


定时器机制

窗口函数


BoltWrapper
open
new
TickThread
StormTuple
processElement
StreamRecord(_tick/_system)
execute

Storm on Yarn



任务部署

- Flink on Yarn
- Perjob模式

```
bin/flink-storm-submit    -hadoop_user_name flink-user    --任务用户
                          -name flink-storm-yarn          --任务名
                          -queue root.online.test         --提交队列
                          -addjar test.jar                --依赖外部jar
                          -jar  flink-storm-test.jar      --任务jar
                          -c com.flink.storm.Test         --任务主类
                          1 a b                           --主类参数
```

用户代码

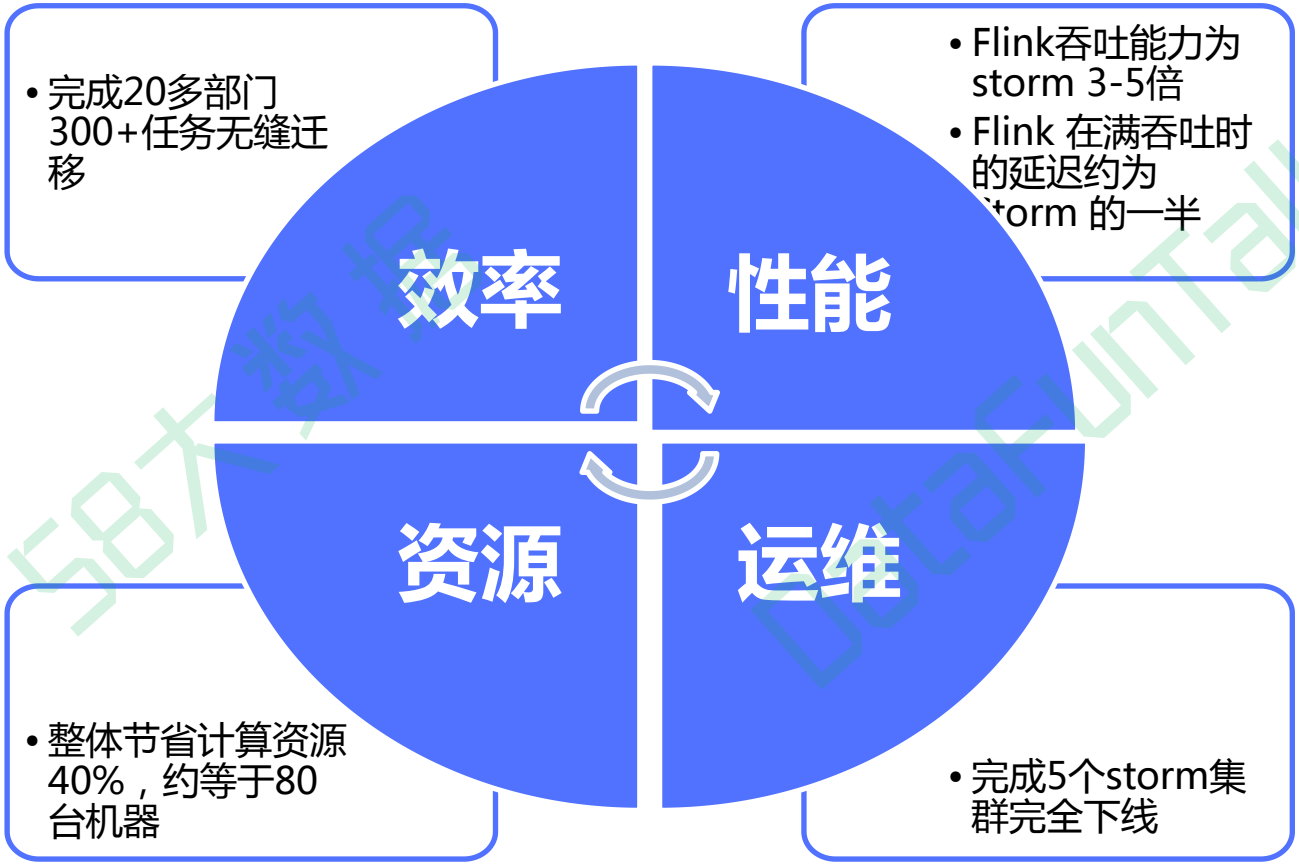
- 支持Local/Cluster两种方式
- 无需逻辑代码调整

```
<dependency>
    <groupId>com.bj58</groupId>
    <artifactId>flink-storm-095</artifactId>
    <version>1.6.0</version>
</dependency>
```

```
if(Boolean.valueOf(mode)){
    LocalCluster cluster = new LocalCluster();
    cluster.submitTopology(prop.getProperty(STORM_TOP_NAME),conf,builder.createTopology());
}else {
    StormSubmitter.submitTopologyWithProgressBar(prop.getProperty(STORM_TOP_NAME), conf, builder.createTopology());
}

if(Boolean.valueOf(mode)){
    FlinkLocalCluster cluster = new FlinkLocalCluster();
    cluster.submitTopology(prop.getProperty(STORM_TOP_NAME),conf,FlinkTopology.createTopology(builder));
}else {
    FlinkSubmitter.submitTopology(args,prop.getProperty(STORM_TOP_NAME), conf, FlinkTopology.createTopology(builder));
}
```

Storm迁移Flink

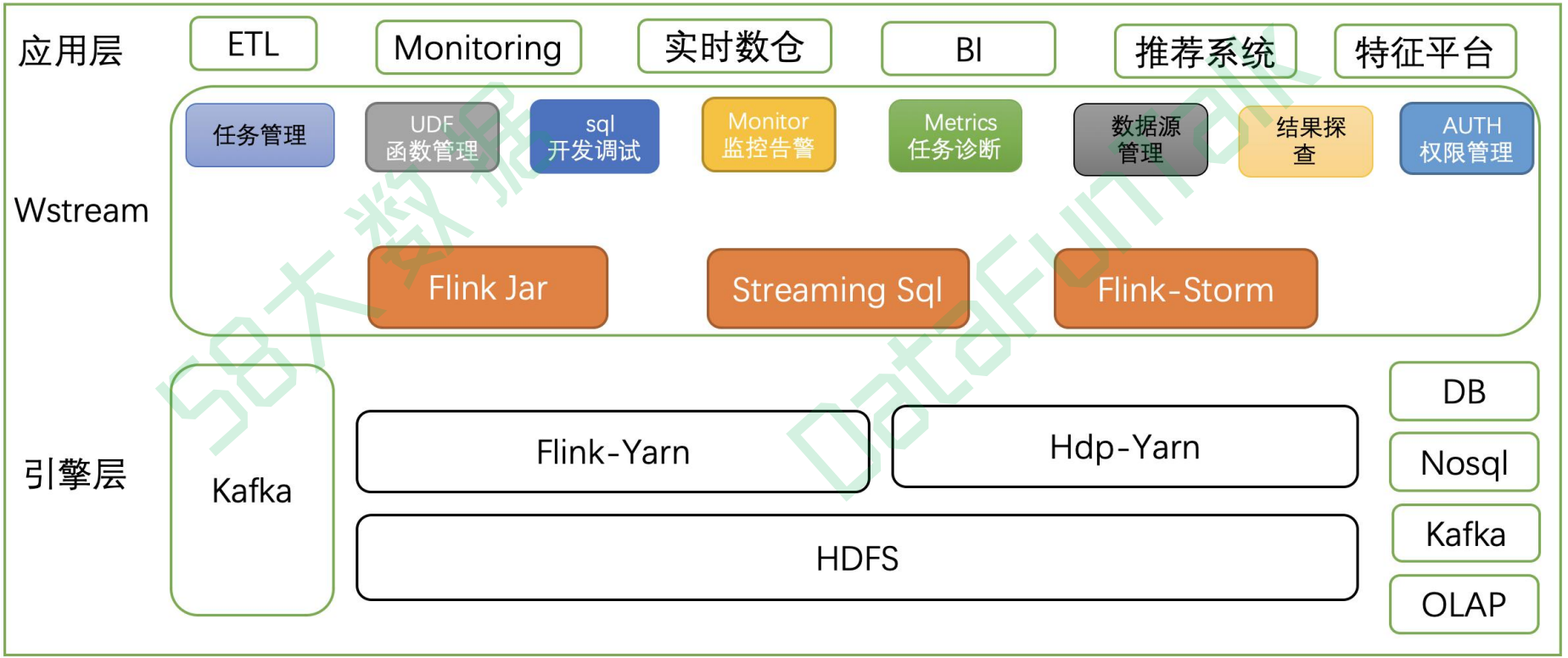


03
Wstream

一站式实时计算平台

概览

通过平台化提高Flink任务管理效率



任务管理

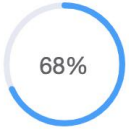
任务生命周期线上化

集群近期 QPS

集群处理总数(条/天) 4,539亿

运行中任务数

493

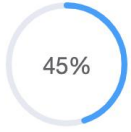


总任务数量 725

可用 CPU 数

617(滨海)

4,300(武清)

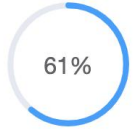


CPU总核数 5,900(滨海) 4,959(武清)

可用内存(G)

9,002(滨海)

17,796(武清)



总内存(G) 23,600(滨海) 19,836(武清)

任务管理

新建

仅我的任务 类型: 请选择任务类型 状态: 请选择任务状态 搜索...

任务名	类型	用户	hadoop账号	日志	状态	更新日期	操作
realtime_ods_app_action_hive	Flink Jar	huili01	hdp_lbg_ecdata_dw	application_1554972497203_12605	RUNNING	2020-02-21 10:57:06	
nginx_middleware_monitor_topology_test	Flink Jar	chenyiping	hdp_teu_op	application_1575281182327_3542	RUNNING	2020-02-21 10:49:23	
HBaseSink10	Flink Jar	daisongchen	hdp_ubu_tech_wei	application_1575281182327_3528	RUNNING	2020-02-21 00:25:57	

状态管理

checkpoint&savepoint

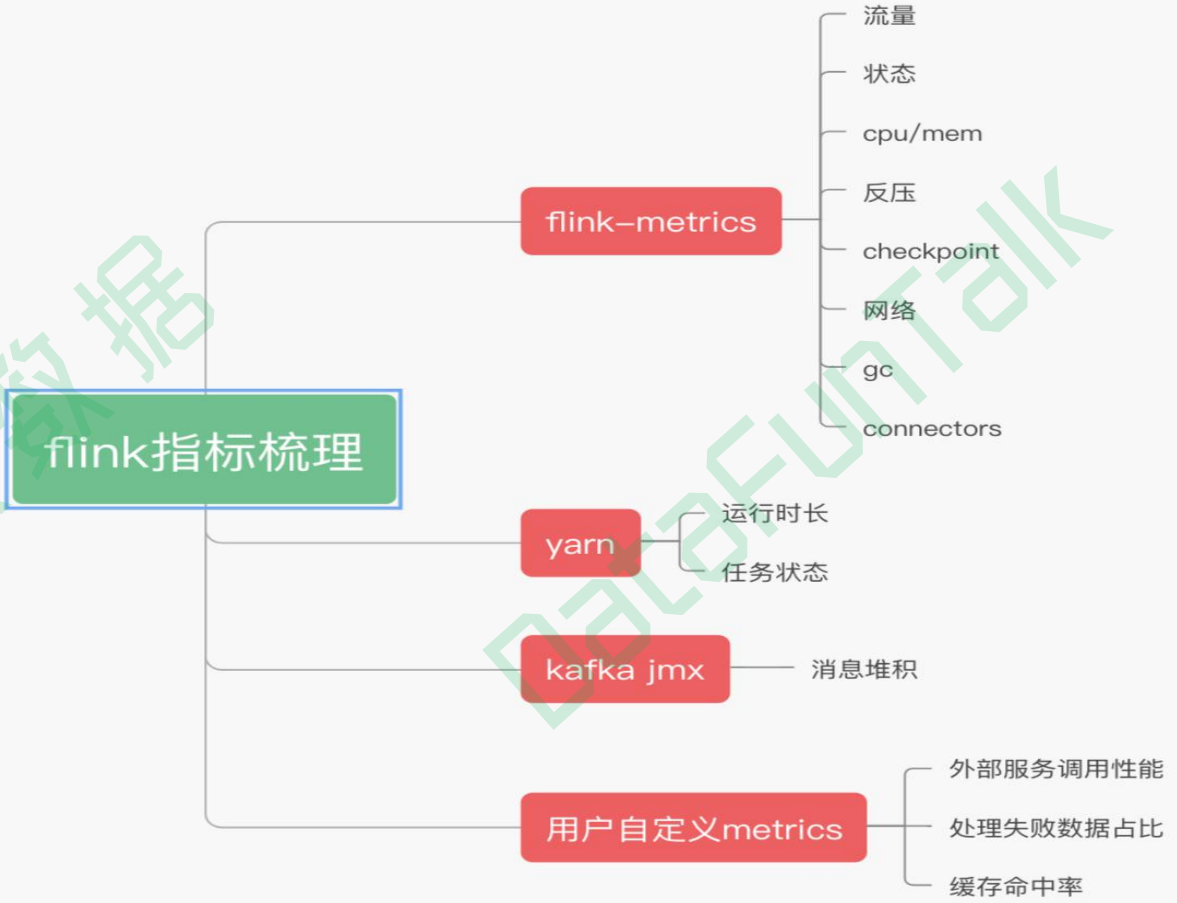
数据表管理

- 配置化管理
- 语法校验

监控指标

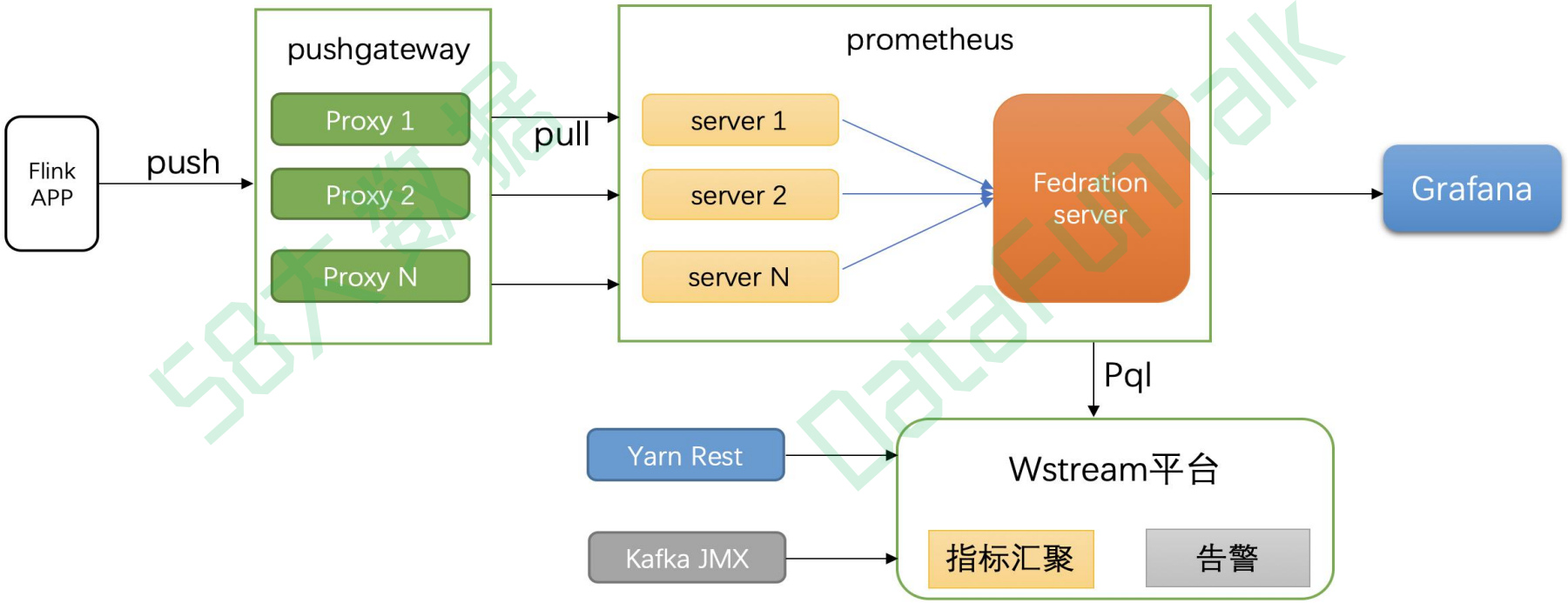
实时任务核心指标

- 稳定性
- 性能
- 业务逻辑



监控架构

原生支持Prometheus



监控告警

核心指标实时告警

基本信息　任务信息　运行参数　告警

任务异常告警

QPS 波动幅度超过 0 % 时告警

CheckPoint 失败 − 0 + 次告警

Flink Task 重试 − 0 + 次后告警

任务延迟(消费堆积)告警

告警策略

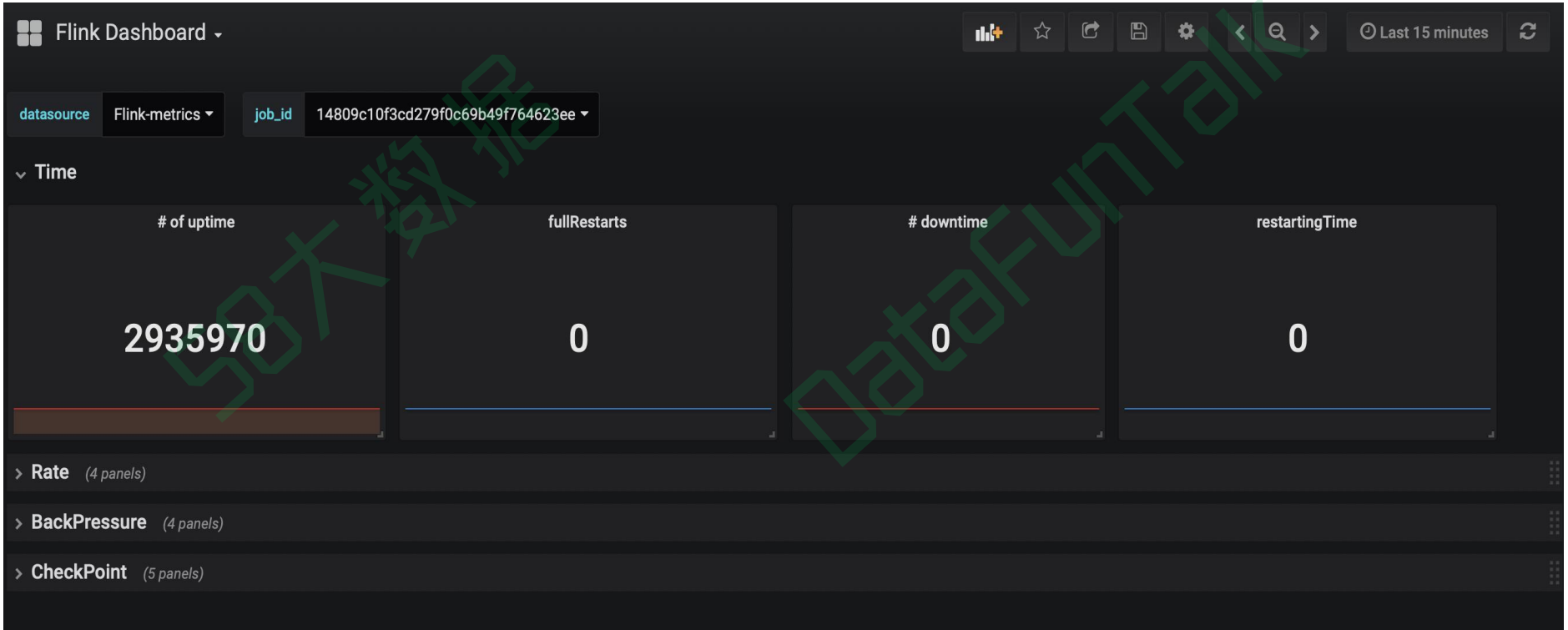
* Kafka 版本 0.8

* 堆积数 1

* 连续次数 3

监控告警

- 全量metrics
- 自定义dashboard
- 定制化告警



THANKS

欢迎关注
【58技术公众号】
更多技术文章等你～
ID：architects_58