FLINK FORWARD # ASIA 2022 ONLINE | 实时即未来

# FLINK 在蚂蚁大规模金融场景的平台建设

李志刚 | 高级技术专家

01
主要挑战

02
架构方案

03
核心技术介绍

04
未来规划

# 01 主要挑战

# 金融场景业务特点介绍

**时效性**

- 安全风控等业务要求时延在秒级
- 业务逻辑经常变更，不能影响时效性
- 上下游链路复杂

**正确性**

- 数据必须保证100%正确
- 实时业务开发需要数据核对
- 数据有问题时，需要进行调试

**稳定性**

- 为提升资源利用率，业务混布
- K8s API server等组件问题，影响pod申请
- 大pod申请时间过长，影响业务稳定性

# 基本情况

78W Core

1.2w+ Job

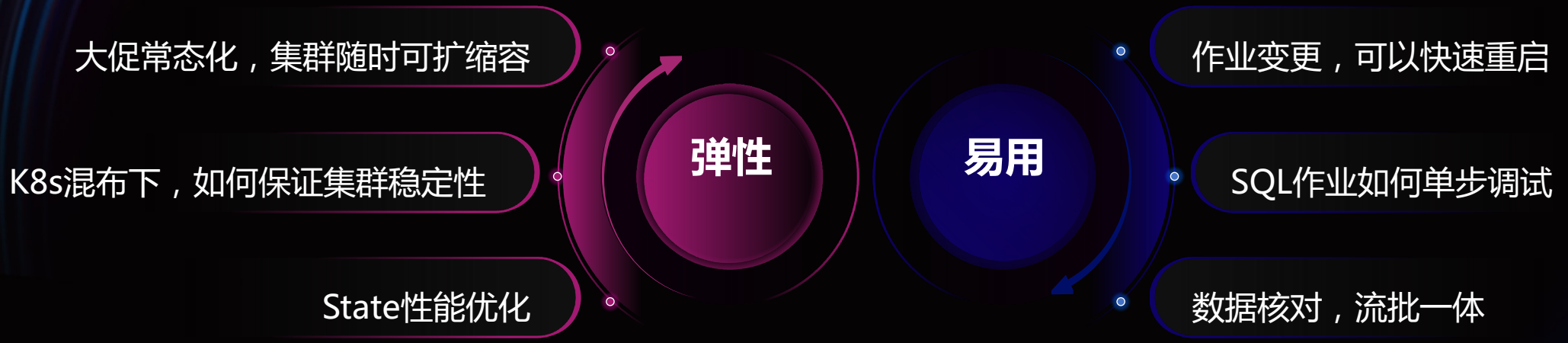
100%云原生集群

每年支持15+次大促活动

每年支持10+次重大活动

（618、双11、双12、北京、五福等）

# 主要挑战


大促常态化，集群随时可扩缩容
K8s混布下，如何保证集群稳定性
State性能优化
弹性
易用
作业变更，可以快速重启
SQL作业如何单步调试
数据核对，流批一体

# 如何应对挑战

**对于易用性方面**

- 全面改造平台，热启动技术解决启动慢的问题
- 调试SQL代码像在IDEA调试JAVA代码一样，解决排查数据问题难题
- 为解决批和流两套引擎数据核对的难点，在蚂蚁内部提出基于Flink的流批一体开发平台

**对于弹性方面**

- 基于k8s全面进行混布
- 对Flink原生的k8s模式进行改造，可以避免由于k8s的问题导致影响实时业务稳定性

# 02 构架方案

# 蚂蚁实时计算平台框架

平台构架

一套代码：一套代码生成实时、离线两个任务（基于Mix元表 + 虚拟列）

数据源　一套标准SQL　单步调试　热启动

核心技术

内存优化　窗口优化　智能诊断

状态存储　智能化调参　湖仓列存储

Flink runtime　流批一体

工作流调度引擎

转实例　实例运行

破线预警　运行报告

集群模式

K8S

# 03 核心技术介绍

# 热启动技术

## 为什么需要热启动？

- 变更时刻在发生，重启整个作业代价极大
- 从作业提交到运行，需要几分钟，整个服务停止


修改参数需要重启整个作业

修改并发

修改内存

修改UDF

作业规模大

流量突变

影响用户体验

整个启动流程4分钟

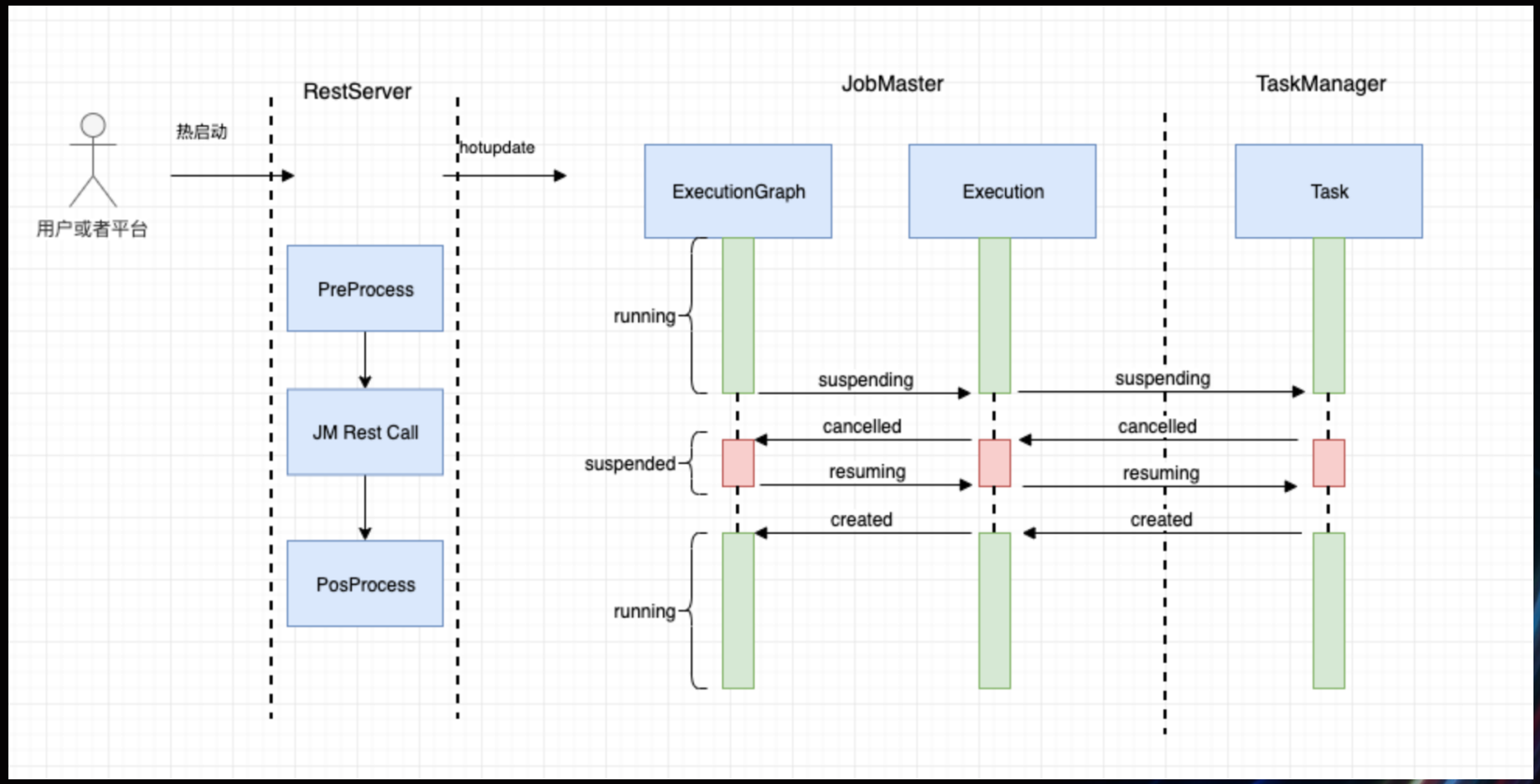
提升资源利用率

用户抱怨Flink产品体验

重启期间无服务

# 热启动技术

解决方案
1. 用户触发
2. 前置处理
3. 请求调用和引擎核心处理
4. 资源释放
5. 后置处理



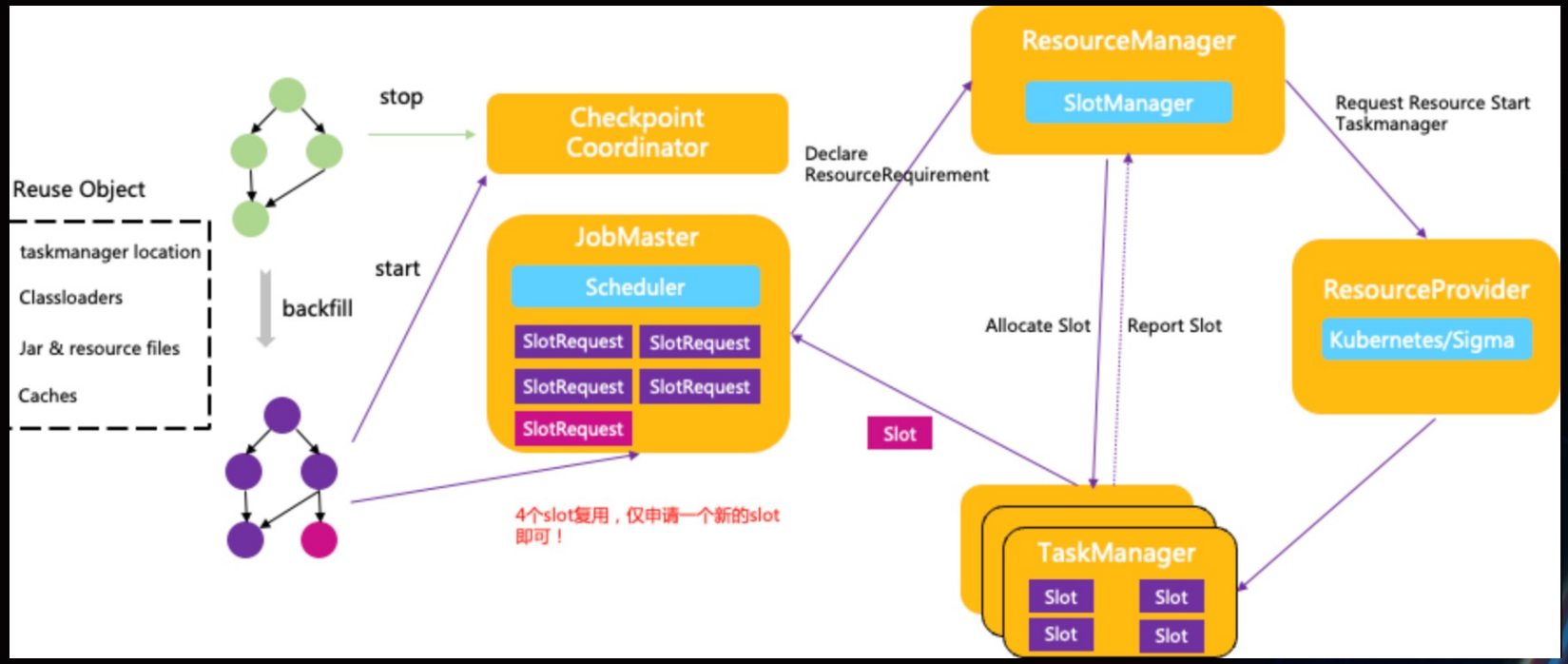
# 热启动技术

处理流程

1、将携带过来的新 jobgraph和旧的 jobgraph进行merge，将旧的 jobgraph中可以复用的数据进行回填到新的 jobgraph中。

2、挂起旧的 jobgraph，挂起的过程中会暂停 checkpoint coordinator，并 cancel 掉作业的task。

3、重新调度 jobgraph，挂起结束后将回填更新得到的新 jobgraph 重新调度启动起来。由于保证了状态兼容，checkpoint的数据也会进行重放。

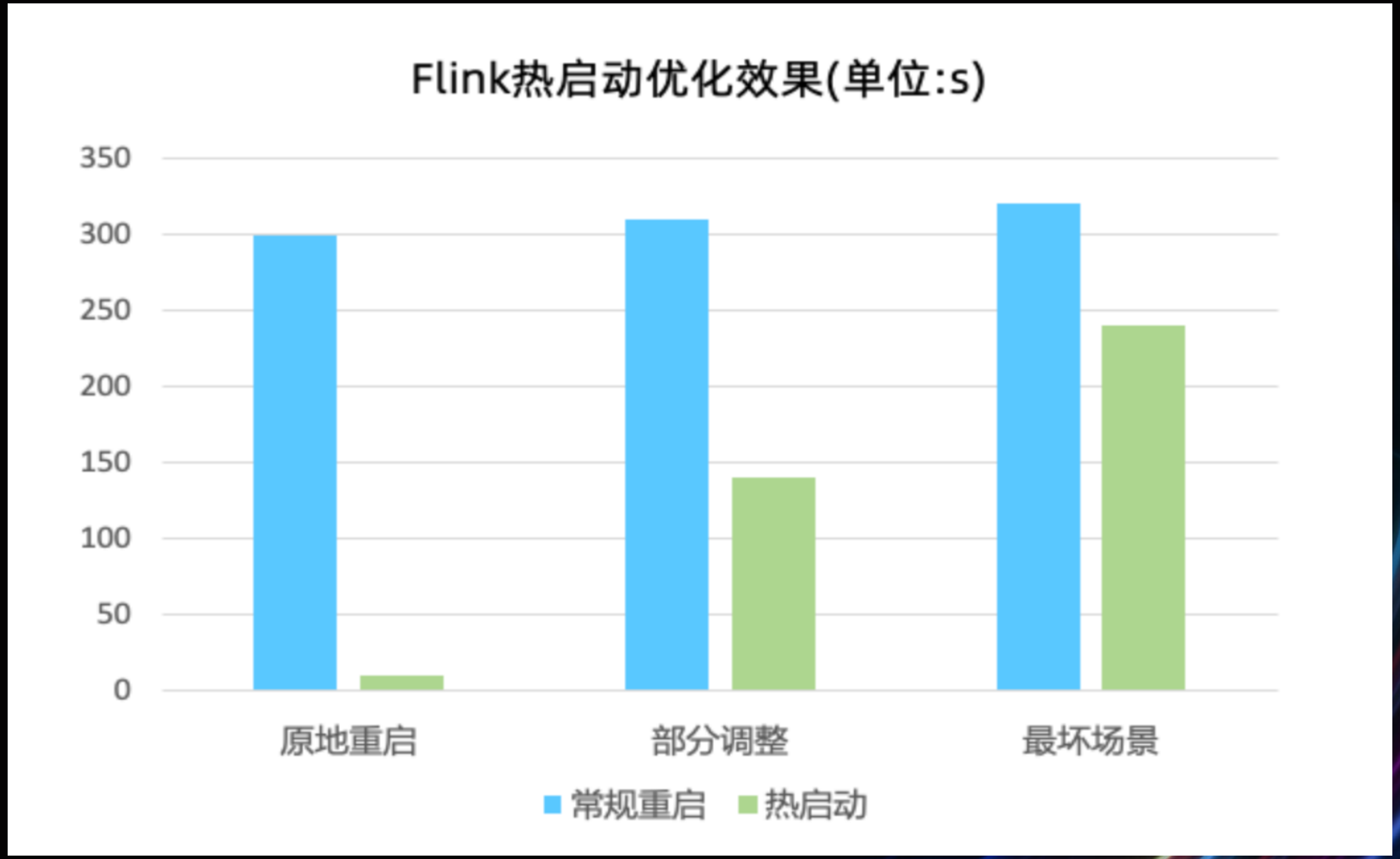


# 热启动技术

经过对近一个月几万次作业启动时间统计

1、大部分作业可以节省90%以上时间

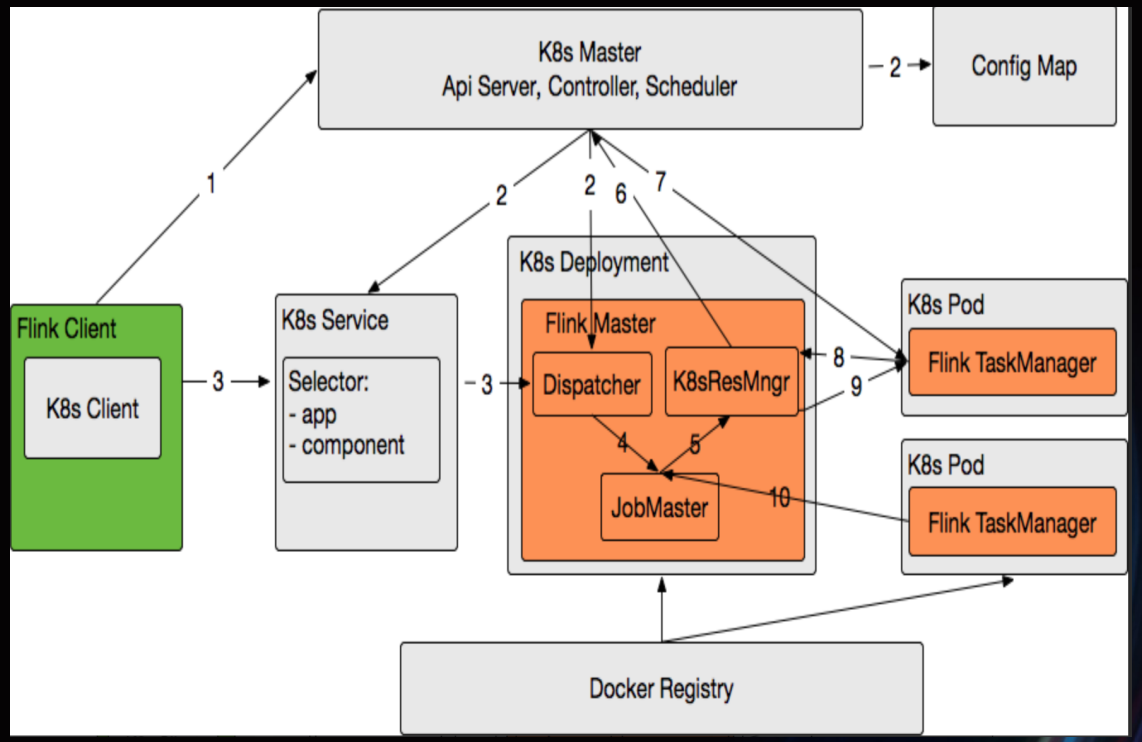
2、用户操作体验大幅度提升



# K8S集群模式

## 为什么需要k8s集群模式？

- 蚂蚁服务都是混布在同一个k8s集群，k8s升级会影响作业发布和运维
- 一个作业申请超过50个pod非常慢，经常超过5分钟
- 申请大pod 32核64GB的经常失败
- 业务突然增加时，可能无法按需扩容
- K8s API server性能瓶颈，大量创建pod非常慢，偶尔出现超时

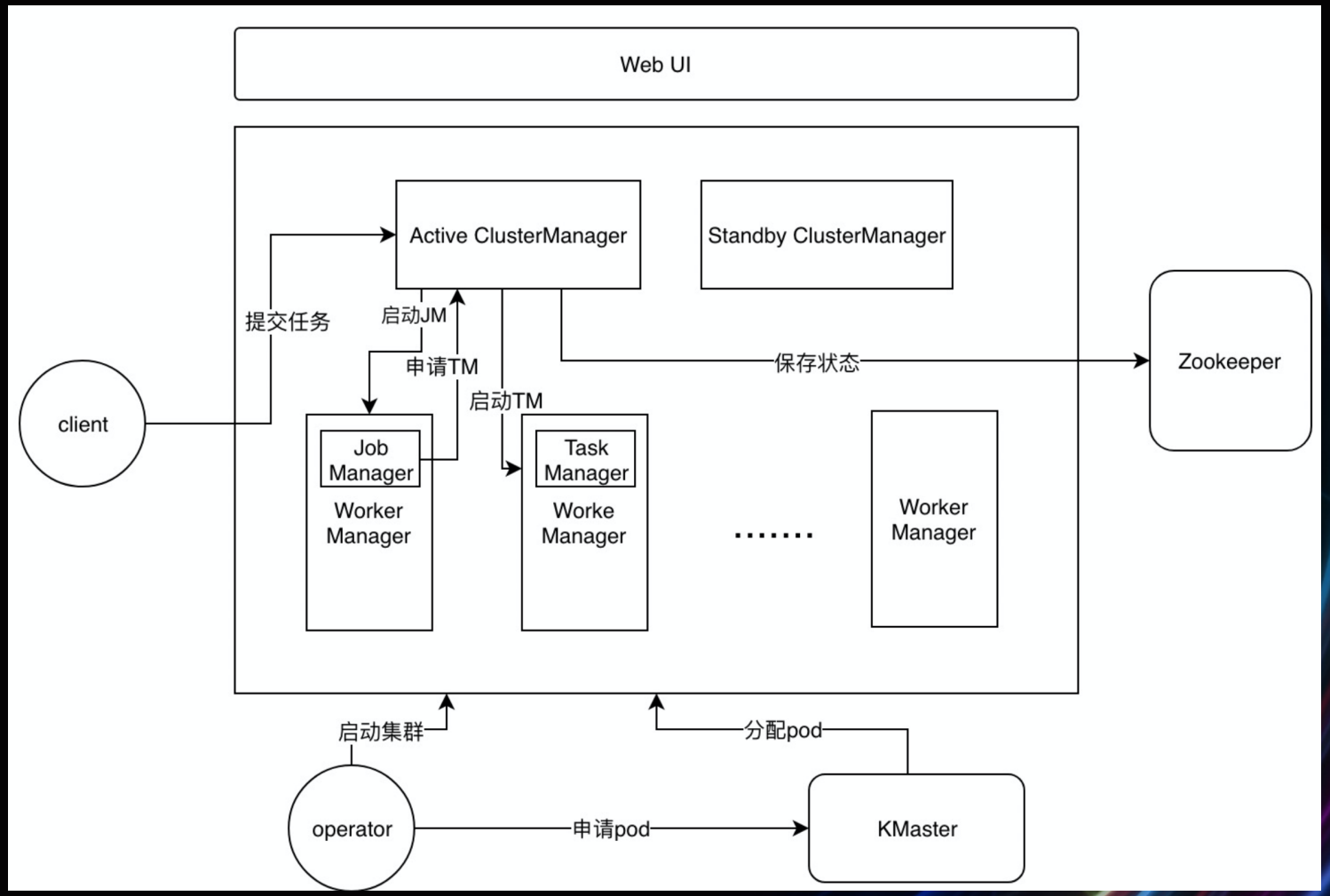



Flink on k8s原生模式

# K8S集群模式

## 系统组件

- Operator：负责和K8s交互，申请pods，并启动Flink集群，在集群运行的过程中，对集群扩缩容。
- Client：通过client提交任务到Flink集群，获取任务的运行状态和停止任务。
- ClusterManager：资源管理中心和应用管理中心，负责处理应用的资源请求，把资源实例上的cpu/memory等资源按调度策略分配给应用。
- WorkerManager：管理单台机器或者容器上的资源，响应资源管理中心的调度，启动和管理应用进程。



# K8S集群模式



## 采用Flink k8s集群模式与原生Flink on k8s相比

- 由于K8S的问题，导致Flink作业出现异常的情况减少95%
- 作业提交时间平均减少50%，操作更加流畅，提升了用户体验
- 通过自定义的调度策略，集群资源的利用提升了5%

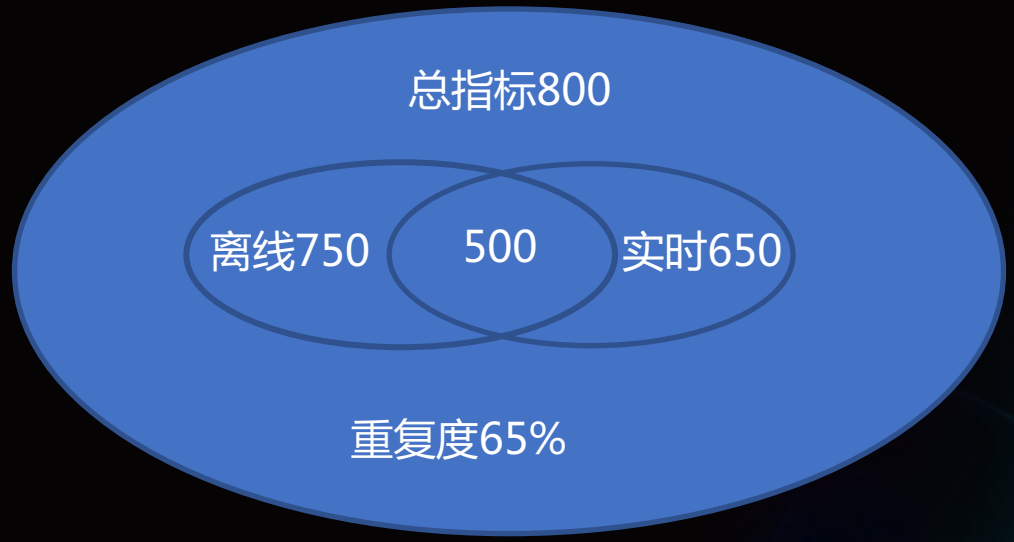
| | K8s 集群模式 | On k8s原生模式 |
|---|---|---|
| K8s组件导致问题 | 1 | 12 |
| 作业启动时间(s) | 50秒 | 100秒以上 |
| 集群资源归一化利用率 | 30% | 25% |

# 流批一体技术

## 当前开发模式的痛点

- 部分业务需要同时开发流作业和批作业，浪费人力
- 由于流和批作业是使用不同引擎开发，导致比对数据困难
- 用户需要掌握两套流批开发语言，而且还要了解两者的细节区别
- 由于流批使用不同的引擎，不同的开发团队，资源难于在不同引擎之间共享

蚂蚁某活动



# 流批一体技术

- 从平台，API到集群都统一，用户只需关注具体业务逻辑
- 流批作业运行在同一个集群模式的集群，优化调度策略提升资源利用率
- Remote shuffle service支持大规模数据

流批一体构架图

# 流批一体技术

流批统一模式下，用户该如何进行SQL作业的调试

- 需要像调试JAVA代码一样，可以看到算子运行中间结果
- 可以运行大数据量，减少OOM风险
- 可以支持大量用户，同时进行调试

**方案一**

1. 在所有算子入口处增加代码trace代码
2. 影响代码优雅，热点

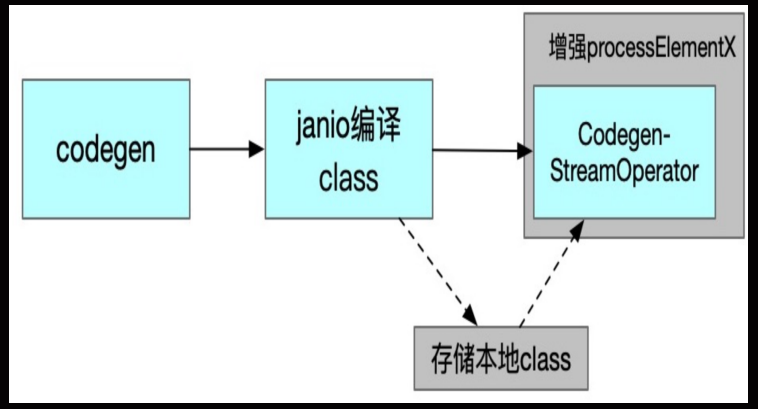
**方案二**

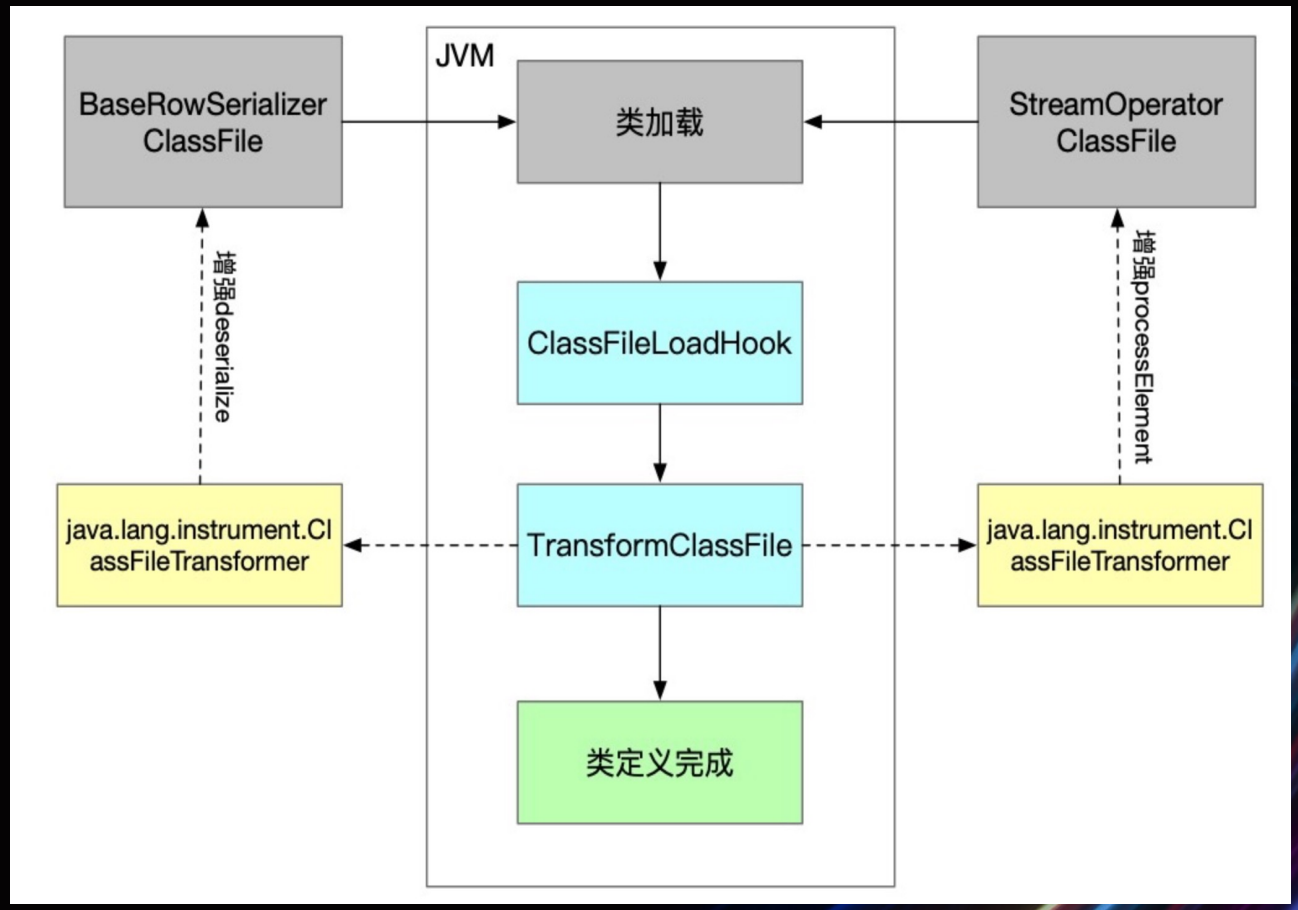
字节码增强

# 流批一体技术

字节码增强—通过JAVA agent来修改类

- 在processElement执行前，通过入参拿到数据，
  写到文件
- 增强BaseRowSerializer反序列方法
- 增加SimpleCompiler



# 04 未来规划

# 未来规划

- 优化Flink批性能、支持全向量化计算
- 基于机器学习的自动化调优
- 发展基于Flink的湖仓技术
- 云化环境下智能化诊断
- 流批混合部署下分时调度，提升利用率

FLINK FORWARD # ASIA 2022 ONLINE | 实时即未来

# THANK YOU

谢 谢 观 看