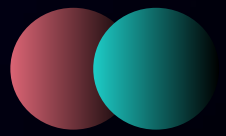
Flink Kubernetes Operator
Flink在云原生的下一站

Flink Kubernetes Operator next step in Cloud Native

陈政羽 | Apache Flink / StreamPark
& Amoro 社区贡献者



01 发展历程

Flink 为什么需要云原生

云原生架构普及

能够提供更高的可扩展性、容错性和弹性，为大数据处理提供更好的支持。未来越来越多的企业和组织将采用云原生架构来构建他们的大数据平台。

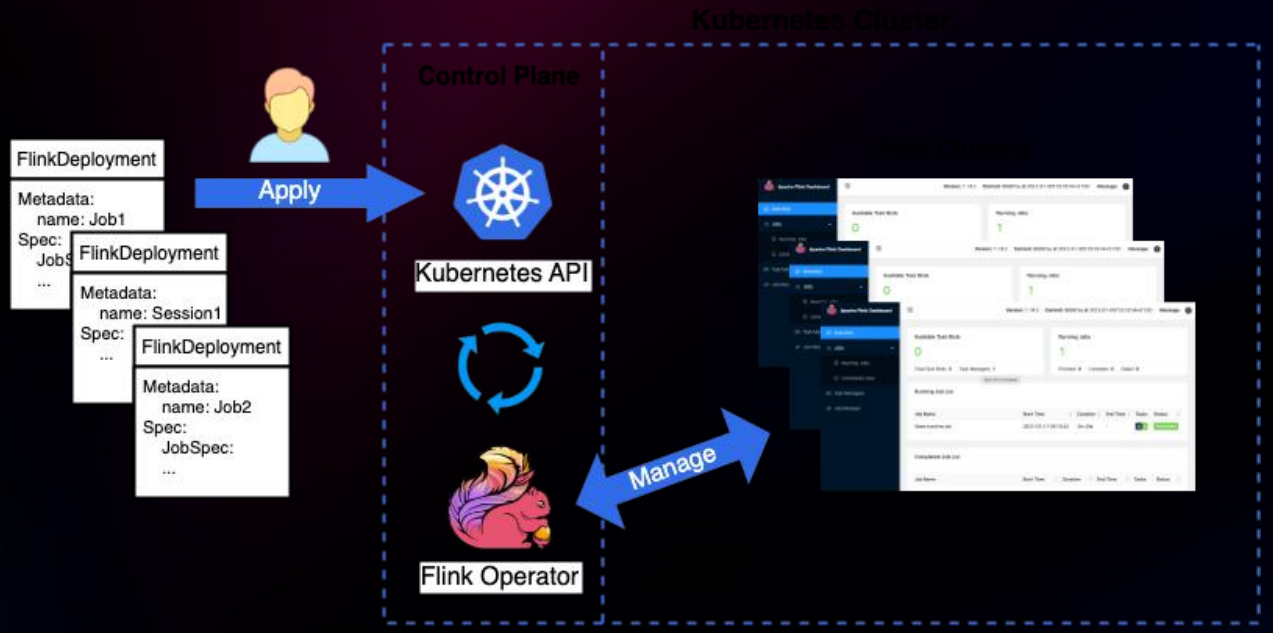
容器化技术完善

容器化技术可以实现更高效的资源利用和快速的部署，同时也降低了维护和管理的成本。未来大数据平台将倾向于采用容器化技术来部署和管理各种大数据组件和应用。

拥抱云原生生态

K8s 拥有几乎最活跃的生态圈，它通过提供标准化的接口定义，促进了各个层次的生态发展，无论是基础运维设施、上层应用管理还是底层的网络、存储等管理中都有非常多的可选方案

Flink Kubernetes Operator 简介



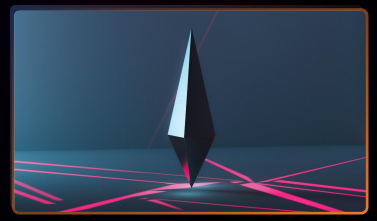
Flink Kubernetes Operator 是一款云原生应用，其主要功能是管理、检测和监控 Flink 作业的部署状态。

在没有 Operator 的情况下，用户需要对 Flink 的部署流程有一定的了解才能完成完整的业务开发，包括启动集群、部署作业和升级作业。当在这些过程中出现问题时，用户必须对 Flink 有相对深入的了解，才能解决这些繁琐的操作。Operator 的出现正是为了解放用户的这些繁琐流程，其主要目标是实现这些流程的自动化，让用户无需关心内部细节即可完成 Flink 作业的部署。

Flink Operator 云原生发展之路

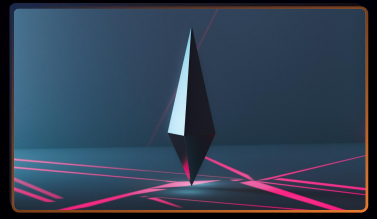
第一阶段

Flink自身支持基础的云原生Flink应用部署，有第三方的Operator支持，从而引出Flink官方Operator的想法



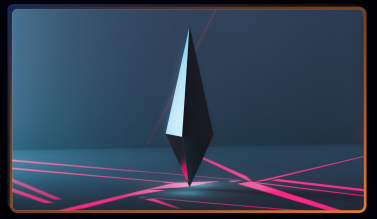
第二阶段

基本的部署和监控 支持 Flink Application 和 Session 部署最低限度部署完整的日志记录和指标集成



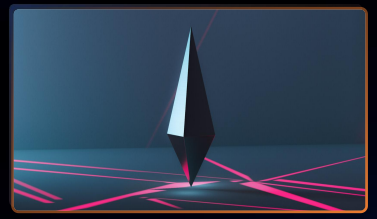
第三阶段

提供第一个生产就绪 API 版本，包括对保存点、检查点的相关管理



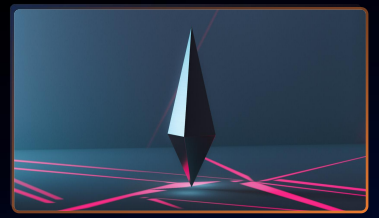
第四阶段

改进Flink作业升级流程、存活探测、事件等，使得Flink部署体验更加流畅



第五阶段

- 支持AutoScaler自动伸缩
  改进AutoScaler伸缩算法，使得缩放获得更好的体验



02 核心功能

一键安装，快速部署作业

自动部署

作业全自动运维

作业SRE自动管控，减少人工干预

云原生化

基于K8S云原生环境

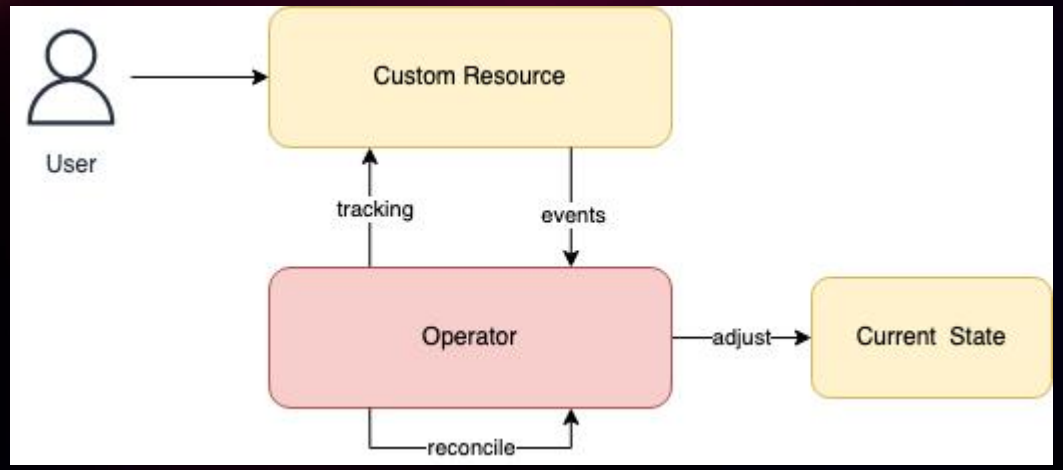
基于Native K8S 环境部署

自动调优

作业根据负载自动调整并行度

根据作业资源，潮汐特征调整并行度

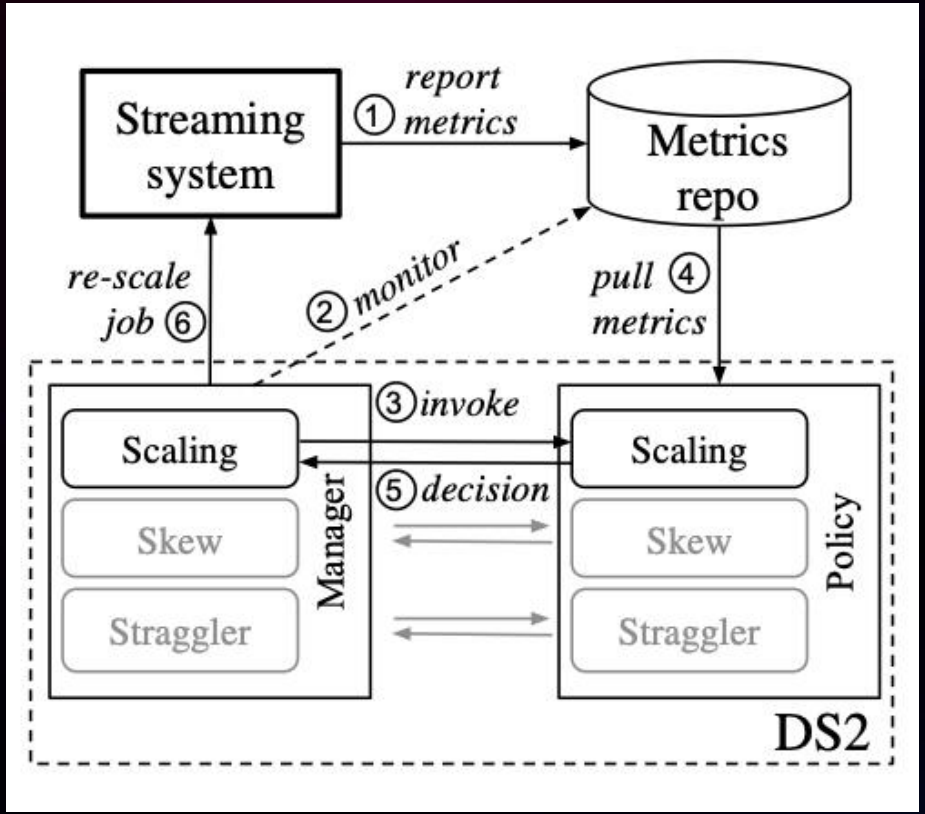
自动部署



用户可以使用 Kubernetes 命令行工具定义一个Flink作业资源描述（如右所示），Operator接收到后将持续跟踪作业部署状态和作业运行状态

```
apiVersion: flink.apache.org/v1beta1
kind: FlinkDeployment
metadata:
  namespace: default
  name: basic-example
spec:
  image: flink:1.17
  flinkVersion: v1_17
  flinkConfiguration:
    taskmanager.numberOfTaskSlots: "2"
  serviceAccount: flink
  jobManager:
    resource:
      memory: "2048m"
      cpu: 1
  taskManager:
    resource:
      memory: "2048m"
      cpu: 1
  job:
    jarURI:
local:///opt/flink/examples/streaming/StateMachin
eExample.jar
    parallelism: 2
    upgradeMode: stateless
    state: running
```

自动调优–功能介绍



社区用户诉求：作业自动调优的目标比较简单，即配置一份在保障作业无延迟的前提下，尽可能的提高作业整体的吞吐量，提高作业的资源利用率，让资源不再成为作业的瓶颈。

从 Flink Operator 1.4 版本开始，已经集成了 autoscaler 模块，为用户提供了基于云原生的自动调优功能。这样，用户可以更加方便地应用自动调优程序来优化流处理作业的性能和资源利用。

自动调优–原理介绍

$$\pi_i' = \lceil \frac{\sum_{j=1}^{n} o_j}{p_i} \cdot \pi_i \rceil$$

π_i' : New parallelism of job vertex i

π_i : Old parallelism for job vertex i

o_j: True Output rate of input j

p_i: True Processing rate of job vertex i

n: Number of inputs from upstream job vertices

基于算子利用率的缩放，我们可以通过以下公式来调整下游算子的并行度：

新并行度 = 当前并行度 * （当前利用率 / 目标利用率）

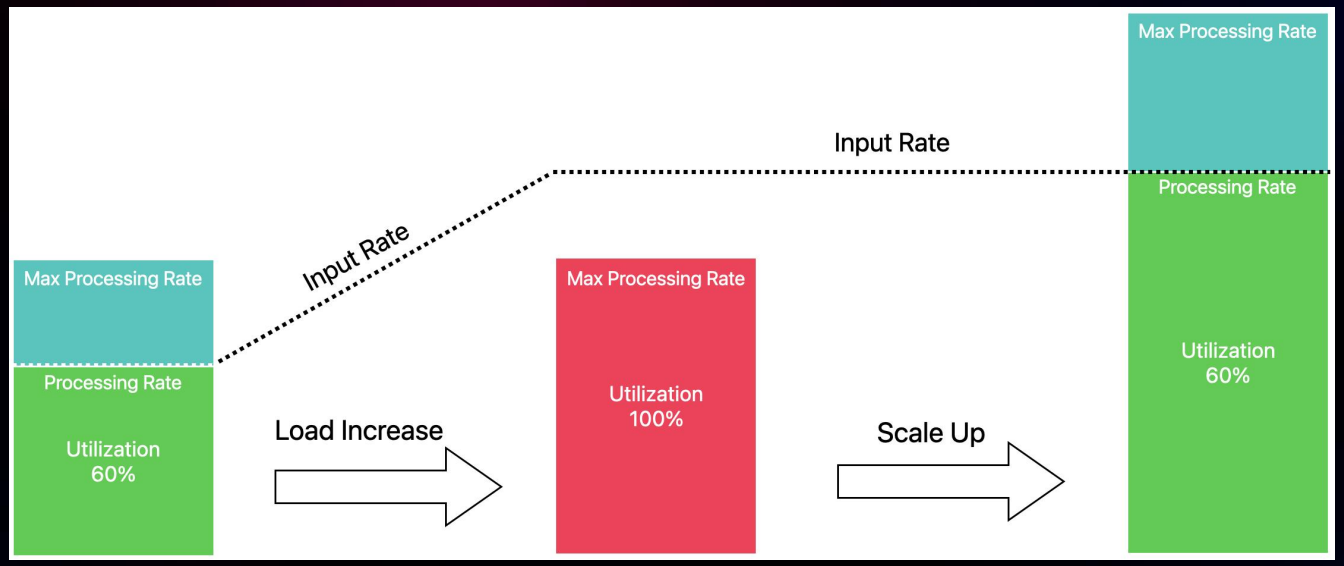
其中：

– 当前利用率表示当前数据源实际处理速率与其理论最大处理速率之间的比率。

– 目标利用率是您希望数据源达到的目标利用率，一般可以根据系统负载和性能要求进行设置。

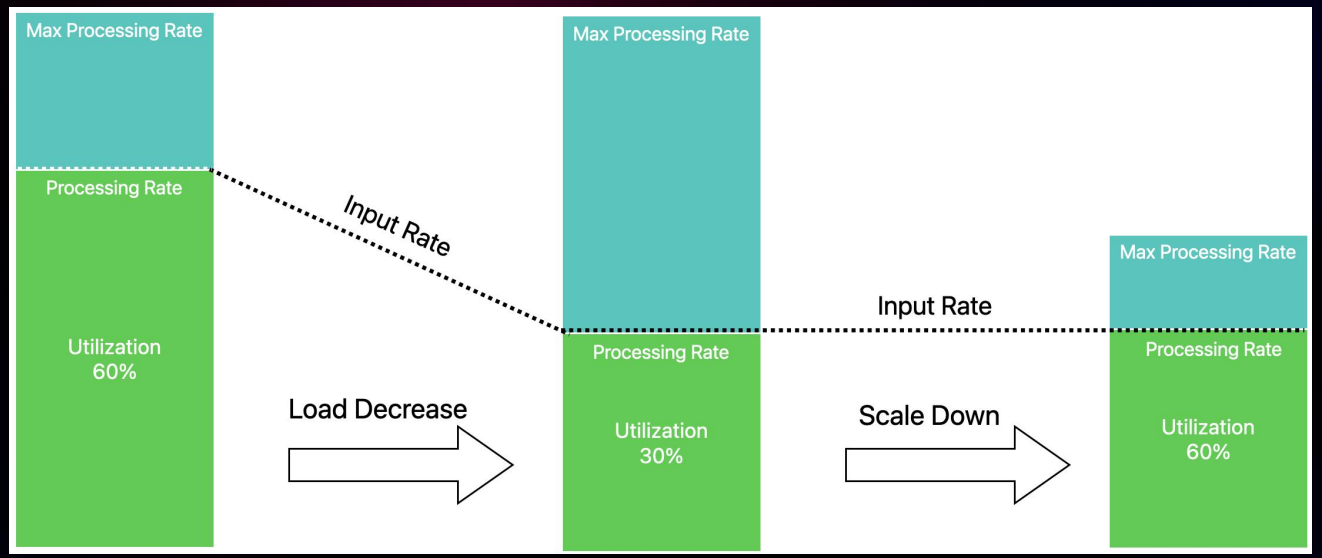
* ASF 2023 有相关主题更加详细深入解析其运作和运行原理

自动调优–扩容



如果用户配置某个作业的目标利用率百分比，例如，将所有作业顶点利用率保持在 60% – 80% 之间。autoscaler则会根据用户配置，使所有作业顶点的输出速率在目标利用率下与其所有下游运算符的输入速率相匹配。这就完成了一个扩容的过程了

自动调优–缩容



随着时间的推移，假设负载洪峰已经过去，数据量输入减少，autoscaler 会根据利用率去调整各个算子的并行度，以匹配当前的数据处理速率。

* 阿里云实时计算产品已经集成了自动调优功能，能更好体验云上弹性伸缩的云原生 Flink 功能

03 云原生实践

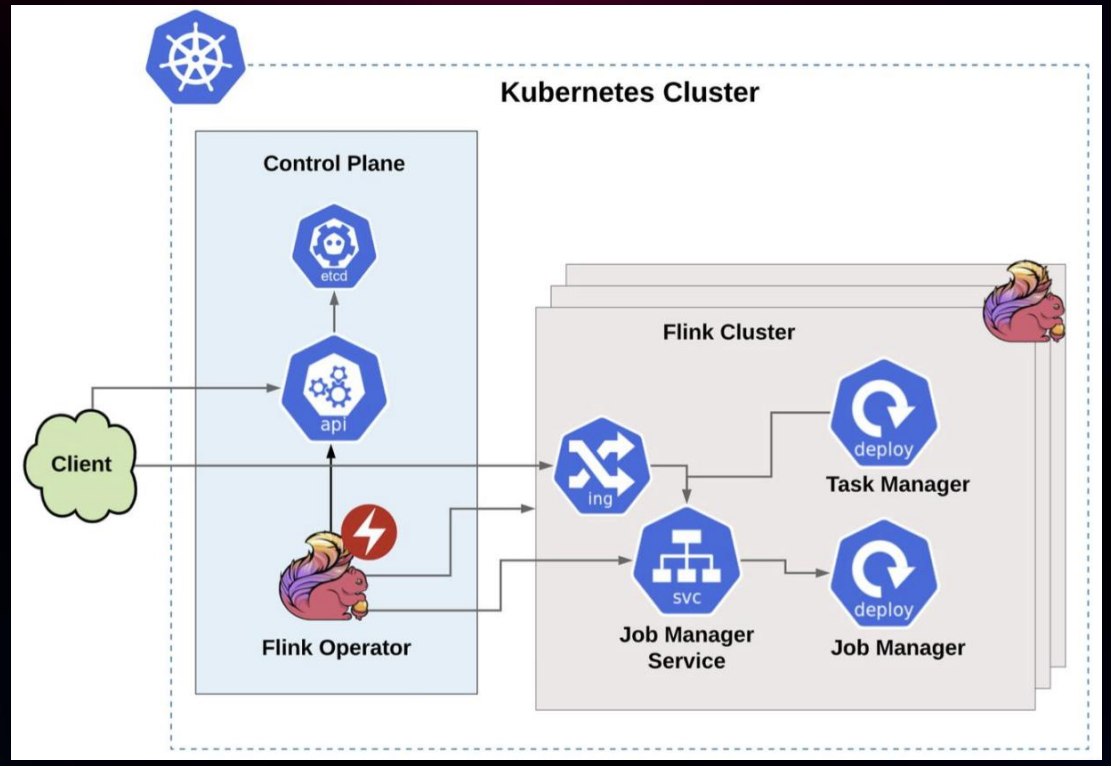
云原生时代关注点

“永远”在线

一键部署

降本增效

云原生SRE部署体验

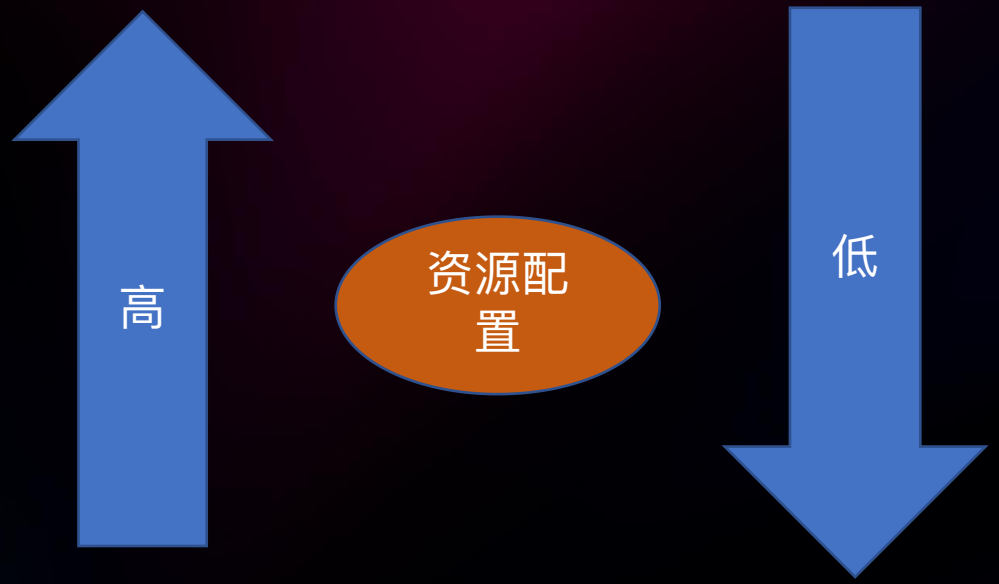


FLIP-212: Flink K8S Operator

- 运行、暂停和删除应用程序
- 有状态和无状态应用程序升级
- 触发和管理保存点
- 处理错误，回滚损坏的升级

* Apache StreamPark 正在集成Operator相关功能，欢迎下载组合体验

降本增效利器–AutoScaler

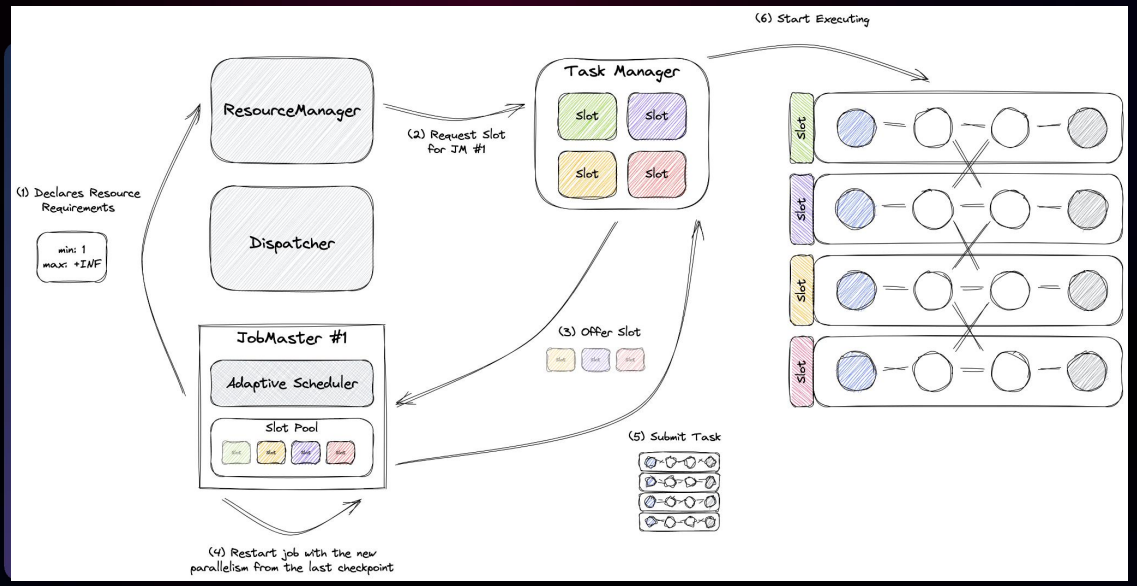


FLIP–271：AutoScaler

- 更好的集群资源利用率，更低的运营成本
- 自动调整算子并行度
- 自动适应作业流量的负载模式
- 提供详细的资源利用率指标

- 资源利用率低，成本过高，无法合理利用云原生技术
- 作业吞吐量低，作业产生背压，导致消息延迟过高，资源不稳定导致容易作业失败，多次重启

作业“永远”在线–不停机更新作业并行度



FLIP–291：Externalized Declarative Resource Management

- 程序断流，业务中断
- 容器重启时间过长，受到Pod销毁、创建时间影响
- 有状态作业 Reload 时间过长
- JobMaster作业重部署

FLINK–31316：Externalized Declarative Resource Management，在最近发布的 Flink1.18 欢迎体验

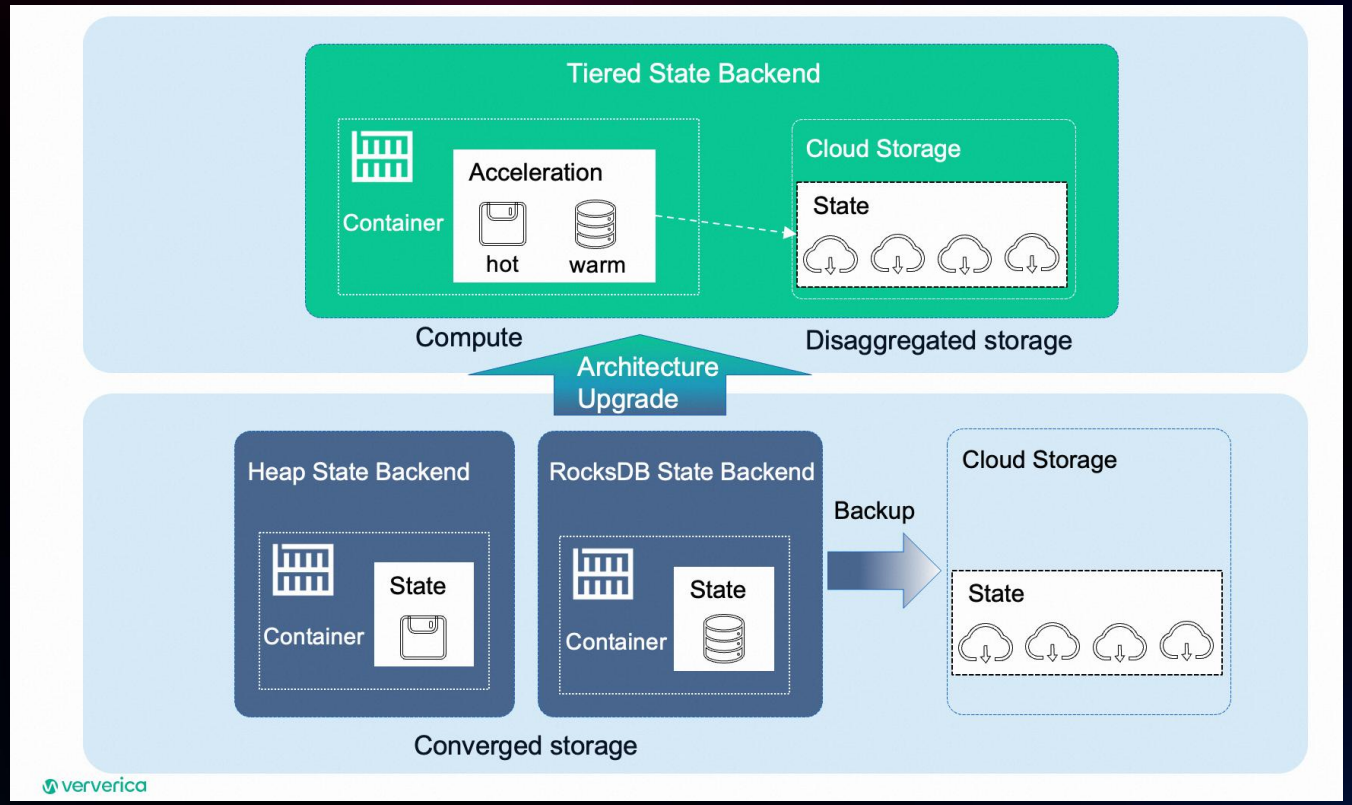
集成化组合使用

SRE部署 + AutoScaler + “不停机”更新作业并行度

= 降低用户成本和门槛使用流式计算

04 未来工作

未来开展工作–云原生后端状态



在 Flink 2.0 中，社区将通过过渡到完全解耦的存储和计算架构来提升 Flink 状态存储管理系统，以满足云原生环境的需求

目前Flink的状态存储系统还没有完全体现存储计算分离的架构。所有状态数据都保存在本a地RocksDB实例中，分布式快照时仅将增量数据传输到远程存储，以保证远程存储完整的状态数据。展望未来，Flink 或将其状态数据完全保存到远程存储，保留本地磁盘和内存专门用于缓存和加速。我们可以称为“分层状态后端”架构的分层存储系统。

未来开展工作-总结


V1Beta1 Api向下一个目标出发
AutoScaler与Operator分离
自动伸缩相关功能改进
K8S
Flink
2.0
云原生的状态管理机制
Remote Shuffle
完善FLIP-291功能

FLINK FORWARD #ASIA 2023 / 实时即未来
REAL-TIME IS THE FUTURE

THANKS

FLINK FORWARD #ASIA .2023