Deploy Apache Flink Natively on YARN/Kubernetes

公司：阿里巴巴

演讲者：任春德 （高级技术专家）

石春晖 （资深技术专家）

Content [内容]

- **Native Integration with YARN** [Flink 与 YARN 的原生融合]
- **Ecosystem & Improvements in Alibaba** [Flink 在阿里巴巴的生态和改进]
- **Native Integration with K8S** [Flink 和 K8S 的原生融合]
- **Result and Future** [效果及未来计划]

Content [内容]

- **Native Integration with YARN** [Flink与YARN的原生融合]
- **Ecosystem & Improvements in Alibaba** [Flink在阿里巴巴的生态和改进]
- **Native Integration with K8S** [Flink和K8S的原生融合]
- **Result and Future** [效果及未来计划]

Flink Standalone Cluster on YARN

- YARN

  Largest? Big Data processing platform

  [最大?的大数据处理平台]

  High Parallel

  [高并行]

  Multiple-tenants [多租户]

- Flink Standalone Cluster on YARN

  Not native integration

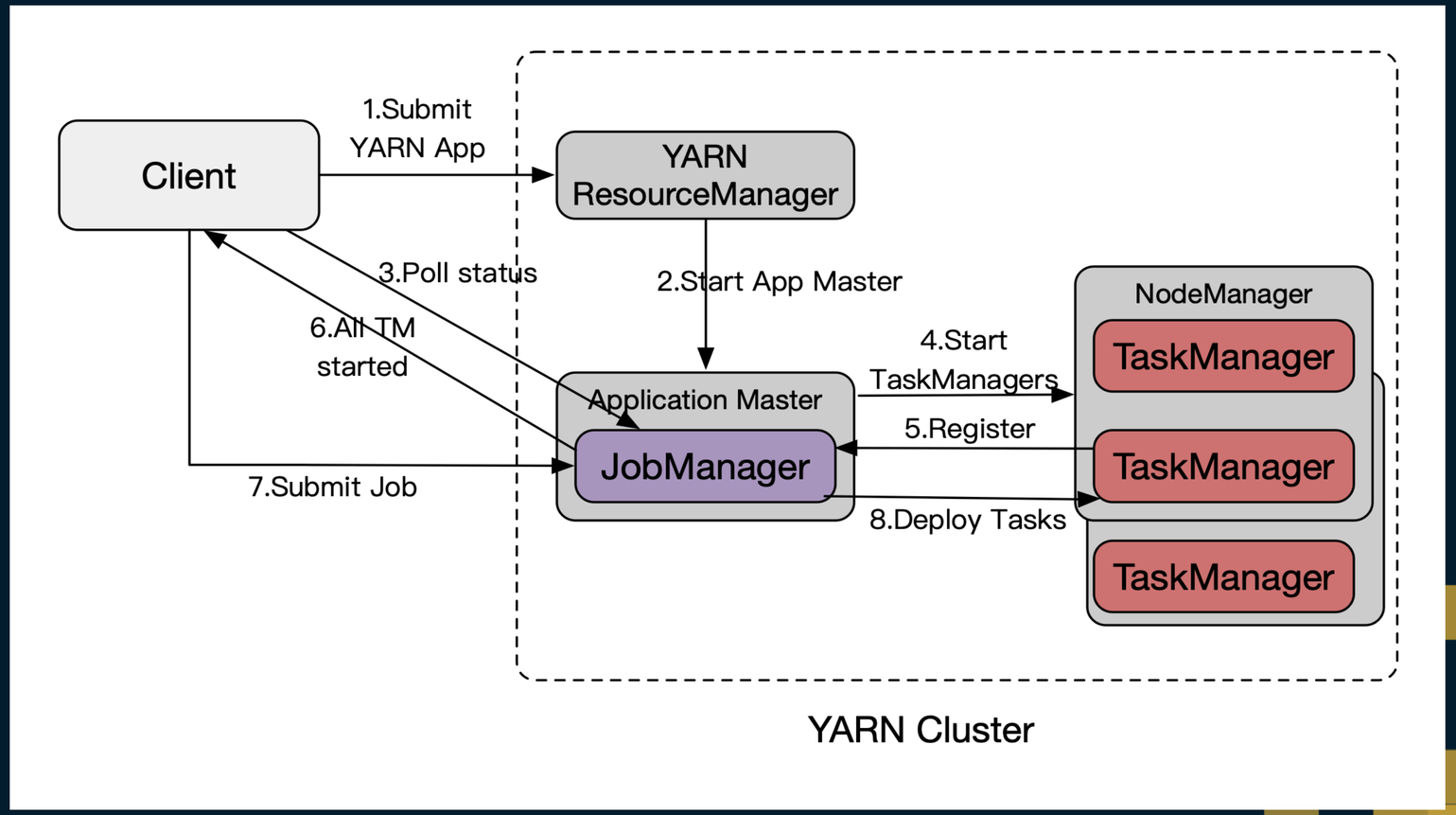
  [缺乏一体化]

  Static Cluster

  [静态的集群]

  Same size Container

  [固定规格的容器]



FLIP-6 - Deployment and Process Model

- ResourceManager

  Request resource from cluster manager

  [向集群管理者请求资源]

  Manage TM

  [管理 TM]

- JobManager

  Schedule tasks of one job

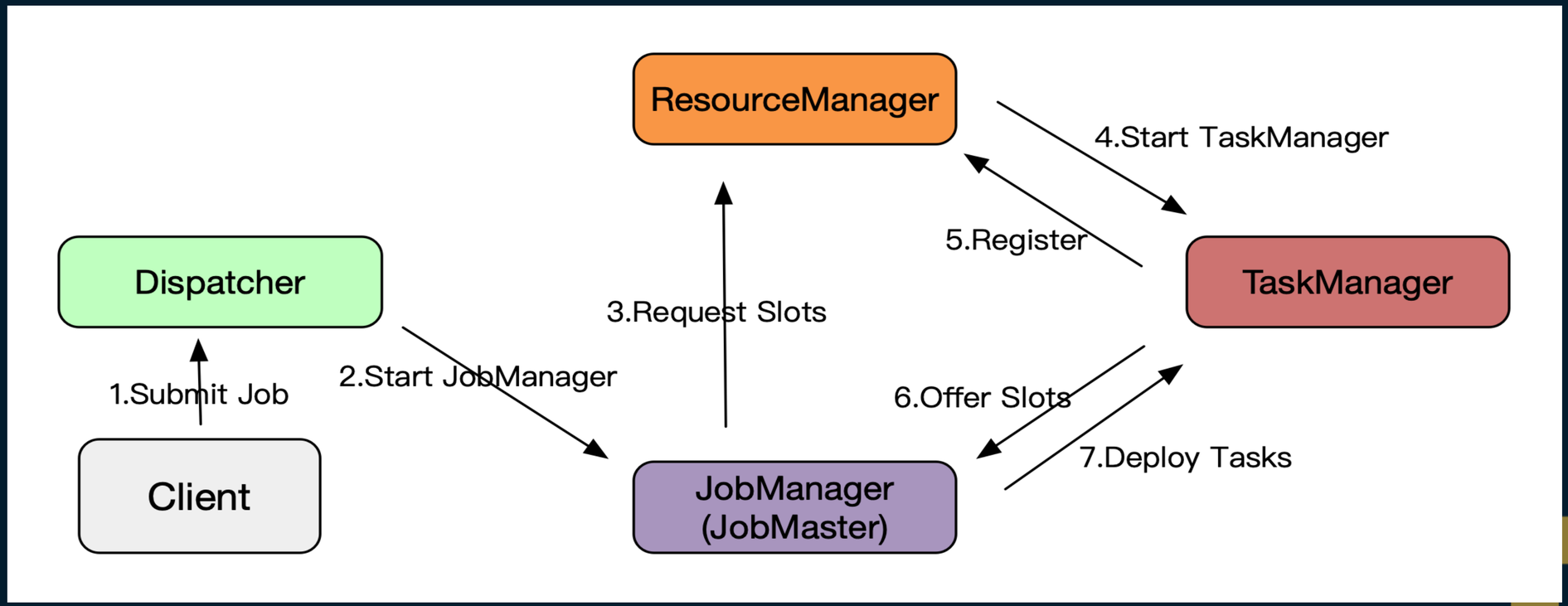
  [调度 job 的 task]

- Dispatcher

  Spawn job

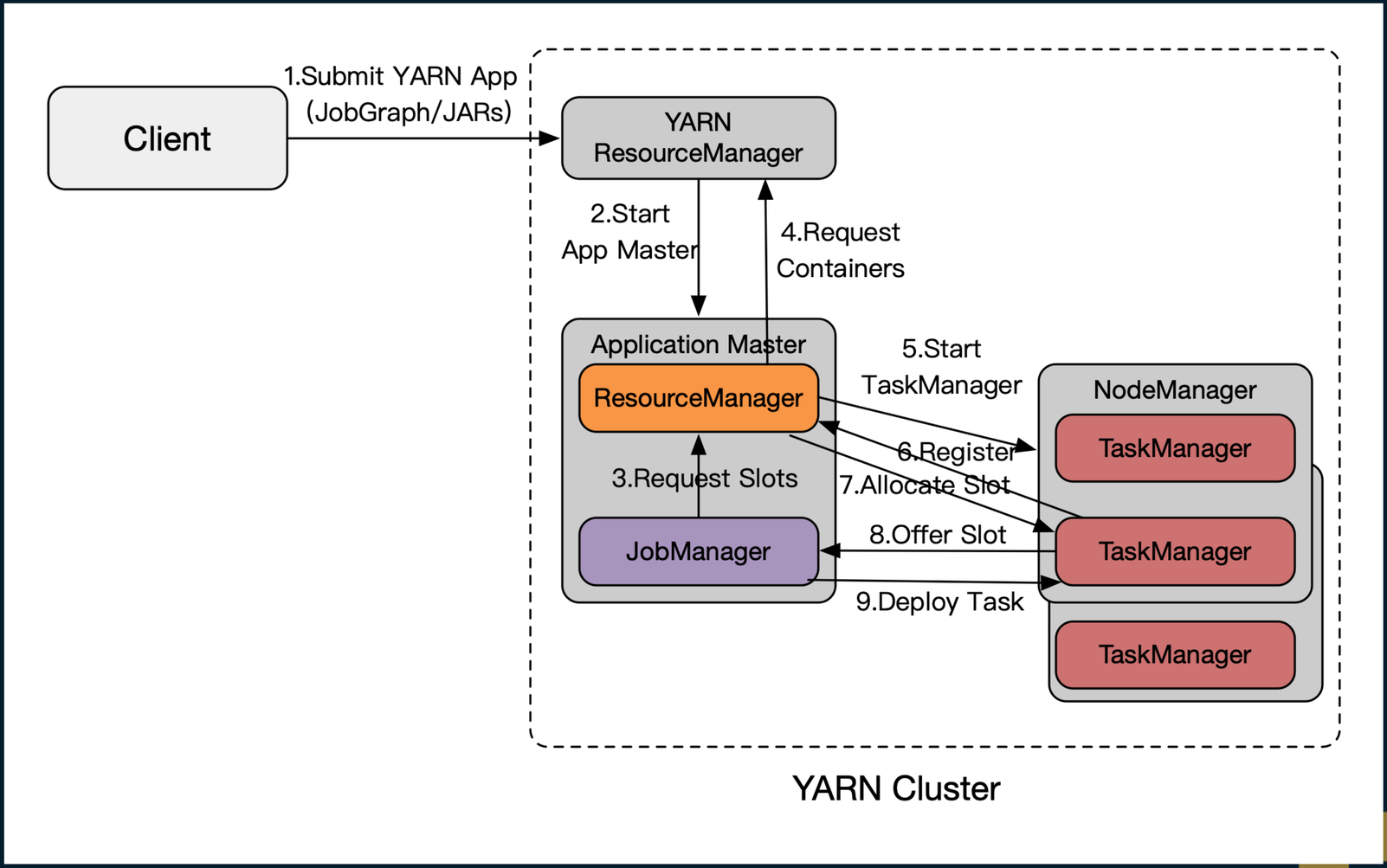
  [生成 job]

- TaskManager



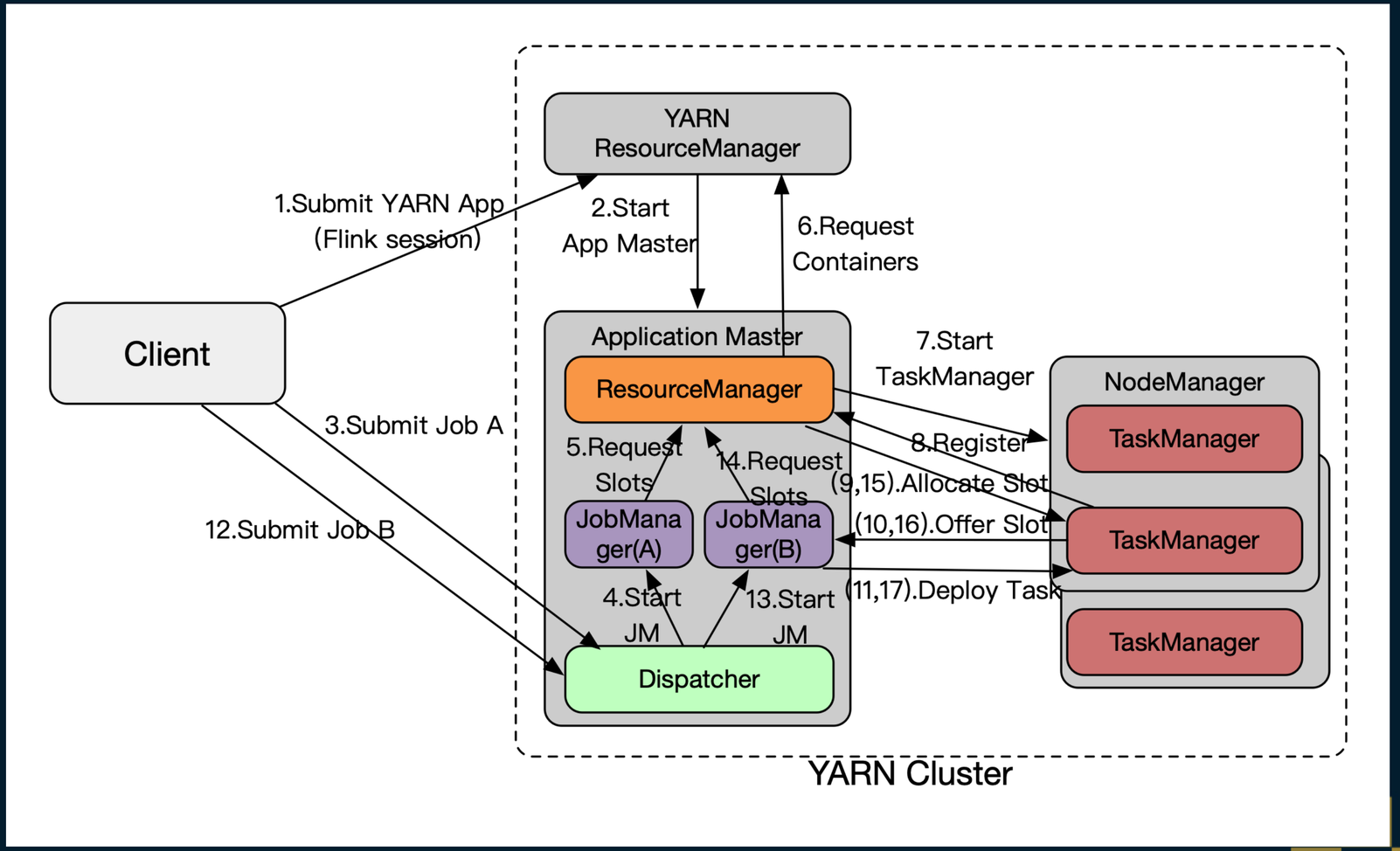
Integration of YARN and Flink – per Job

- JARs via distributed cache
  [jar通过缓存分发]
- No two phase job submission
  [无需2阶段提交]
- Dynamic resources allocation
  requested containers as needed and released when not used
  [按需请求和释放容器]



Integration of YARN and Flink – session

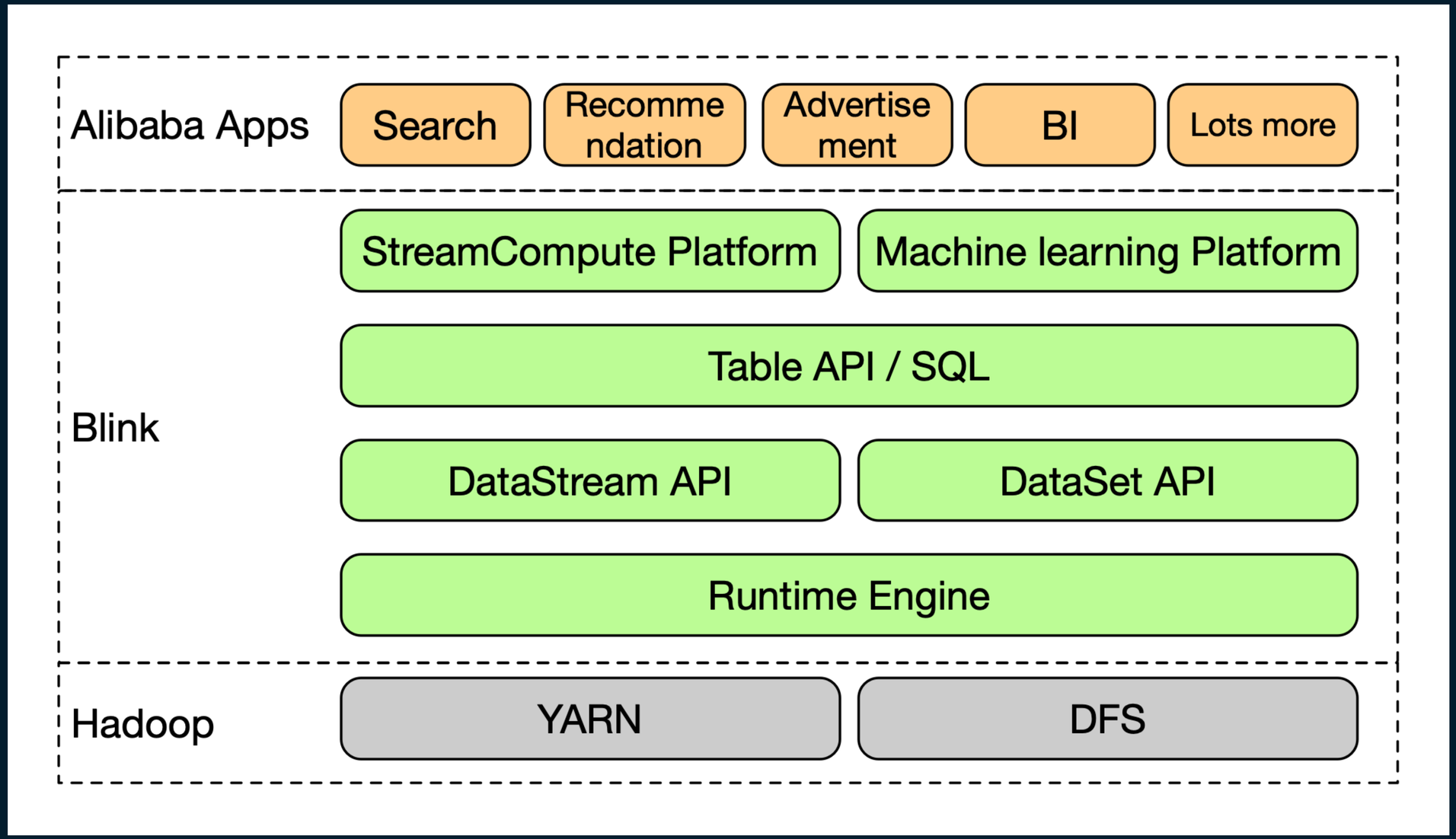
- Only Start the AM
  [只启动应用 Master]
- Dispatcher created by Session Entrypoint
  [Session入口启动 Dispatcher]
- Multiple JobManager
  [多个 JobManager]
- Resource Reuse
  [资源重用]



Content [内容]

- **Native Integration with YARN** [Flink与YARN的原生融合]
- **Ecosystem & Improvements in Alibaba** [Flink在阿里巴巴的生态和改进]
- **Native Integration with K8S** [Flink和K8S的原生融合]
- **Result and Future** [效果及未来计划]

Flink Ecosystem in Alibaba



Resource Profile

- Slot

  Waste of resources VS Out of Memory [权衡资源浪费和内存不足的矛盾]
- Resource Profile [资源]

  Specify CPU & Memory requirements for individual operators

  [指定每个operator的CPU和内存需求量]

  Extended Resource Map，e.g. GPU

  [可扩展的资源Map, 如GPU]

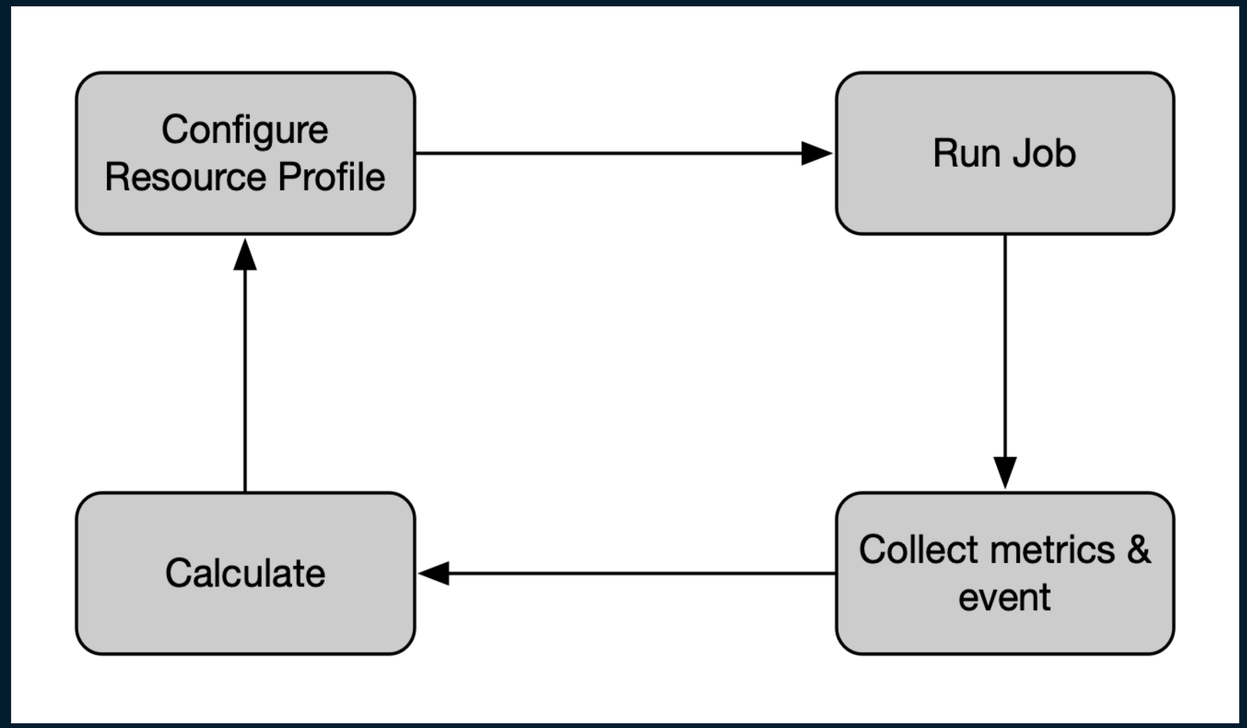
  ResourceManager allocates containers according to resource profiles

  [RM按照资源profile申请容器]
- Cons: difficult to configure the resource profile

  [不足：难以配置资源profile]

Auto Configure Resource Profile

- Configuration Calculating automation
  [配置计算自动化]
- Principle [原理]
  Measurable [可测量]
  Stable status [稳定状态]
  Constant average event processing time
  [不变的消息平均处理时间]
  Constant Input / Output TPS
  [不变的输入/输出TPS比例]
  Task parallelism =
  target TPS / max (single task TPS)
- Autoscaling
  [自动伸缩]



YARN External Shuffle Service

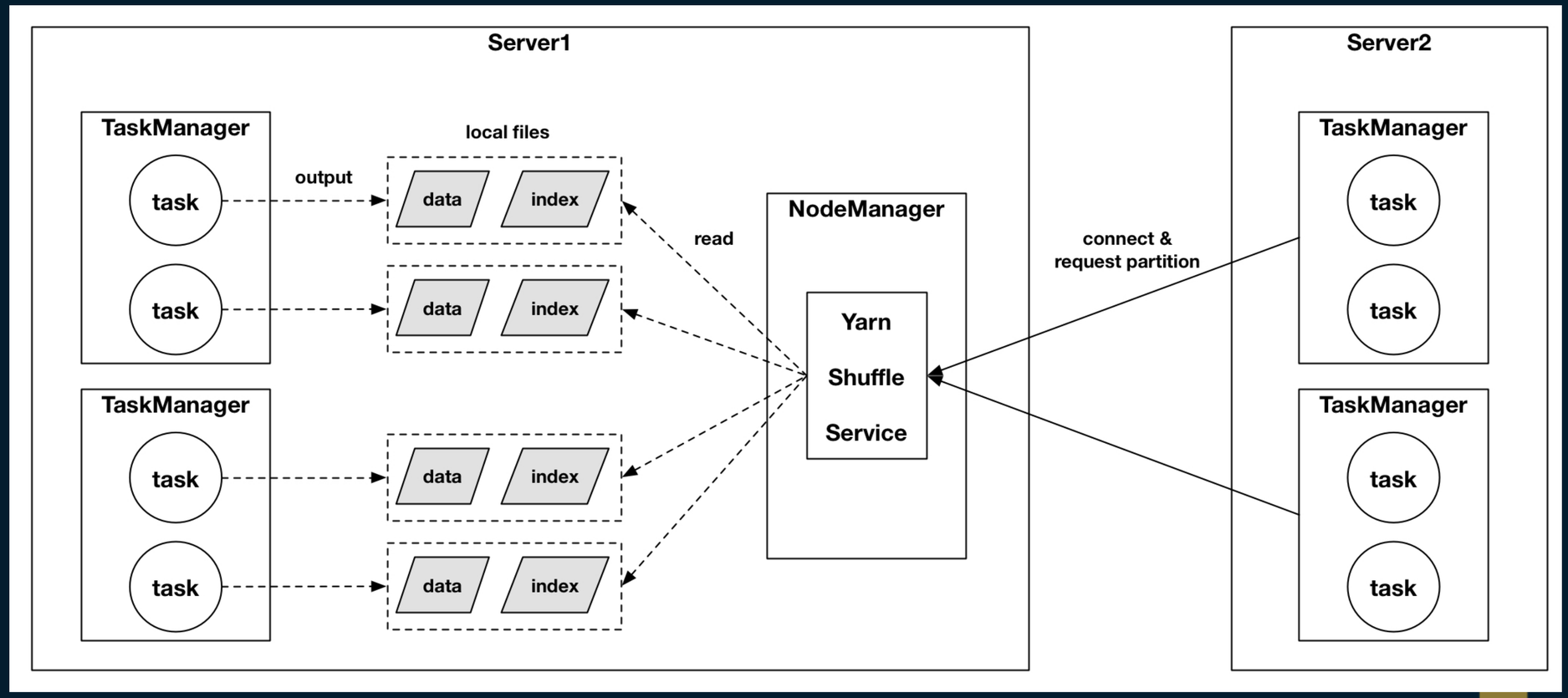
- Batch job [批作业]
- Pipeline shuffle [管道式shuffle]

  Task completed but not release slot

  [task结束但未释放slot]

- External Shuffle Service [外部的Shuffle服务]

  Reduce waste of resources [减少资源浪费]



Content [内容]

- **Native Integration with YARN** [Flink与YARN的原生融合]
- **Ecosystem & Improvements in Alibaba** [Flink在阿里巴巴的生态和改进]
- **Native Integration with K8S** [Flink和K8S的原生融合]
- **Result and Future** [效果及未来计划]

Introduction to Kubernetes

- Container [容器]

  Container provides os-level virtualization and resource isolation

  [容器提供OS层的虚拟化和资源隔离]
- Container Orchestrators [容器编排]

  Challenges to use containers at scale: life cycle management, discovering, consuming, and monitoring

  [大规模使用容器的挑战：容器生命周期管理，（容器里)服务的发现，使用和监控]
- Container Orchestrators provides: Declarative Configurations; Rules and Constraints; Provisioning on Multiple Hosts; Service Discovery and Health Monitoring

  [容器编排器提供： 描述性配置；规则和限制；多主机资源分配；服务发现；服务监控]
- Kubernetes by Google is the most popular Container Orchestration Platform

Componenets in Kubernetes [K8S 的内容]

- Control Plane [控制层]

API server: provides RESTful interface to query and modify cluster states

etcd: distributed K-V store

Scheduler: assign nodes to pods

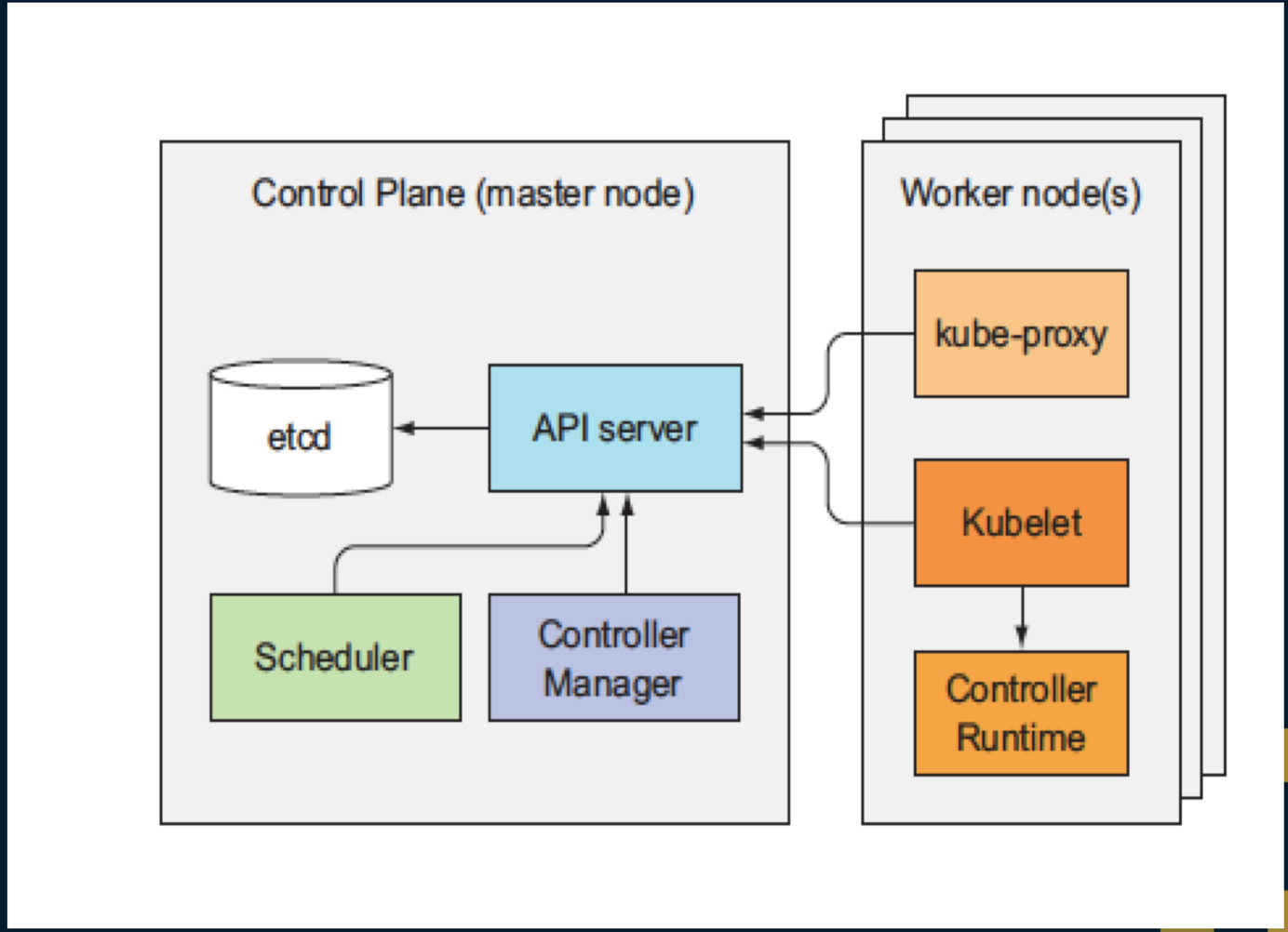
Controllers: to ensure the state of a cluster is desired

- Worker nodes [工作节点]

kube-proxy: for clients to connect services on the node

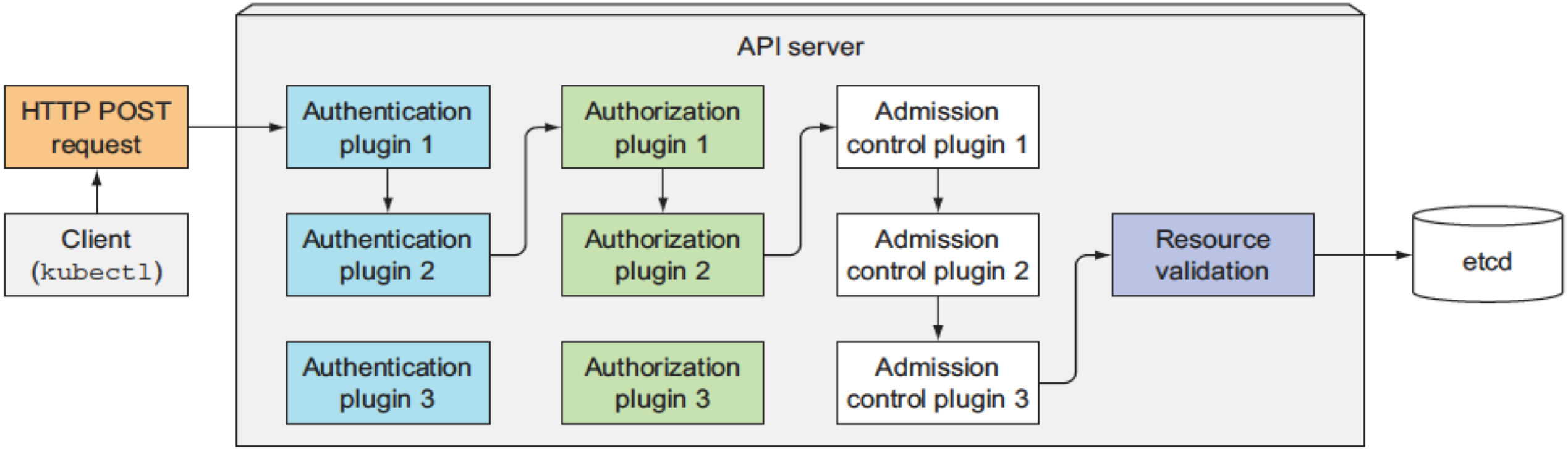
kubelet: start, monitor, stop pod's containers and notify API server

Container Runtime: docker, rkt, etc



Security in Kubernetes [K8S 的安全模块]

- Authentication: k8s Authenticate API requests with certificate, bearer tokens, openID connect tokens; K8s authn can be integrated with LDAP, SAML, etc. by using authenticating proxy, and authentication webhook.
- Authorization: RBAC, ABAC, Node, etc.



Popularity of Container Orchestrators

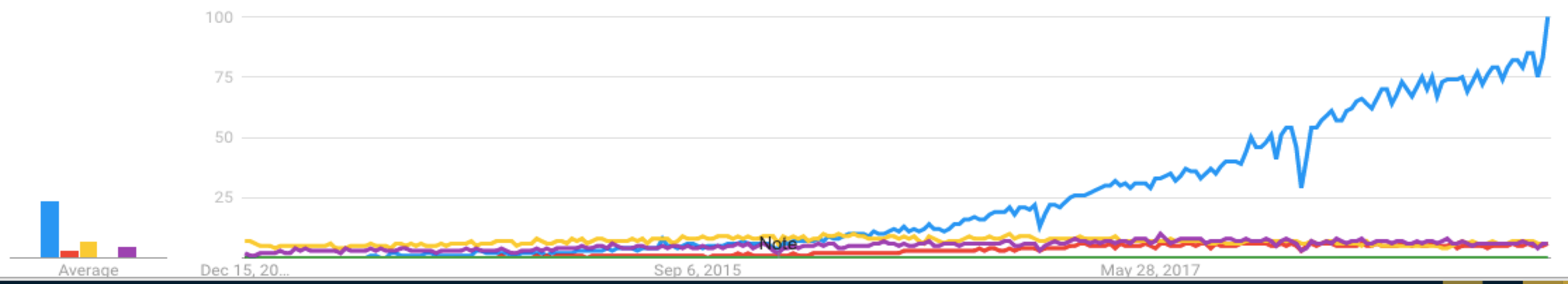
FLINK FORWARD

Kubernetes	Docker Swarm	Mesos	CoreOS Fleet	Cloud Foundry
Search term	Search term	Search term	Search term	Software

Worldwide ▾ Past 5 years ▾ All categories ▾ Web Search ▾

Note: This comparison contains both Search terms and Topics, which are measured differently. LEARN MORE

Interest over time

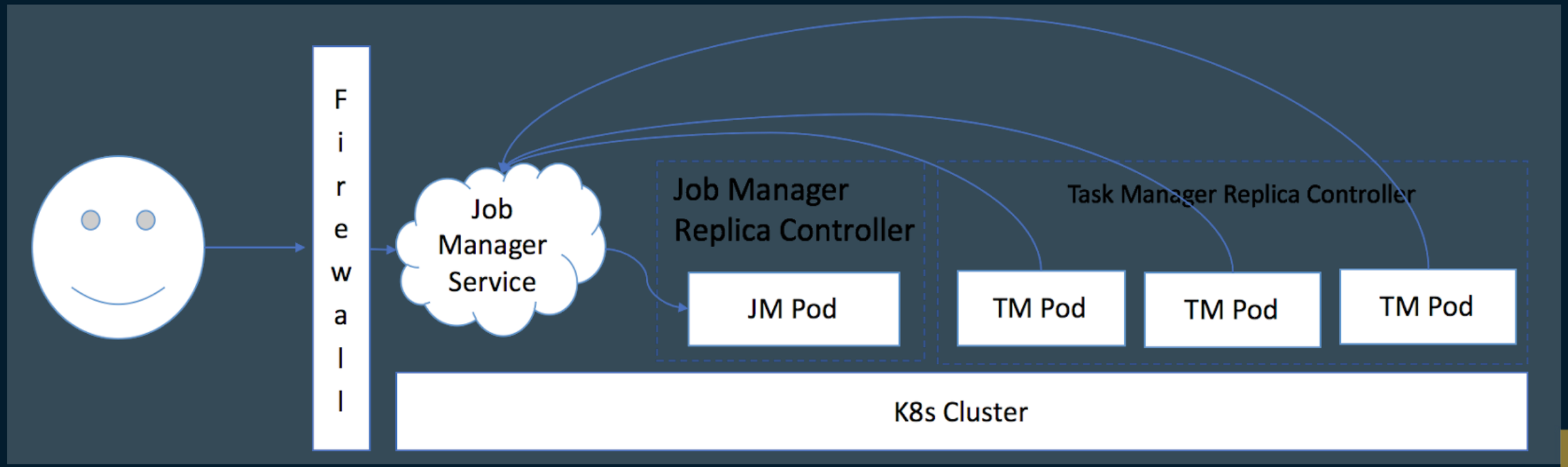


Why Flink on Kubernetes

- Run everywhere: Kubernetes is available on all cloud platform
- Native resource manager & dynamic allocation
- Multi-tenant, resource isolation and quota management
- Leverage security features of Kubernetes

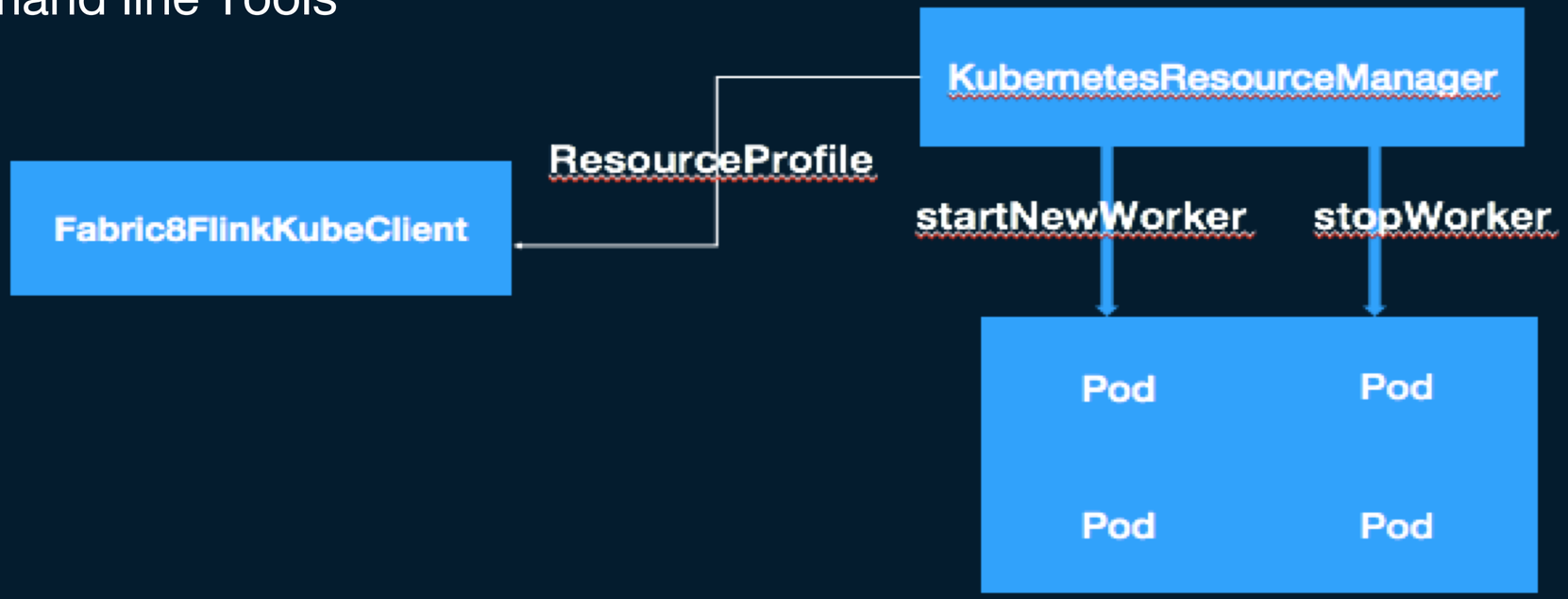
Integration of Kubernetes and Flink

- Flink Native Integration with Kubernetes
  Implement another ResourceManager in Flink that access Kubernetes natively.
  Job Manager will be the Application Master to request resources(Pod)
  Two kinds of Pods are defined: JM Pod, TM Pod



Integration of Kubernetes and Flink

- K8s Resource Manager Module
  - cluster descriptor
  - resource manager
  - mock test framework
- Docker Image for both cluster pod and taskmanager pod
- Command line Tools



Content [内容]

- **Native Integration with YARN** [Flink与YARN的原生融合]
- **Ecosystem & Improvements in Alibaba** [Flink在阿里巴巴的生态和改进]
- **Native Integration with K8S** [Flink和K8S的原生融合]
- **Result and Future** [效果及未来计划]

Project Status [项目进度]

- Targeted for: Flink 1.8
- Design Doc (to be continued)
- Umbrella JIRA: FLINK-9953
- First PR: https://github.com/apache/flink/pull/7144

Future Directions

- Dynamic Resource allocation
  The benefit of ‘native integration’
- High Availability of Flink with Kubernetes
- External Shuffle Service


THANKS