阿里巴巴云原生专场

FaaS & Cloud Native 函数计算的云原生之旅

演讲人：常率 阿里云技术专家

观看视频回放

Agenda

1. 云原生和 FaaS
2. Serverless/FaaS 和容器生态
3. 函数计算的云原生可观测性

云原生的业务价值

1. 更快的上市速度（Time-to-market）
    - 原生属性基本满足生产要求（Production ready）
        - 高可用
        - 自动扩缩容
        - 资源隔离、安全
    - 发布周期短、频率高
2. 成本优化
    - 运维成本
    - 资源利用率

While there are many benefits provided by cloud native technologies, 52% of respondents ranked faster deployment time as the biggest benefit of using cloud native projects in production. This was followed by improved scalability at 45%, and cloud portability and improved availability tied with 39%. Interestingly, cost savings ranked the lowest on a weighted scale.

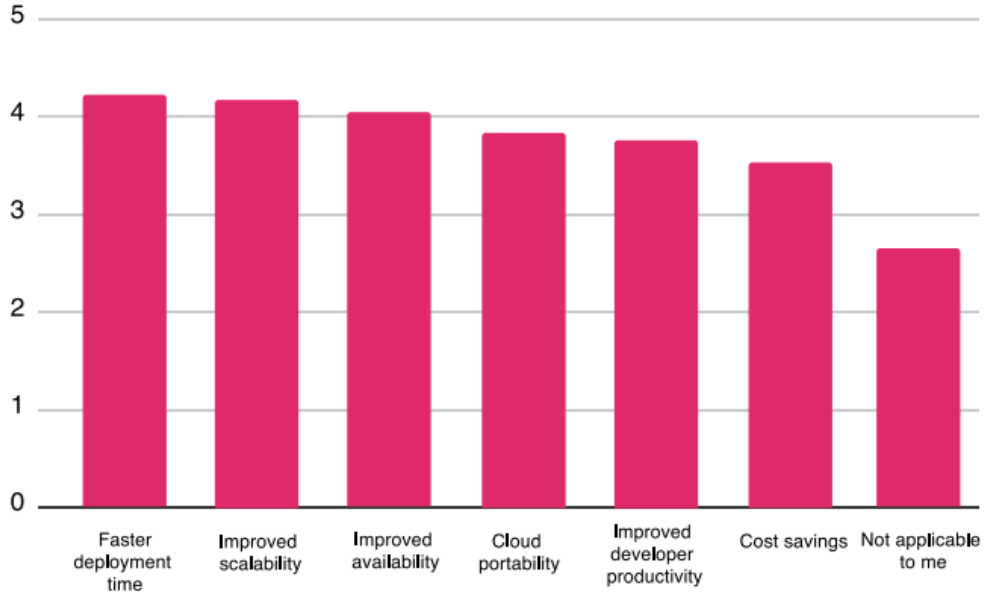
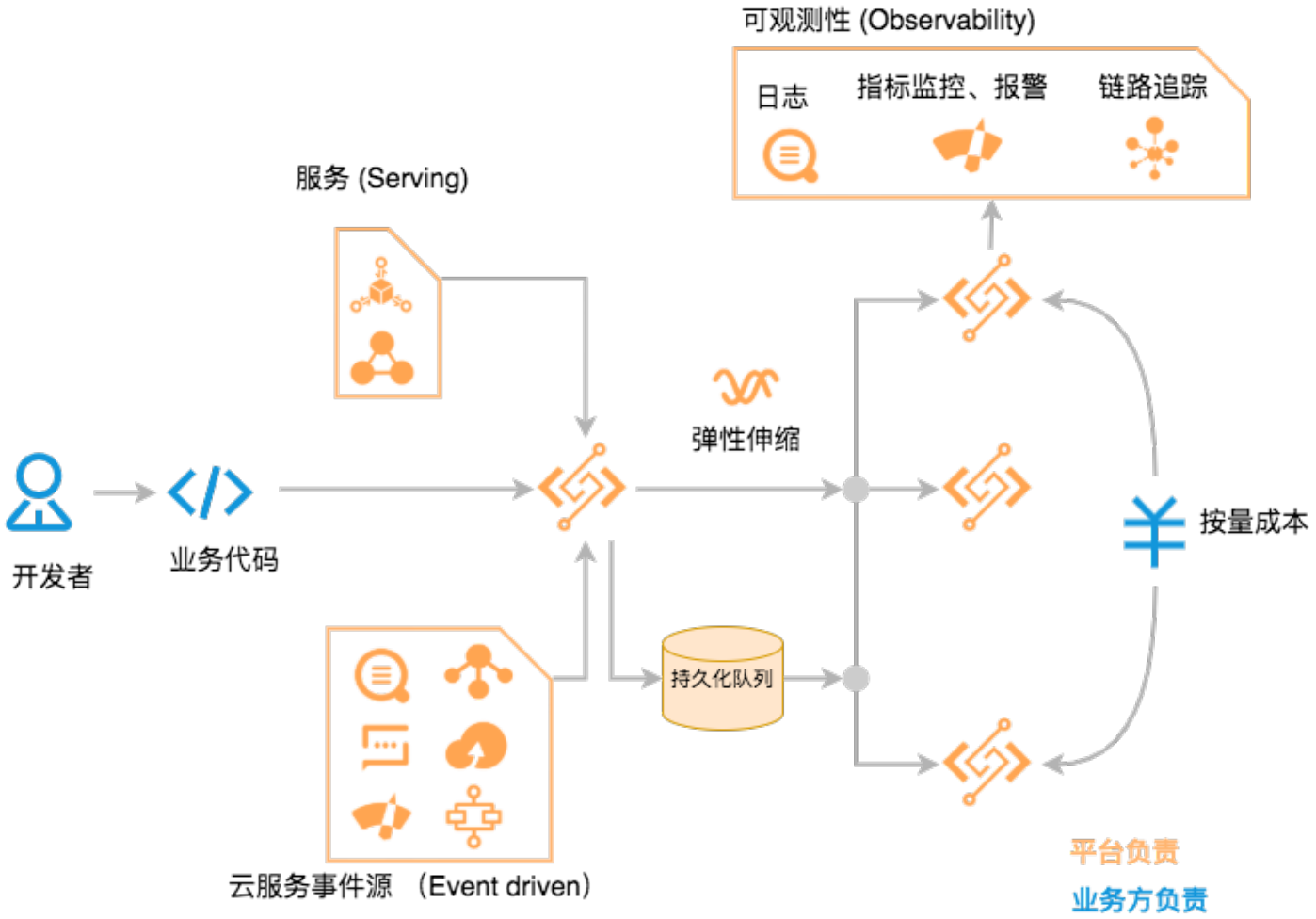


Image source: CNCF SURVEY 2019

Serverless.FaaS

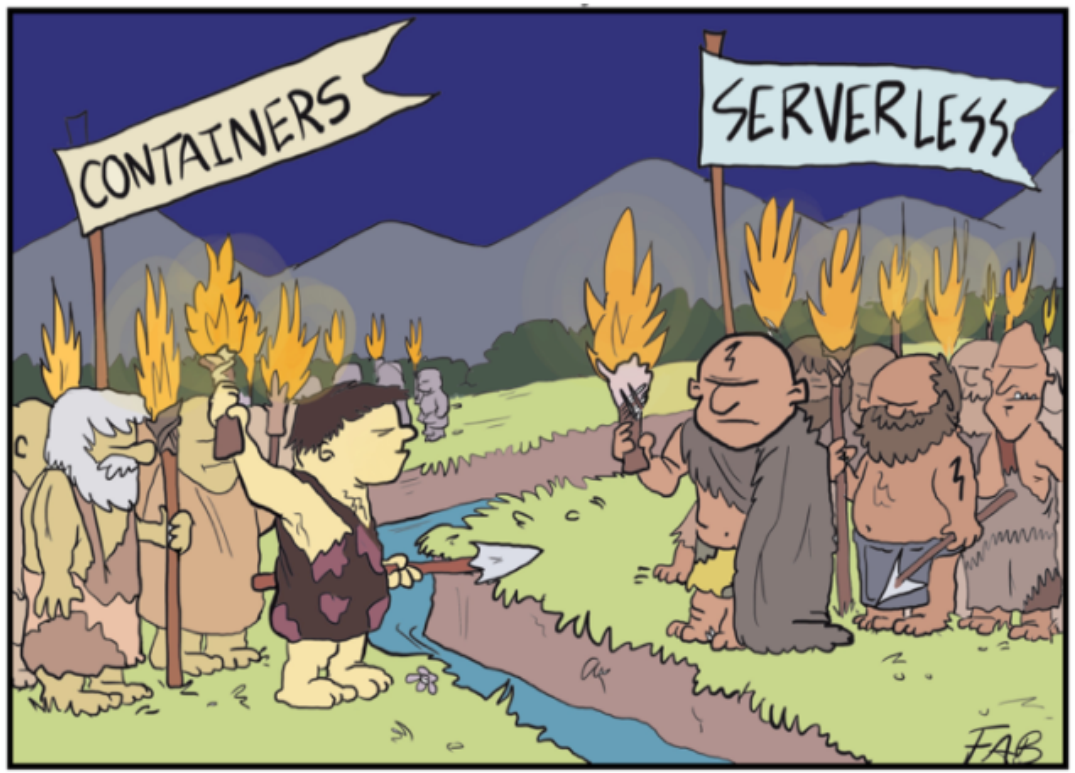
- Time-to-market:
  - 端对端整合，production ready
  - 云服务集成
  - 专注于业务逻辑开发
- 成本优化
  - 仅为实际使用（请求粒度）付费
  - 平台负责运维成本
- Serverless 是云原生重要组成部分
- FaaS 是 Serverless 的一种服务形式



固有印象

Containers

- 任意语言、依赖、二进制
- 易迁移、无锁定
- 完善标准的工具生态 e.g. CI/CD
- 灵活：硬件种类、规格
- Layer 机制提速开发，简化分享和复用



The two tribes regarded each other suspiciously in the glow of their blazing production environments.

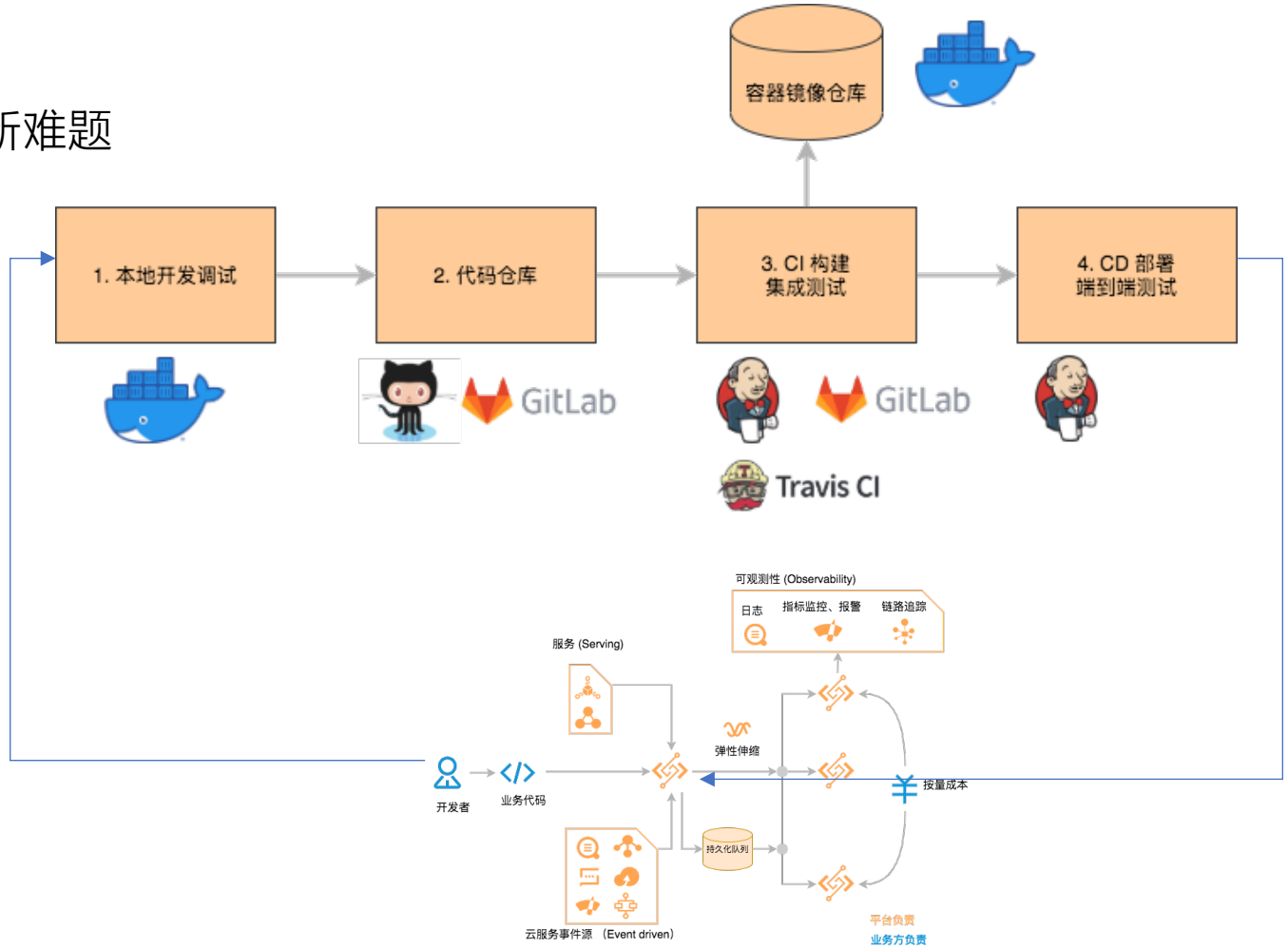
Image source: by Forrest Brazeal https://faasandfurious.com/46

FaaS

- 云服务集成
- 免运维
- 快速伸缩
- 事件驱动：
  - 响应云服务事件
  - 按需、请求粒度成本
  - 伸缩

Time-to-market 的全貌

- CI/CD 是快速交付上市的关键
- FaaS 简化“上线后”，却为“上线前”引入新难题
- **难点在交付物**（代码压缩包）：
  - 特定的编译环境
  - 代码包大小限制
  - 全量构建
  - 无标准的版本管理
  - 缺少工具和生态
  - 缺少标准的分享和引用机制

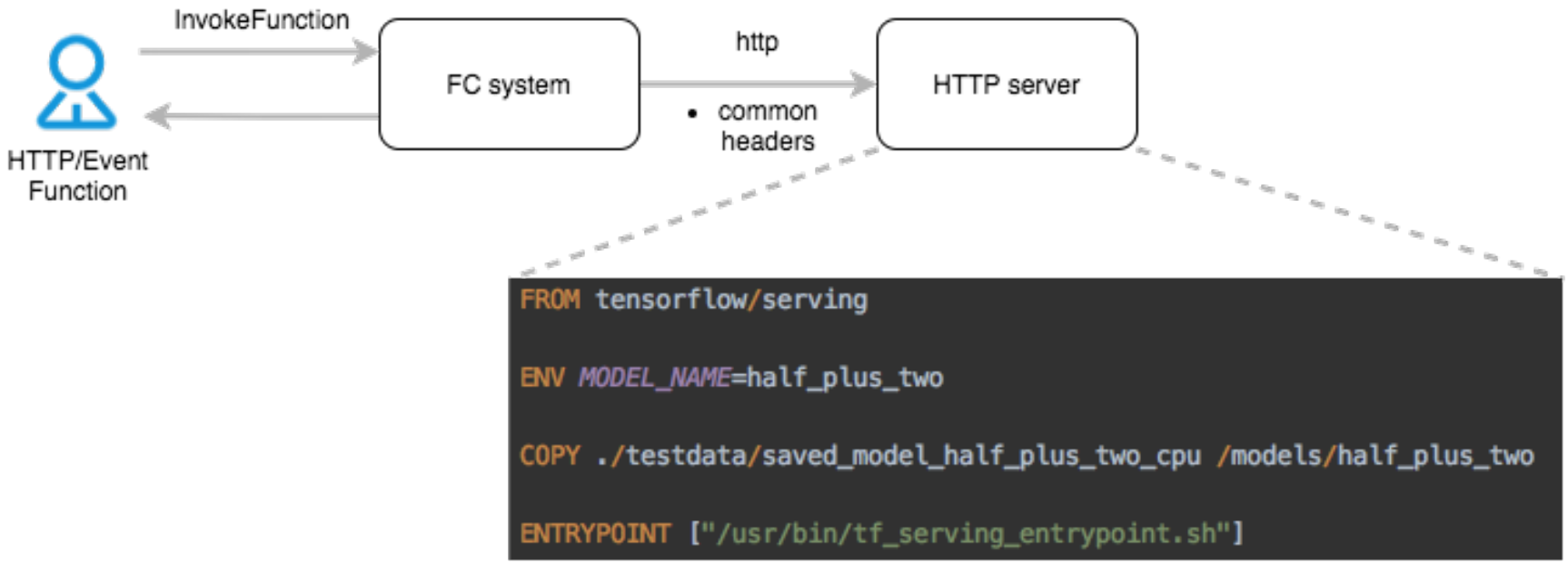


我有适用于 FaaS 的场景，但是

- 我需要：
  - 使用熟悉的开发、调试工具
  - 复用统一的 CI/CD 流水线
  - FaaS 提供环境无法编译的二进制
  - 在 FaaS 和 Kubernetes 用统一交付物
  - 需要使用 nodejs14, .NET Core 3.1，FaaS 平台不支持
  - 复用或基于开源的容器镜像
  - 以镜像的形式分享我的代码
  - 增量构建、上传、下载镜像
  - 我的代码不想和云厂商绑定（vendor-lockin）

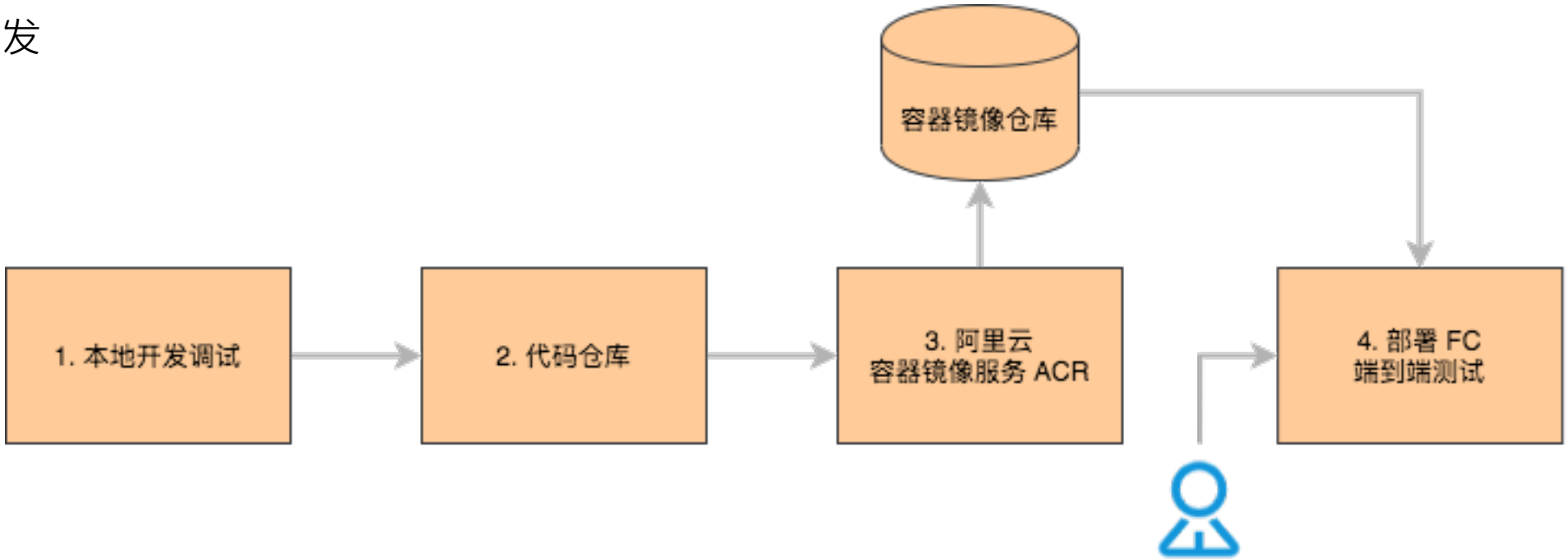
函数计算支持自定义镜像

- 新的 runtime: custom-container
- 工作原理：



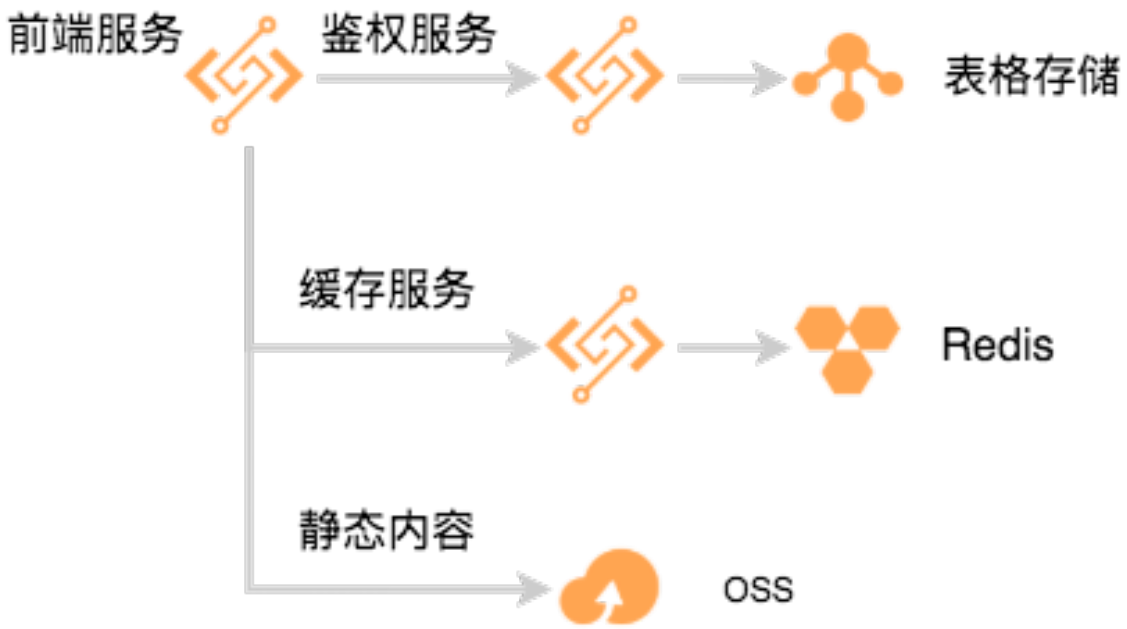
Demo

1. 复用已有开源镜像（tensorflow/serving）
2. Git push -> build -> push -> 升级函数
3. 依赖和业务逻辑代码分层
4. 事件触发



FaaS 可观测性

1. 事件触发 -> 与众多云服务交互
2. 函数单一责任最佳实践，函数互相调用形成应用
3. 内部对开发者是黑盒
4. 分布式系统调查链路延迟问题：
    1. 日志：关联难度大，缺少标准埋点库
    2. 指标：经聚合后精度降低，不易发现问题



函数计算云原生 Tracing 能力

1. 和阿里云 ARMS 打通
2. 函数代码中使用开源 Jaeger tracing 埋点，指标自动收集到链路追踪服务
3. 使用 FC SDK 调用，无需埋点自动添加 span
4. 跨 FC 间调用，trace 透传至被调用 FC，支持继续埋点
5. FC 内部冷启动等系统延迟可观测

Demo

函数计算的云原生之旅

- FaaS + 容器镜像帮助业务更快、更早地体验 Serverless 优势
- FC 新增 tracing 功能，形成可观测性完整套件
- 未来 FC 会和云原生生态更紧密结合，进一步简化应用的 Serverless 迁移、可观测、分享以及复用

THANKS