

ServiceMesh发展趋势：云原生中流砥柱



蚂蚁金服-中间件-服务与容器团队



讲师介绍

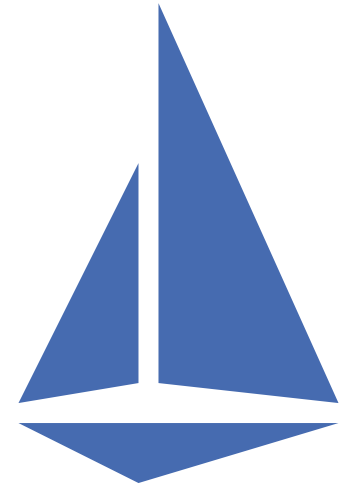
- 敖小剑，蚂蚁金服高级技术专家
- 资深码农，专注于基础架构，Cloud Native 拥护者，Service Mesh布道师。
- 曾在亚信、爱立信、唯品会等任职，对基础架构和微服务有过深入研究和实践。
- 目前在蚂蚁金服中间件团队从事Service Mesh、Serverless等Cloud Native产品开发。



- ❖ Service Mesh产品动态
- ❖ Service Mesh发展趋势
- ❖ Service Mesh与云原生

Istio1.1 Release

Istio 最近的版本发布



0.8

2018-06-01

- Istio历史上第一个LTS版本
- Istio历史上变动最大的一个版本



1.0

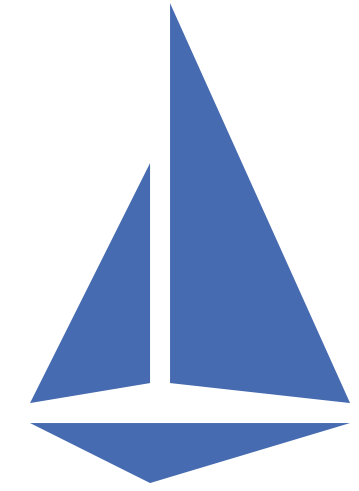
2018-07-31

- Product Ready

1.0.6
1.1.0-snapshot6
1.1.0-rc6



9个月

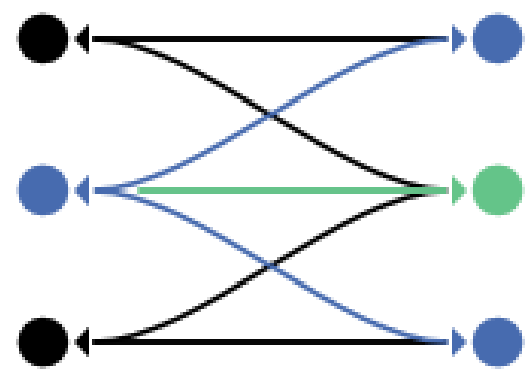


1.1

2019-03-20

- Enterprise Ready

Istio 1.1 新特性



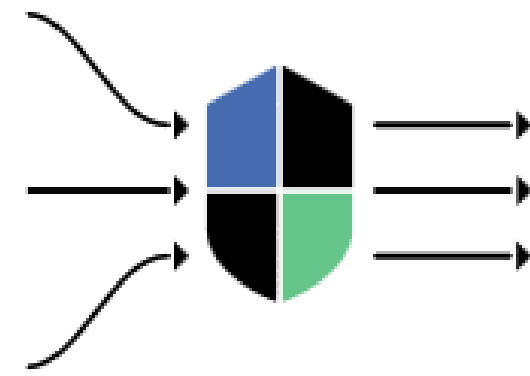
流量管理

- 新增 Sidecar CRD
- 限制服务可见性: `exportTo`
- 区域感知路由
- 大幅改进的多集群路由
- 缺省开放 Egress 通信
- 更新了 ServiceEntry 的资源
- 弃用 Istio Ingress



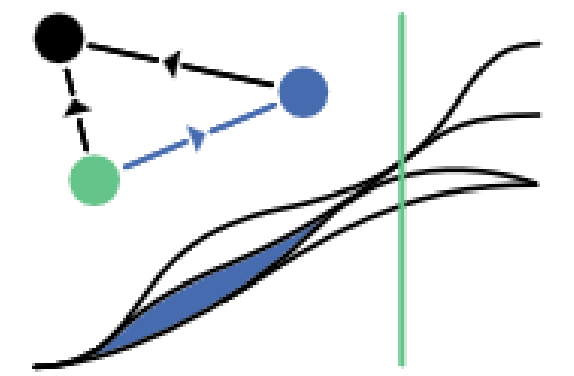
安全

- Readiness and Liveness 探针
- 集群RBAC配置
- 基于SDS的身份设置
- 对 TCP 服务提供鉴权支持
- 最终用户组授权管理
- Gateway上外部证书管理
- 集成Vault PKI
- 自定义的可信域



策略和遥测

- 缺省关闭 Mixer 策略检查
- Kiali 替代 ServiceGraph
- 性能改进, 降低开销
- 控制请求头和路由
- 进程外适配器生产可用
- 多方面增强 Tracing 的能力
- 默认的TCP指标

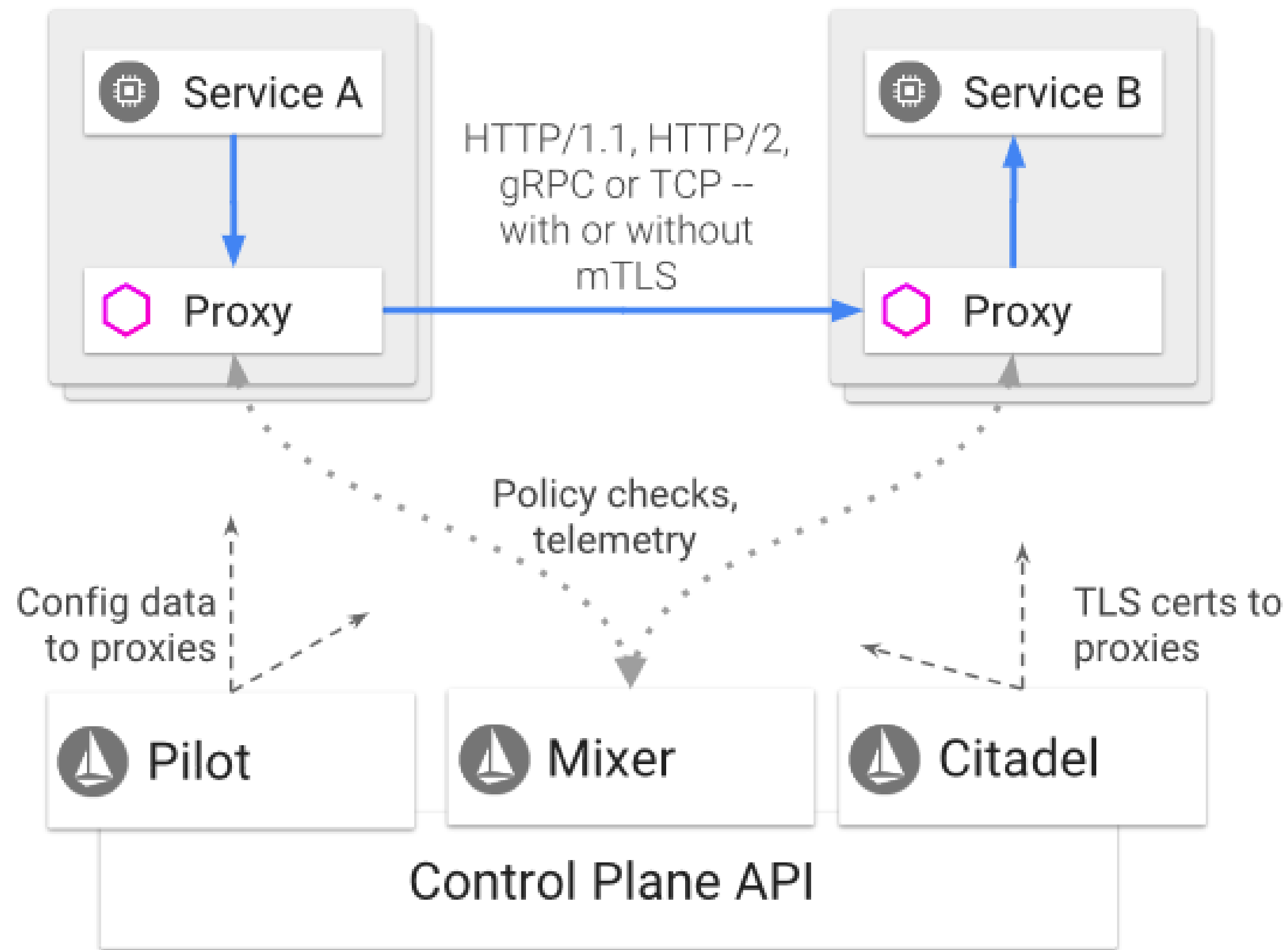


配置管理

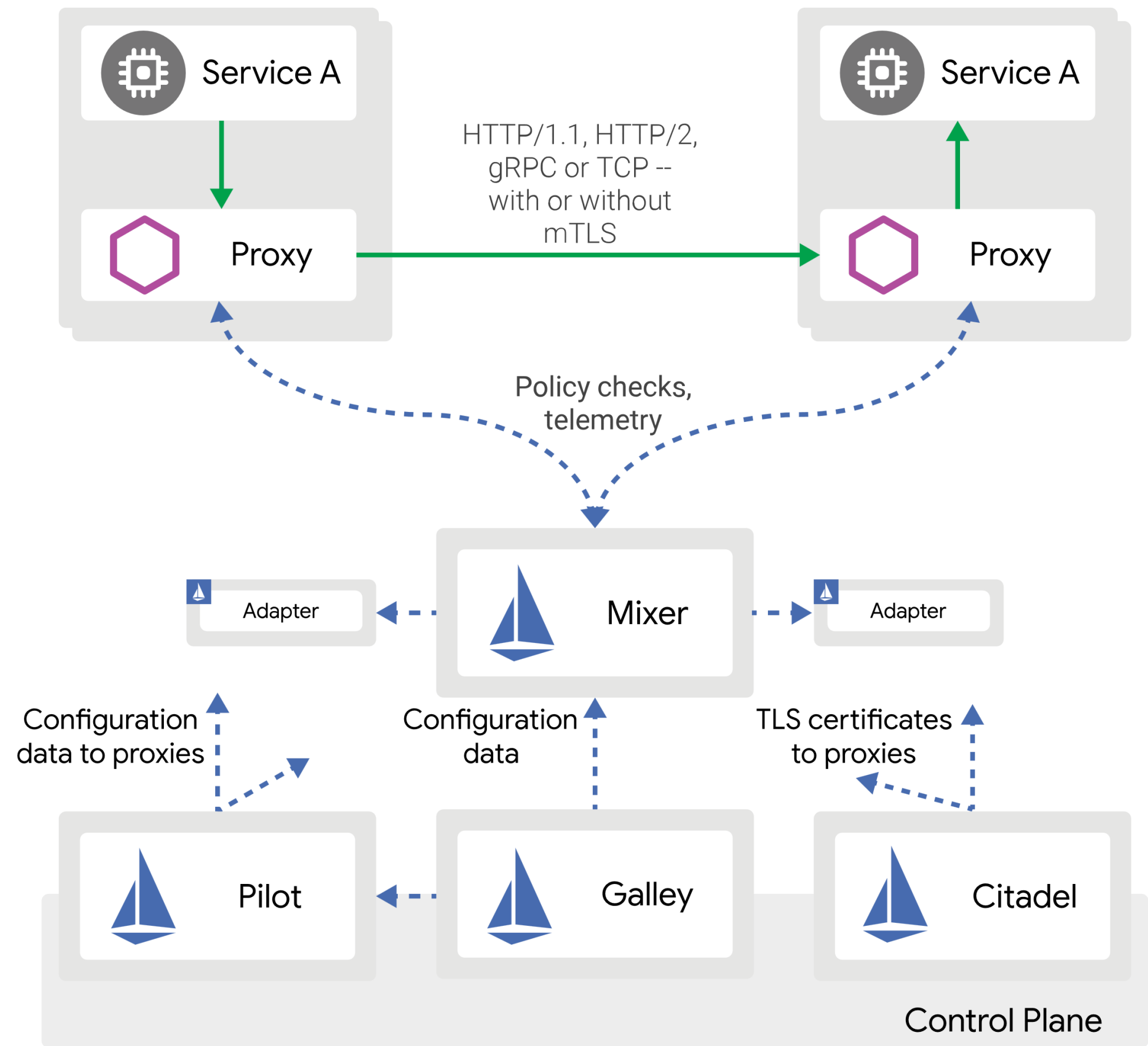
- Galley负责配置管理和分发机制
- 引入MCP协议 (进行中)

Istio1.1 架构变化

Istio 主要组件和架构



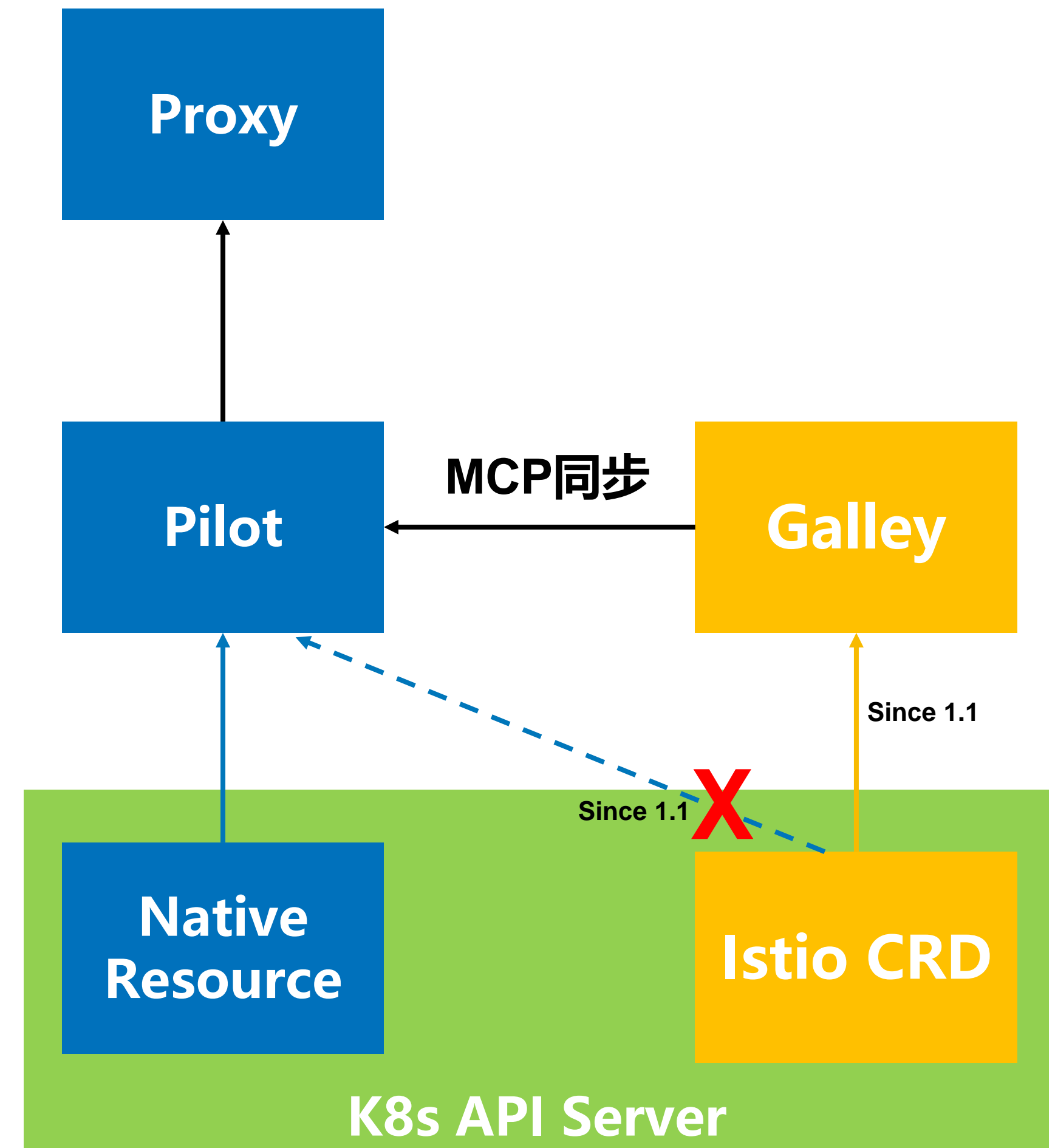
Istio 1.0



Istio 1.1

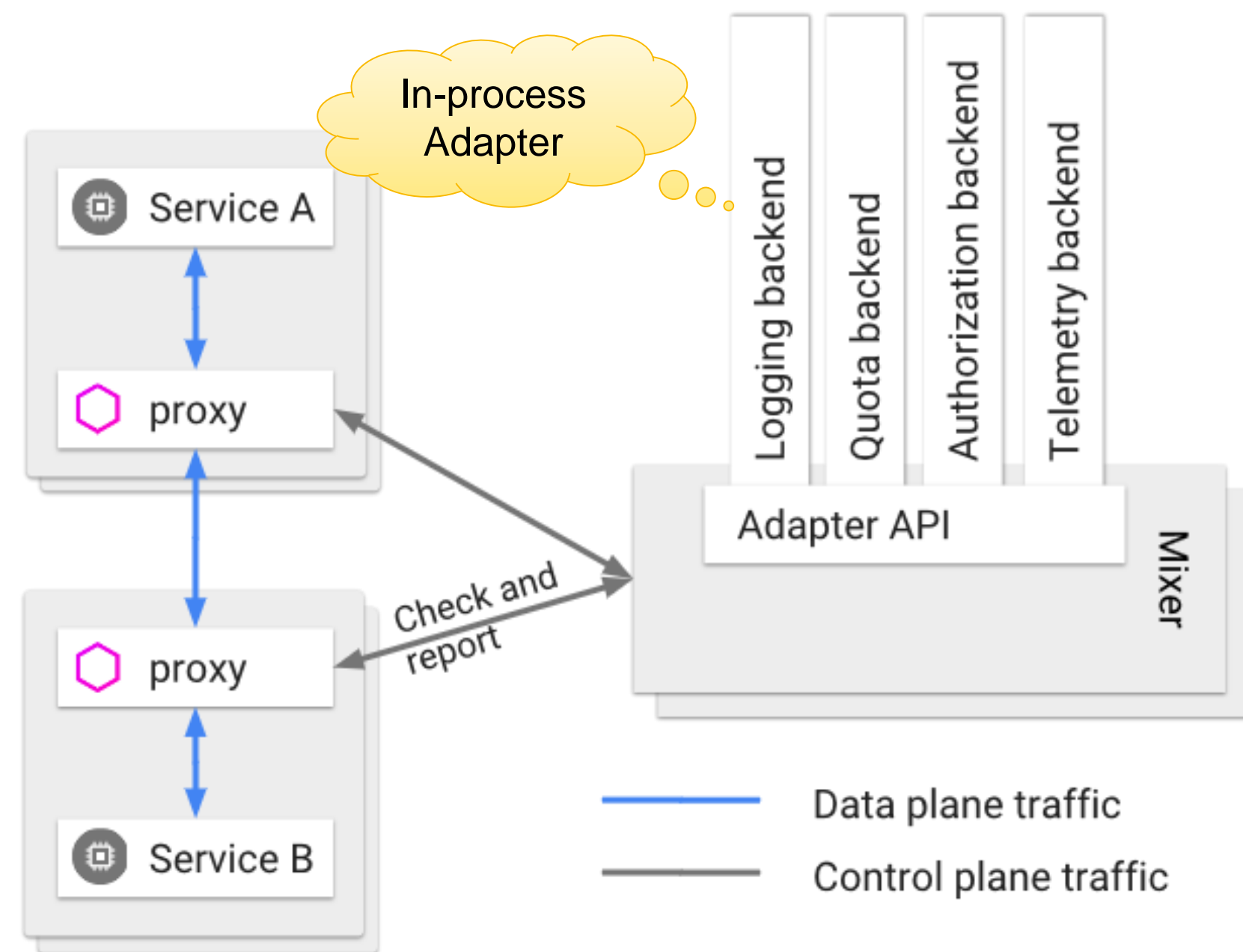
Galley / Pilot

- Galley 分担 Pilot 的职责 (进行中)
 - Istio CRD 由 Galley 读取, Pilot 通过MCP协议
 - K8s 原生资源 (Service/Pod), 暂时还是由Pilot读取
- Pilot简化 (进行中)
 - 目标是和k8s解耦, 只和Galley通信
- 引入MCP协议
 - 受xDS协议启发, 用于在Istio各模块之间同步数据

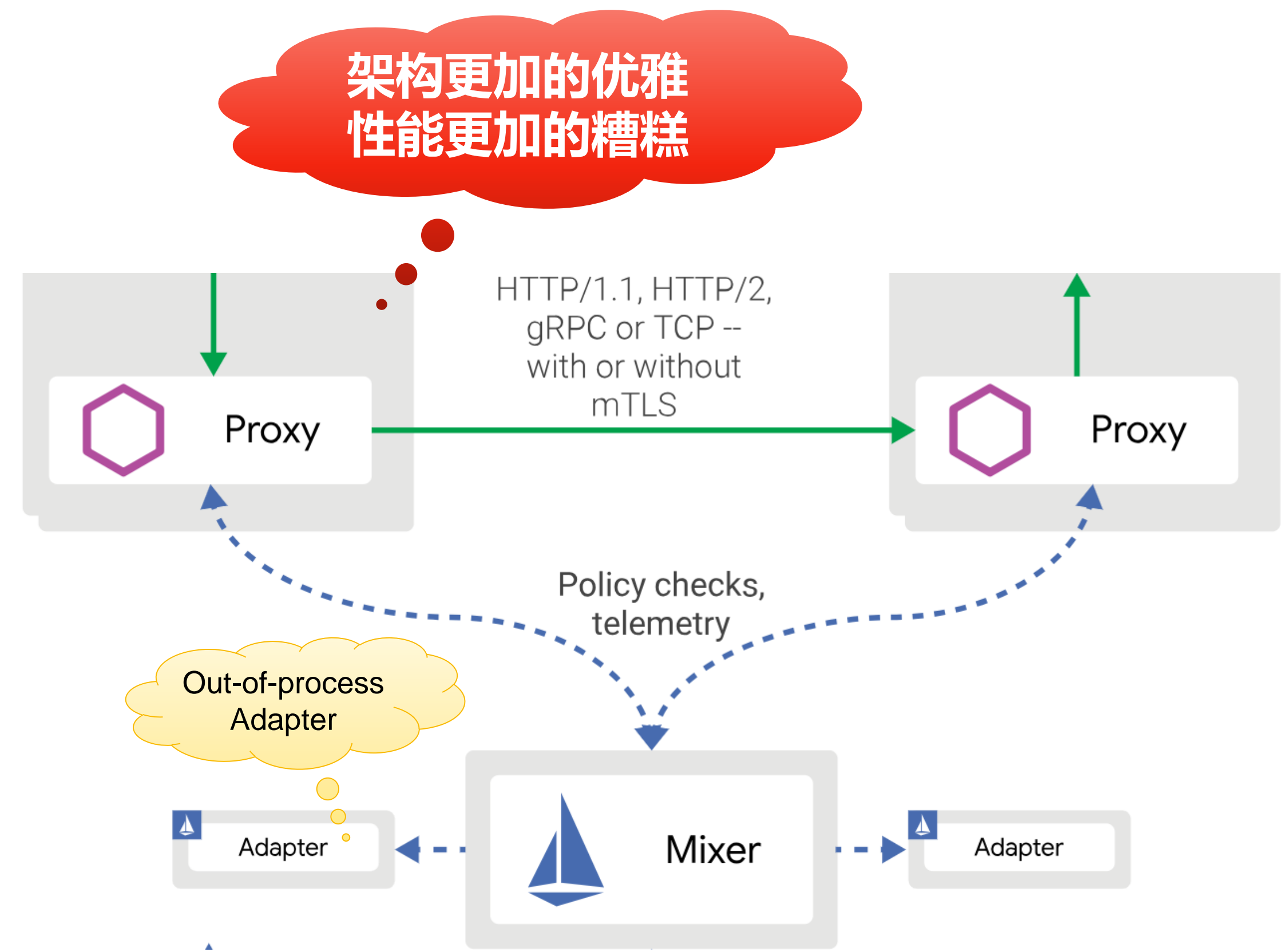


Mixer

- Out of Process Adapter
 - Istio 1.1 推荐使用
 - 预计下一个版本将弃用 In-Proxy Adapter
 - 所有Adapter都将改为 Out-of-process adapter



Istio 1.0



Istio 1.1

Mixer v2 规划中

- 合并Mixer到Envoy中
- 用c++重写Adapter
- 后续准备引入Web assembly (WASM)

- 目前状态: Review 中

通用数据平面API

- Universal Data Plane API (UDPA)
- 创意来自 Envoy, 实现为 xDS API
- xDS v2 API 是数据平面API的事实标准
- CNCF 组织 UDPA working group
 - 初始成员来自包括 Envoy 和 gRPC 项目的代表
 - UDPA会以xDS v2 API 为基础
 - 蚂蚁金服在积极参与
- 目前还处于非常早期的筹备阶段



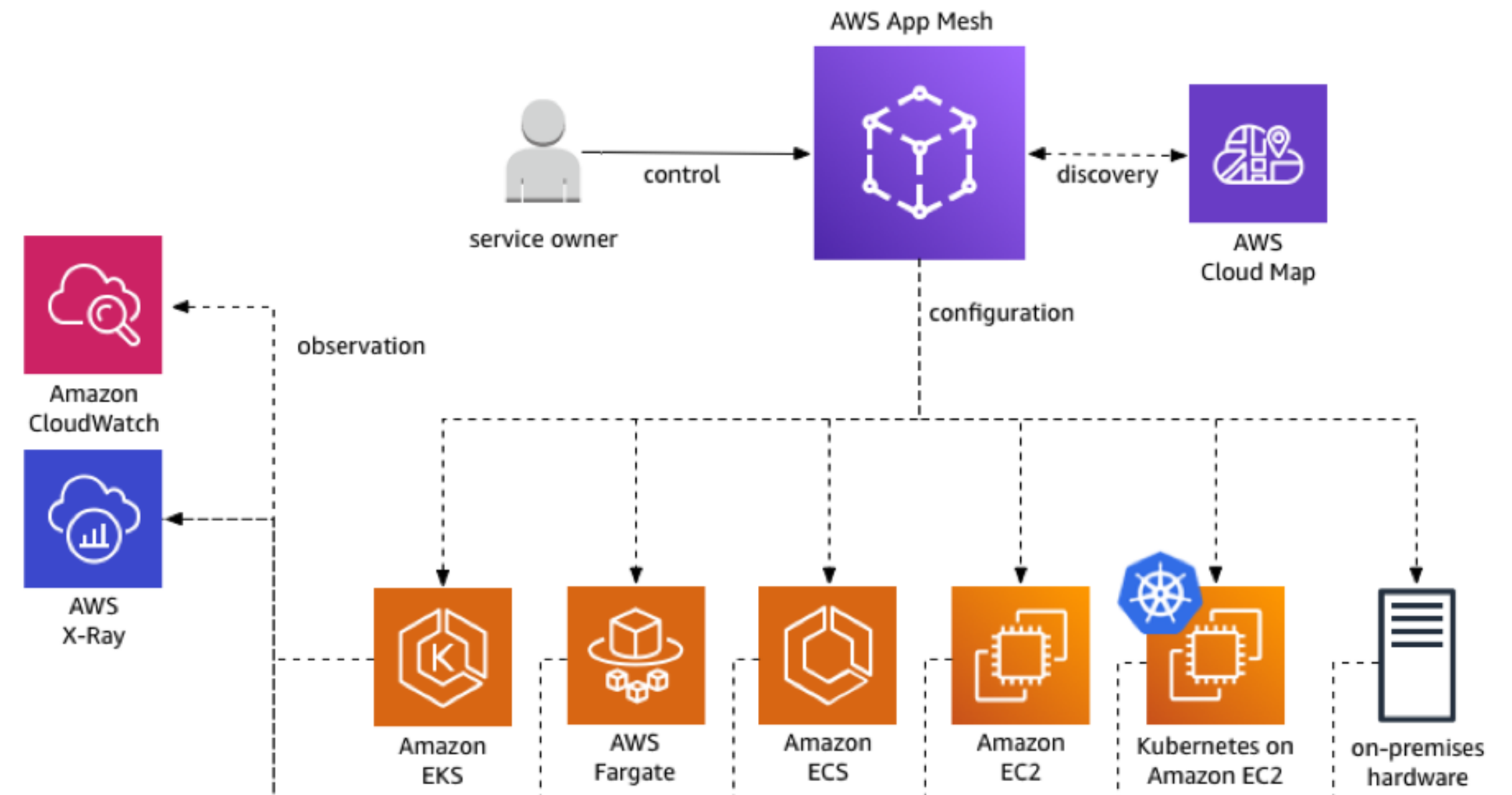
Linkerd2

- 目前开源产品中唯一正面对抗Istio
- 最新稳定版本 2.3 发布于 2019-04-17
- Buoyant 最近的B轮融资来自Google

AWS推出 App Mesh



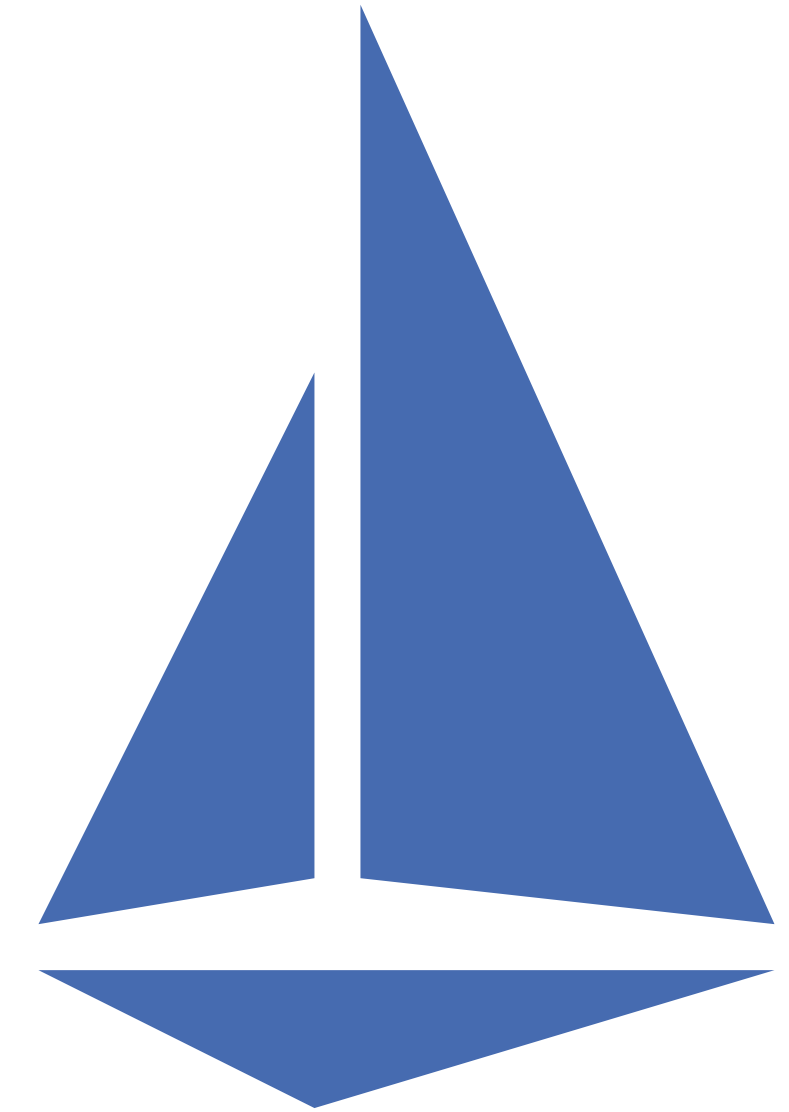
- 2019年4月, AWS宣布App Mesh GA
- 远景: AWS原生服务网格
- 与AWS完全集成
 - 网络 (AWS cloud map)
 - 计算 (Amazon EC2和AWS Fargate)
 - 编排工具 (AWS EKS, Amazon ECS和EC2上客户管理的k8s)
- 数据平面采用 Envoy
- 支持VM和容器



Google推出 Cloud Service Mesh

- 之前: Istio on GKE
 - 2018年底, GKE上线一键集成Istio
 - 提供遥测、日志、负载均衡、路由和mTLS 安全能力
- GCSM: Istio的完全托管版本
 - 提供Istio开源版本的完整特性
 - 集成 stackdriver

在 GKE 管理控制台上点
"Enable Istio"

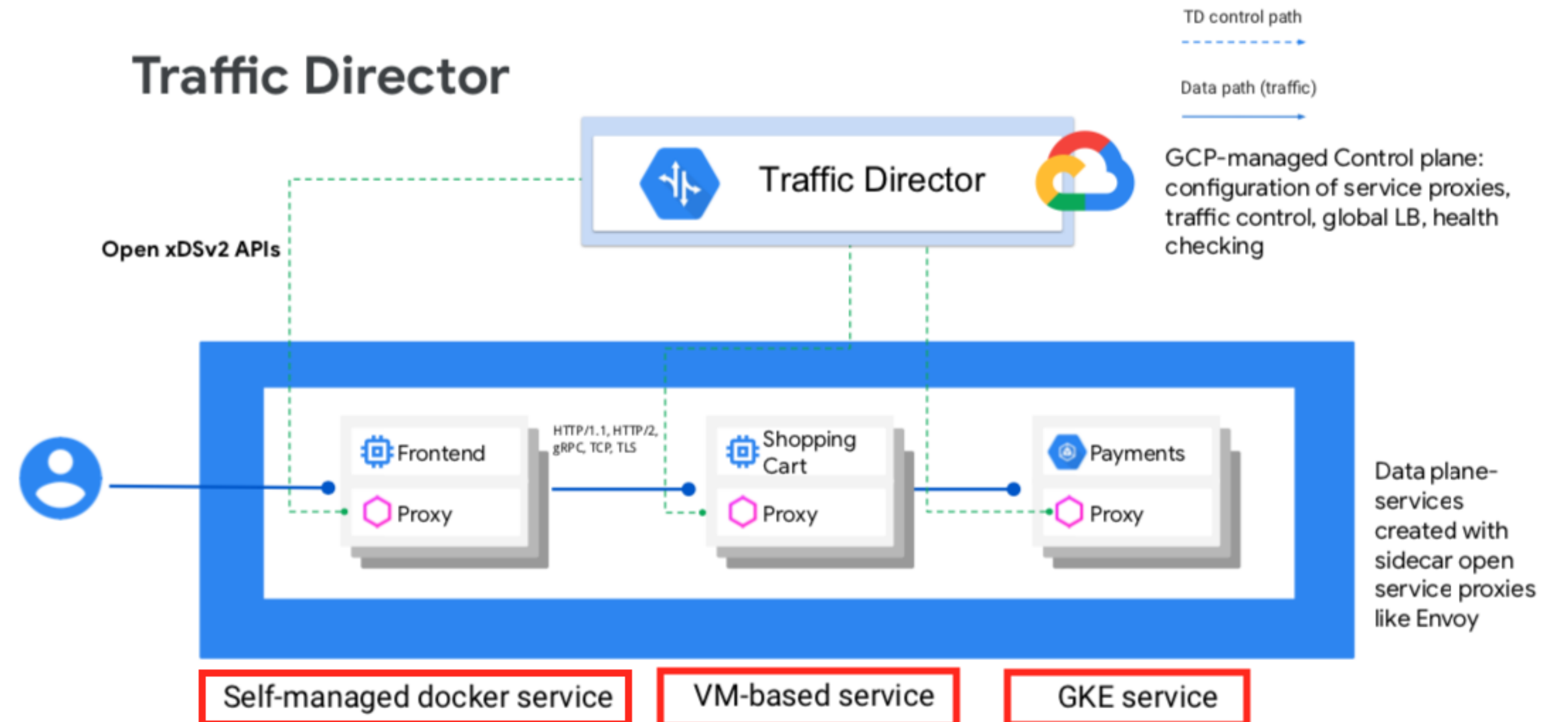


Google推出Traffic Director

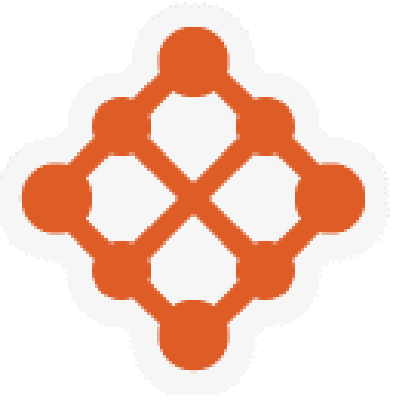


- 完全托管的服务网格流量控制平面 (beta)

- 支持全局负载均衡
- 适用于虚拟机和容器
- 混合云和多云支持
- 集中式健康检查
- 流量控制
- 基于流量的自动伸缩



微软推出Service Fabric Mesh

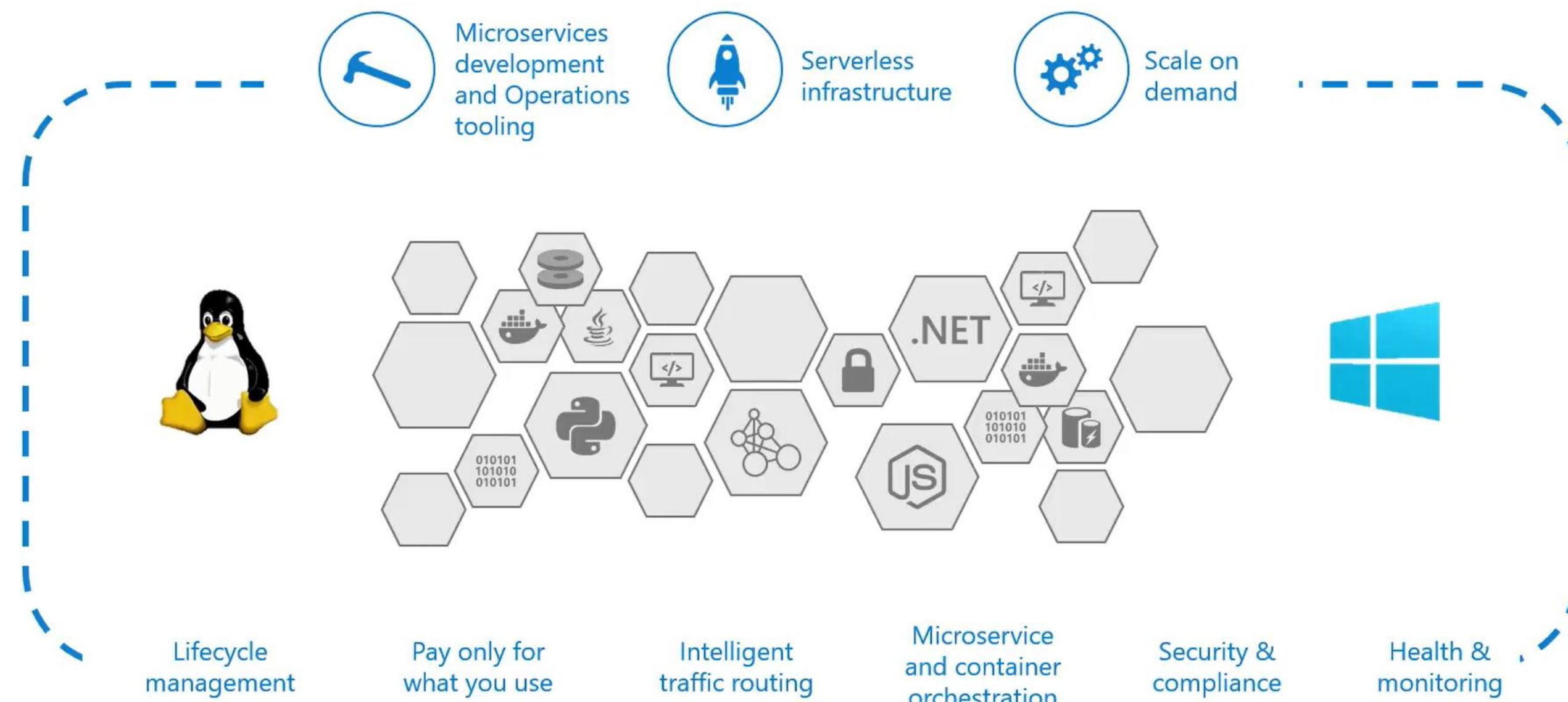


- Azure Service Fabric是Microsoft的微服务框架
- 设计用于公共云，内部部署以及混合和多云架构
- Service Fabric Mesh是Azure完全托管的
- 2018年8月推出，预览版

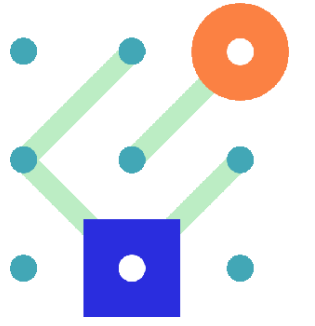


Azure Service Fabric Mesh

A fully-managed microservices platform for business critical applications



最新! 微软推出Service Mesh Interface



- SMI 是在 Kubernetes 上运行服务网格的规范




- 定义了由各种供应商实现的通用标准
- 使得最终用户的标准化和服务网格供应商的创新可以两全其美
- SMI 实现了灵活性和互通性

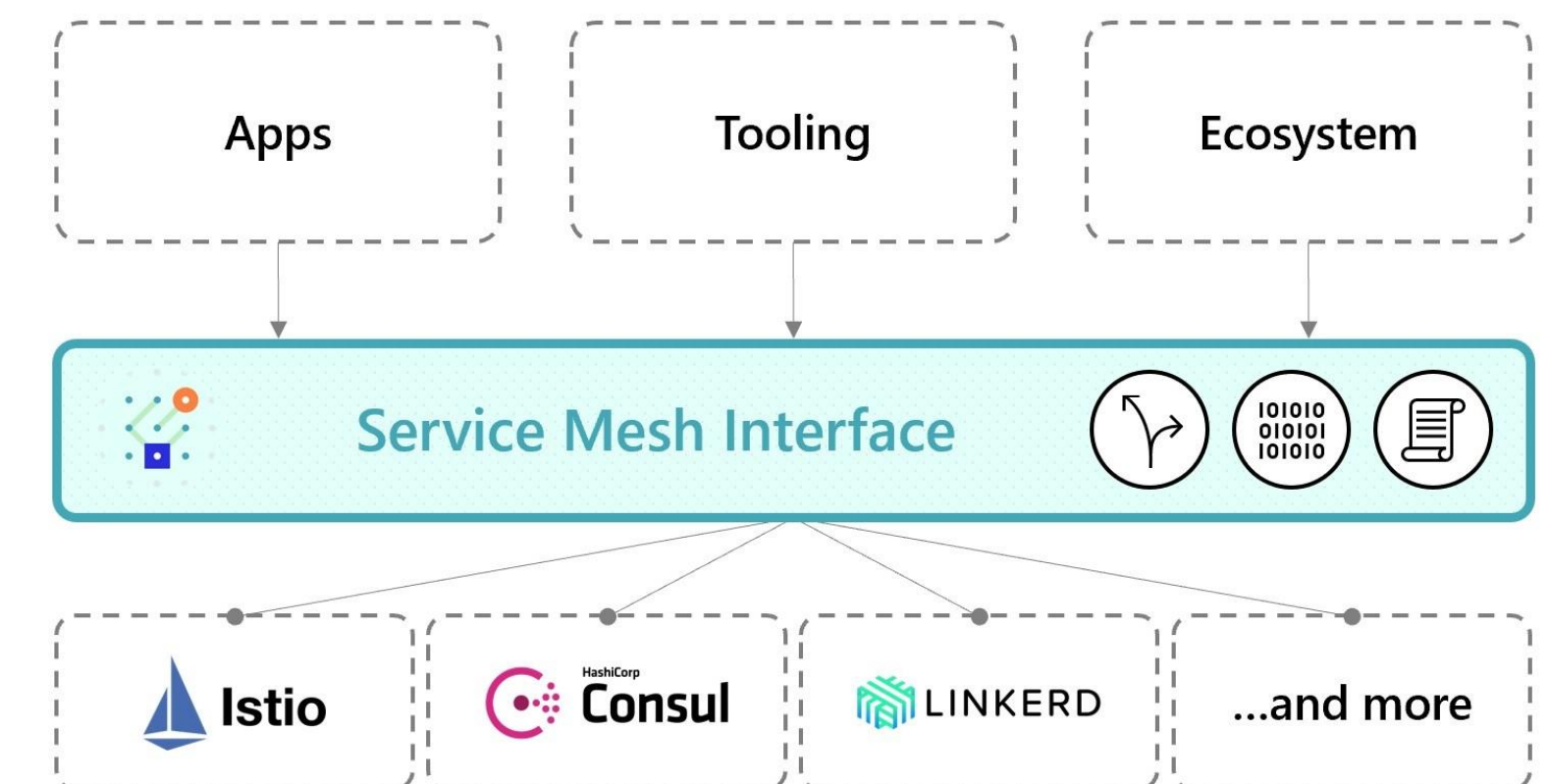
- SMI是一个开放项目

- 由微软, Linkerd, HashiCorp, Solo, Kinvolk和Weaveworks联合启动;
- 并得到了Aspen Mesh, Canonical, Docker, Pivotal, Rancher, Red Hat和VMware的支持。

Service Mesh Interface (SMI)

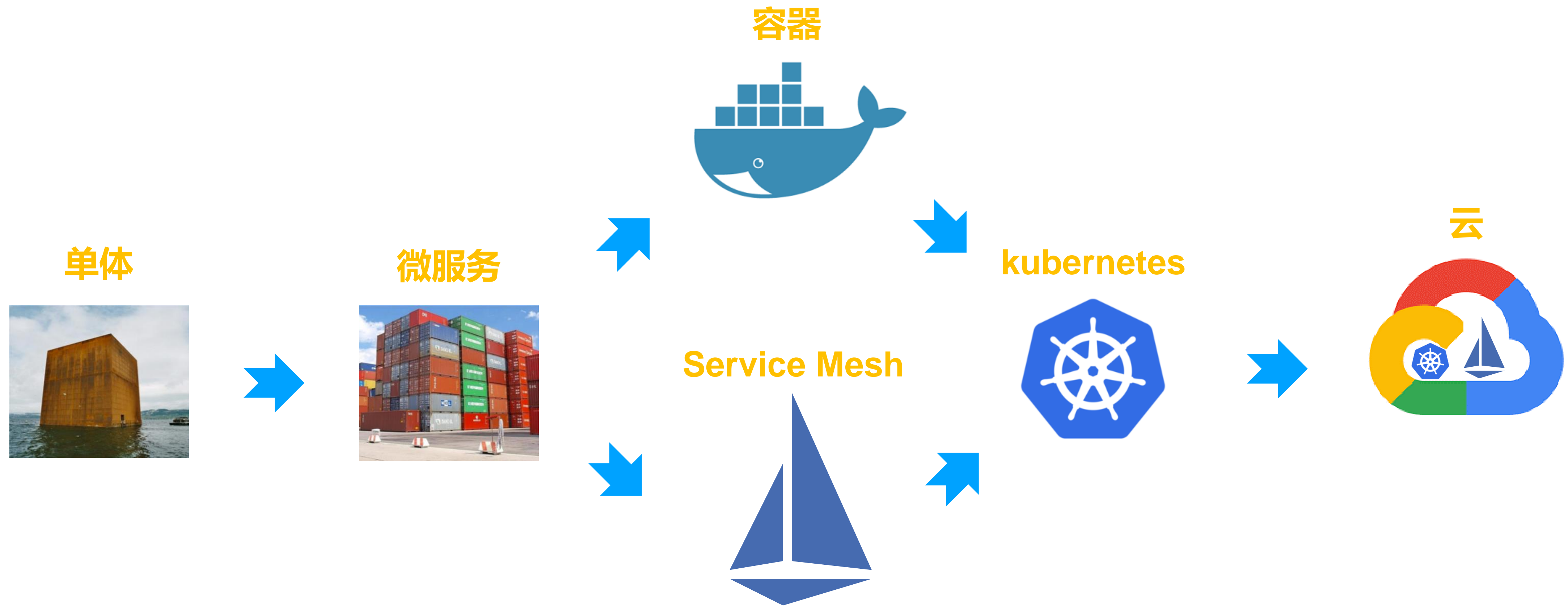
A Kubernetes interface that provides traffic routing, traffic telemetry, and traffic policy

-  **Standardized**
Standard interface for service mesh on Kubernetes
-  **Simplified**
Basic feature set to address most common scenarios
-  **Extensible**
Support for new features as they become widely available



- ❖ Service Mesh产品动态
- ❖ Service Mesh发展趋势
- ❖ Service Mesh与云原生

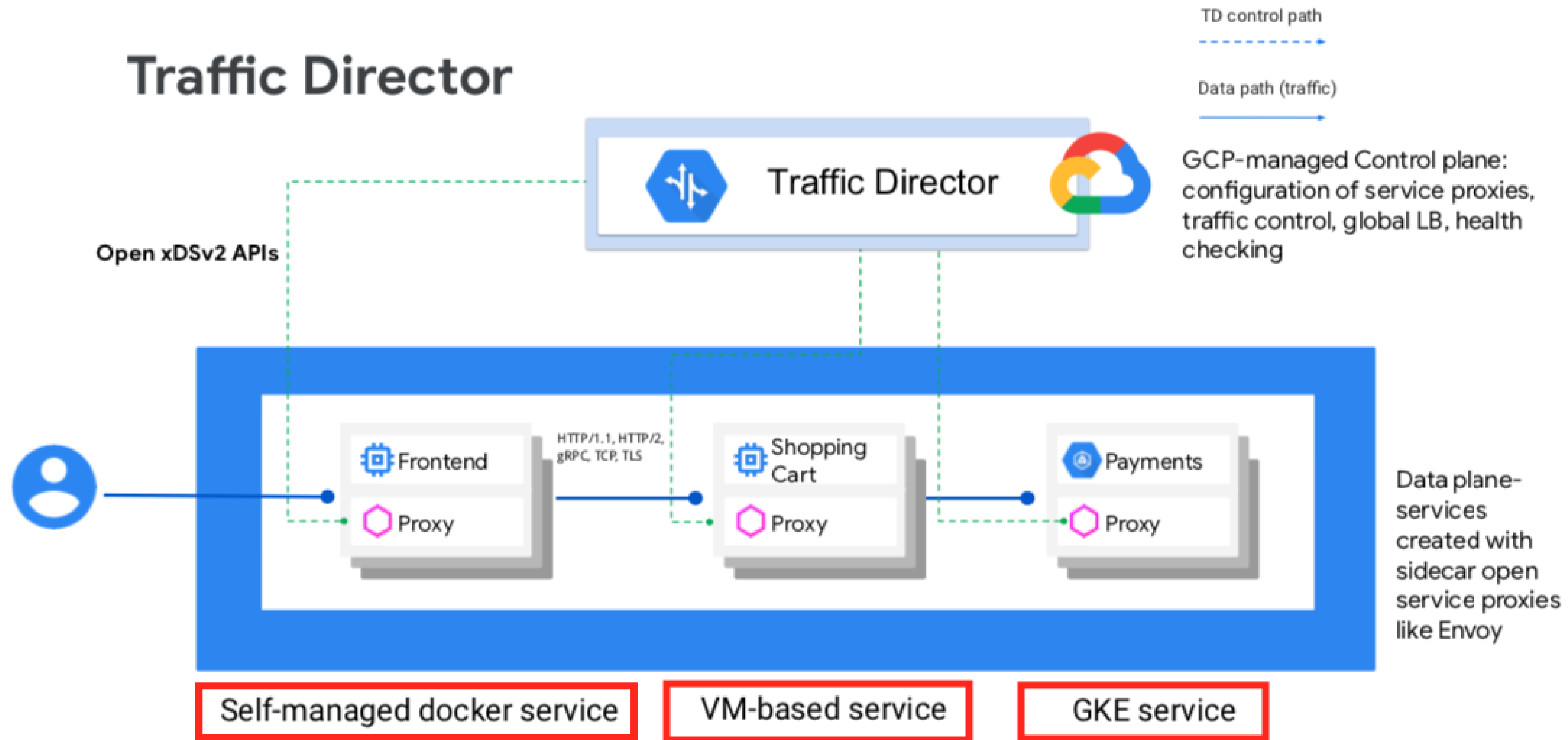
趋势1：上云+托管



几乎所有的主要公有云提供商都在提供（或者准备提供）

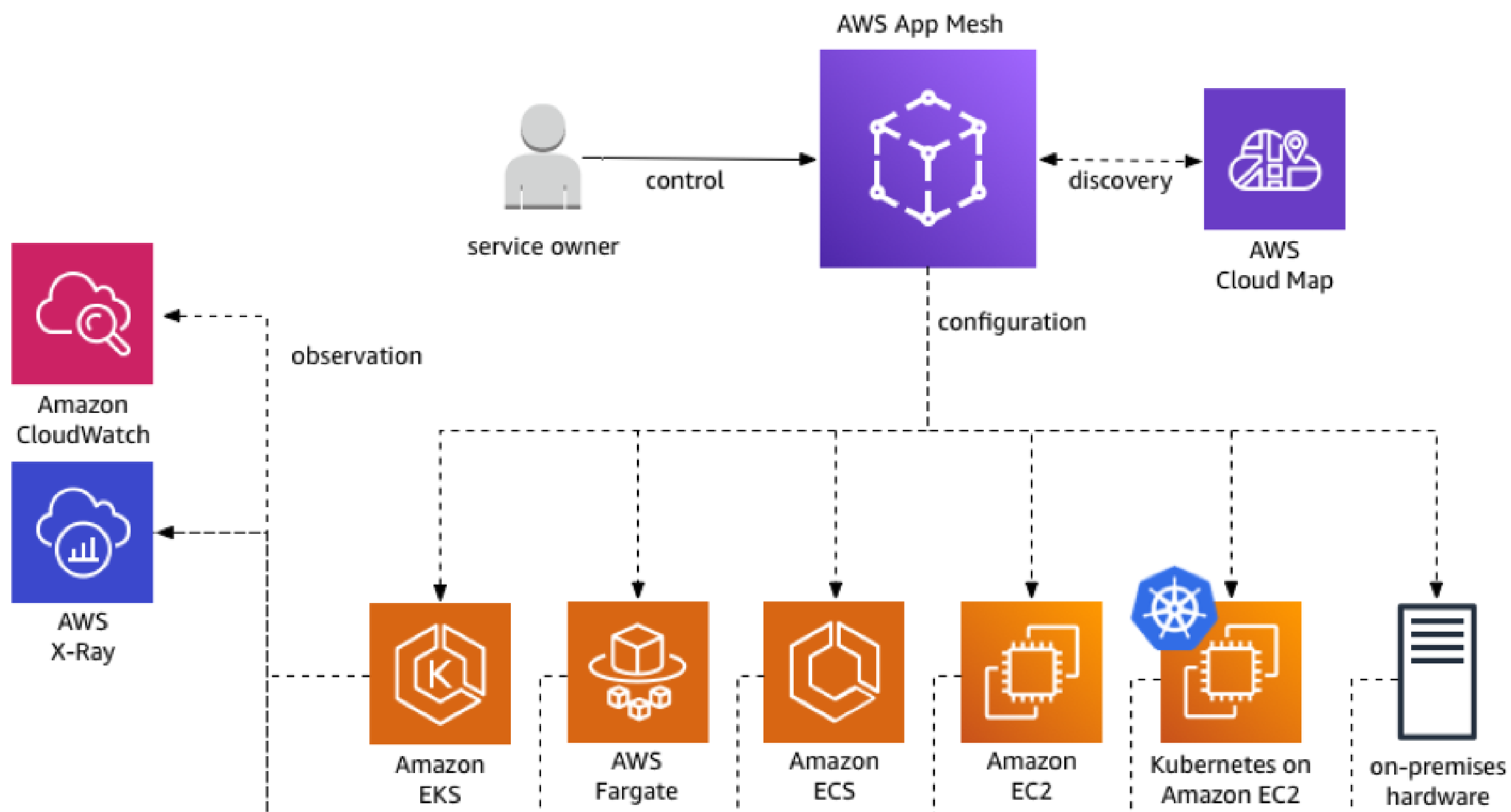
Service Mesh托管方案

趋势2：VM和容器混用



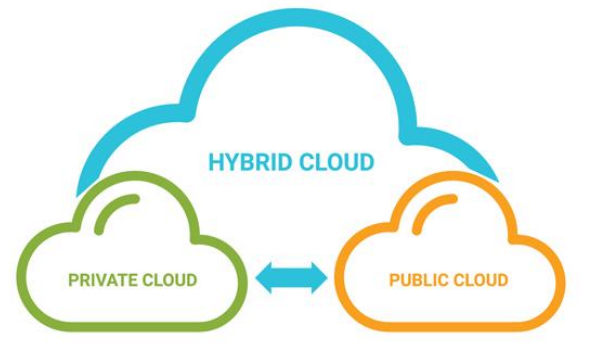
Google Traffic Director

VM和容器混用

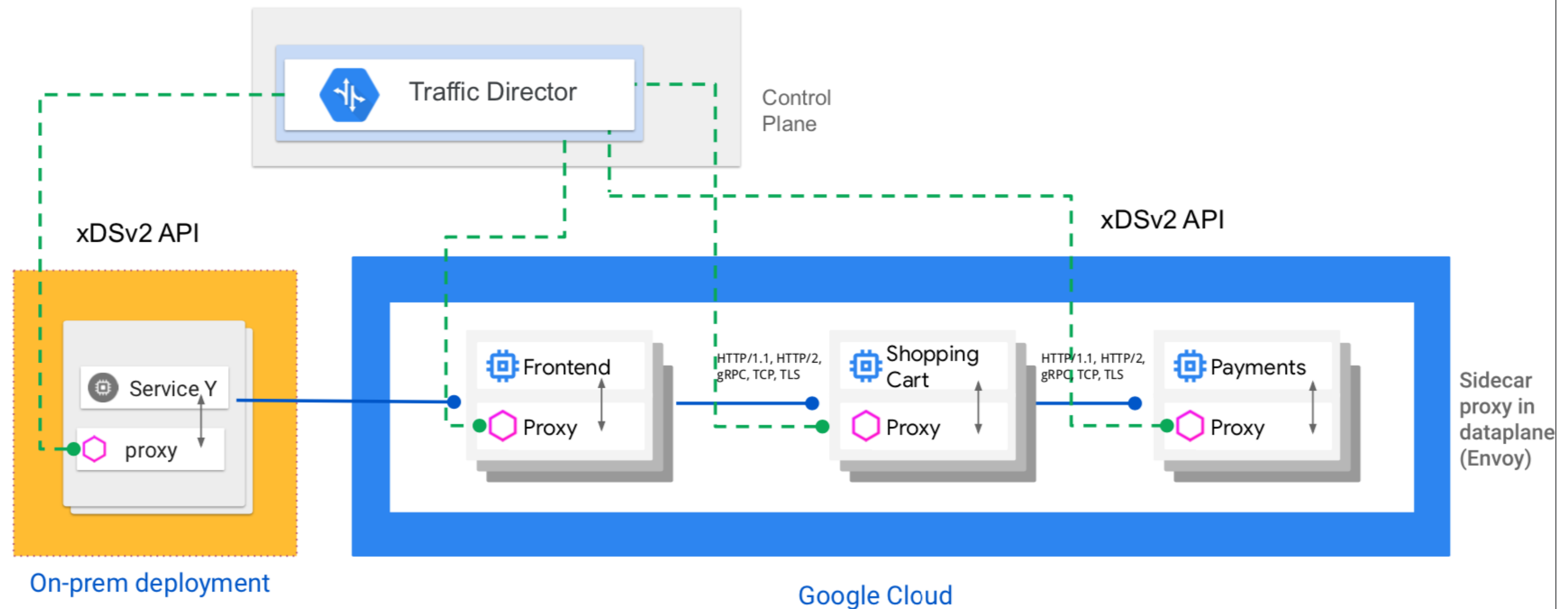


AWS App Mesh

趋势3：混合云和多云支持

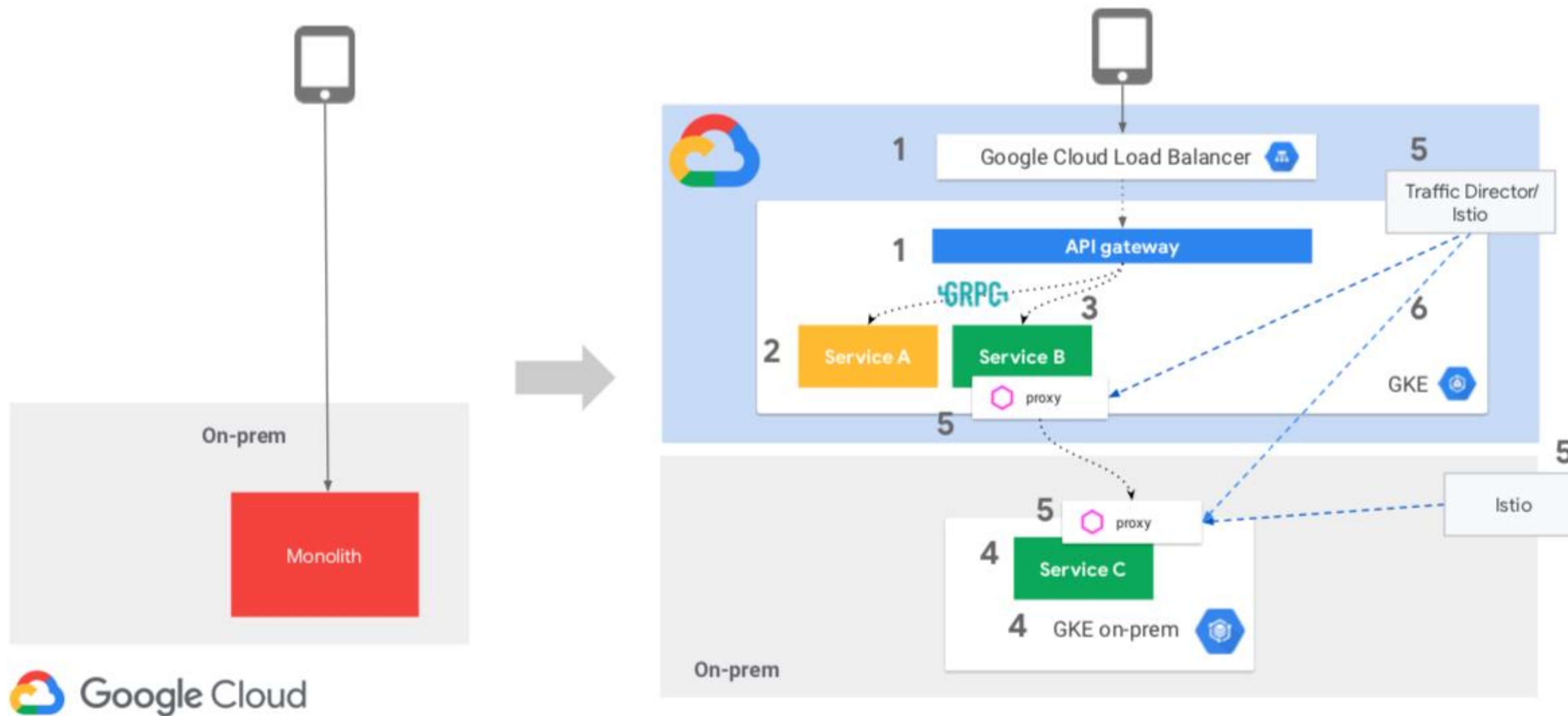
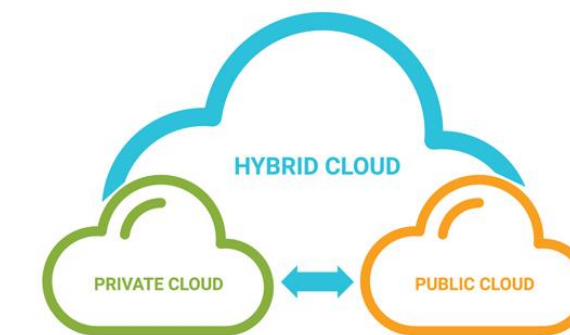


- Google Cloud: All in Hybrid Cloud!



Google Traffic Director支持混合云

混合云和多云支持



应用改造示例

单体 → 微服务
私有云 → 混合云

趋势4：和 serverless 的结合



应用减负
安全性
路由能力
策略执行
流量管理

Service Mesh

关注点：服务间通信

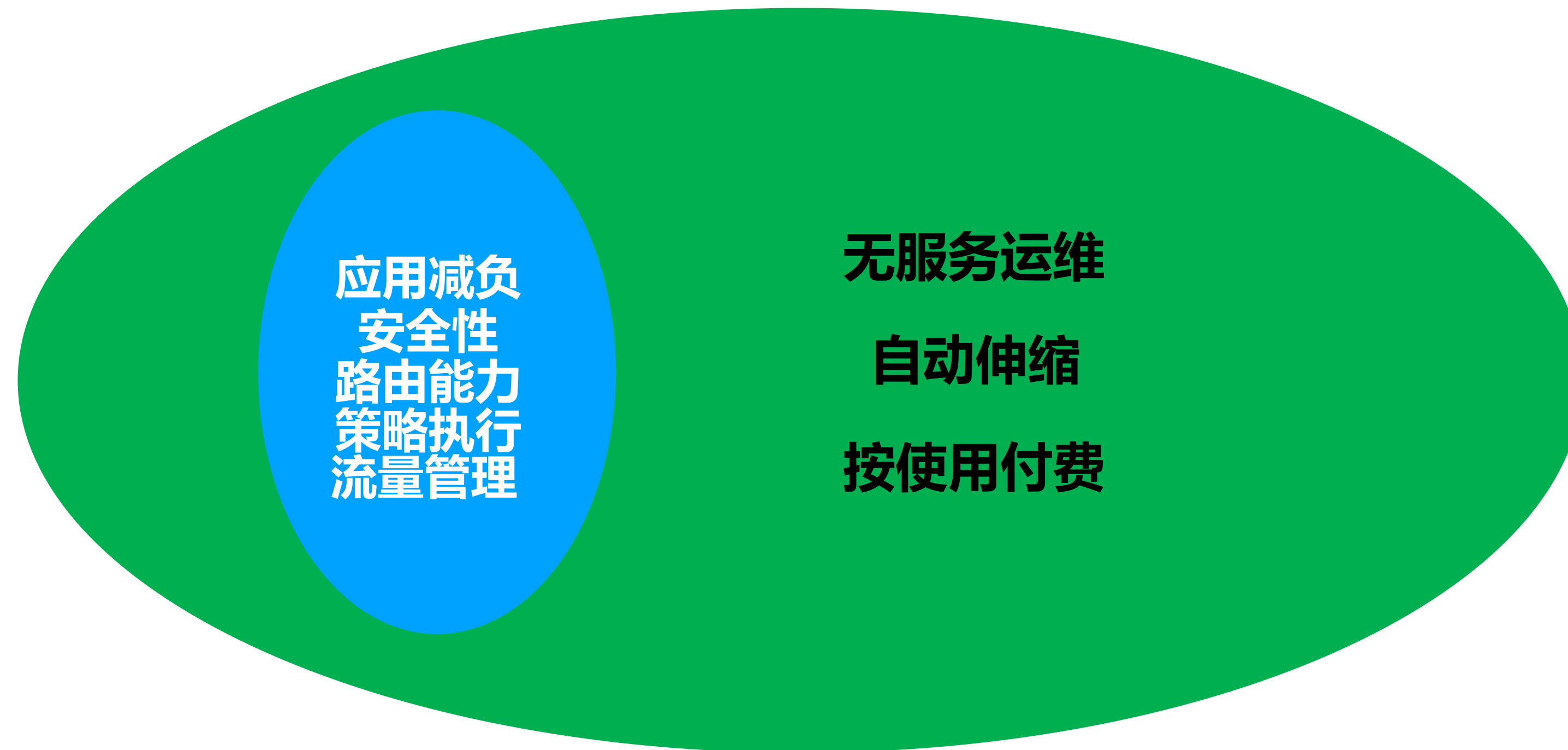


无服务运维
自动伸缩
按使用付费

Serverless

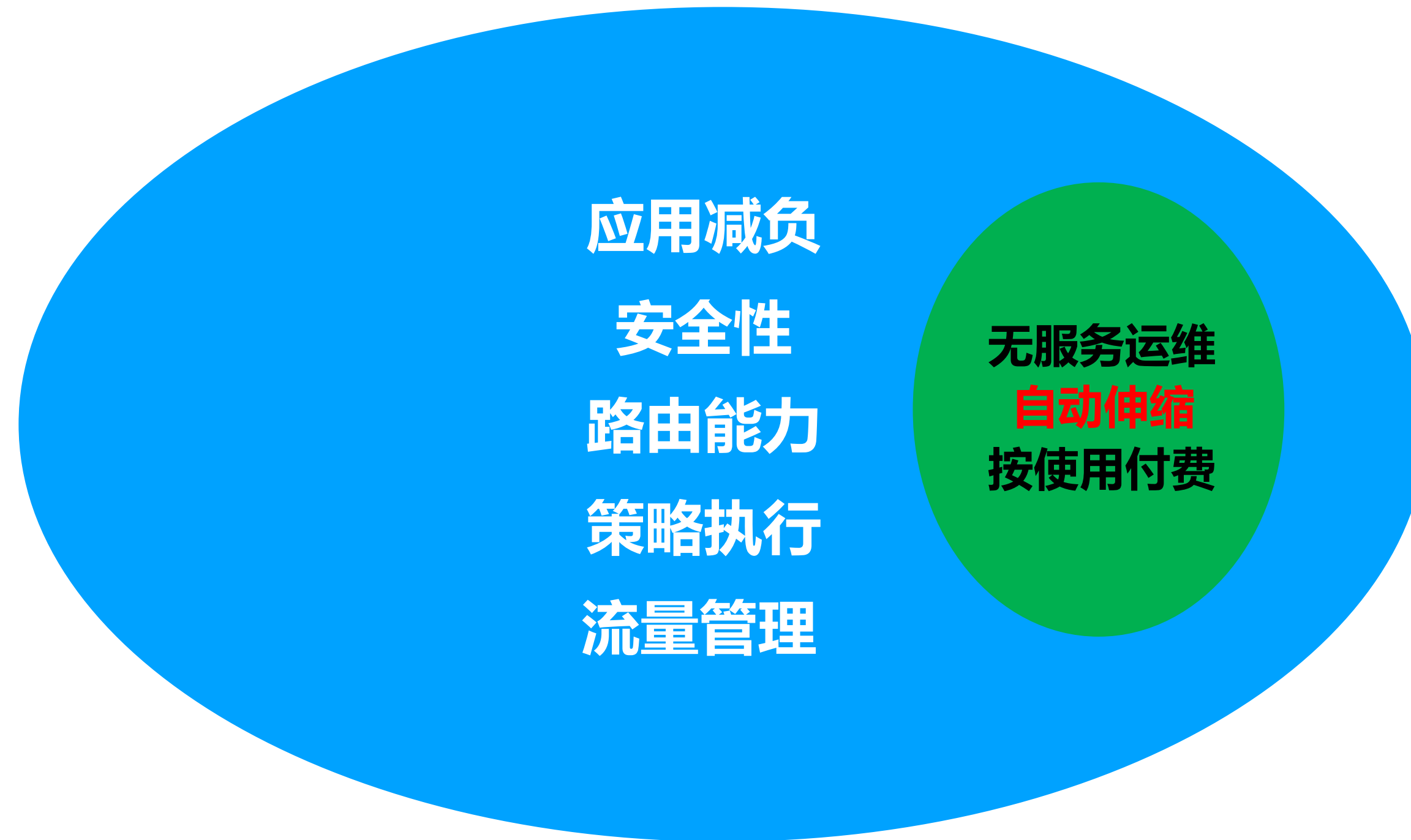
关注点：服务运维

融合方式：serverless中引入Servicemesh



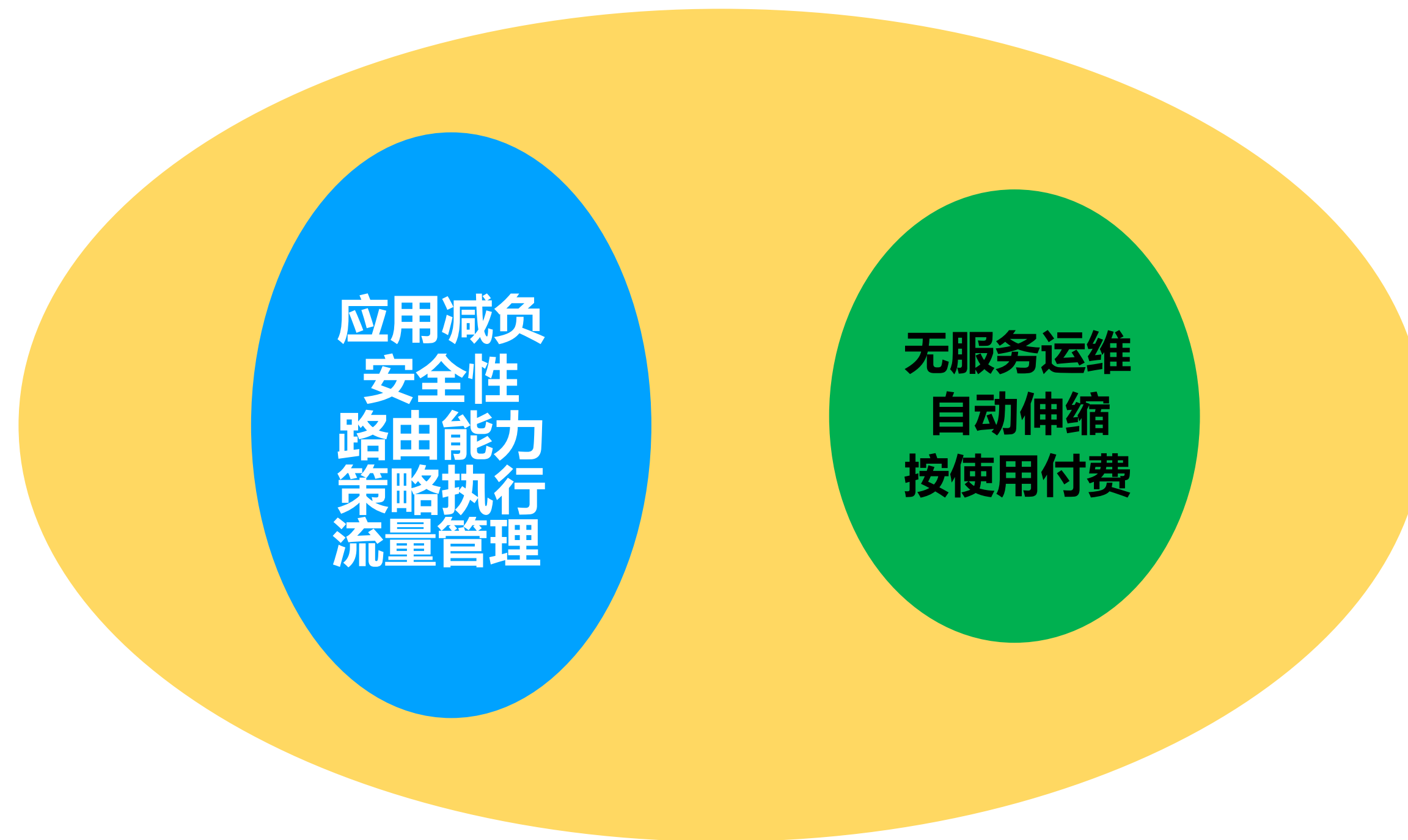
代表项目：Knative / Google Cloud Run

融合方式：Servicemesh中引入serverless



代表项目：Google Traffic Director

未来形态展望：合二为一的新型服务模式



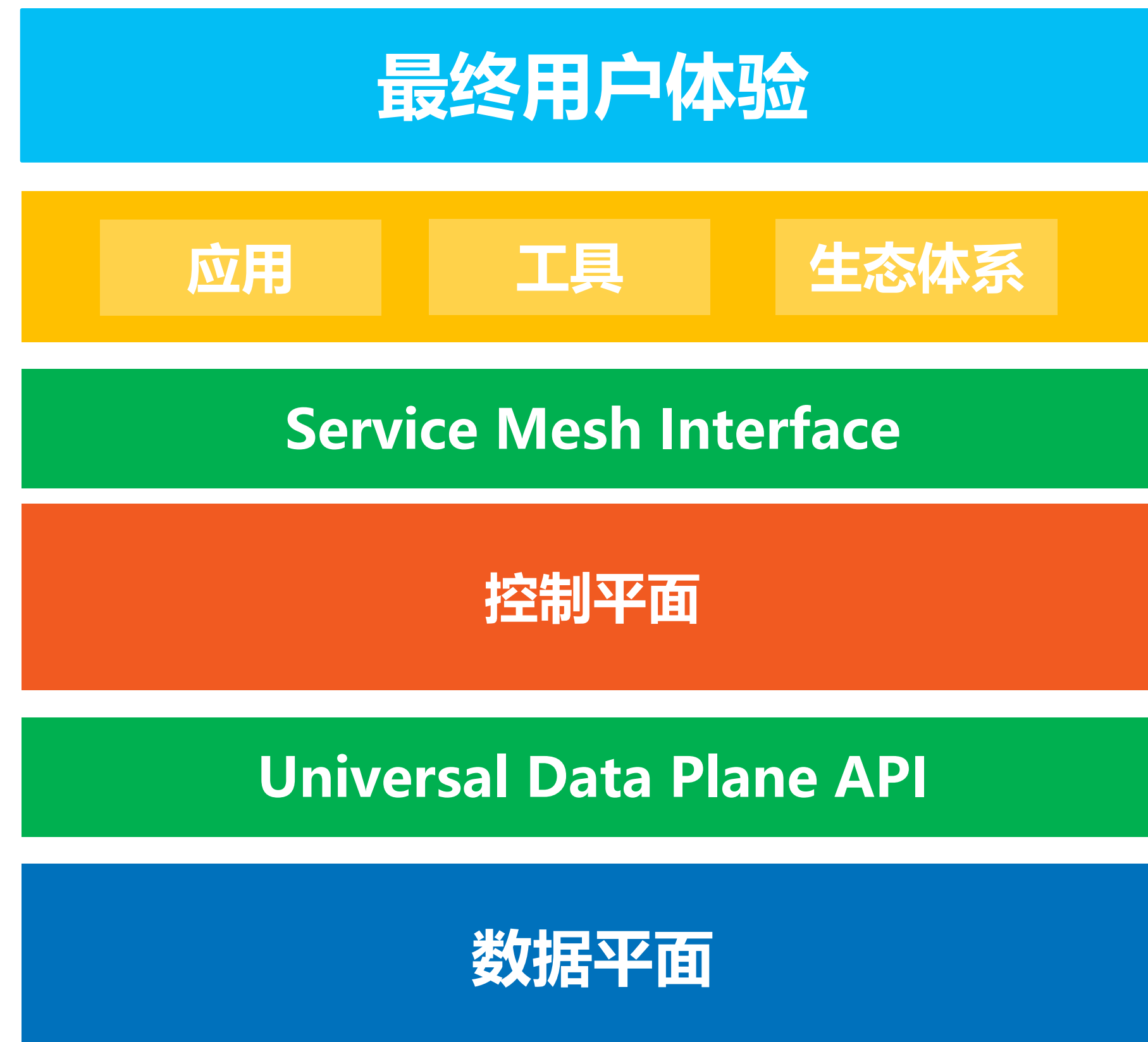
服务间通讯：ServiceMesh
服务运维：Serverless



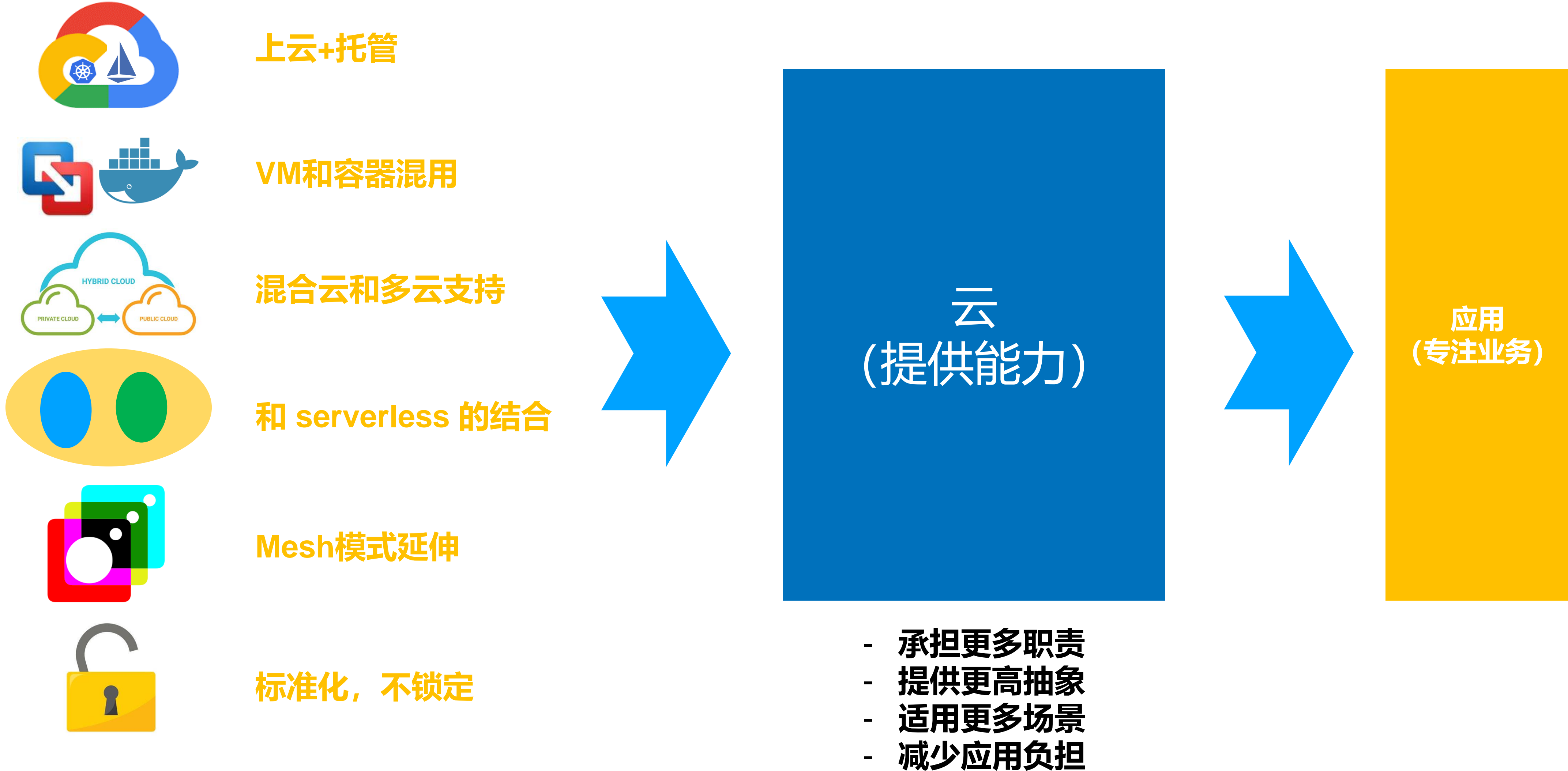
趋势5： Mesh模式延伸

- Service Mesh模式的核心
 - 原理：客户端SDK剥离，以Proxy独立进程运行
 - 目标：能力下沉，应用减负 → 应用云原生化
 - 特征：网络访问 + 客户端SDK
- 不仅仅适用于服务间同步通讯
 - Database Mesh: 数据库访问
 - Message Mesh: 消息
 - Cache Mesh: 缓存

趋势6：标准化，不锁定



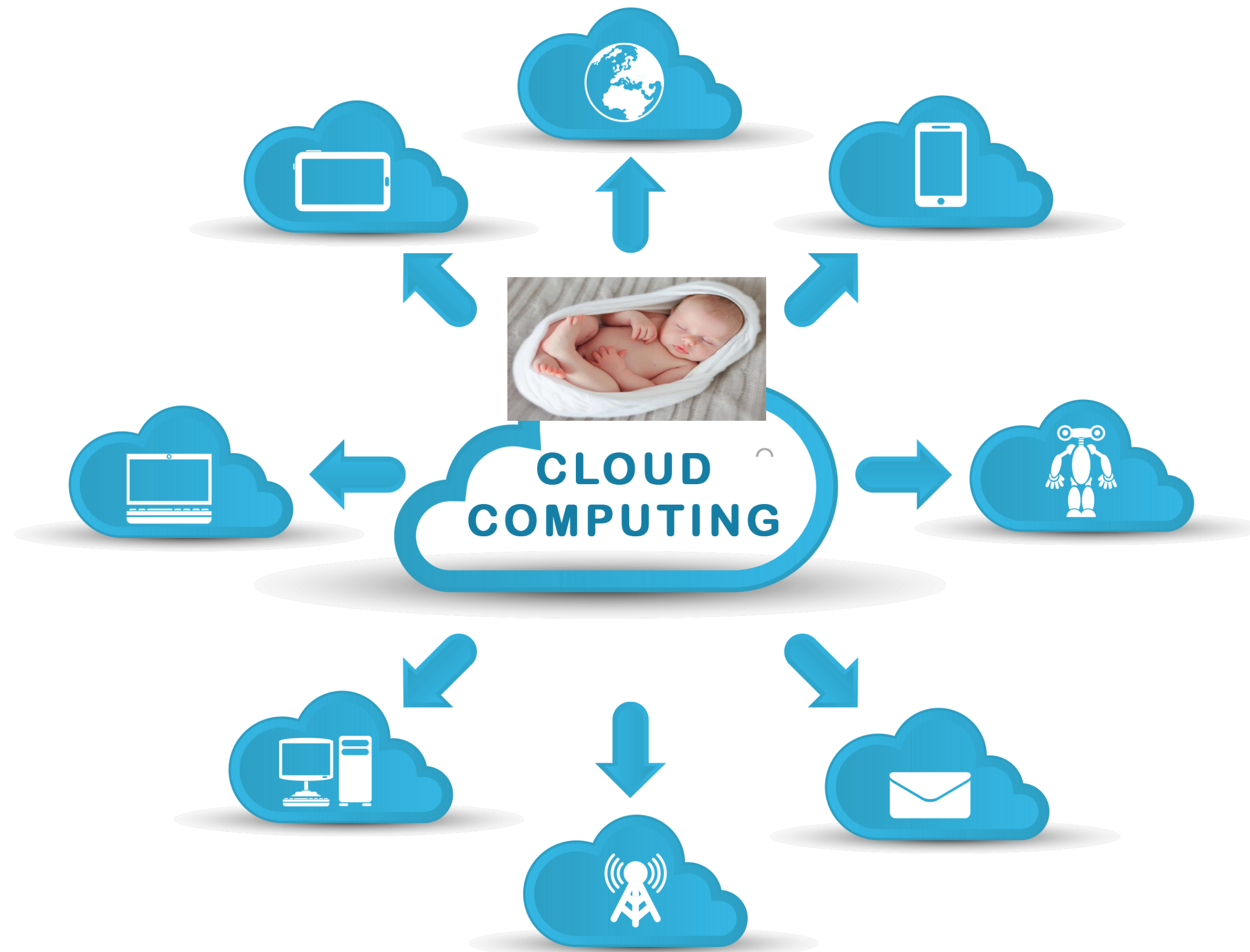
Service Mesh发展趋势分析



- ❖ Service Mesh产品动态
- ❖ Service Mesh发展趋势
- ❖ Service Mesh与云原生

什么是云原生?

云原生 → 原生为云设计



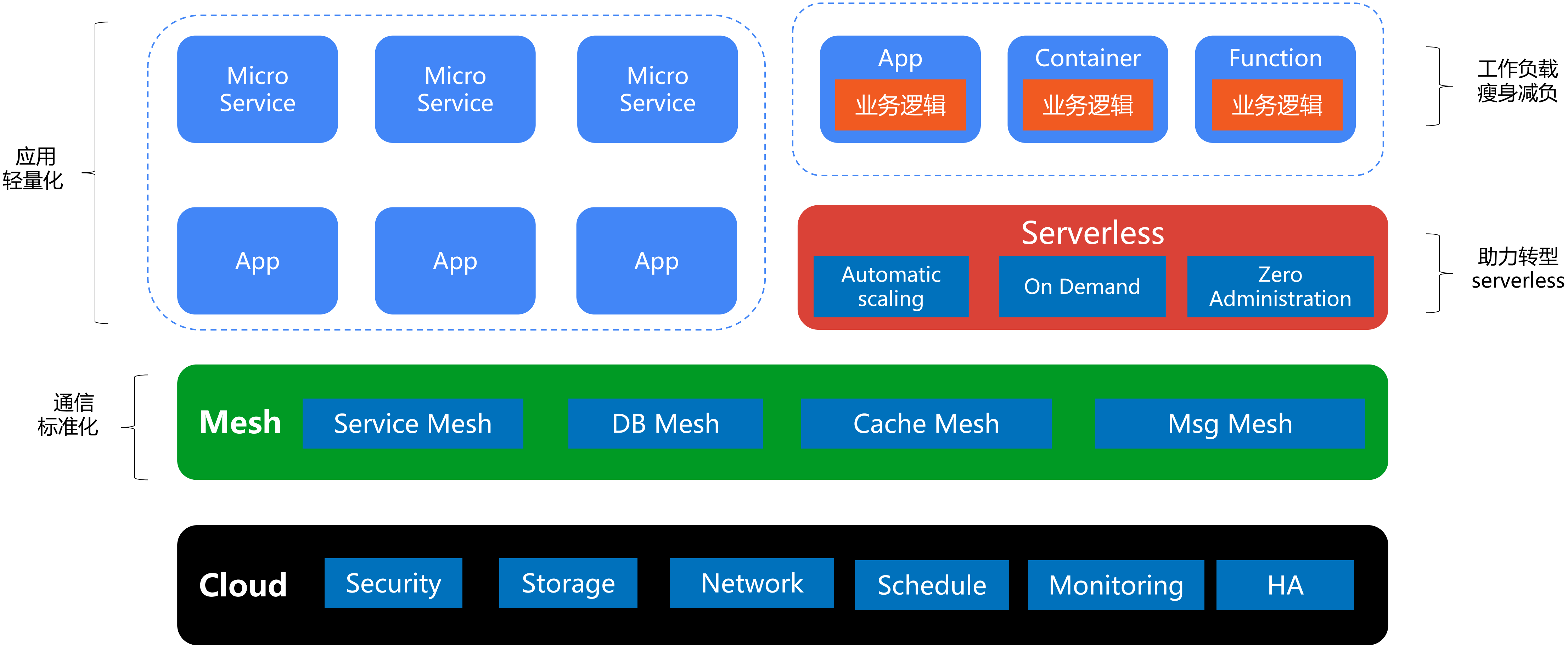
云原生：应用原生被设计为在云上以最佳方式运行，充分发挥云的优势。

Service Mesh的核心价值

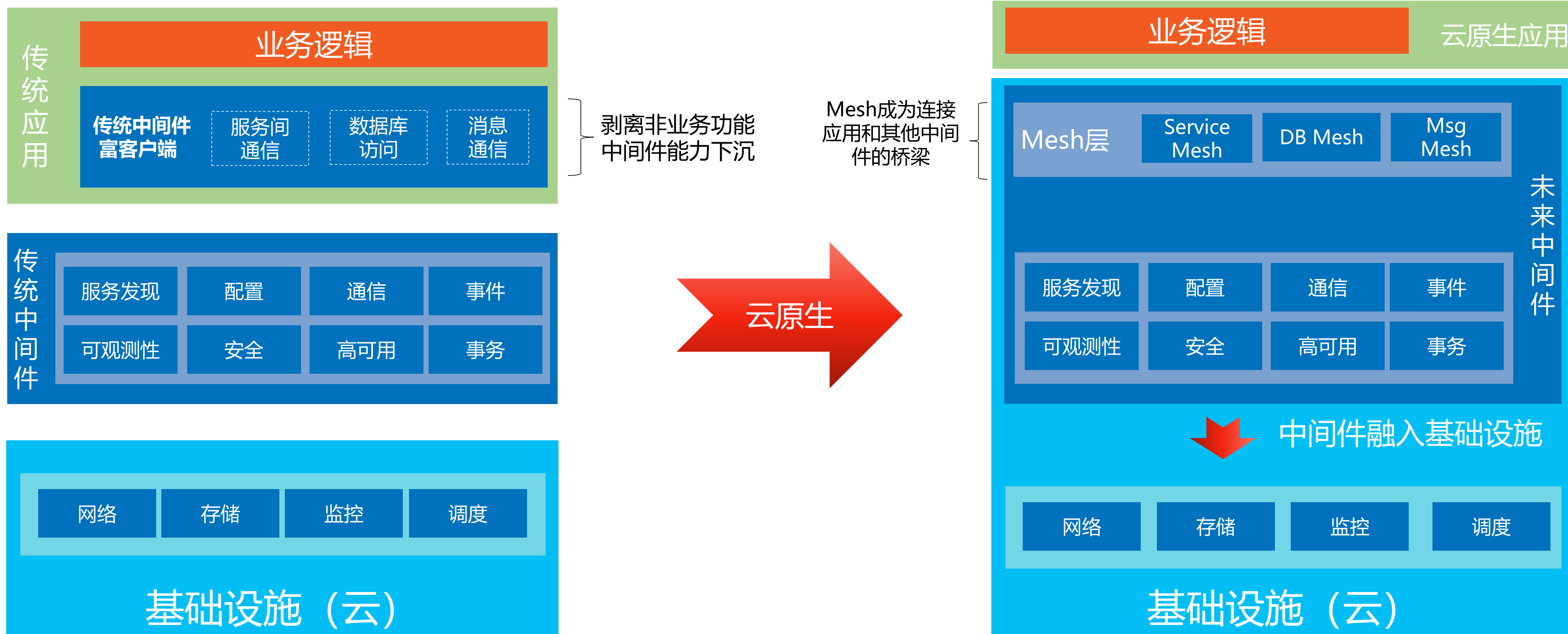
实现业务逻辑和非业务逻辑的分离

- 为下沉到基础设施提供可能
- 为上云提供可能
- 为应用轻量化提供可能

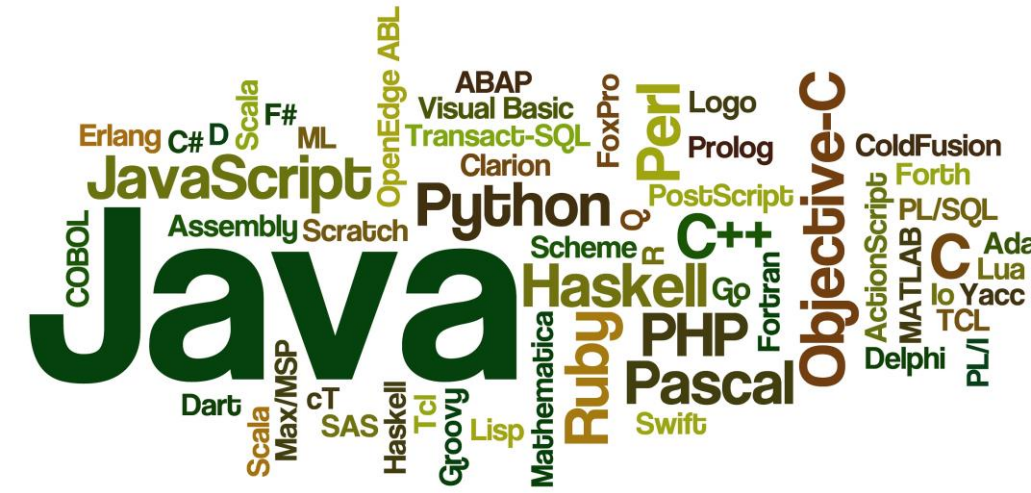
Mesh化是云原生落地的关键步骤



中间件的未来：Mesh化，融入基础设施



Service Mesh发展展望



多语言支持困难



类库升级困难

原始动力



上云+托管

VM和容器混用

混合云和多云支持

和 serverless 的结合

Mesh模式延伸

标准化, 不锁定

新动力, 新方向, 新目标

欢迎参加 SOFAShark 云原生工作坊

KubeConf 上海

日期: 2019年6月24日

时间: 9:00-16:00

地点: 上海世博中心

金融级分布式架构 SOFA...

410人



扫一扫群二维码，立刻加入该群。

请使用钉钉扫描二维码



SOFAShark (Scalable Open Financial Architecture Stack) 是蚂蚁金服自主研发并开源的金融级分布式架构，包含了构建金融级云原生架构所需的各个组件，是在金融场景里锤炼出来的最佳实践。

参加此次 Meetup 您将获得：

- 基于 SOFAShark 快速构建微服务
- 金融场景下的分布式事务最佳实践
- 基于 Kubernetes 的云原生部署体验
- 云上的 Service Mesh 基本使用场景体验
- 基于 Serverless 轻松构建云上应用

Thank you!



Kubernetes &

CloudNative