

智能时代的新运维

CNUTCon

全球运维技术大会

蚂蚁金服 SOFAMesh 在多语言 上的探索实践

黄挺（鲁直）

蚂蚁金服高级技术专家、SOFA 开源负责人

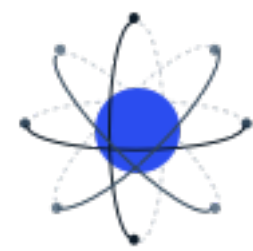
主办方 **Geekbang** **InfoQ**
极客邦科技

TABLE OF

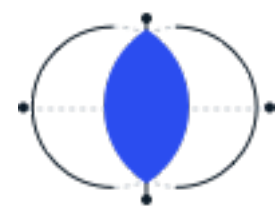
CONTENTS 大纲

- 多语言在蚂蚁金服的发展
- 蚂蚁金服 SOFAMesh 介绍
- SOFAMesh 多语言方案介绍
- 用 SOFAMesh 解决多语言问题的技术要点

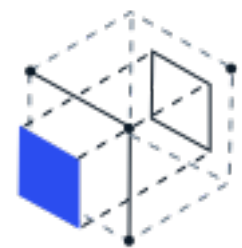
蚂蚁金服多语言发展



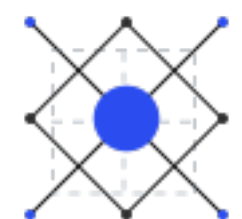
SOFA 微服务体系



SOFA 分布式事务

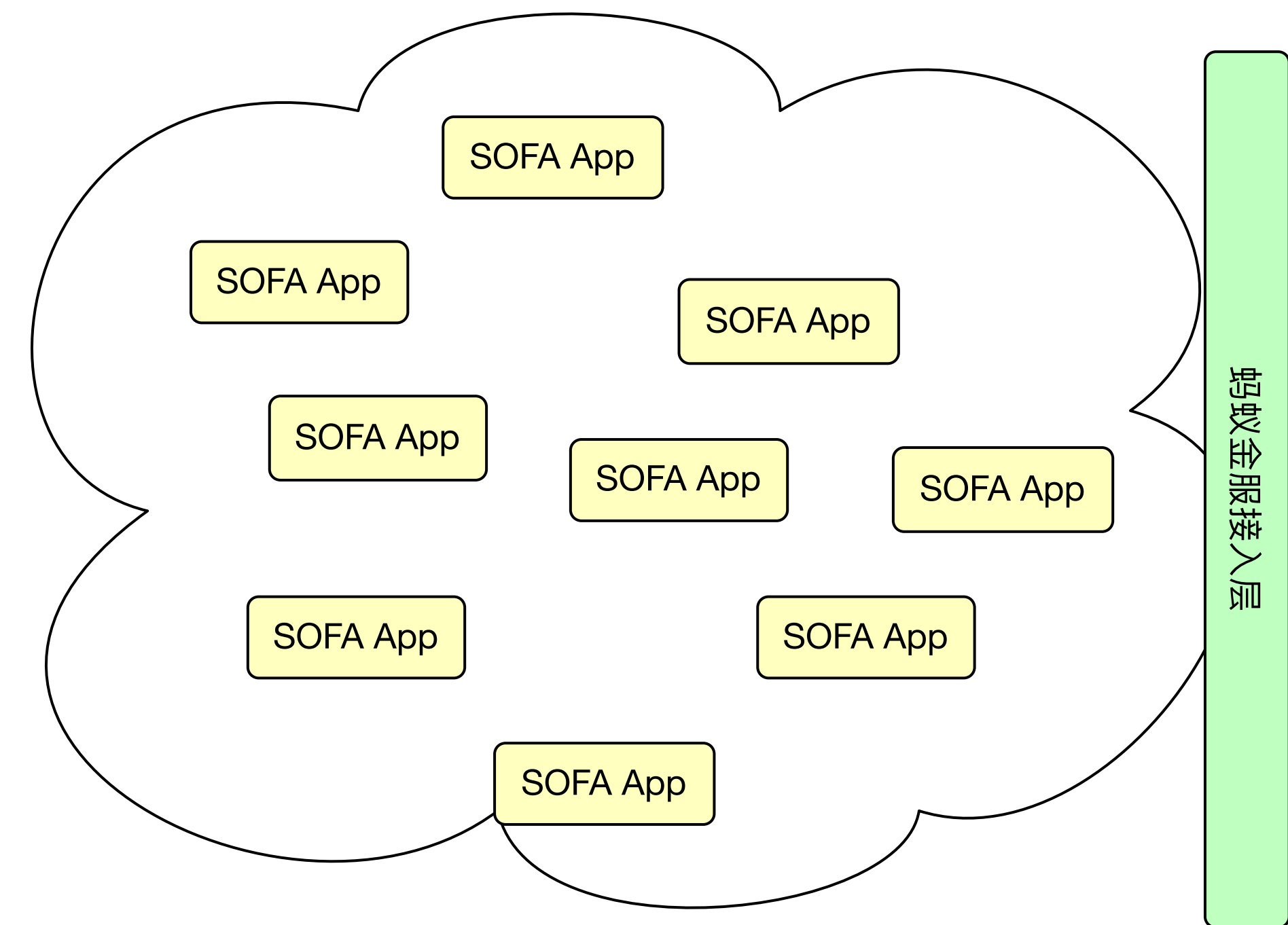


SOFA 消息中间件



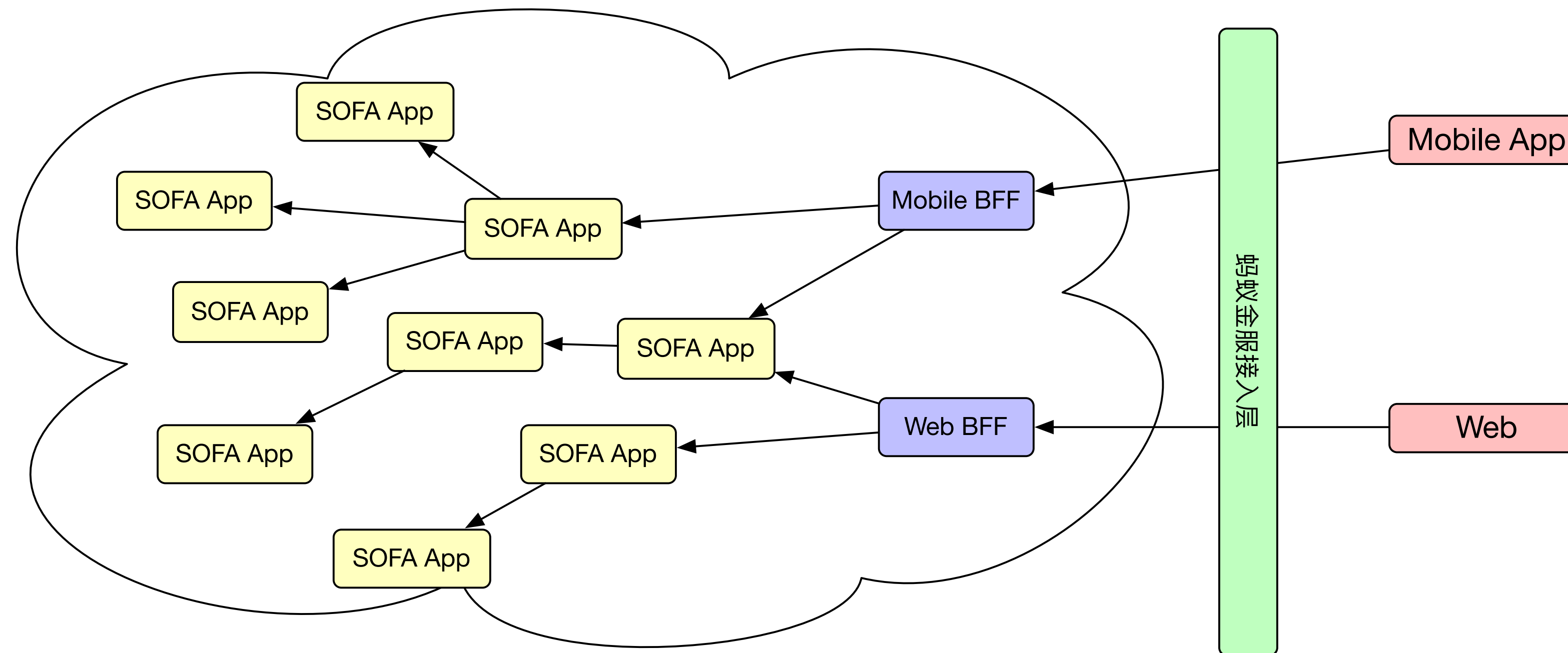
SOFA 数据访问代理

<http://github.com/alipay>



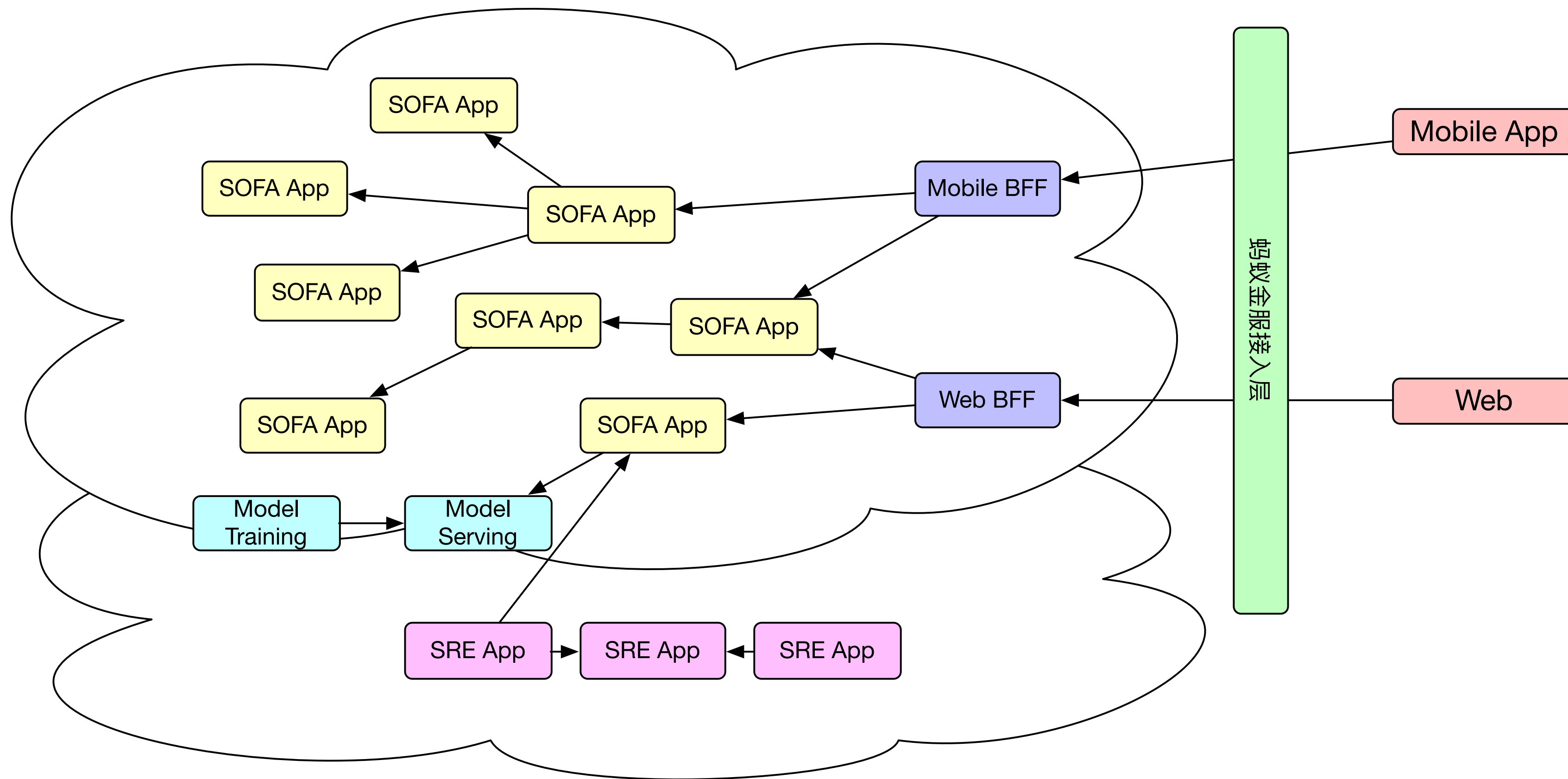
以 SOFA 中间件为中心的 Java 技术体系

蚂蚁金服多语言发展



EggJS (NodeJS) 作为 BFF 层

蚂蚁金服多语言发展



- C++/Python 作为 AI 领域的第一选择被引入
- 大量 SRE 相关系统采用 Python 来构建

多语言发展的问题

基础设施重复投入：NodeJS 这一层技术设施重复投入，导致大量的重复的消耗。

以 Java 为中心：功能特性研发往往不会考虑多语言的场景，大多数中间件/框架以 Java 为中心，被其他语言的使用推广构建了很大的壁垒。

多语言发展的问题

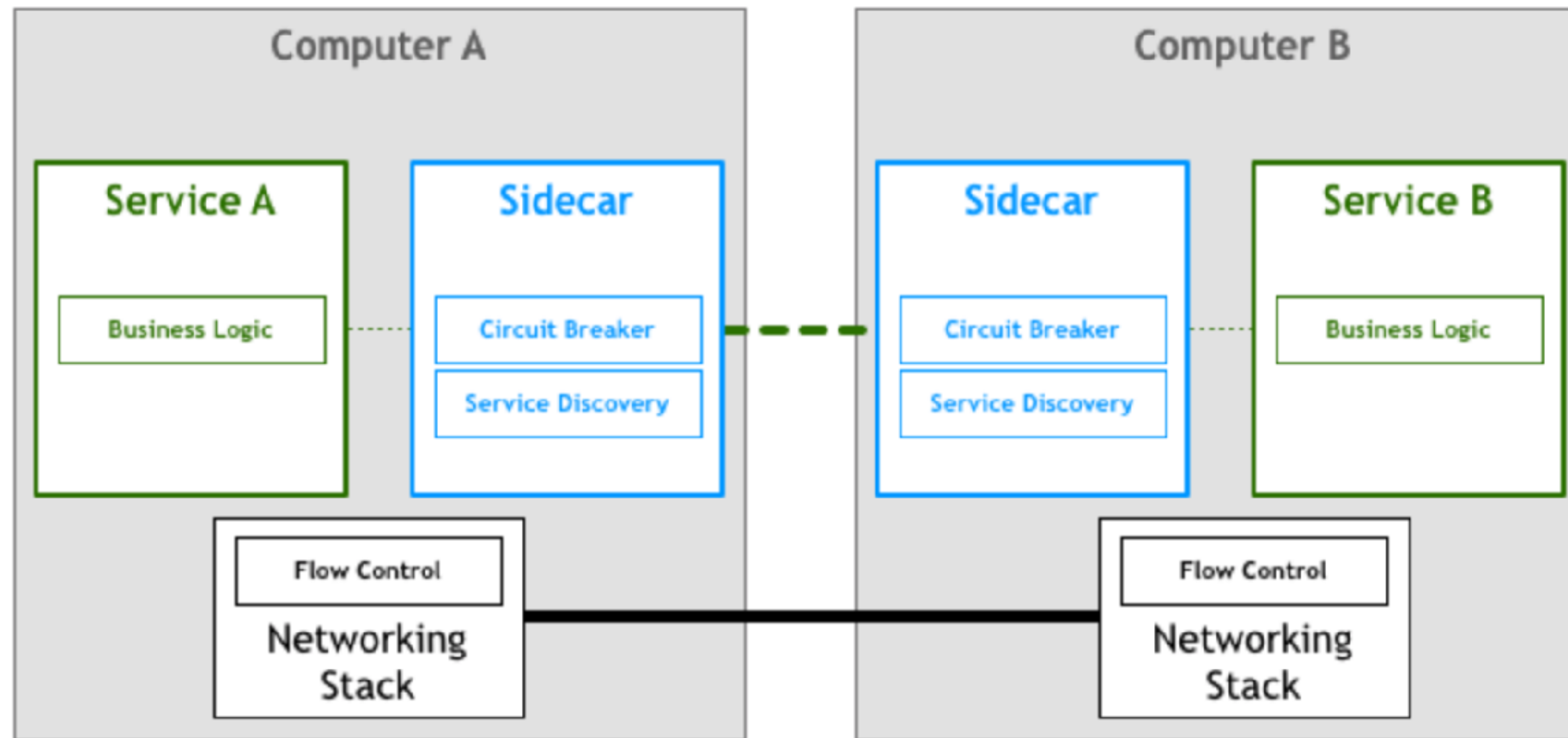
基础设施重复投入：NodeJS 这一层技术设施重复投入，导致大量的重复的消耗。

一次实现，到处可用

以 Java 为中心：功能特性研发往往不会考虑多语言的场景，大多数中间件/框架以 Java 为中心，被其他语言的使用推广构建了很大的壁垒。

保证语言的中立性

Service Mesh

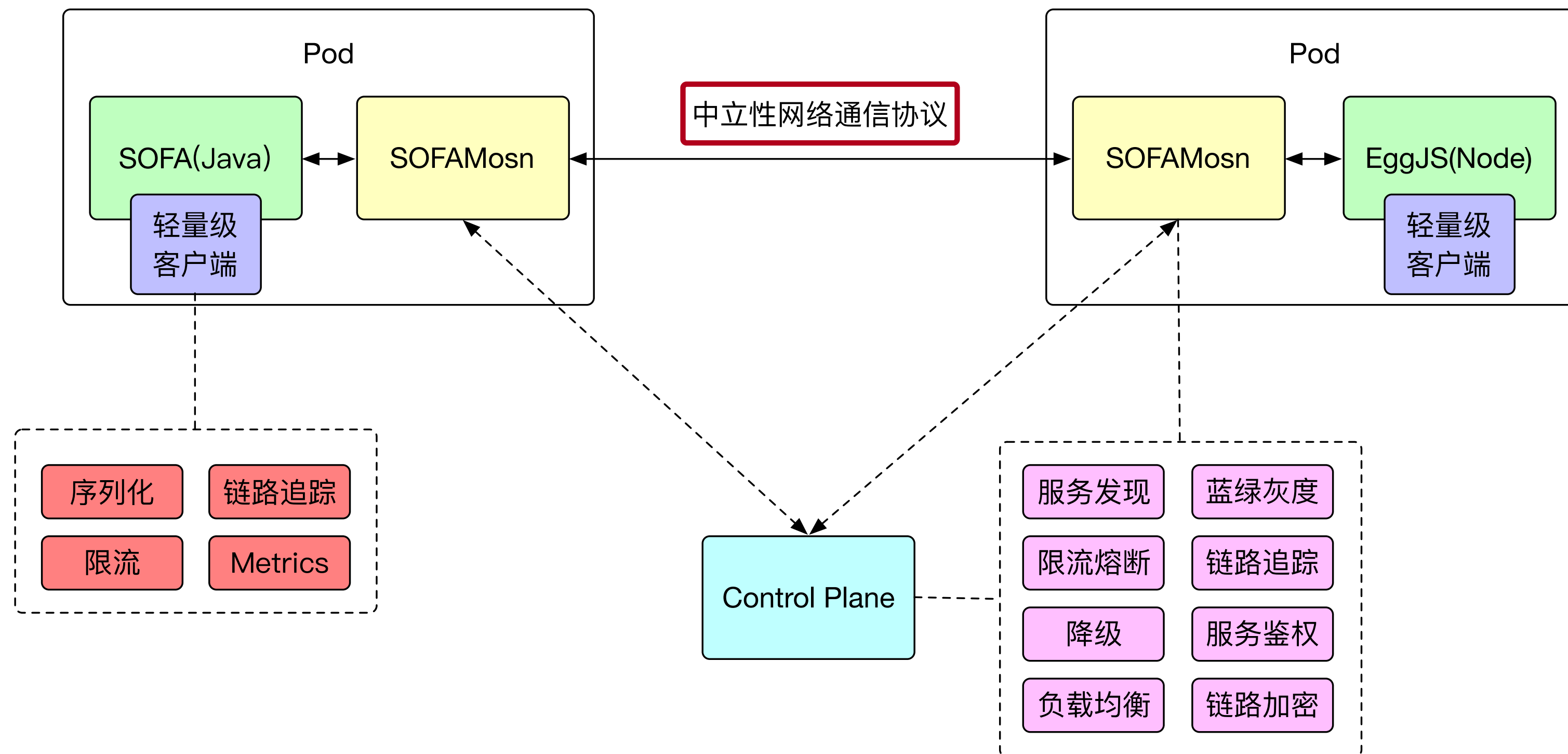


图片来源: http://philcalcado.com/2017/08/03/pattern_service_mesh.html



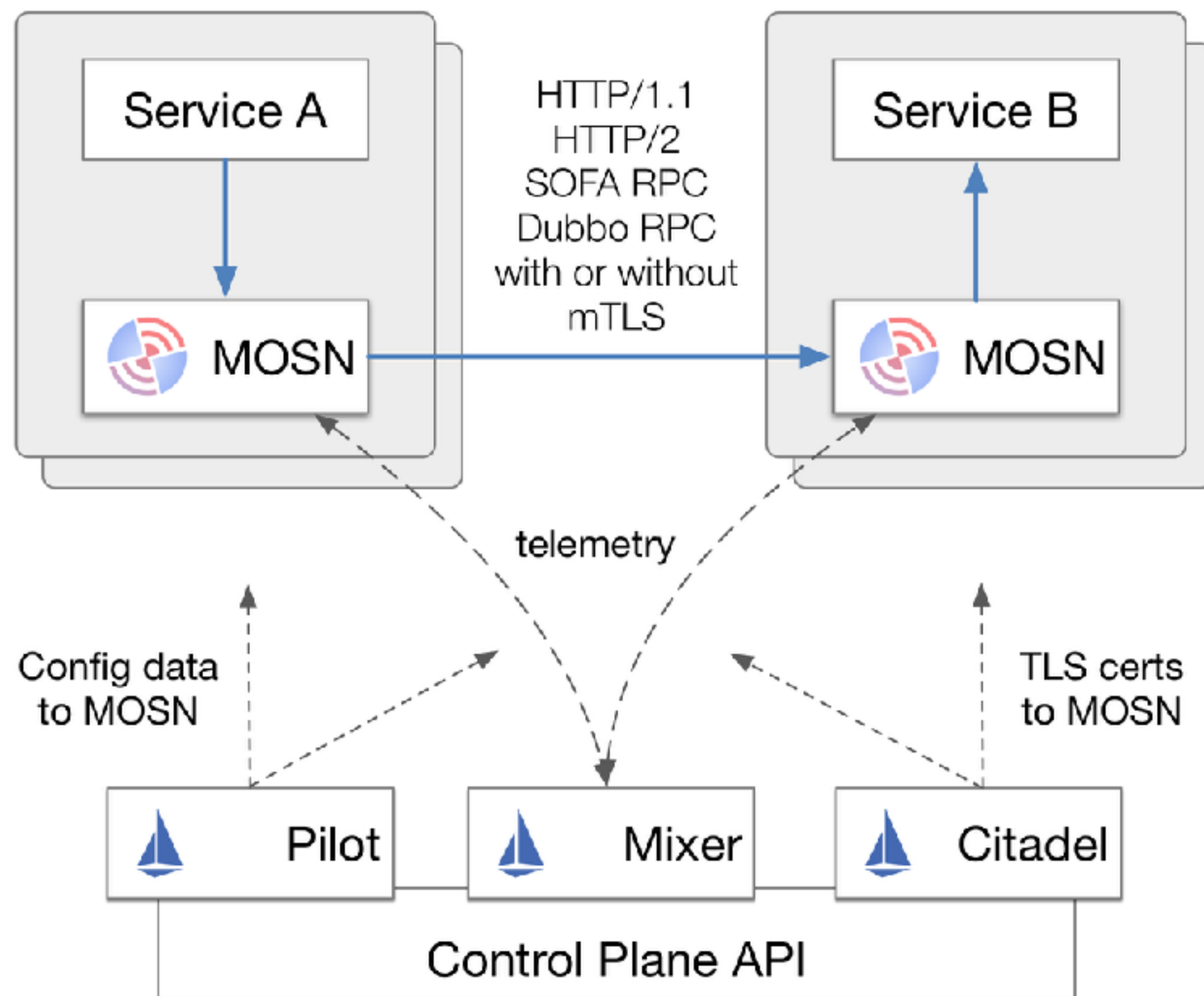
Service Mesh 实际上是将网络调用**抽象**到基础设施层

SOFAMesh 在多语言下的目标



语言中立的高效的通信协议
能力尽量往 Mesh 中沉淀
客户端尽量保持轻量化

SOFAMesh



跟随社区

- fork 自 Istio
- 紧跟 Istio 的最新版本
- 开源，反哺上游

实践检验

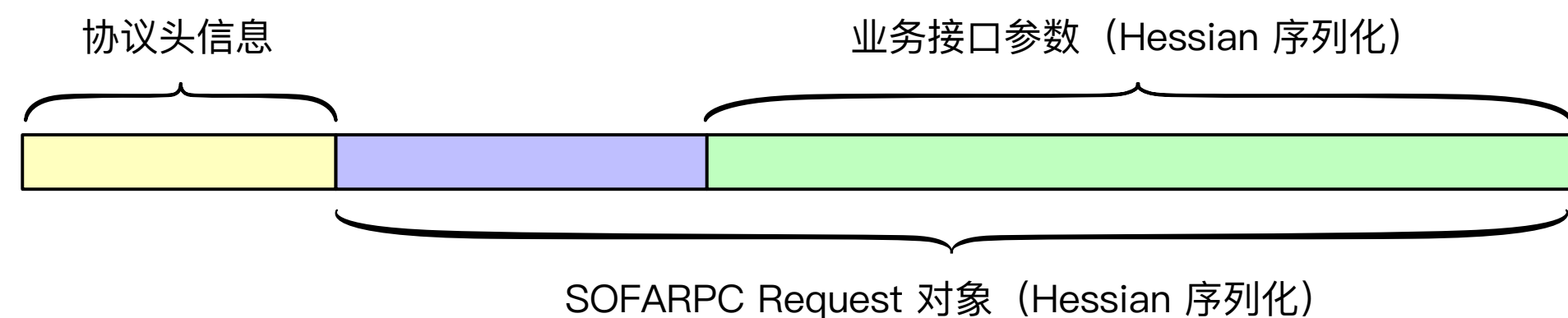
- 在实际生产落地中，发现问题，解决问题
- 在解决问题的过程中，寻求创新
- 扩展 Istio，弥补其不足的地方

SOFAMesh 解决多语言问题中的 技术要点

通信协议：网络调用中的**灵魂**，需要保持良好的语言中立性和扩展性

通信协议设计

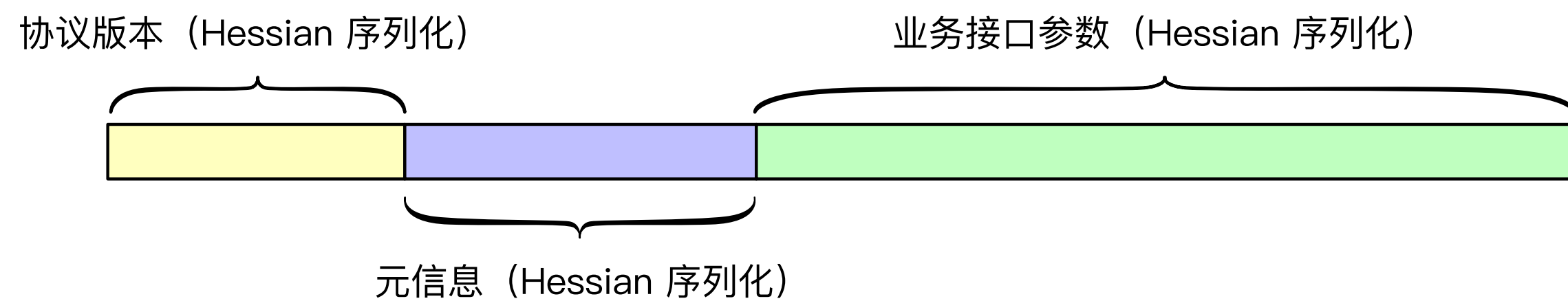
SOFARPC 早期的协议设计:



- **序列化协议不通用**, 采用了 Hessian 作为序列化协议。
- **元数据和业务参数封装在一起**, 解出元数据需要将业务参数一起解出, 对于 Sidecar 的消耗较高。
- **协议扩展性差**, 要扩展一些字段, 只能将对象放到 SOFARPC 的 Request 对象中

通信协议设计

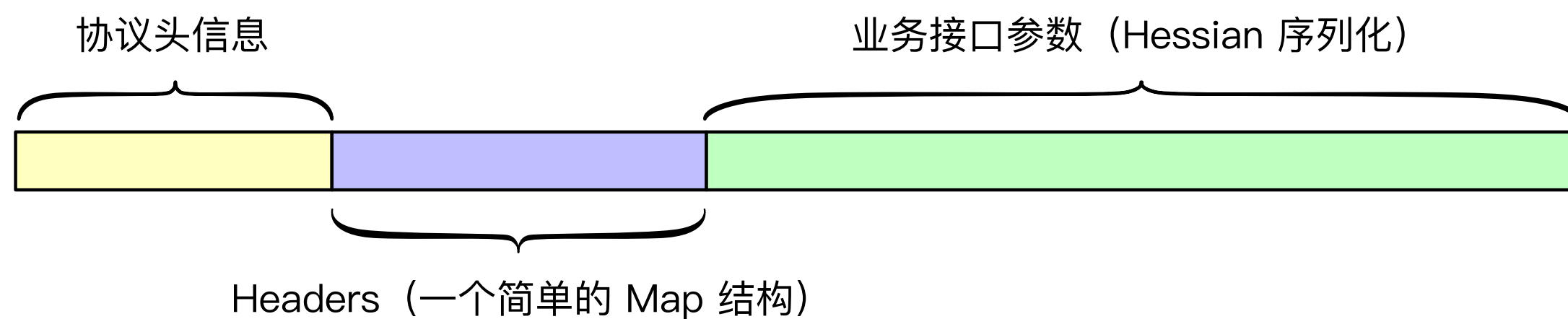
Dubbo 协议设计:



- **元信息是一等公民**: Sidecar 获取元数据只需要把 Headers 解出来就可以, 不用解业务接口参数。
- **Hessian 是超一等公民**: 从版本到元信息都必须依赖 Hessian, 多语言客户端和 Sidcar 实现较为麻烦
- **协议扩展性差**: 元信息需要 Hessian 序列化

通信协议设计

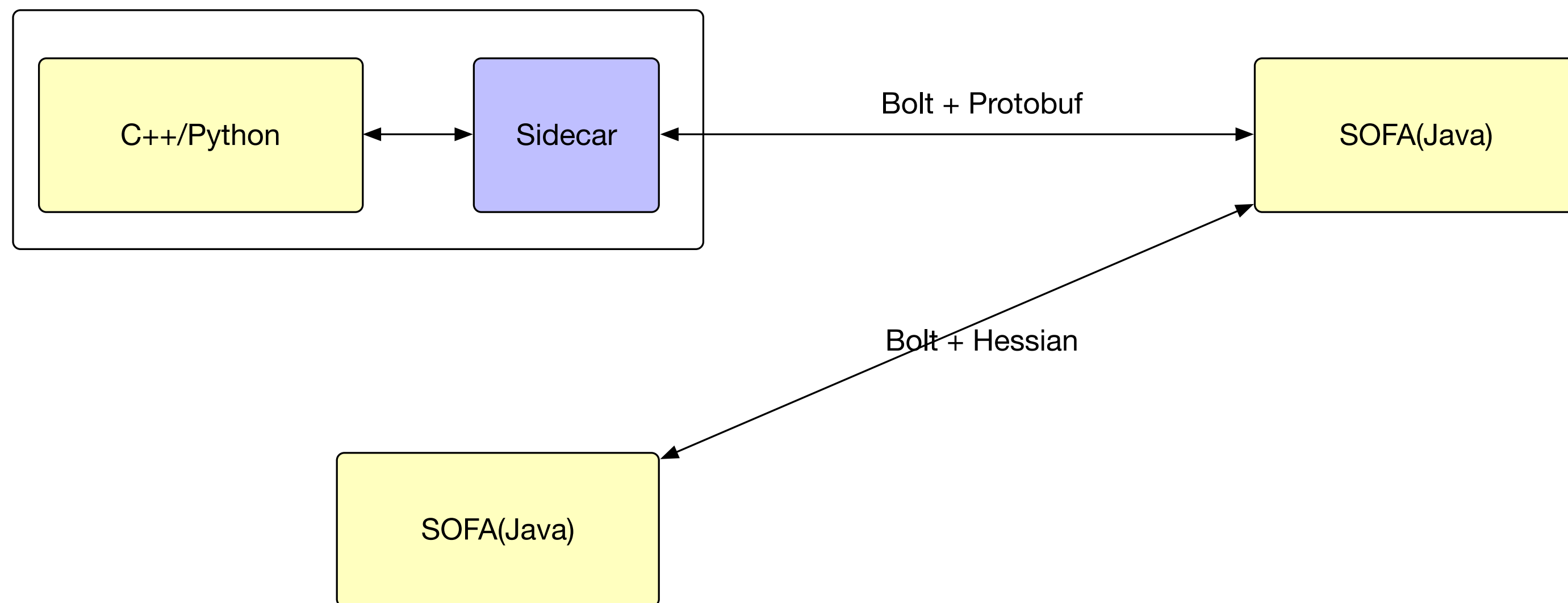
SOFARPC Bolt 协议设计:



好处

- **元数据是一等公民:** Sidecar 获取元数据只需要把 Headers 解出来就可以, 不用解业务接口参数。
- **元数据序列化形式简单:** 元数据直接采用简单的 Map 结构做序列化, 非常简单, Sidecar 和多语言客户端都可以轻松实现, 扩展简单。

序列化协议



Hessian 的问题:

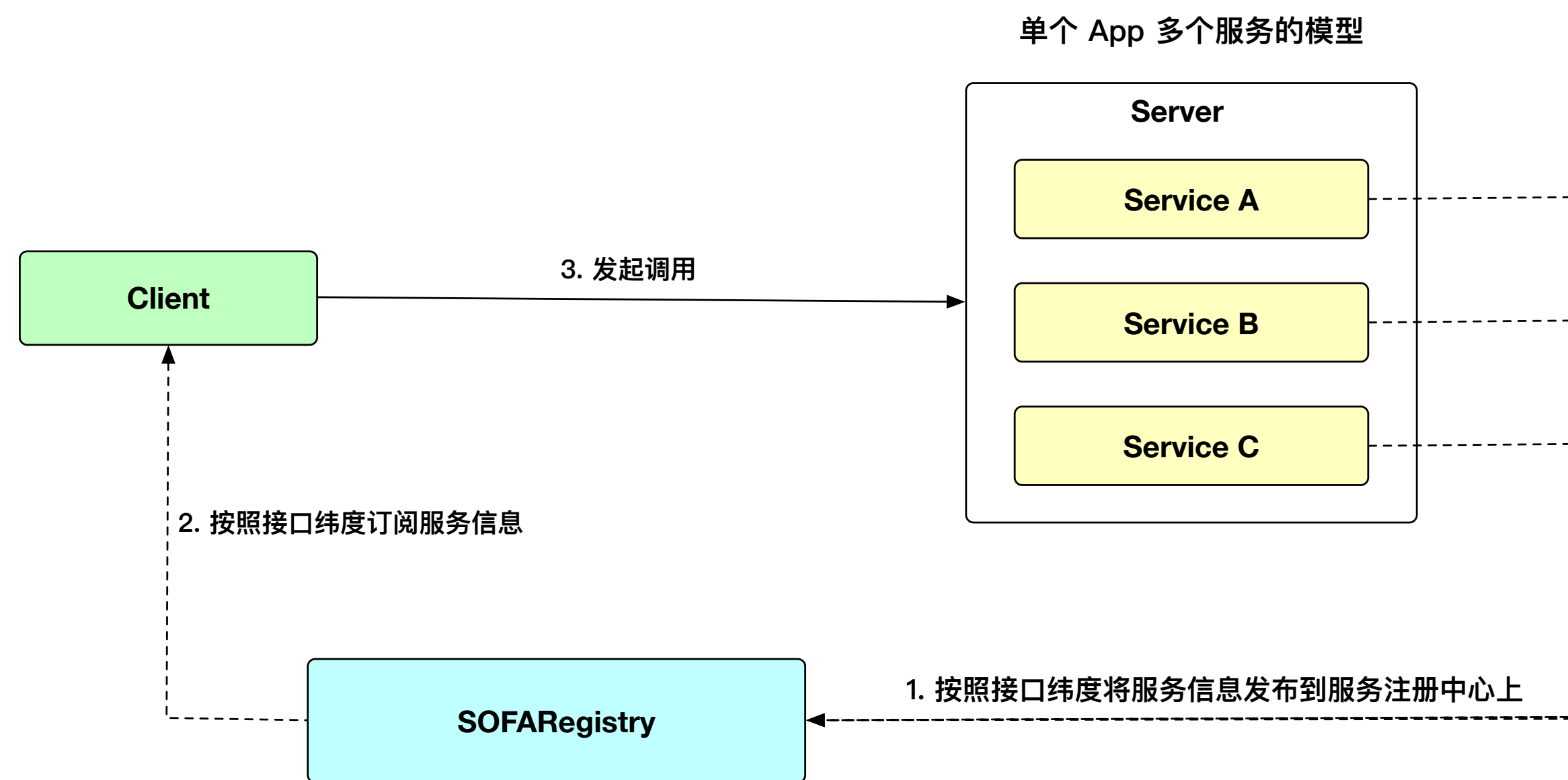
- 多语言支持较少
- 部分特性语言绑定
- 几乎没有人维护了

```
<sofa:service>
  <sofa:binding.bolt>
    <sofa:global-attrs serialize-type="pb"/>
  </sofa:binding.bolt>
</sofa:service>
```

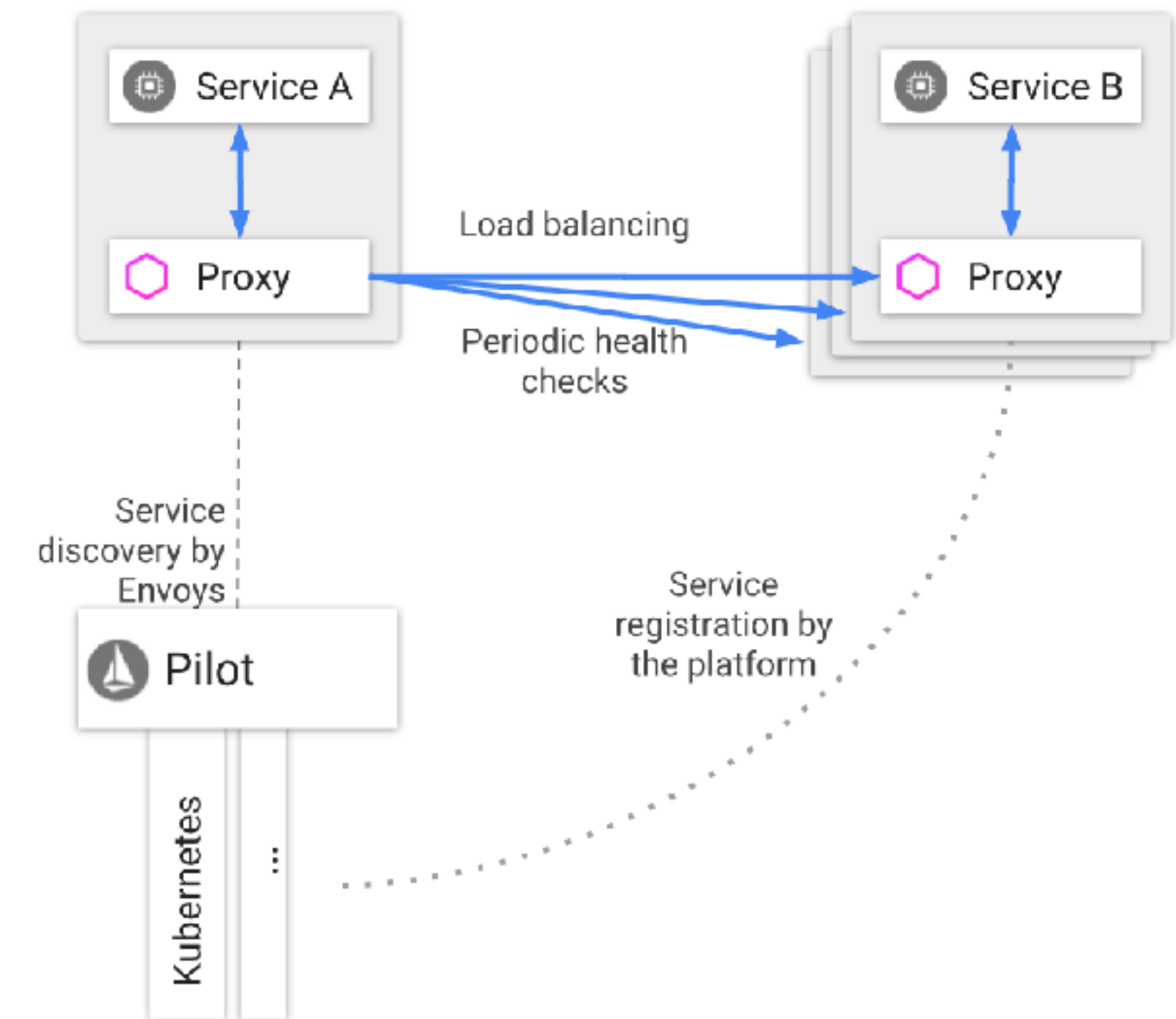
服务发现

服务发现是分布式 RPC 的**最基本能力**，尽量保证和语言无关

服务发现模型 —— SOFARPC vs Service Mesh

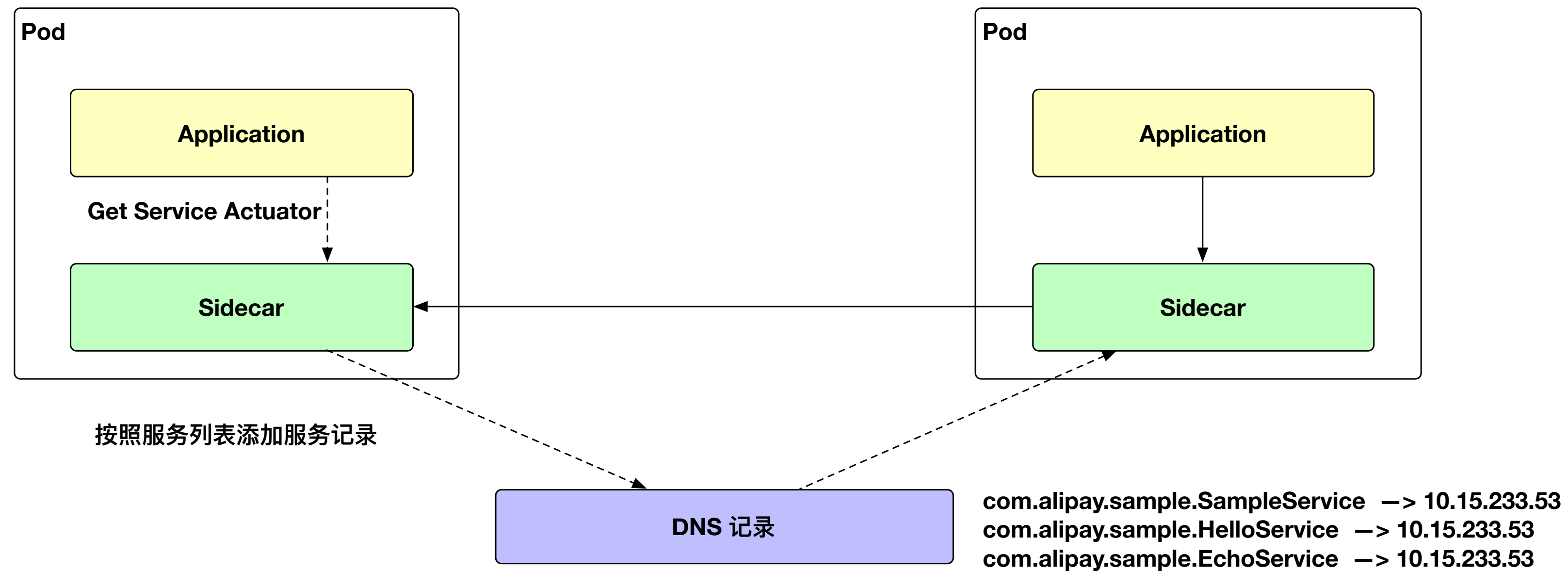


SOFARPC 服务发现模型



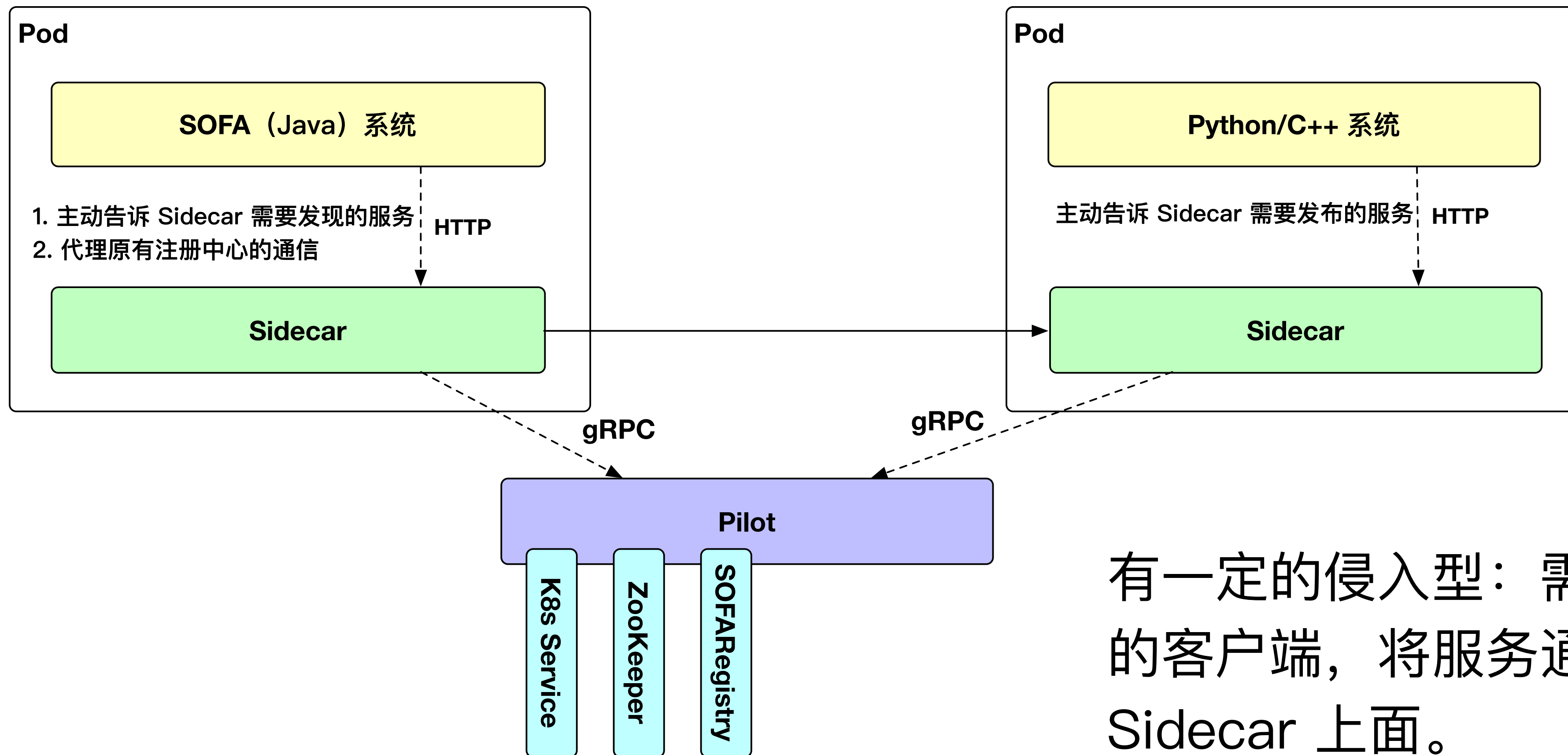
Istio 服务发现模型

服务发现 —— 演进方案一



依赖于 App 提供的 Actuator 的能力，并不是所有的 App 都有这个能力
DNS 存在 TTL，服务发现可能并没有及时。

服务发现 —— 演进方案二



有一定的侵入型：需要实现一个轻量级的客户端，将服务通过简单的方式注册到 Sidecar 上面。

轻量级客户端

当 Service Mesh 无法完全解决问题的时候，就搞一个**轻量级，足够简单**的客户端吧。

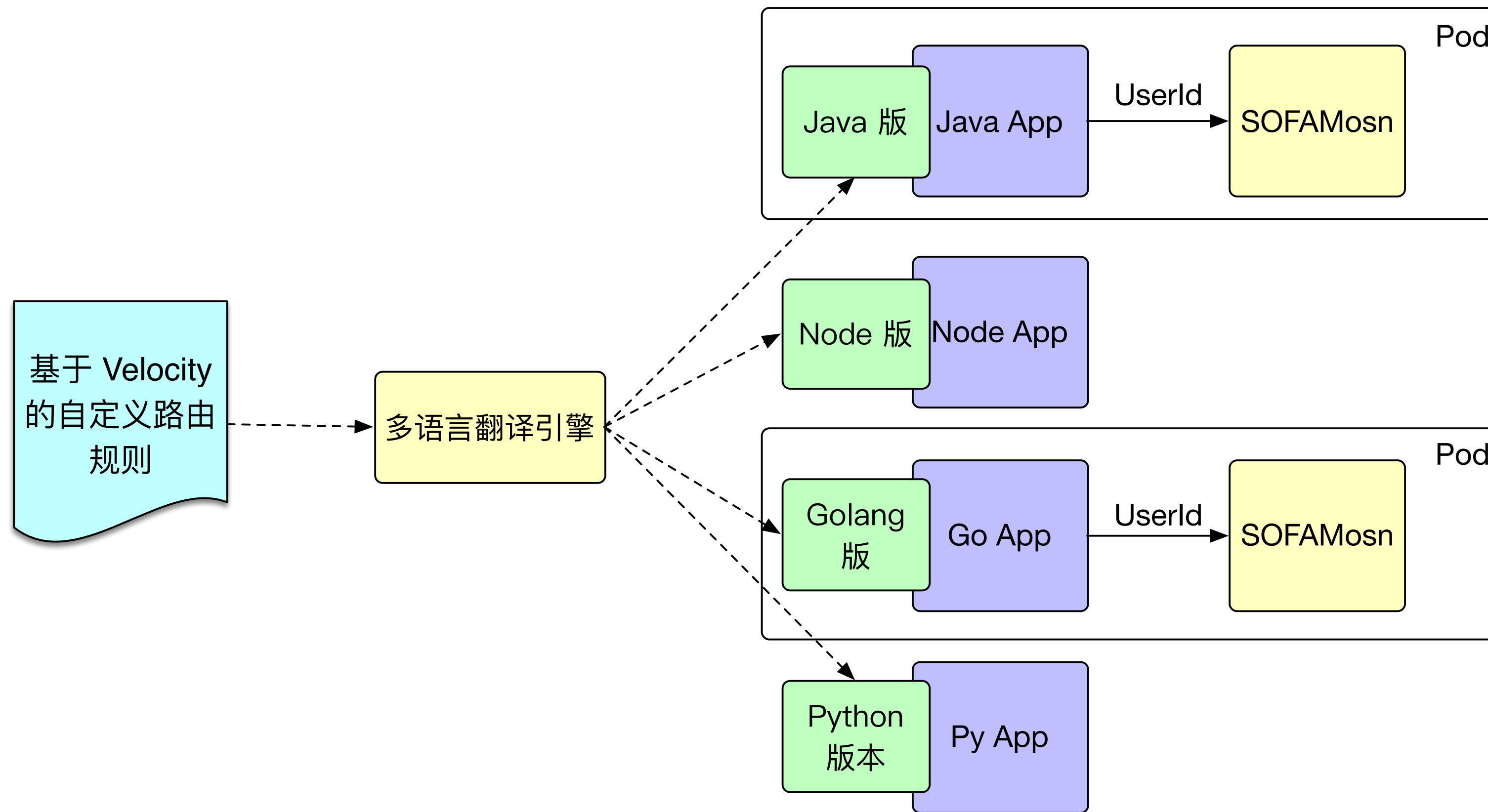
SOFARPC 中以注解驱动的单元化的路由方式

```
@ZoneRoute(uidGenerator = "com.alipay.account.util.AccountUidGenerator")
public interface AccountService {
    //增加
    Result increase(String userId, Double balance);
    //扣减
    Result decrease(String userId, Double balance);
}

public class AccountUidGenerator implements UidGenerator {
    public String generateUid(Method method, Object[] args) {
        //参数校验及异常处理略.....
        String userId = (String) args[0];
        //从用户ID截取末两位作为分片ID
        return userId.substring(userId.length()-2);
    }
}
```

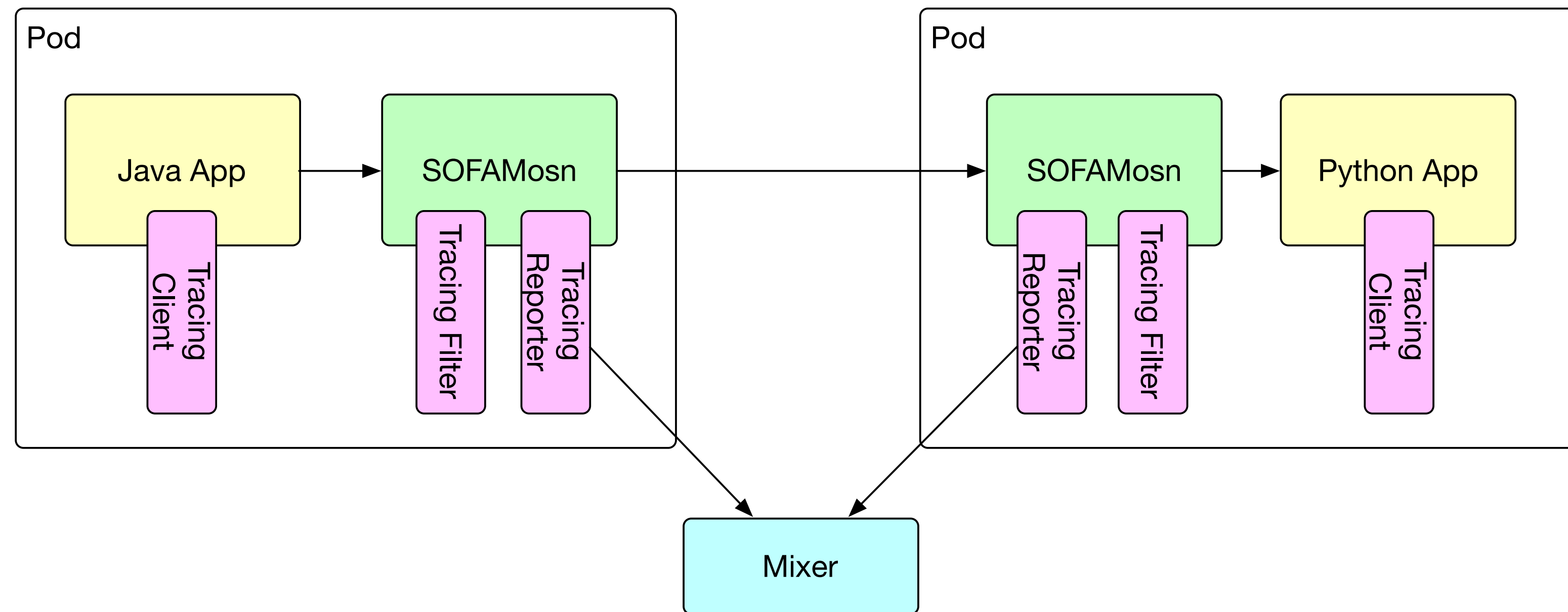


面向多语言的单元化路由方式



- 通过一个语言中立的脚本生成多语言的路由规则。
- 脚本的执行上下文严格控制，传入必要的参数，提供必要的工具类。

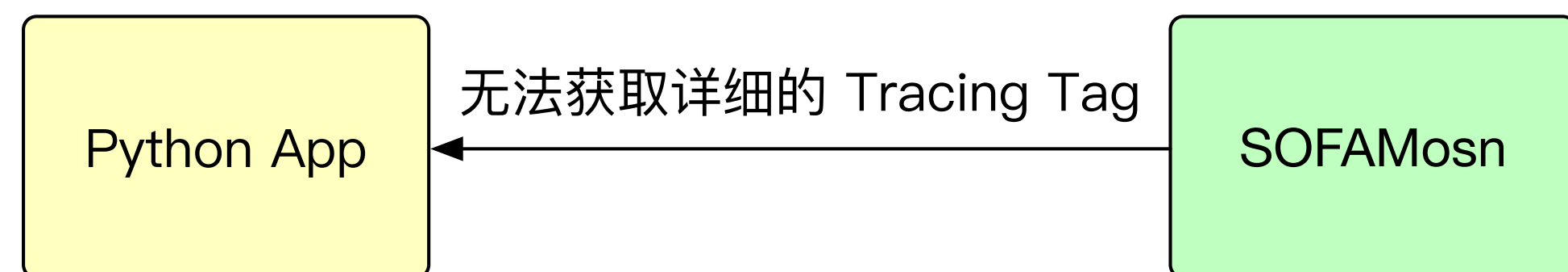
分布式链路追踪



即使有 SOFAMesh 的存在，依旧需要各个语言的分布式链路追踪的客户端

分布式链路追踪

当 Sidecar 作为 Downstream 的时候，详细的 Tracing 的数据可以通过通信协议携带回来



当 Sidecar 作为 Upstream 的时候，请求回到客户端之后的 Tag 无法获取到，忍痛割爱

小结

- 做好多语言，关键是保持语言中立，保证一次编写，到处可用
- 架构很美好，现实很骨感，落地需要做好**妥协**准备。
- Service Mesh **不是银弹**，必要的时候采用轻量级客户端作为辅助。

开源



- SOFAMosn: <https://github.com/alipay/sofa-mosn>
- SOFAMesh: <https://github.com/alipay/sofa-mesh>
- 其他 SOFA 项目: <https://github.com/alipay>

公众号：金融级分布式架构

个人微信：khotyn



智能时代的新运维

CNUTCon
全球运维技术大会

THANKS

主办方 **Geekbang** **InfoQ**
极客邦科技



AiCon

2018.12.20-23 / 北京·国际会议中心

AI商业化下的技术演进实战干货分享

京东：智能金融

景驰科技：自动驾驶

阿里巴巴：NLP

清华人工智能研究院：机器学习

今日头条：机器学习

Twitter：搜索推荐

AWS：计算机视觉

Netflix：机器学习



扫码了解详情

在不同业务场景下架构师如何升级？

听100+位讲师的优质 架构实践分享

来自Google、Netflix、LinkedIn、BAT等...

2018年12月7-8日 北京·国际会议中心



扫码了解大会详情

QCon

全球软件开发大会

北京·2019

更多技术干货分享，北京站精彩继续
提前参与，还能享受更多优惠

识别二维码
查看了解更多

2019.qconbeijing.com



深入剖析 Kubernetes

Kubernetes 原来可以如此简单



扫码了解更多



张磊

Kubernetes 社区

资深成员与项目维护者