

分布式应用运行时落地实践

郭智勇

CONTENTS 目录

- 1 云平台概述
- 2 分布式运行时
- 3 实践分享
- 4 展望

01 CHAPTER

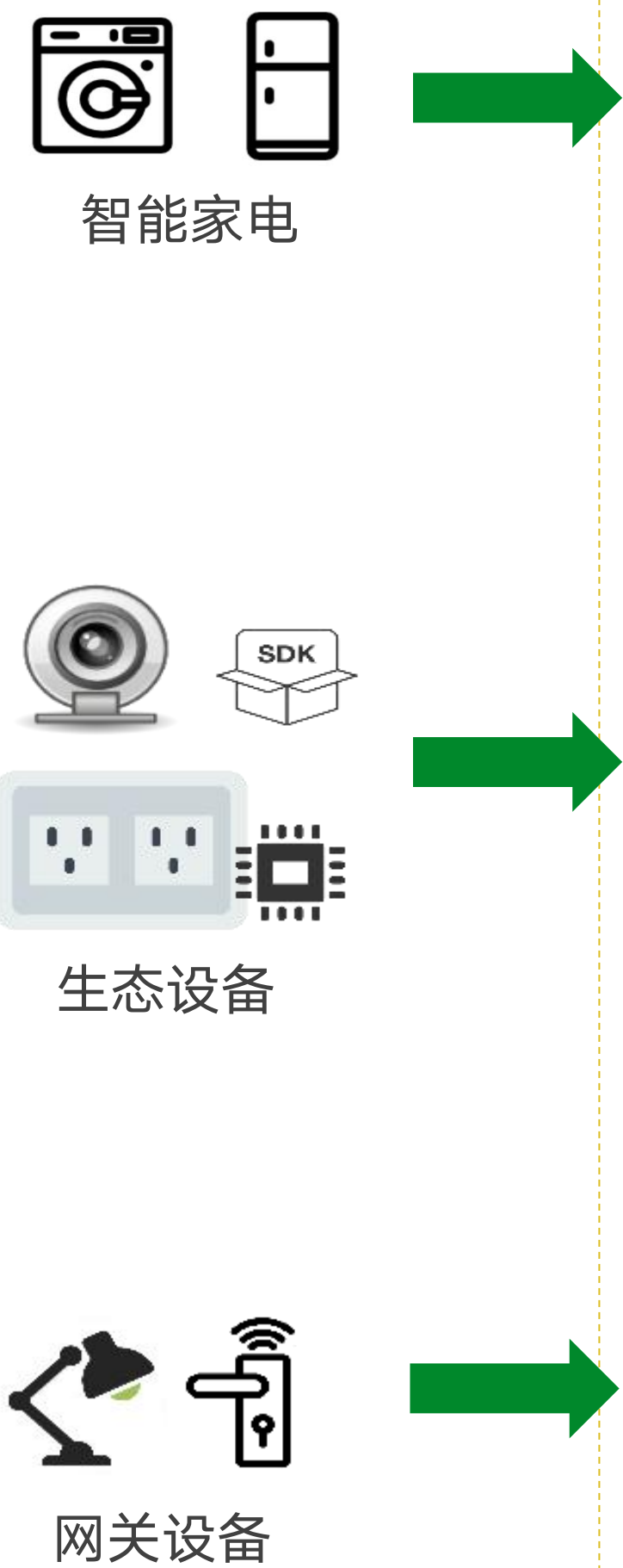
云平台概述

• 平台概览

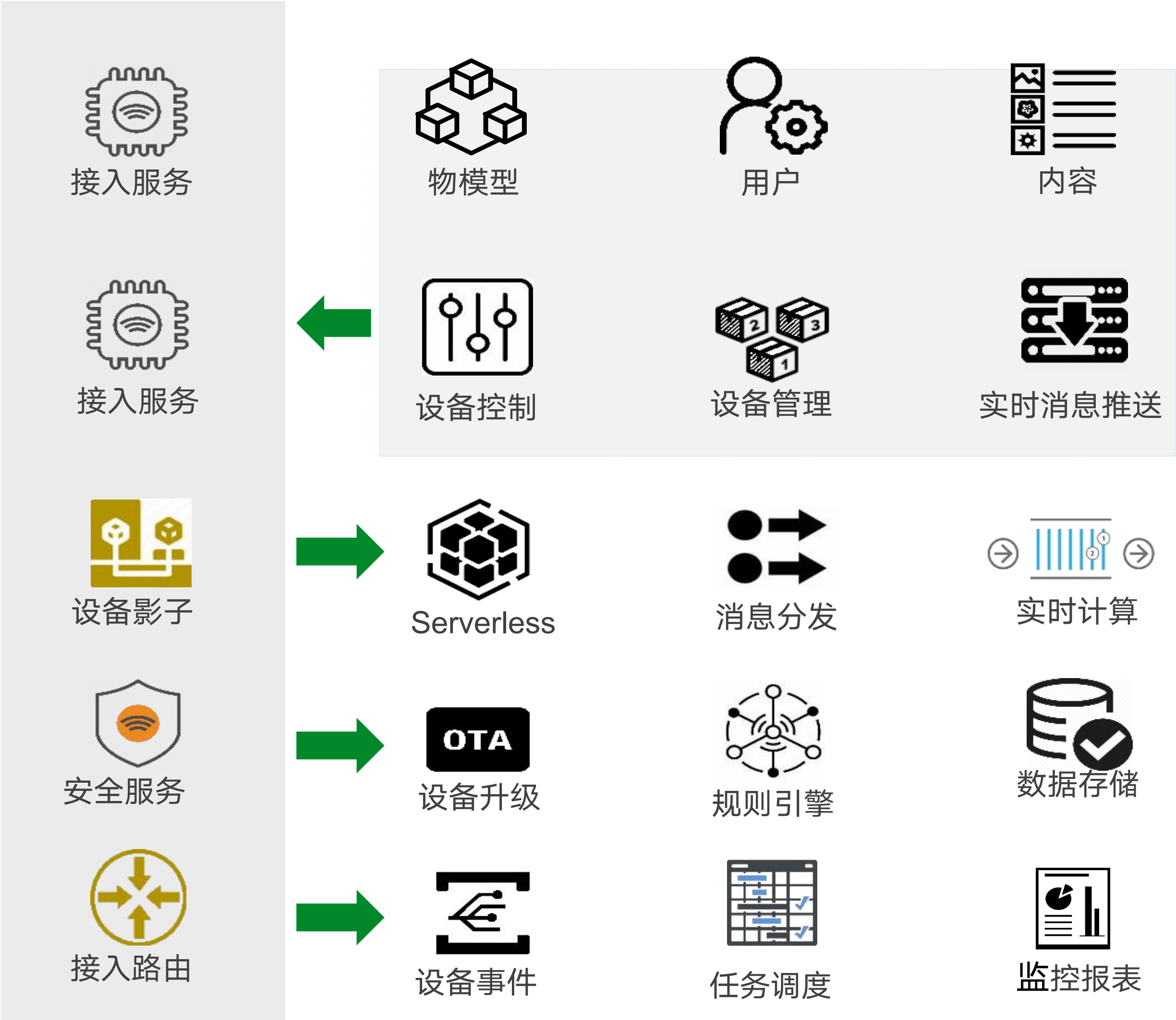
• 技术挑战

平台概览

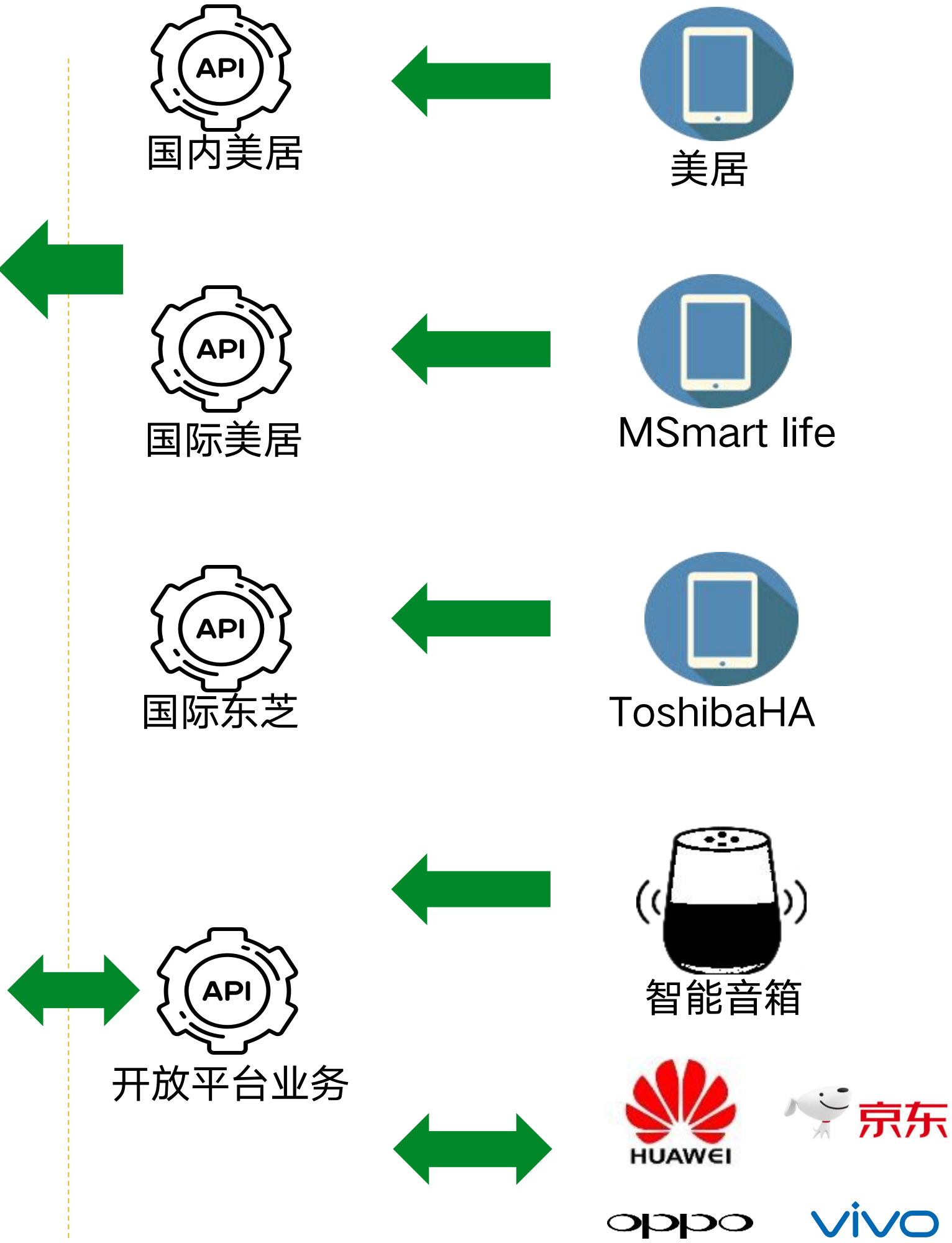
智能设备



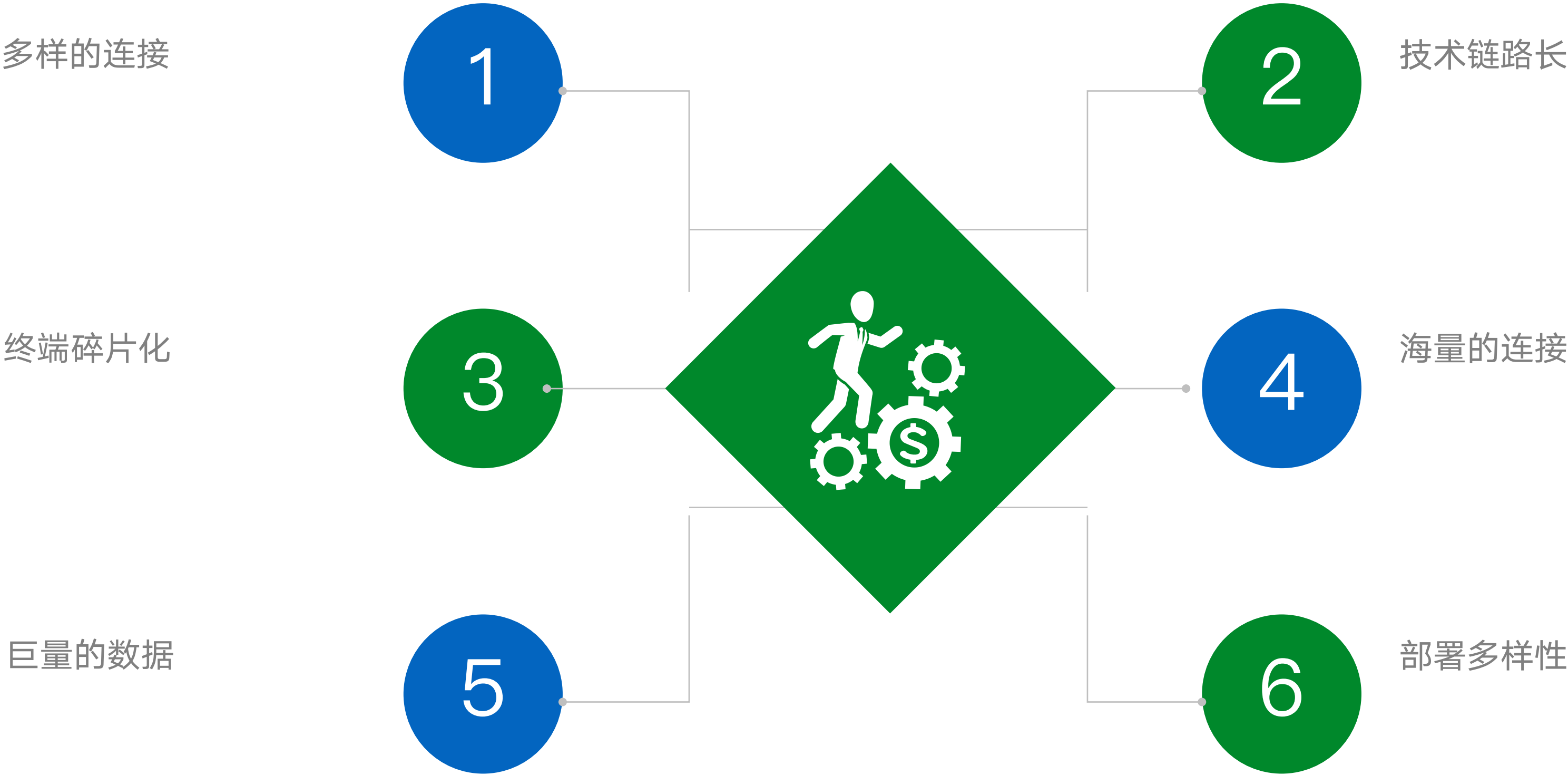
核心服务



业务应用



技术挑战



欢迎加入

02 CHAPTER

分布式运行时

- 现实问题
- 多运行时
- dapr

现实问题



服务代码中引入越来越多的SDK，有一些与云厂商强相关，不同的云厂商提供的SDK不一样，多云部署时需要调整代码（例如：KMS服务）

服务开发使用的语言各不相同，有java、c++、python、go,一些公共的库需要编写多种版本，升级维护需要投入不少人力



技术链路长，关联的部门多，联调、测试、压测大家都需要有一个稳定的环境互不干扰

服务的治理，服务间的限流、降级、熔断；服务与外部的交互；整个调用链路的追踪，出现问题时如何快速定位



多运行时

业务运行时和技术运行时分离；
业务运行时专注于业务的逻辑实现；
技术运行时关注于业务所依赖的技术中间件能力。二者分离开，各自维护，但又相互依存。

开源社区框架：



infoq.com:

[《Multi-Runtime Microservices Architecture》](#)

将各种各样的分布式能力外移到独立 Runtime，最后和应用 Runtime 共同组成微服务，形成所谓的“Multi-Runtime”（多运行时）架构

从实际出发，选用Dapr,在此基础上扩展自己的业务组件，满足自身的需求，同时回馈社区

Dapr APIs

Application code

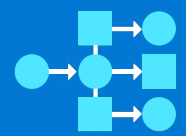
Microservices written in

Any code or framework...



HTTP API

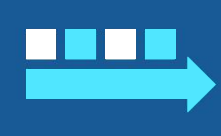
gRPC API



Service-to-service invocation



State management



Publish and subscribe



Resource bindings and triggers



Actors



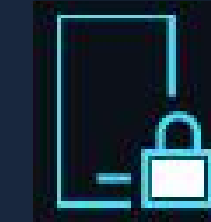
Observability



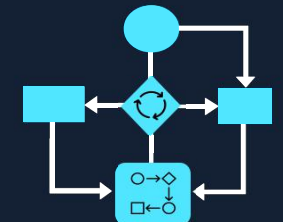
Secrets



Configuration



Distributed Lock



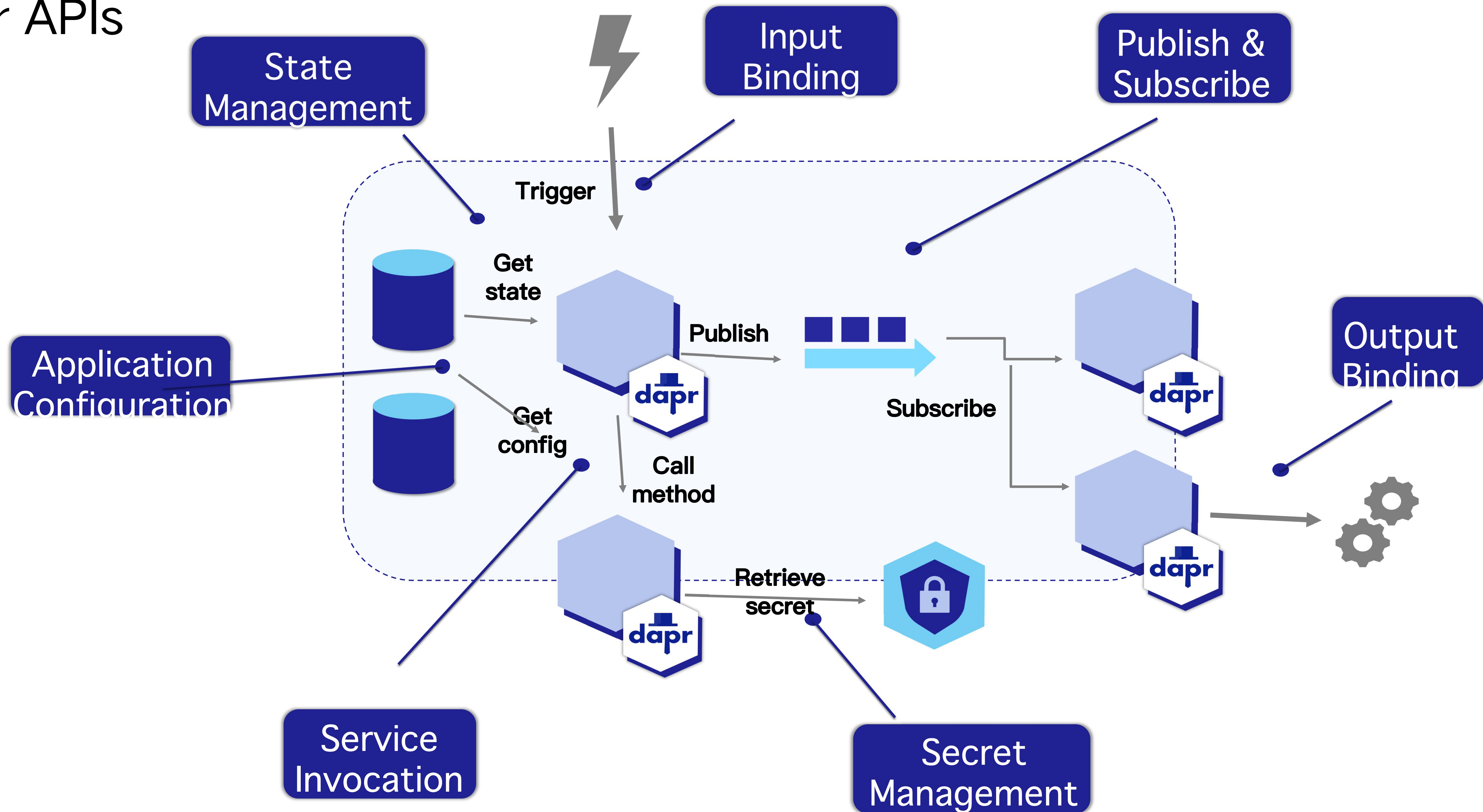
Workflow

Any cloud or edge infrastructure



virtual or physical machines

Dapr APIs

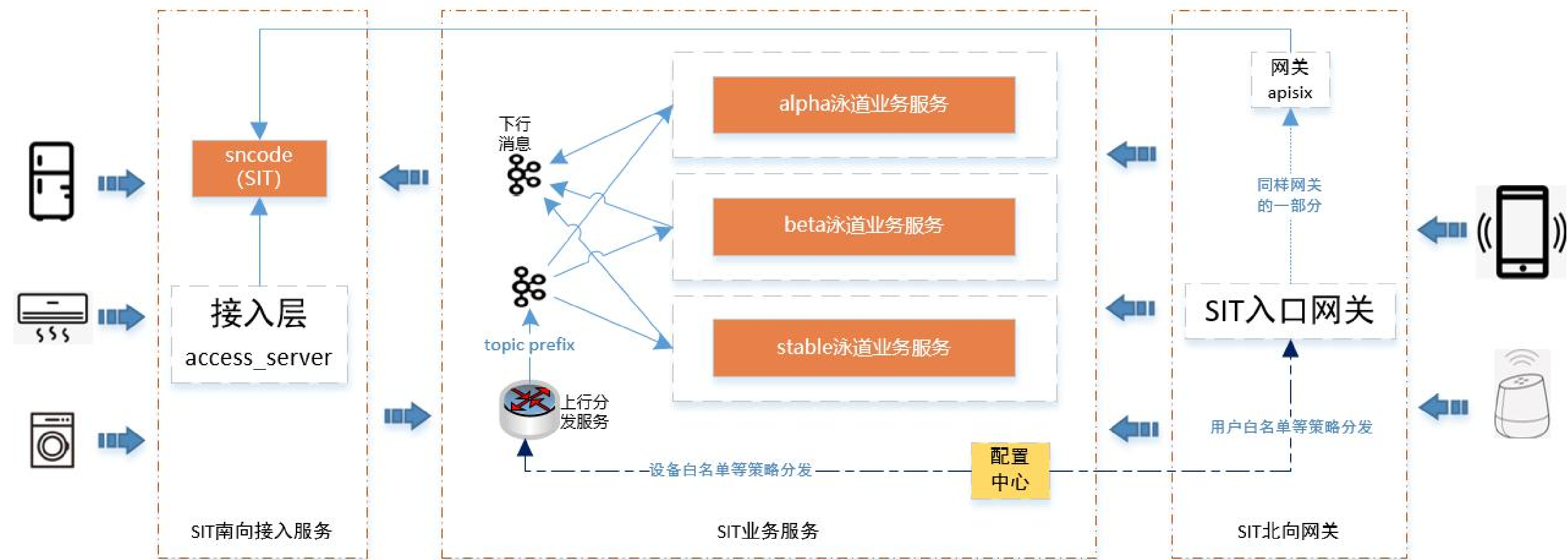


03 CHAPTER

一些实践分享

- 多泳道环境隔离
- 多云配置管理
- 服务治理

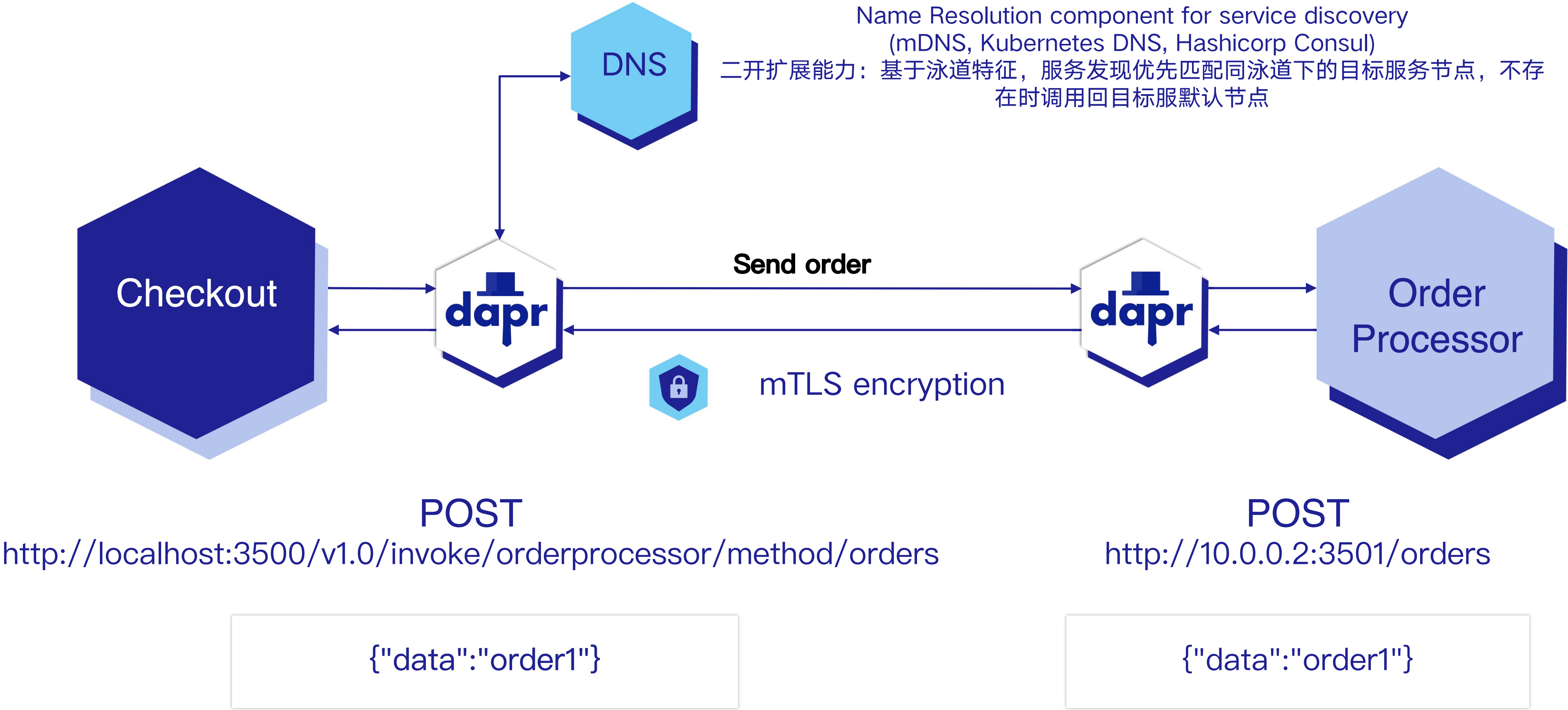
1.1 多泳道环境隔离-全览



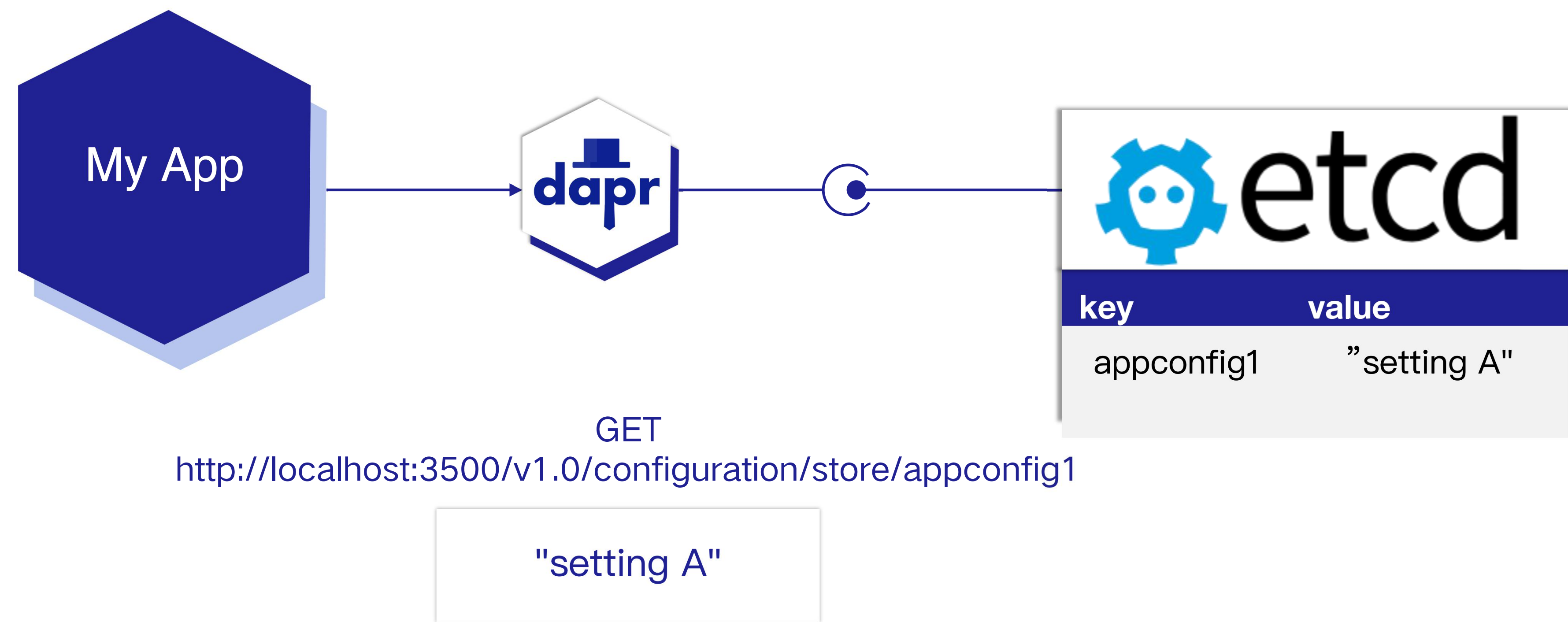
南向消息
设备上行，按照设备白名单将上行消息按泳道投递，分发到MQ的不同前缀topic
设备下行，无需处理，接入层只需要一个

北向业务
在网关层通过用户白名单将业务请求按泳道投递，实现不同来源（开发人员、品质、事业部等）用户分别投递到对应的泳道。

1.2多泳道环境隔离-服务间调用示例

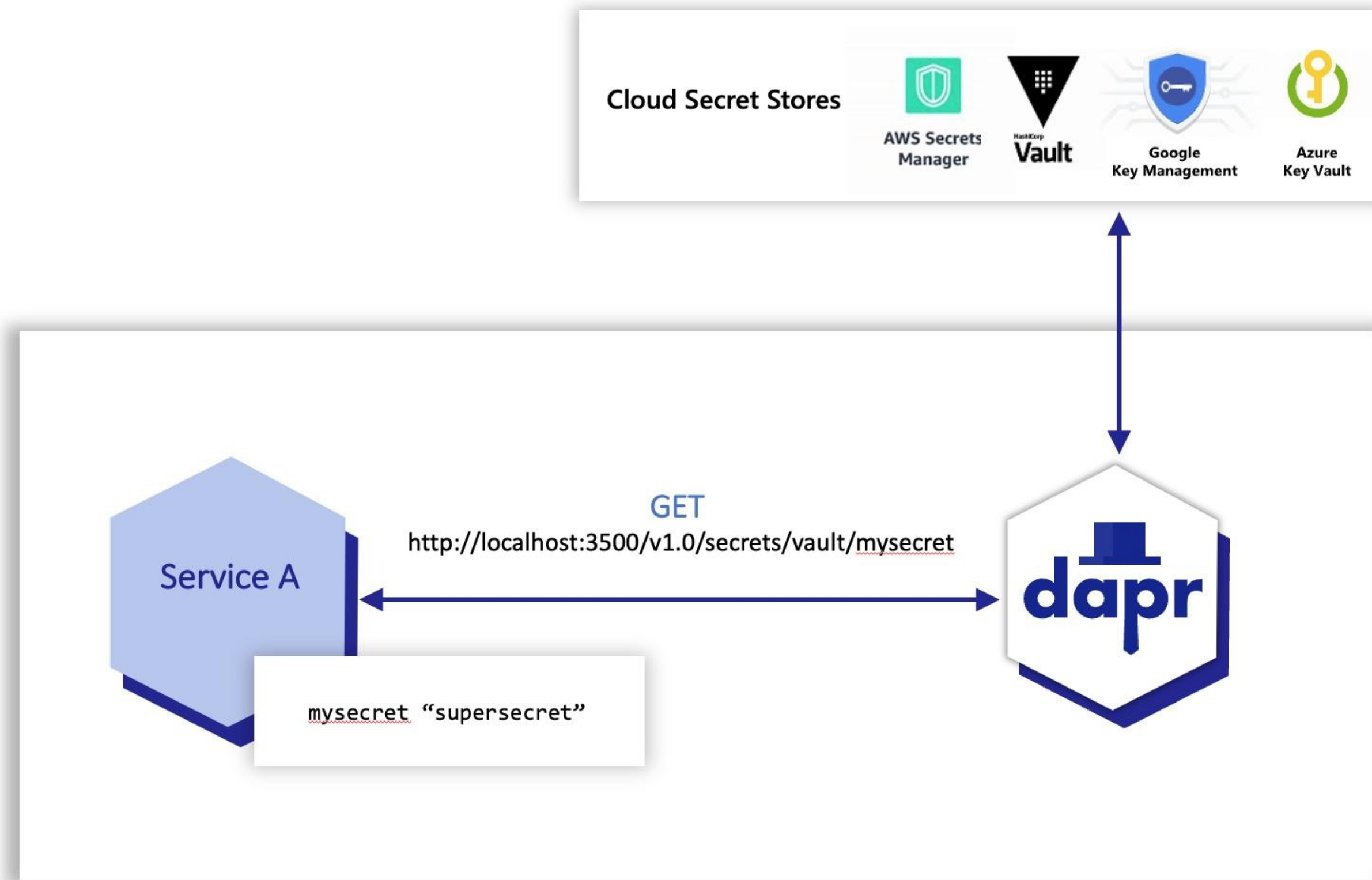


2.1 配置管理-业务配置



二开扩展支持ETCD方式的存储
结合自研配置管理后台，实现配置的增、删、改、查、发布
基于http api交互，业务侧中不关注具体的配置存储方式（etcd\redis\mysql），不依赖业务具体的开发语言(java\go\c++\python)，不需要集成sdk，可主动读取配置
业务提供回调路由地址，也可实现及时接收配置的热更新

2.2配置管理-安全配置



二开扩展支持对接不同云服务厂商的密钥管理云服务或自建的KMS服务
业务侧不需要集成sdk,与开发语言无关
服务的移植性大大提升

3.1 服务治理-更多探索

混合部署下的服务发现

可观察性

统一的限流、降级

云原生演进



展望



谢谢

想一想，我该如何把这些
技术应用在工作实践中？

THANKS