蚂蚁集团平台工程规模化实践

李大元
蚂蚁集团

Agenda

01 背景

02 传统 PaaS 的困境

03 平台工程会是答案吗？

04 我们的实践

05 架构与技术

06 产品建设与文化推广

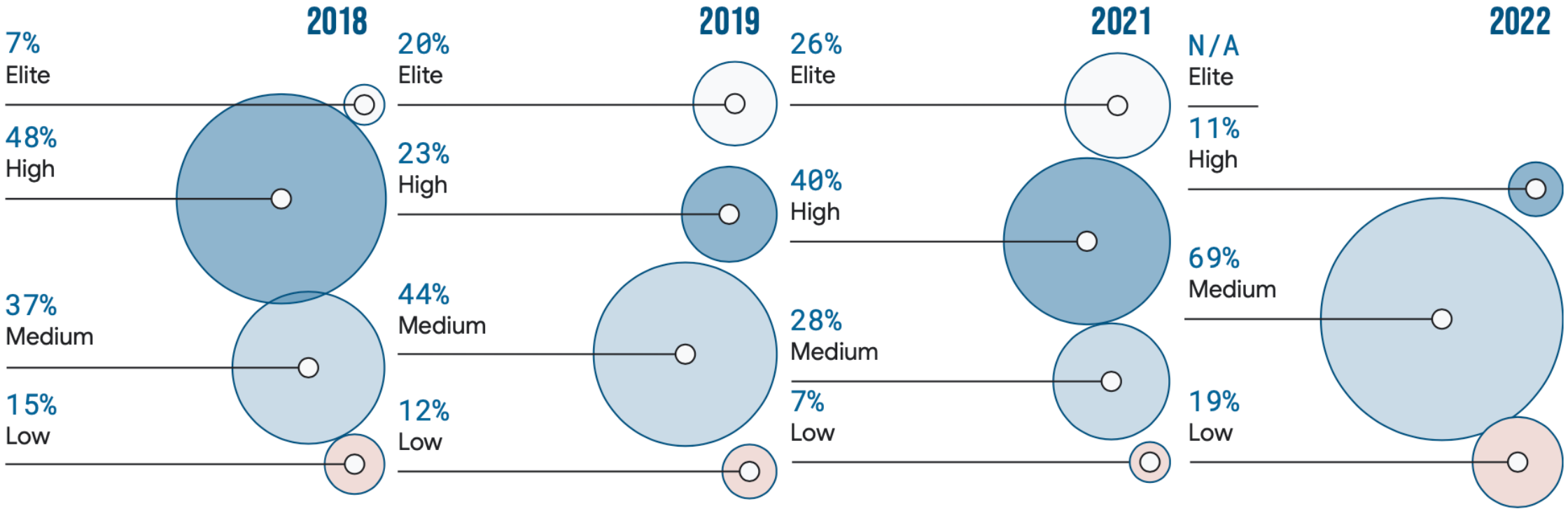
07 一些挑战

08 在蚂蚁和其他行业的实践

背景

行业趋势

软件交付运维效率[1]



高级别与精英级别的比例下降。行业整体运维效率在下降

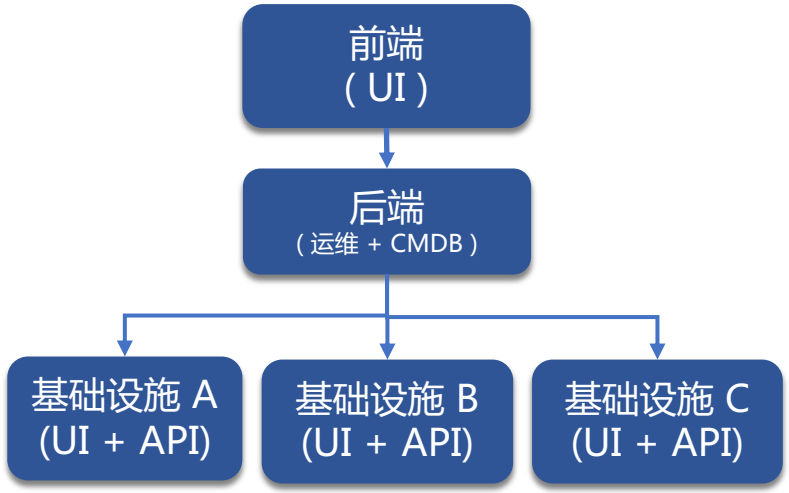
[1] Google 2022 Accelerate State of DevOps Report

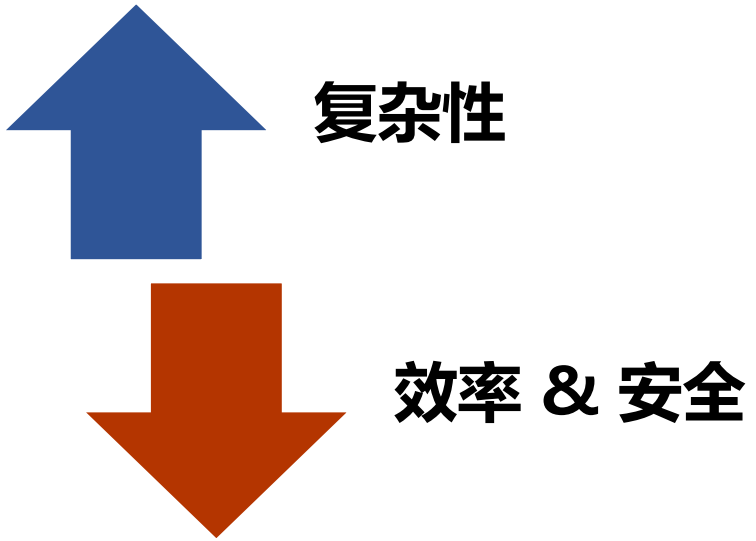
背景

我们已经有了什么？（与自动驾驶比较）

等级		描述	产品
L0	无自动化	手动运维	Hands🤘
L1	运维人员辅助系统	通过脚本辅助运维人员	Shell
L2	部分自动化	通过平台产品解决特定场景的运维问题	Heroku/云管平台
L3	有条件的自动化	平台通过声明式运维可以做一些简单的决策. 仍然需要人工介入	Kubernetes
L4	高度自动化	完全声明式运维，大多数场景下不需要关注过程	AIOps
L5	完全自动化	不需要人工参与，不需要运维，运维职业不复存在	?

传统 PaaS 的困境





Developer

- 高认知负担
- 需求被满足时间变长

Platform

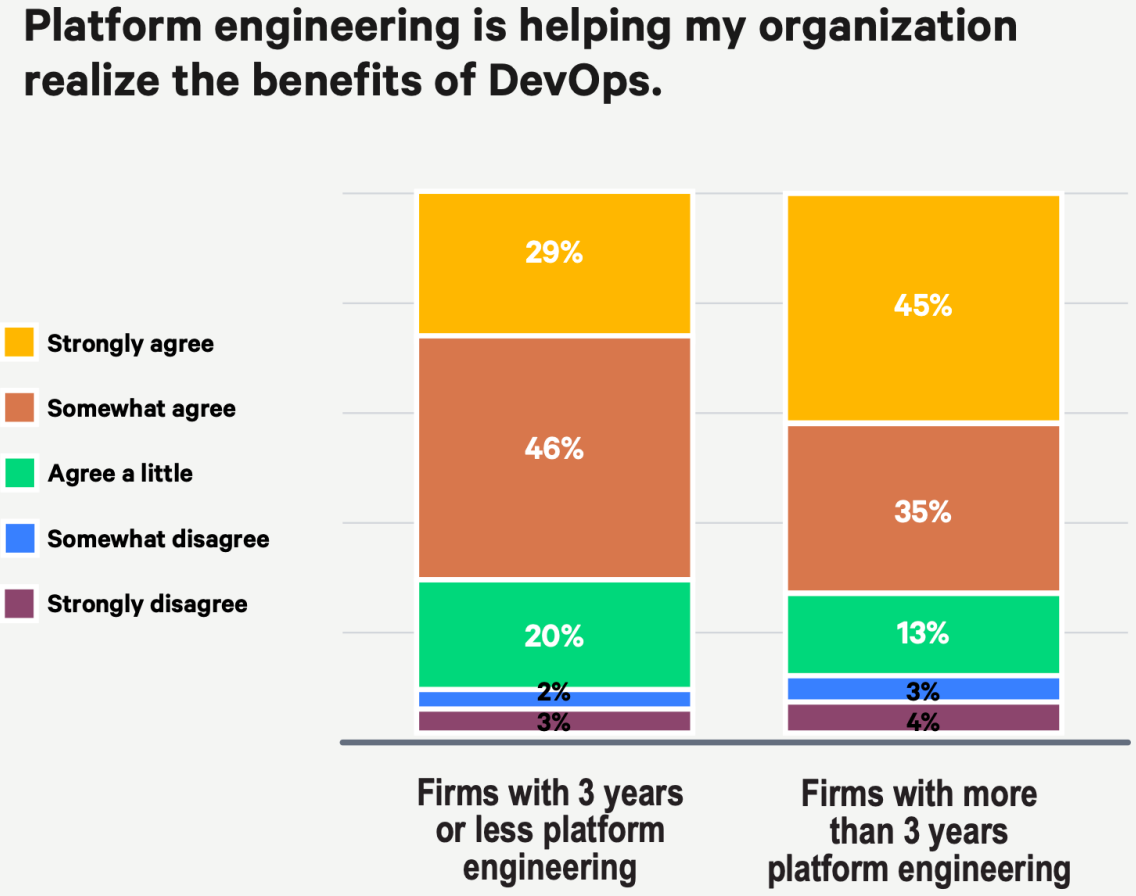
- 需求的种类与数量大幅增加
- 与基础设施团队沟通成本大幅增加

Security

- 多个独立的基础设施平台导致很高的风险敞口
- 生命式运维很强大也很危险

平台工程会是答案吗

平台工程是设计和构建内部开发者平台的**学科**，用于解决日益复杂的运维需求与企业效率与安全之间的**矛盾**



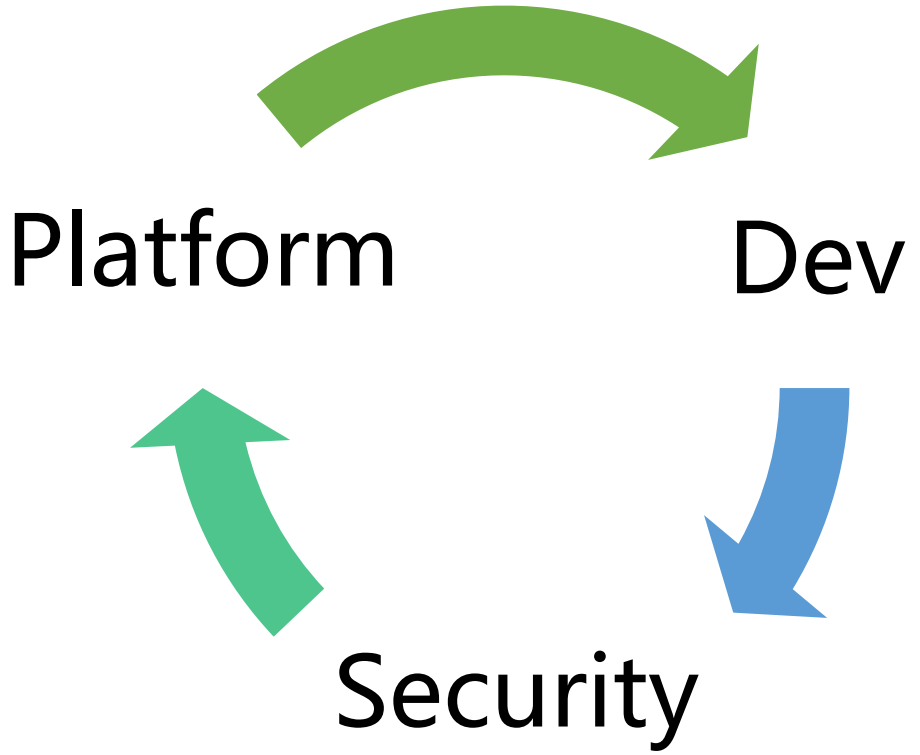
DevOps 没有死，

平台工程是新的 DevOps

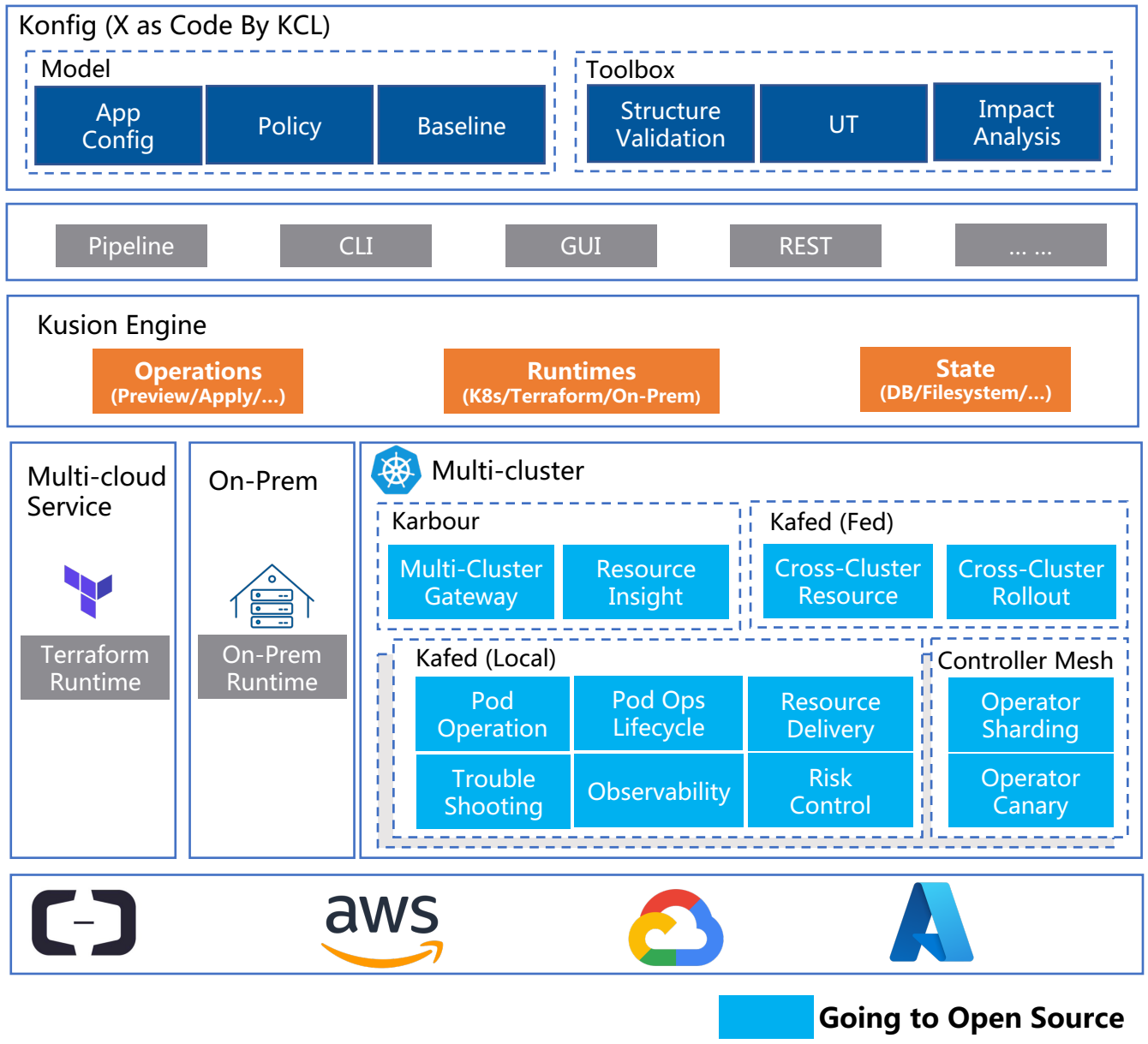
Puppet 2023 State of DevOps Report

我们的思考

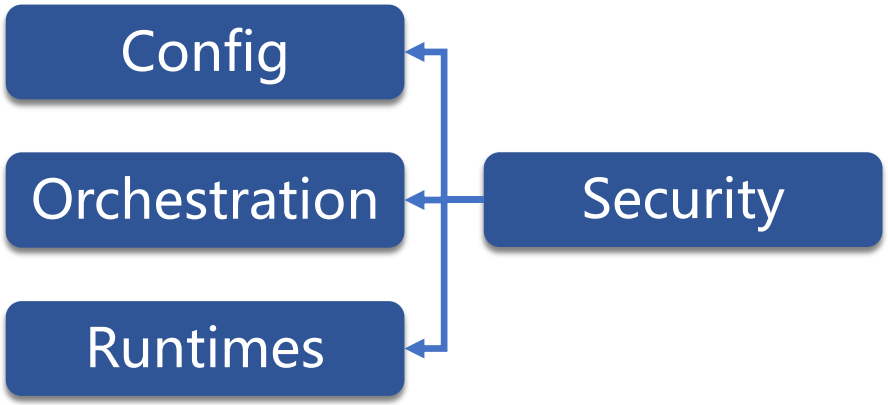
- **减少用户认知负担:** 通过灵活的高层级抽象，屏蔽基础设施复杂性
- **转变生成关系:** 研发通过 PaaS 提供的自服务能力，可以自助满足自己的需求
- **风险左移:** 通过策略规则，尽可能的在运维早期发现、解决问题



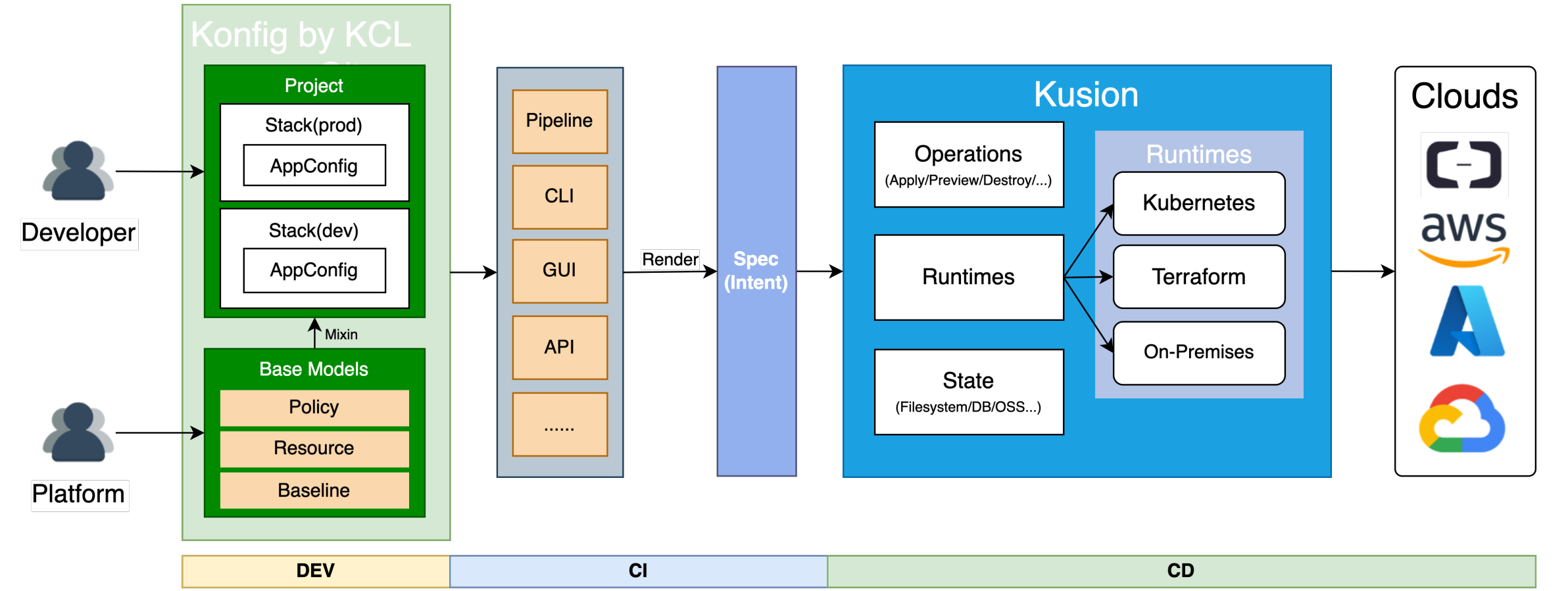
我们的实践— KusionStack 架构



助你更快、更安全的构建**自己的**
Internal Developer Platform



我们的实践— KusionStack 工作流



架构与技术

Konfig: 存放运维意图的 git 仓库，也是 Platform 与 Developer 的协作平台

```
cd appops/wordpress && tree
.
├── README.md
├── base                                // Common configuration for all stacks
│   └── base.k
├── dev                                 // Stack directory
│   ├── ci-test                         // Test data
│   │   ├── settings.yaml
│   │   └── stdout.golden.yaml
│   ├── kcl.yaml                        // Compile configuration for current stack
│   ├── main.k                          // App Configs matained by App Dev
│   ├── platform.k                      // App Configs matained by Platform Dev
│   └── stack.yaml                      // Stack metadata
└── project.yaml                        // Project metadata
```

Platform

Developer

架构与技术

KCL: 运维配置、策略 DSL

```
import base.pkg.kusion_models.kube.frontend

appConfiguration: frontend.Server {
    image = "howieyuen/gocity:latest"
}
```

Spec

Deployment

Service

ConfigMap

Database

Monitor

… …

前端模型（Developer）

后端模型（Platform）

K8s/clouds/自建基础设施

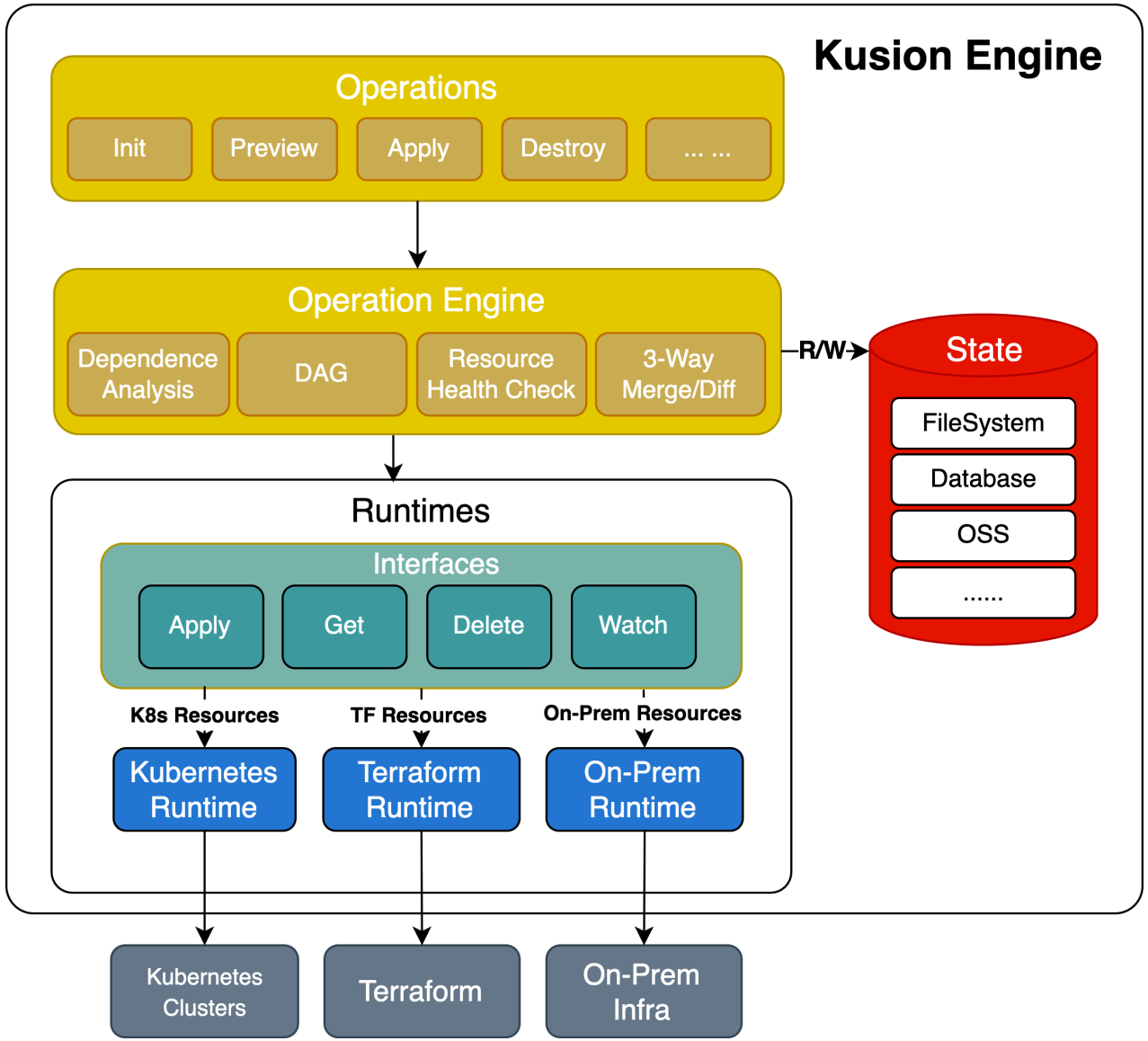
架构与技术

Kusion Engine: 平台引擎，负责所有运维操作

Operations: 为所有 Kusion 命令提供资源管理、编排和 3-way live diff 等核心功能

Runtimes: 代表由 Kusion 管理的基础设施，它通过统一的接口与异构基础设施进行交互。

State: 真实资源在 Kusion 中的映射，Kusion 管理资源的必要数据



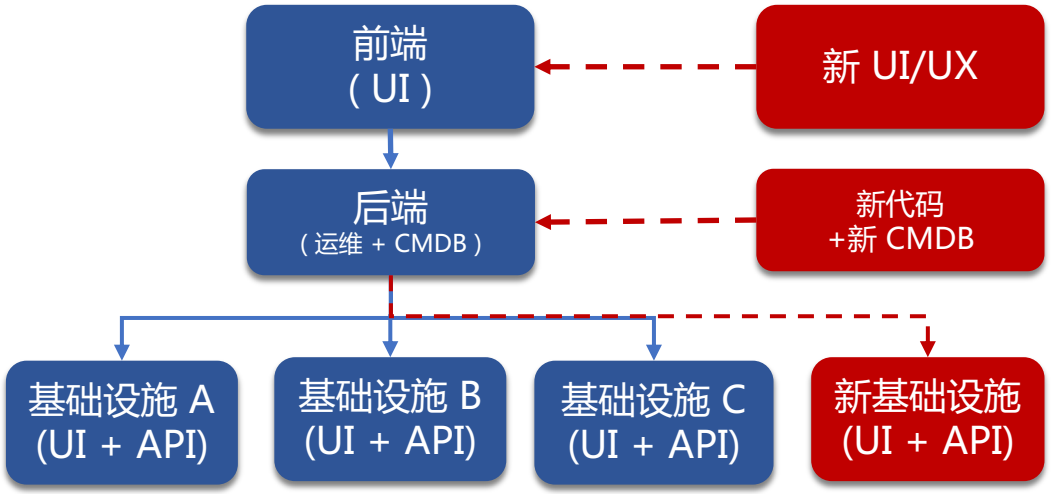
Highlights

- **以应用为中心:** 在同一个地方、以统一的方式，管理应用所需要的所有的运维操作
- **研发自服务:** 研发可以利用平台提供的能力，自助实现自己的需求
- **风险左移:** 在运维前期保证安全性，使操作更有信心。
- **K8s 友好:** 提供友好的可观测性与排障能力，让云原生运维更简单

使用案例 — 新增一种云资源类型

经典 PaaS

1. **Platform** 设计实现新 UI/UX
2. **Platform** CMDB 添加新资源元数据，编写胶水代码调用基础设施的新 API
3. **Platform** 处理基础设施返回结果，屏蔽基础设施细节，抽象为用户视角的模型

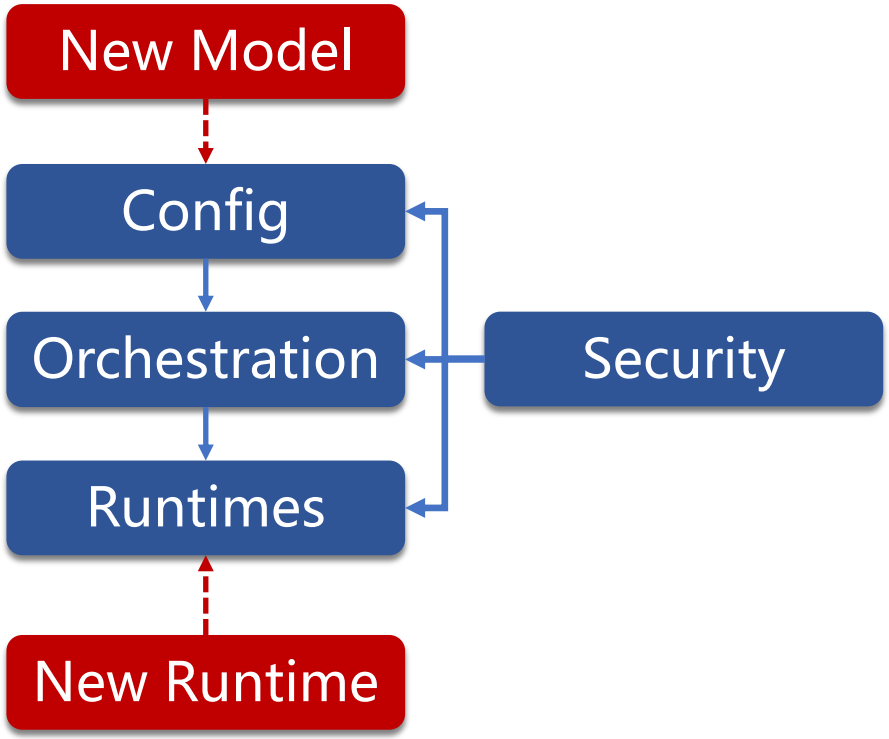


使用案例 — 新增一种云资源类型

KusionStack

1. **Infra/SRE** 设计新配置模型
2. **Infra team** 编写新 Runtime
   (or 复用现存 TF provider)

转变生成关系：PaaS 提供能力，Infra 直接服务 Developer



使用案例 — 代码细节

Developer 配置

1. 以应用为中心
2. 减少用户认知负担

```
# defination of wordpress application frontend model
wordpress: frontend.Server {
    # specify application image
    image = "wordpress:4.8-apache"

    # use cloud database for the storage of wordpress
    database = storage.DataBase {
        # choose aliyun_rds as the cloud database
        dataBaseType = "aliyun_rds"
        dataBaseAttr = storage.DBAttr {
            # choose the engine type and version of the database
            databaseEngine = "MySQL"
            databaseEngineVersion = "5.7"
            # choose the charge type of the cloud database
            cloudChargeType = "Serverless"
            # create database account
            databaseAccountName = "root"
            databaseAccountPassword = option("db_password")
            # create internet access for the cloud database
            internetAccess = True
        }
    }
}
```

使用案例 — 代码细节

Platform 配置

1. 基础设施相关配置
2. 策略与规则
3. 依赖定义

```
aliyunVPC = alicloud.AlicloudVPC {
    vpc_name = _alicloudResourceName
}
provider = [*provider, alicloud_backend.VPCRender(aliyunVPC).provider]

aliyunVswitch = alicloud.AlicloudVswitch {
    vpc_id = _alicloudDependencyPrefix + alicloud_config.alicloudVPCMeta.type + ":"
    + aliyunVPC.vpc_name + ".id"
    vswitch_name = _alicloudResourceName
    zone_id = alicloud_config.alicloudProviderMeta.region + "-h"
}
provider = [*provider, alicloud_backend.VswitchRender(aliyunVswitch).provider]

if config.database.dataBaseAttr.cloudChargeType == "Serverless":
    assert config.database.dataBaseAttr.databaseEngine == "MySQL",
    "databaseEngine must be set to MySQL when creating a serverless instance"
    aliyunDBInstance = alicloud.AlicloudDBInstance {
        engine = config.database.dataBaseAttr.databaseEngine
        engine_version = config.database.dataBaseAttr.databaseEngineVersion
        instance_type = "mysql.n2.serverless.1c"
        instance_charge_type = config.database.dataBaseAttr.cloudChargeType
        instance_name = _alicloudResourceName
        vswitch_id = _alicloudDependencyPrefix + alicloud_config.alicloudVswitchMeta.type
        + ":" + aliyunVswitch.vswitch_name + ".id"
        category = "serverless_basic"
        security_ips = ["0.0.0.0/0"]
        serverless_config = [alicloud.serverlessConfig{}]
    }
```

Demo

https://kusionstack.io/docs/user_docs/getting-started/usecases/deliver-the-wordpress-application-on-kubernetes-and-clouds

产品建设与文化推广

Developer 的收益

用户调研

- 调查问卷
- 用户访谈
- 焦点小组

数据分析

- 用户行为分析
- 数据挖掘分析
- A/B 测试

运营

- 周期性 release note
- 用户原声跟踪
- 布道宣传

领导层的支持

- 价值导向
- 行业趋势
- 整个组织的收益

一些挑战

- 研发体验
- 用户友好的文档
- 新用户上手

- 配置正确性保证
- CI 流水线效率
- CMDB 服务化

- Code GPT
- 异常检测
- 智能决策

在蚂蚁和其他公司的实践

1K/day	10K+/day	1 : 9	100K+
Pipelines	KCL Compilations	Plat : Dev	Commits
600+	5.7K+	1.2M+	10M+
Contributors	Projects	KCL Codes	YAML

Adopted by

HUAWEI

KYLIGENCE®

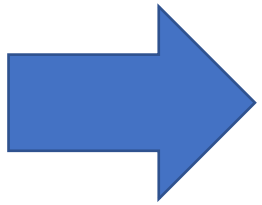
有赞

在蚂蚁和其他公司的实践

有赞容器化改造

KusionStack 技术栈

自建适配流水线



应用建模

版本化发布

降本增效

37

Projects

138

Clusters

98%+

Container

欢迎加入我们

- Web Site
  - https://kusionstack.io/
- Github
  - https://github.com/KusionStack/kusion
  - https://github.com/KusionStack/KCLVM
  - https://github.com/KusionStack/konfig
- Twitter
  - @KusionStack

微信小助手

Thank You

李大元