

# 知乎测试环境建设

知乎-QA：徐实

知乎

# 测试环境的介绍

- 测试环境痛点和需求
- 知乎测试环境背景
- 知乎的测试环境介绍

知乎

# 测试环境的痛点

资源管理

部署频繁

多人干扰

线上数据污染

测试覆盖不全

质量和效率 一路下行

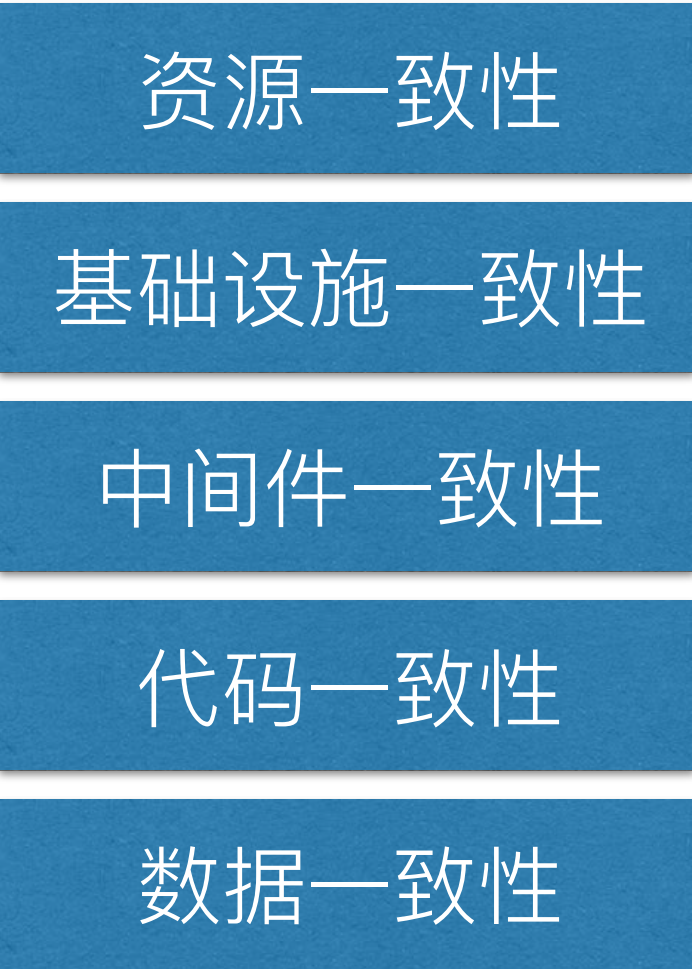
知乎



# 测试环境的需求



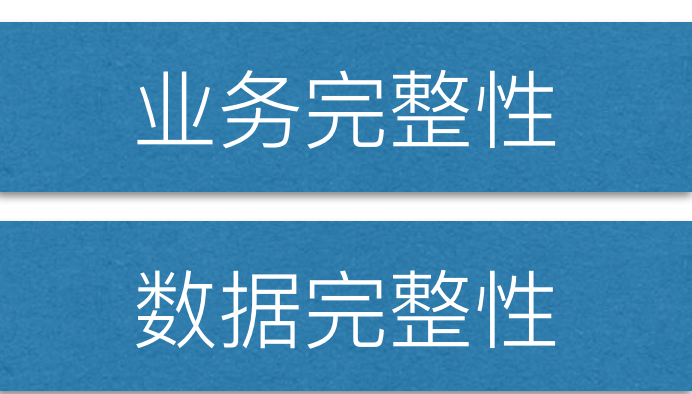
一致性



隔离性



完整性



稳定性



资源申请

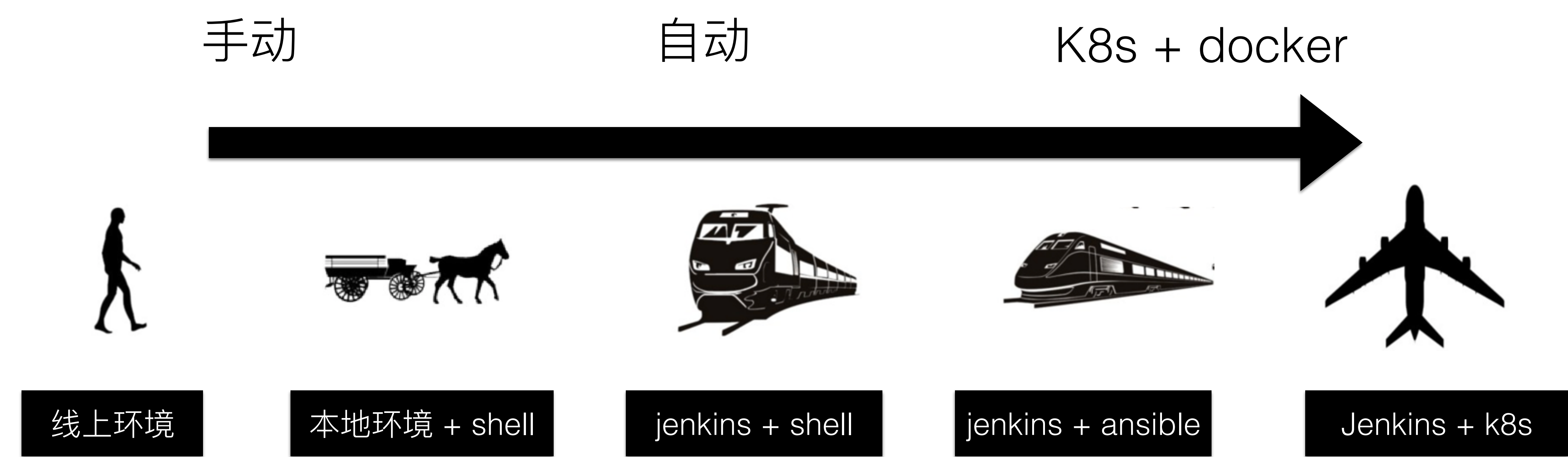
环境初始化

服务可用性

服务发现

数据维护

# 测试环境演进





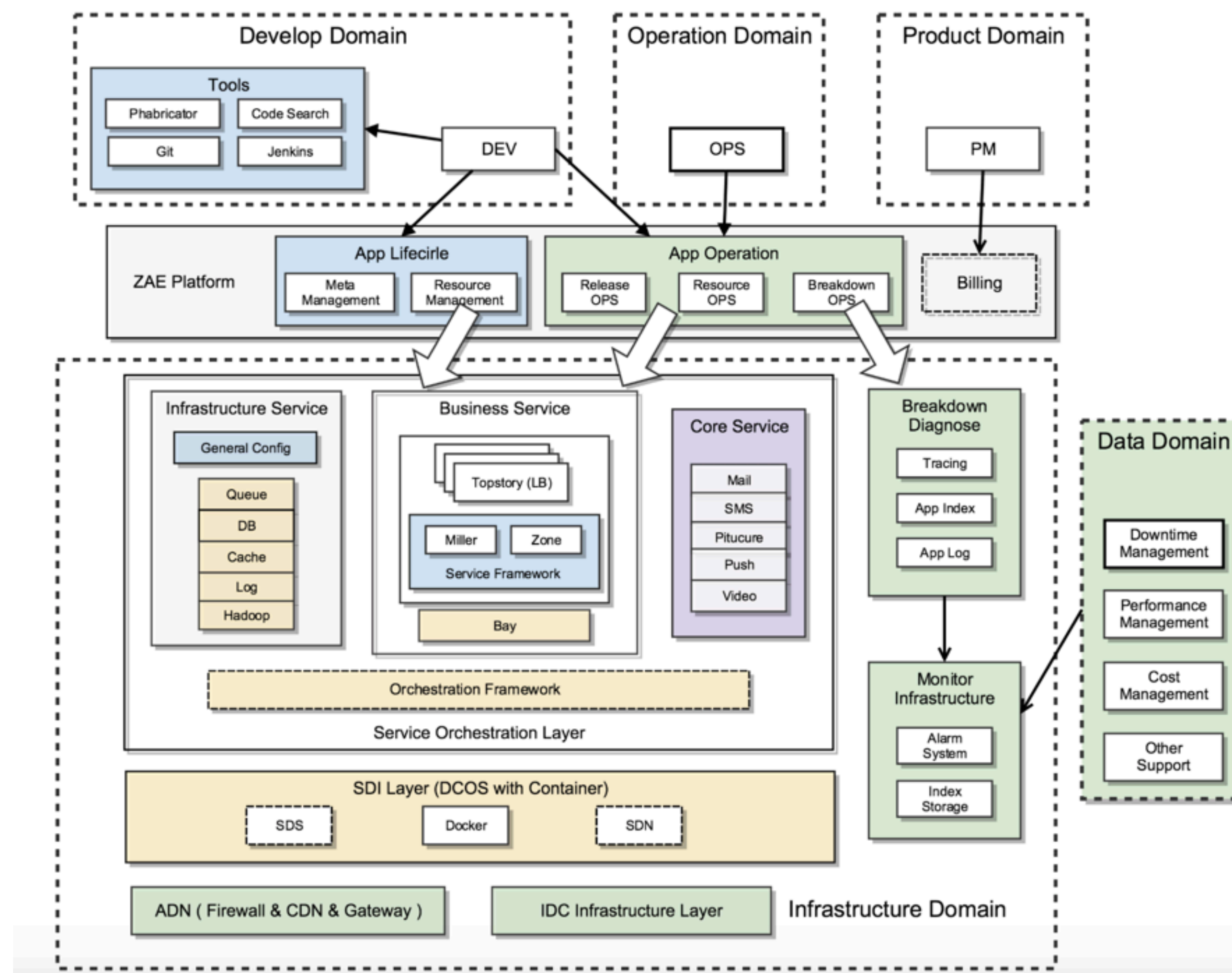
# 背景：知乎的容器化建设

知乎在 2016 年已经完成了全量业务的容器化，并在自研容器平台上以原生镜像的方式部署和运行，并在后续陆续实施了 CI、Cron、Kafka、HAProxy、Twemproxy 等系列核心服务和基础组件的容器化。

站在巨人的肩膀上：

Docker：标准化，快速化，隔离性，易扩展

K8s：自动化运维，快速调度



知乎

# 知乎现有环境

- 线上环境
- 办公室环境
- 测试环境
- 分支部署环境
- 联调环境

部署阶段设置	
	测试环境 已默认启用，并会自动部署
	办公室环境 已禁用
	金丝雀 已启用
	金丝雀-2 已启用
	生产环境1 已启用



知乎

# 知乎现有环境的问题？

- 线上环境
- 办公室环境：数据与线上共用，容易脏数据/造成影响
- 测试环境：一份环境，多人共用
- 分支部署环境
- **联调环境**

知乎

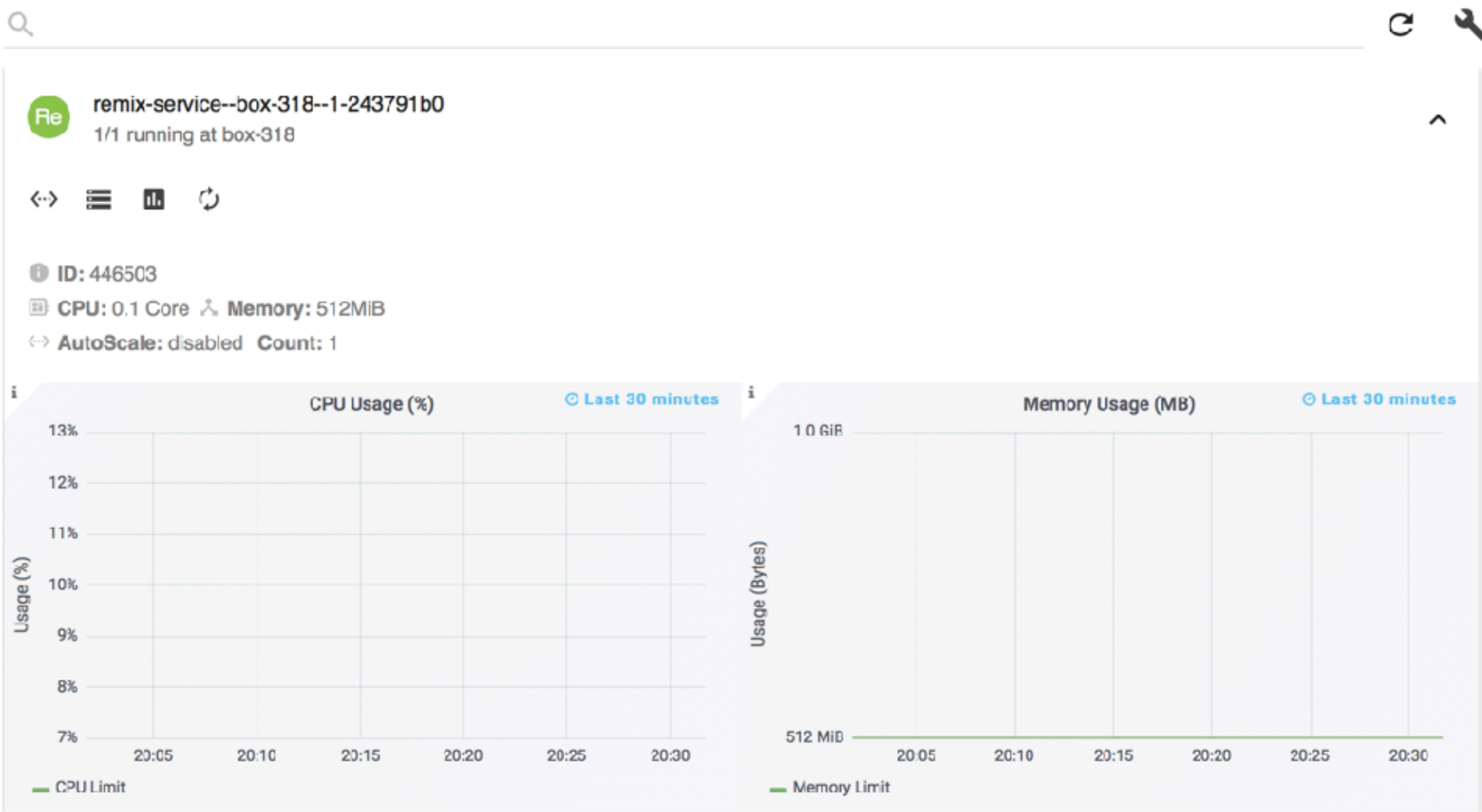


# 知乎测试环境：一致性/稳定性

- 资源由 ZAE 统一分配和管理
  - 每次测试环境的构建对应容器镜像的构建
- 稳定性：资源监控，弹性伸缩，故障恢复

```
20:18:11 创建测试数据库: test_40e31c9cf82e4f2e9c2969t...
[Pipeline] }
[Pipeline] // ansiColor
[Pipeline] echo
20:18:11 create database test_40e31c9cf82e4f2e9c2969t...
[Pipeline] echo
20:18:11 mysql -h127.0.0.1 -uroot -pjenkins -e 'create database test_40e31c9cf82e4f2e9c2969t...'
[Pipeline] sh
20:18:11 [zhihu-... flow-std] Running shell script
[Pipeline] echo
20:18:12 mkdir -p /data/cache/apps/zhihu-.../master
[Pipeline] sh
20:18:12 [zhihu-... std] Running shell script
[Pipeline] ansiColor
[Pipeline] {
[Pipeline] echo
20:18:12 生成 Dockerfile
```

测试环境    生产环境1    生产环境2 (危险勿动)



编辑联调应用

应用名称  
market-webapp

代码来源  
当前生产环境版本

复制离线任务 ☐

使用独立资源 (Beta) ☒

自动复制存储数据 (Beta) ☐

删除    取消    保存

知乎

# 知乎测试环境：一致性

- 保证测试环境资源与线上资源一致
  - 资源个数一致（线上有哪些资源，测试环境就有哪些资源）
  - 资源名称一致（可选，方便使用资源发现）
- 保证测试环境服务（如容器、定时任务等）运行正常
- 保证 NGINX 配置与线上一致
- 保证各 HTTP/RPC 服务访问正常
- 迁移到资源发现
- 保证依赖的服务也满足以上需求

知乎

# 知乎测试环境：一致性

- 使用 git 维护数据库的变更记录
- 每次变更会有对应的 MR Request
- 数据库的表结构线上、联调环境同步
- 每次可创建新 DB 容器

部署阶段设置

-  测试环境  
已默认启用，并会自动部署
-  办公室环境  
已禁用
-  金丝雀  
已启用
-  金丝雀-2  
已启用
-  生产环境1  
已启用



zhihu-database ▾

Database schema in Zhihu.

Project ID: 1571

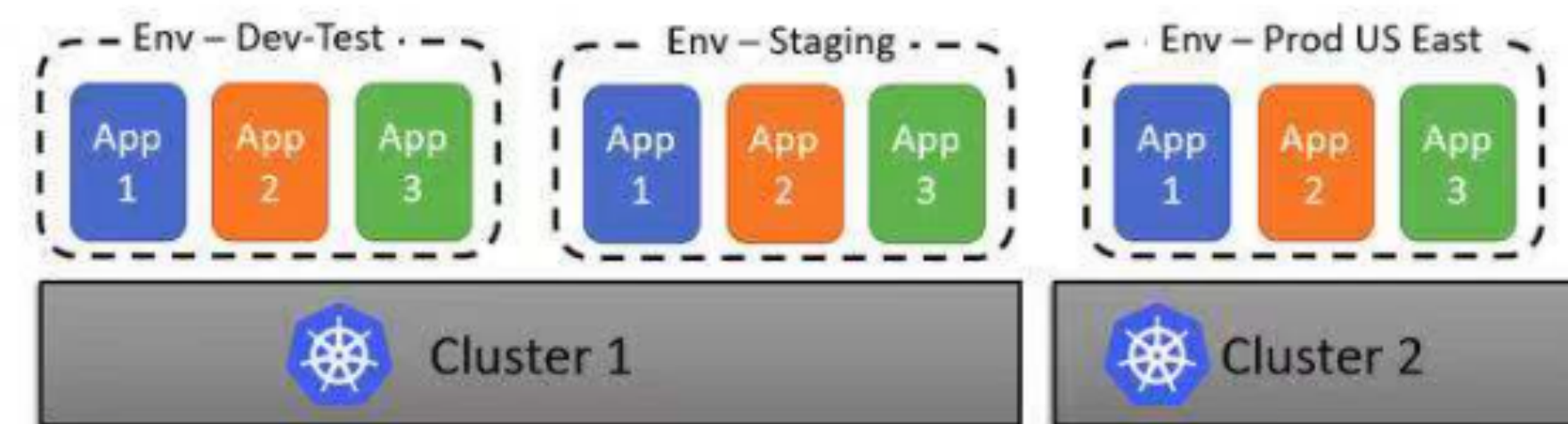
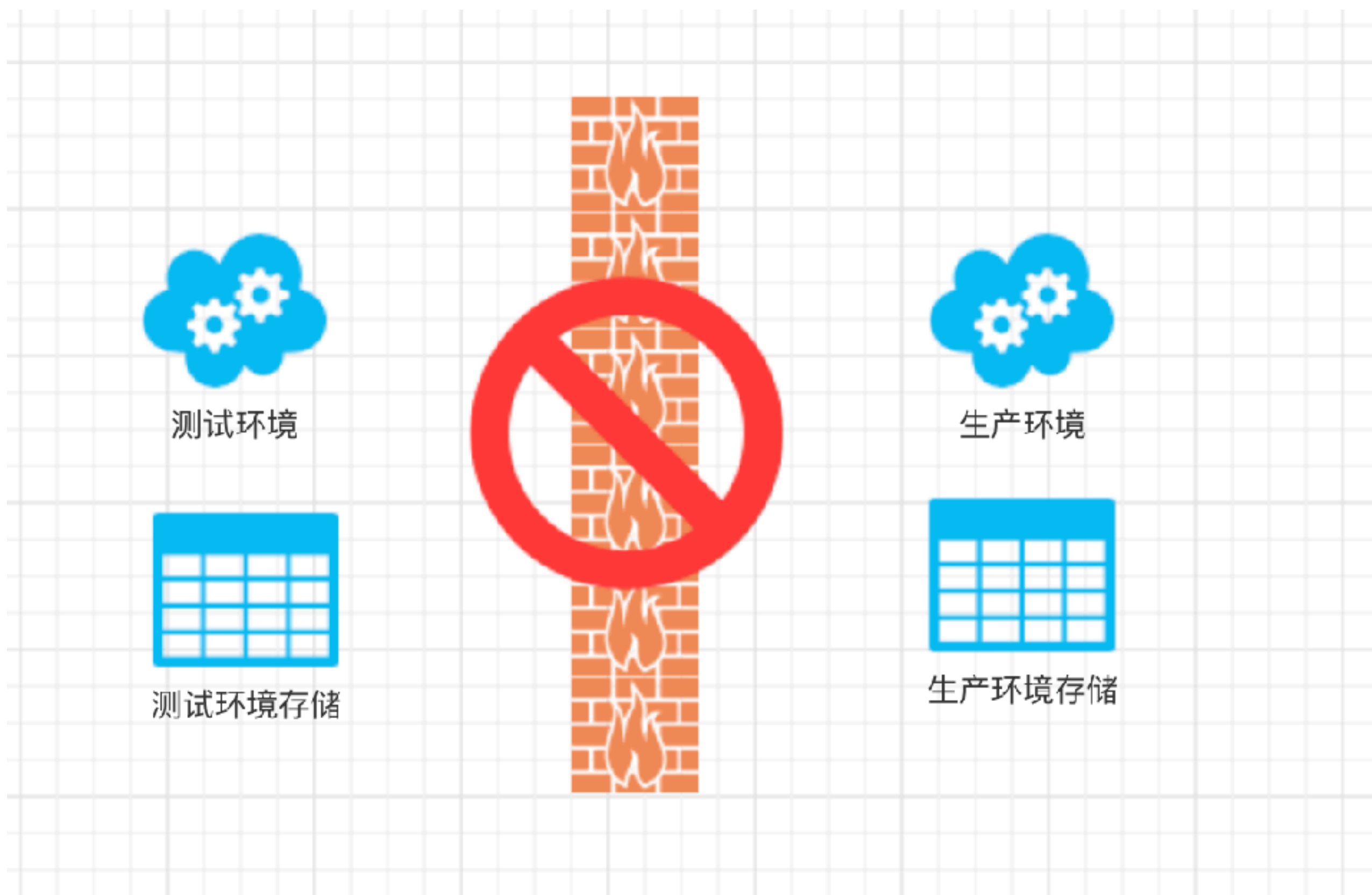


知乎



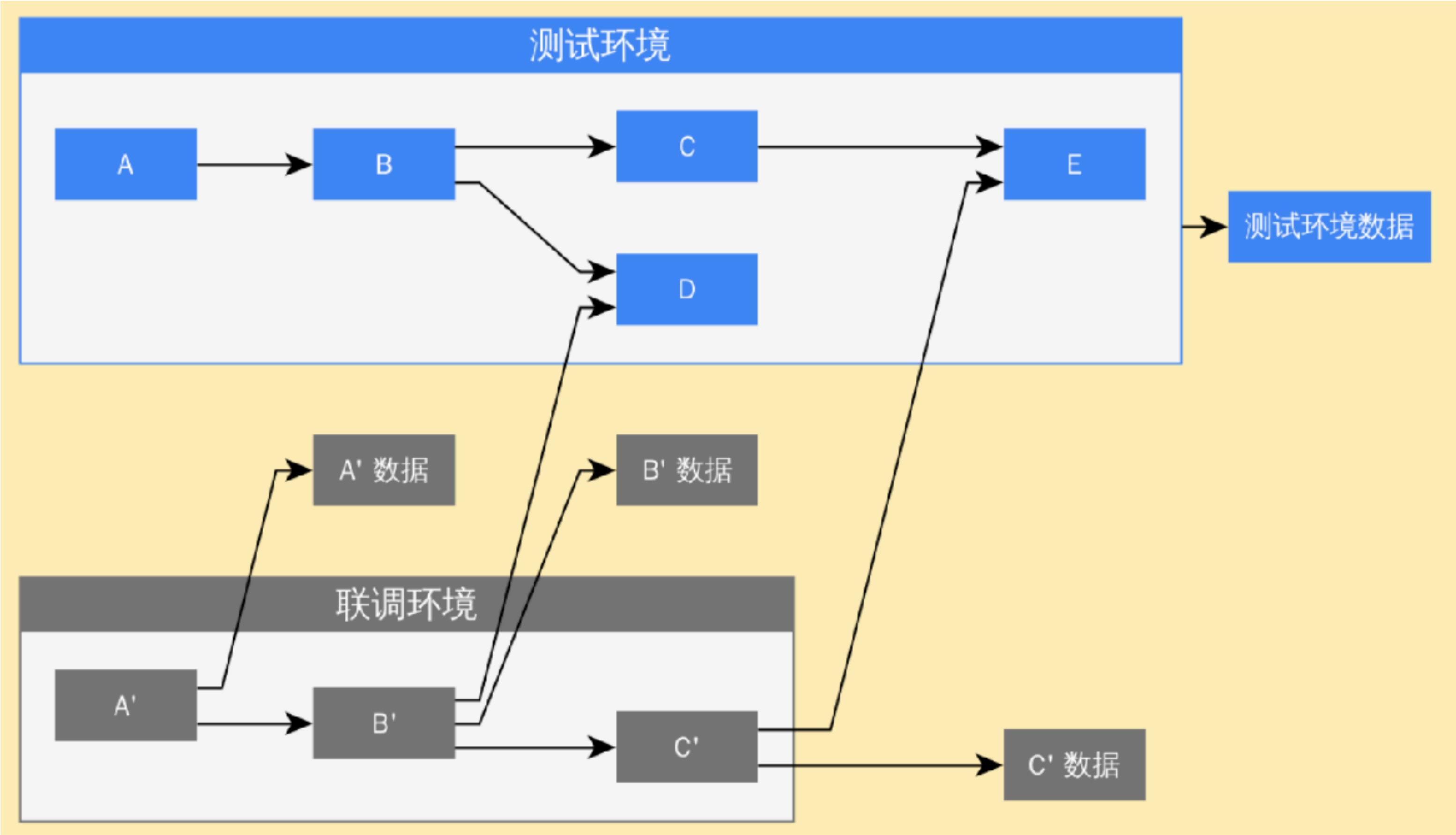
# 知乎测试环境：隔离性

Testing 环境为与生产环境在网络和数据层面完全独立的环境



知乎

# 知乎测试环境：测试环境部分隔离



- 服务发现:
- ENV
- DNS

# 知乎测试环境：测试环境部分隔离

自动为联调应用创建一套独立于原本测试环境的资源，  
并可以自动从原本的测试环境复制一份相同的数据。  
本次联调测试所有数据都会写入新创建的存储中，  
而不影响测试环境的存储。  
测试资源伴随测试环境销毁。

添加联调应用

应用名称

代码来源

当前生产环境版本

使用独立资源

自动复制存储数据

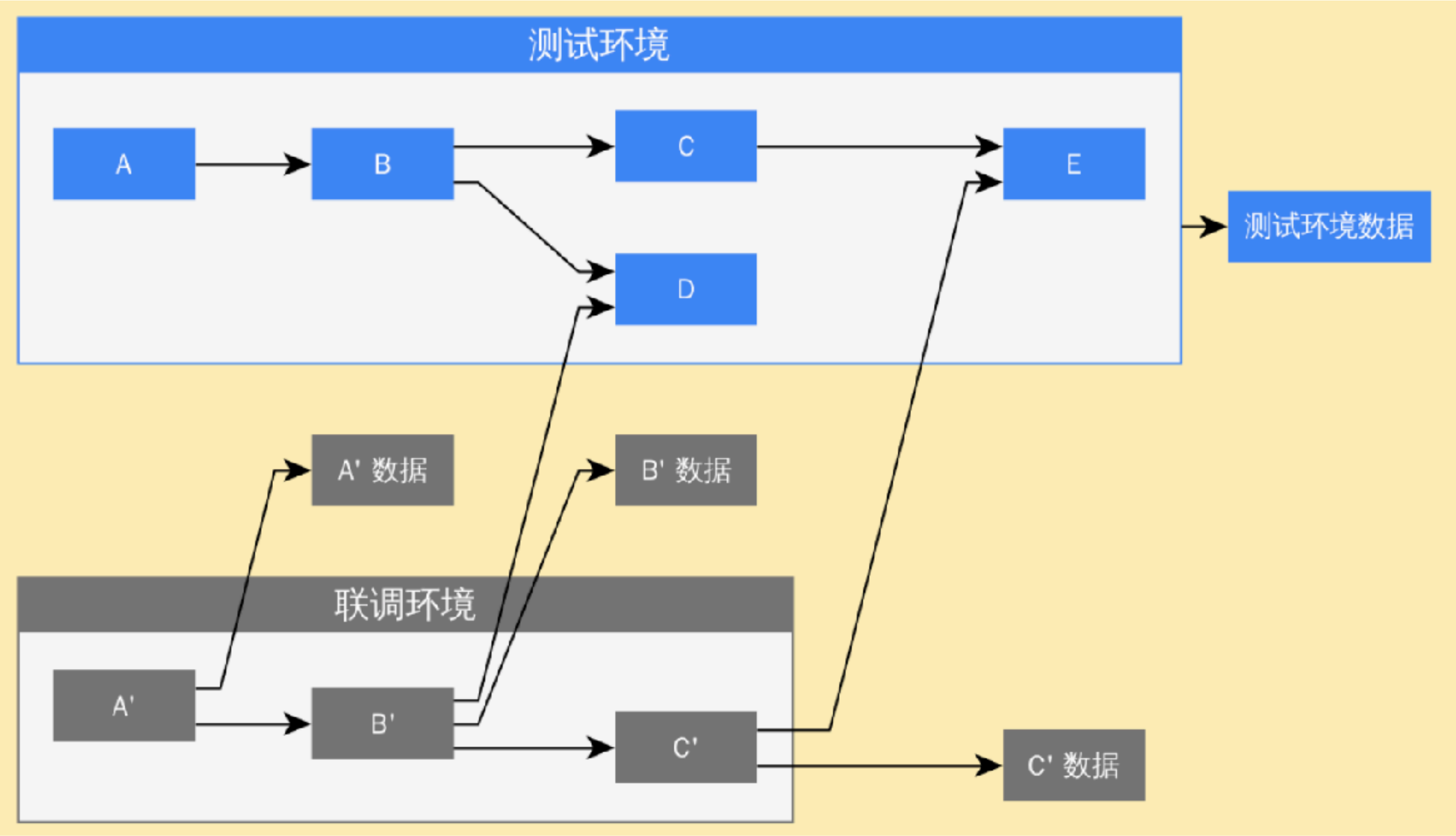
取消

保存

知乎



# 知乎测试环境：业务完整性



# 知乎测试环境：完整性

基础数据：

- 测试基准环境各业务维护基础数据可用性

业务数据完整性：

- 直接使用 MySQL 语句等方式操作存储资源，来注入数据
- 使用业务外层 HTTP 接口来创建业务数据 (自动化)

避免数据污染

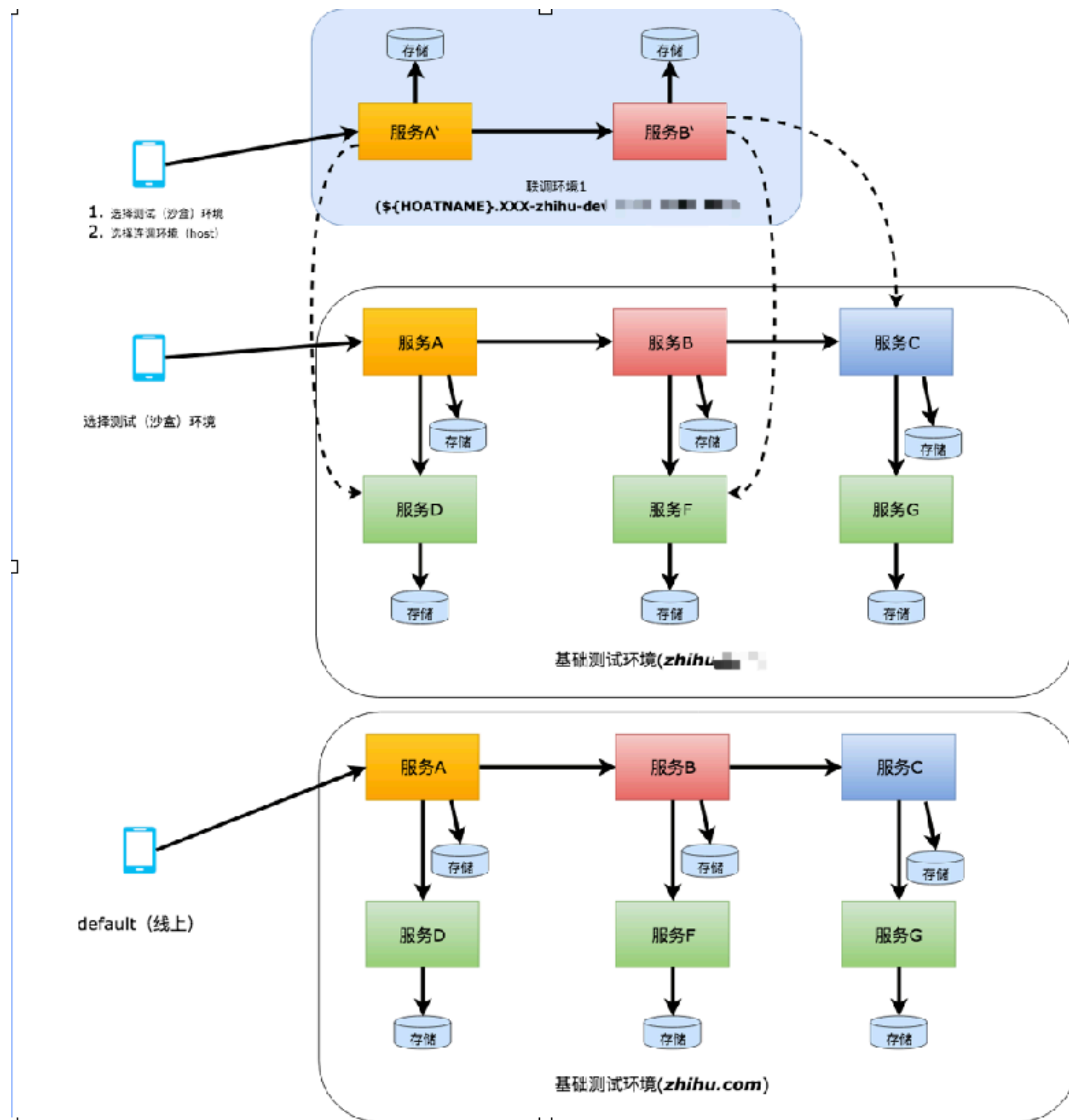
定期测试环境重置，以保证测试环境数据的持久可用：

- 定期（例如每天凌晨）重置测试环境所有应用的业务数据
- 执行初始化脚本，对所有数据进行重建

知乎

# 客户端与测试环境交互

- 基础环境
  - 服务代码、存储结构与生产环境一致
  - 系统使用脚本访问接口产生数据
- 联调环境
  - 开发分支代码改变触发重新部署
  - 存储结构发生变化触发数据库更新
  - 系统使用脚本访问接口产生数据或从基础环境拷贝数据



知乎

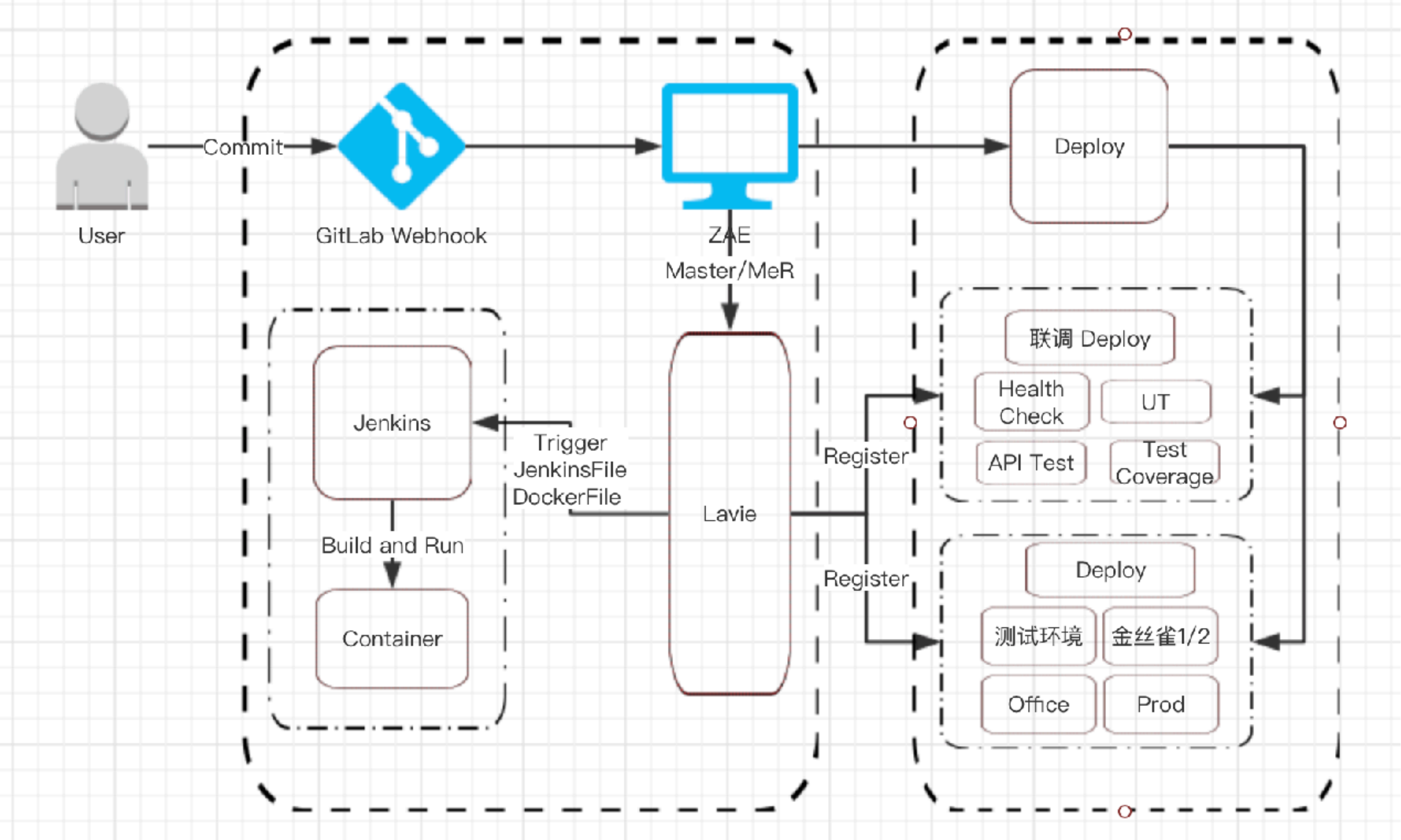


# 知乎测试环境：环境治理步骤

- 确定要完善的业务线
- 确定核心业务场景
- 确定各业务线 QA 和开发接口人
- 确定各业务线核心场景负责人
- 各业务线开发保证应用在测试环境的可用性
- 各业务线开发提供数据注入接口
- -----
- 各业务线 QA 根据接口完成初版注入数据脚本
- ZAE 提供能力，定期重置测试环境数据
- 各业务线 QA 验证本业务线测试环境可用性
- 数据平台提供测试环境离线集群
- 各业务线开发保证应用离线业务可用性

知乎

# 知乎测试环境：持续集成





个人微信

微信搜索「知乎招聘」公众号

THANKS



# 知乎

发现更大的世界