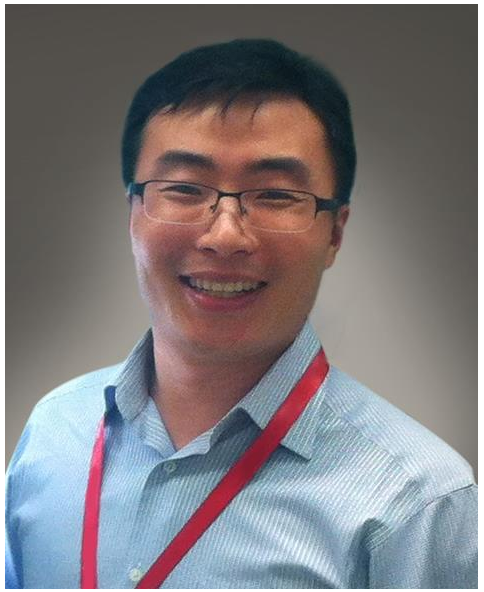
案例分享
从传统.NET单体到.NET Core微服务架构

马洪喜 行云创新CTO

行云创新CTO—马洪喜

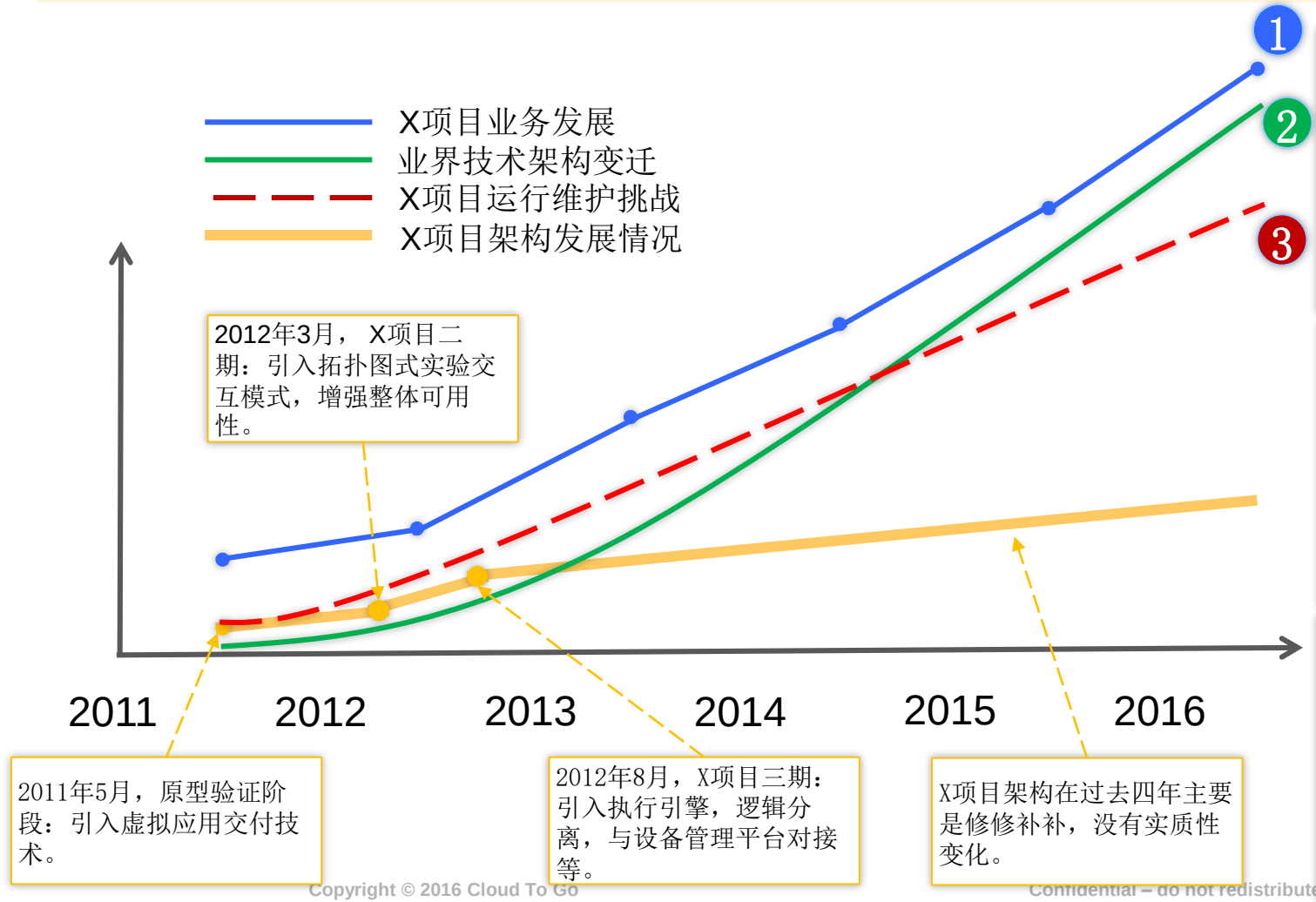
- ✓ 过去十年一直聚焦于云计算产品研发和项目交付
- ✓ 为最早一批金融、电信行业云计算提供咨询服务
- ✓ 热衷于参加社区活动，连续三年荣获微软MVP 称号
- ✓ 曾供职于 Oracle 、Citrix 、Rancher Labs
- ✓ 现任行云创新 CTO，负责公司开发云产品研发和交付



从一个真实的案例说起

X项目曾经面临的挑战

X项目是基于.NET构建的一套复杂的远程实验交付系统。自2011年以来，从最早的远程实验接入，到后来的的复杂实验业务场景交付，其重要性和担当的业务价值越来越大，但其技术架构发展较为缓慢，无法高质量满足多变的业务需求，在运行和维护中由于陈旧架构导致的问题较为突出，也不符合互联网化大背景下的技术架构发展方向。



挑战一：业务需求越来越难以满足。X项目交付量从最初几千班天，发展到几万班天，要处理多种业务。业务需求越来越复杂、差异化明显且具有多变性。相对地，X项目架构在过去四年内并未做任何升级改造，客户需求多以“打补丁”的方式满足，难以继续满足未来业务发展需求。

1

挑战二：部分技术已经逐渐落后于行业发展。从行业趋势来讲，企业业务正面向互联网化转型，在“高频率版本发布”、“更加注重用户体验”等目标驱动下，企业架构更多的开始学习BAT，采用开放平台和技术，如 Linux, Apache, MySQL,Docker等；同时，在开发运维模式上，企业正在由传统的开发和运维分离，向开发运维一体化即DevOps转变。

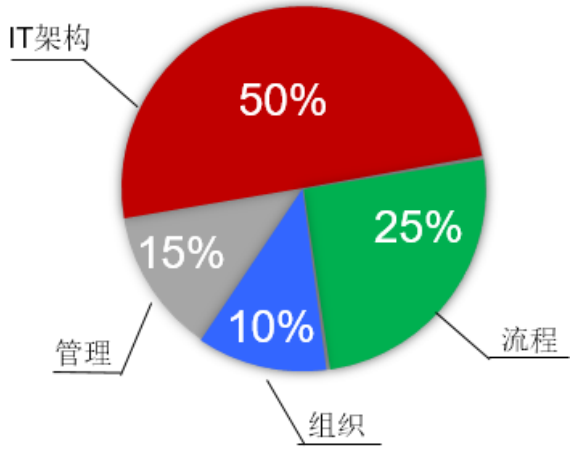
2

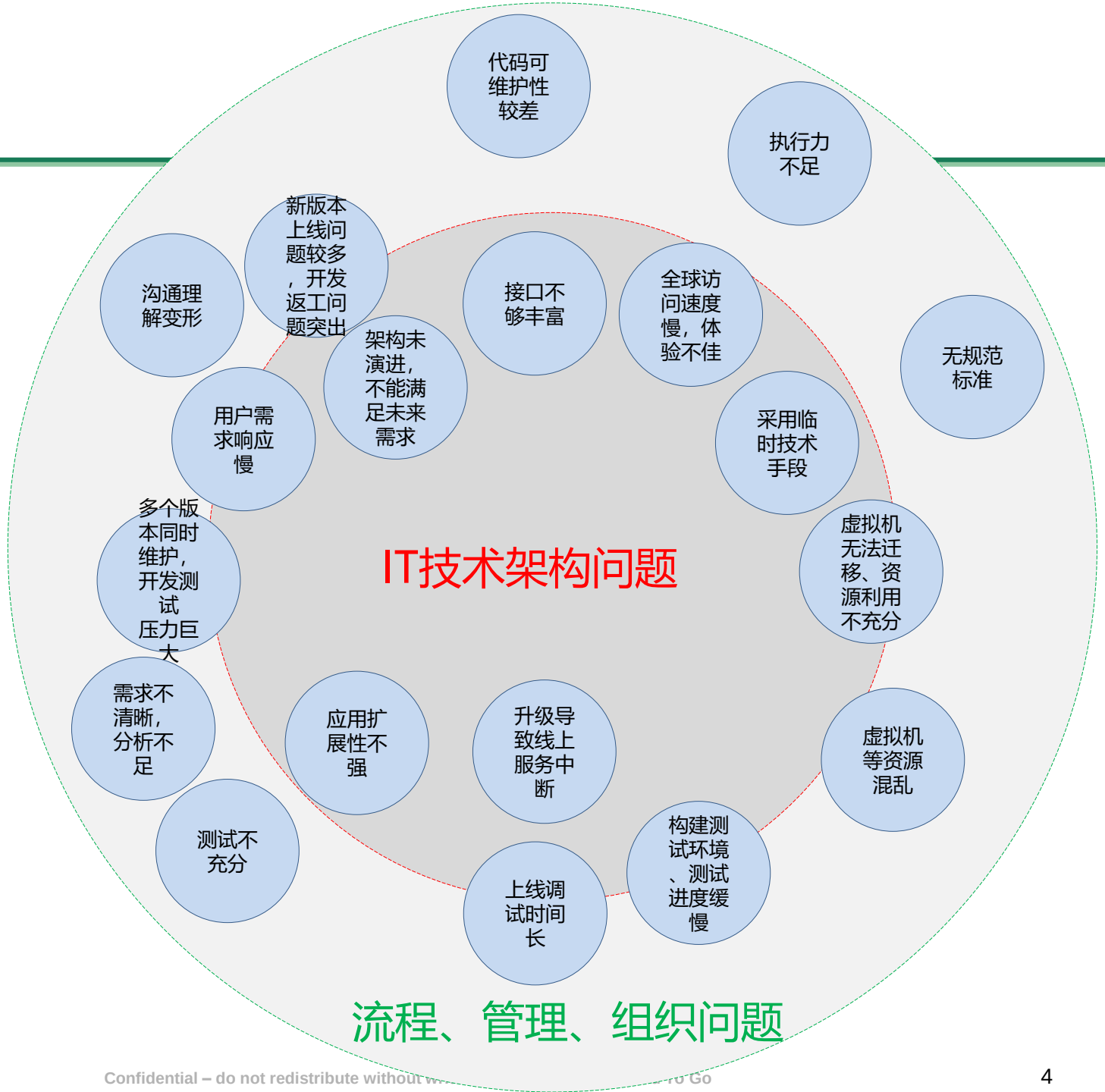
挑战三：运维压力越来越大。当前架构涉及到的组件全部是自行部署和维护，平台、数据库、监控、应用配置、大数据等，对运维团队提出了较高的要求。具体表现为：底层组件无法跟随上层业务压力变化而变化，出现问题后定位过程时间长且难度高。

3

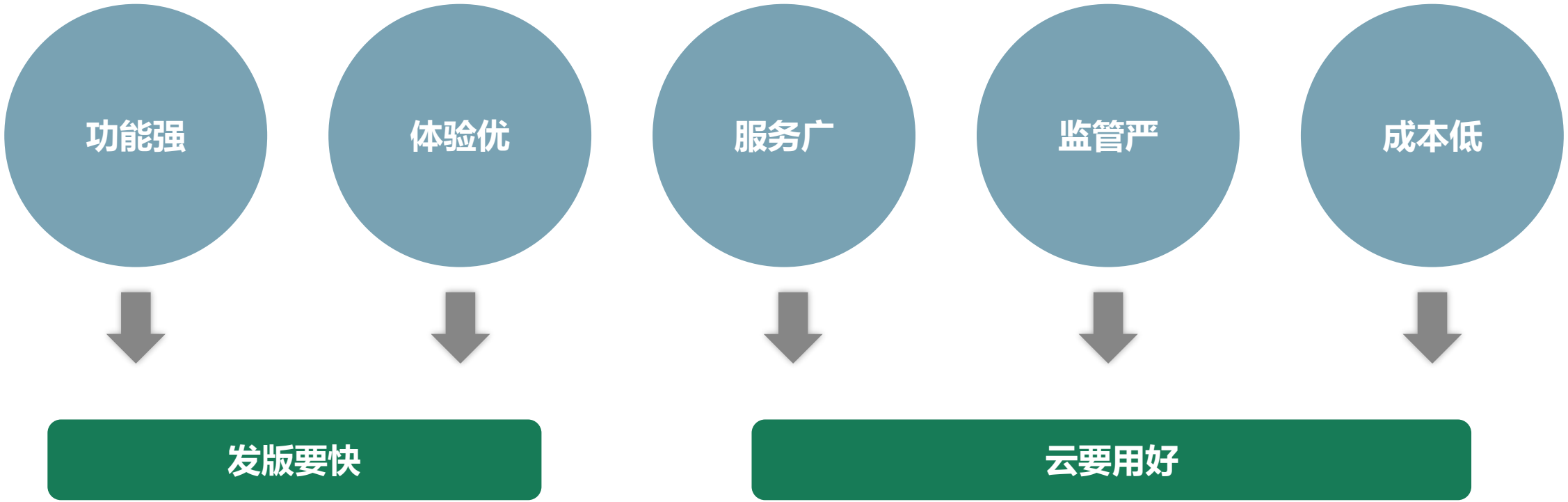
项目问题具体分析

2016年8月，行云顾问组对X项目进行了评估，与包括管理组、需求分析、PM、架构师、开发、测试、运维等多位同事进行了访谈，特别就不同角色对项目的痛点进行了识别，其中技术架构与流程相关问题较为突出，组织和管理方面问题也占有一定比例。

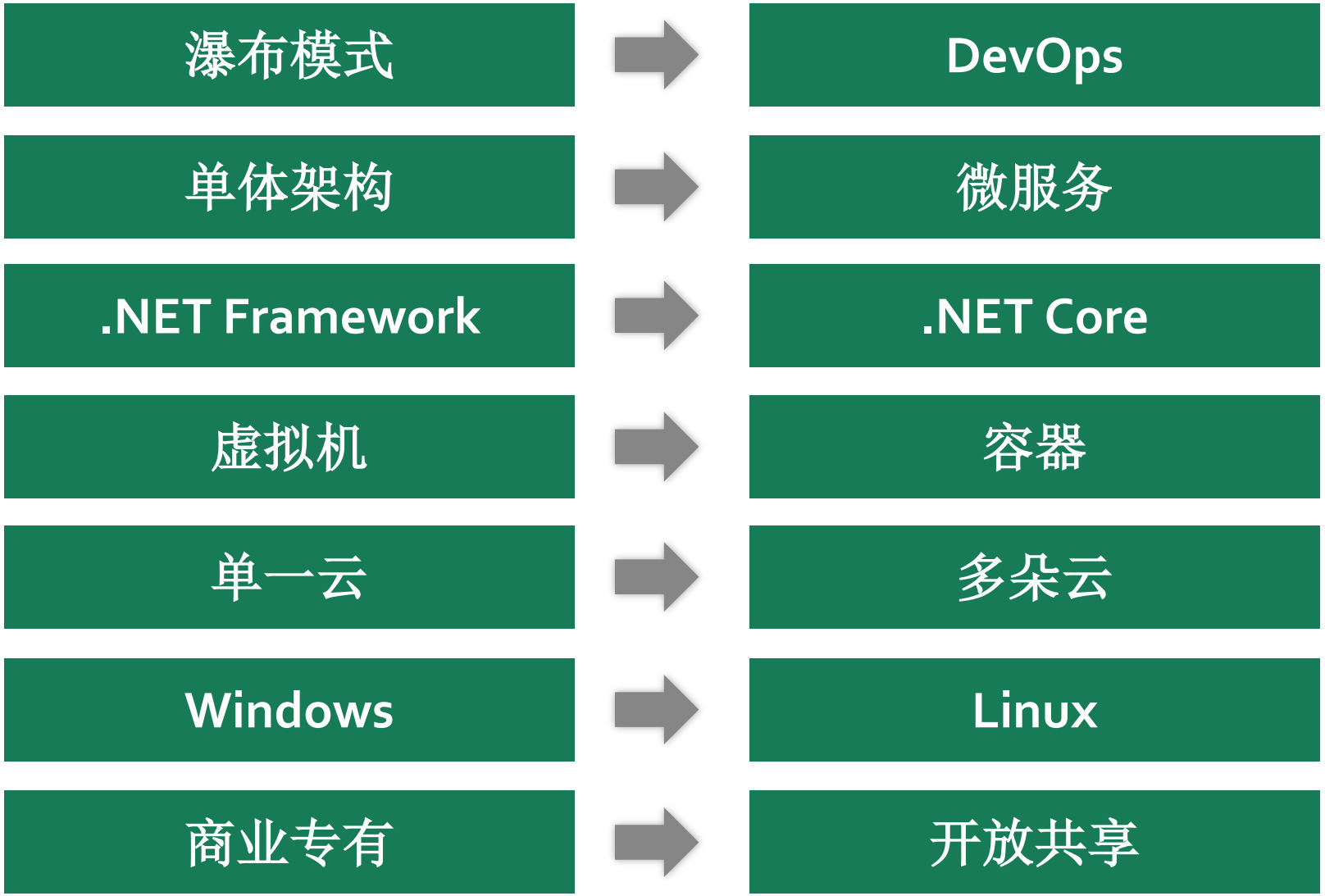




X项目业务发展诉求

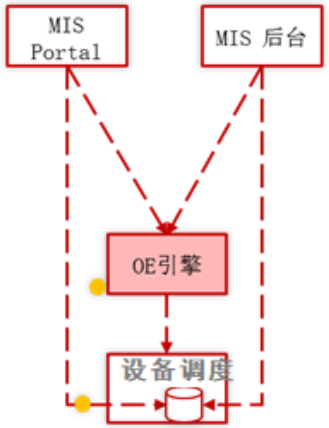


技术解决思路



微服务是X项目重要的技术架构变革

传统的单体应用架构与DevOps方法格格不入，特别是X项目经过数年发展，其单体架构已经成为项目发展的关键阻碍。



模块间依赖性强，相互影响

架构修修补补，越发复杂

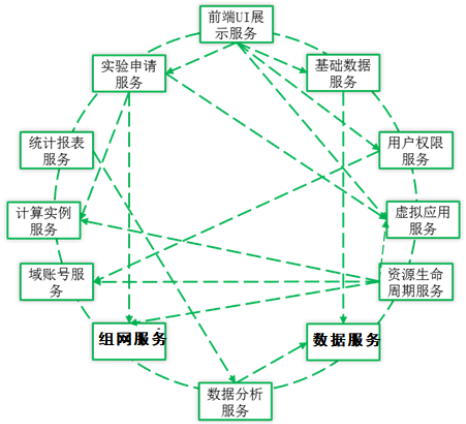
核心技术组件成为发展瓶颈

无法有效演进，阻碍对需求的快速匹配

测试成本高，测试不充分，质量无法保证

上线时间长，成本高，问题较多

单体架构



模块间松耦合，相互独立

架构灵活扩展，清晰明了

按功能点打散，解除瓶颈

持续演进，快速达成需求

测试范围明确，促成自动化测试提高效率

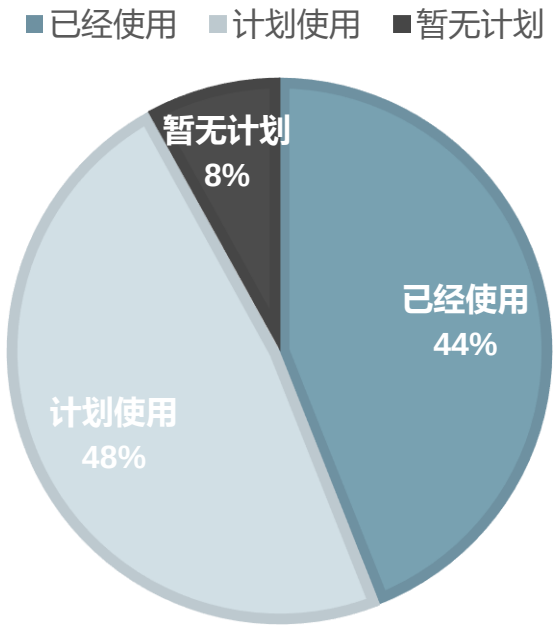
只上线变更的服务，通过自动化手段形成持续部署能力

DevOps
持续交付

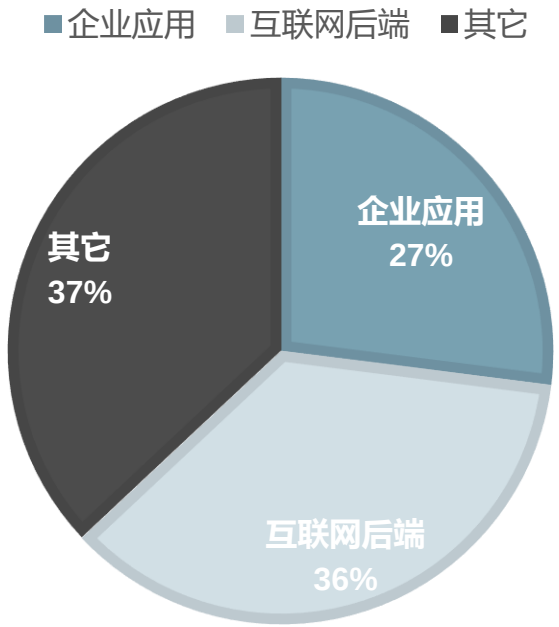
微服务架构

.NET Core是X项目重要的技术实现变革

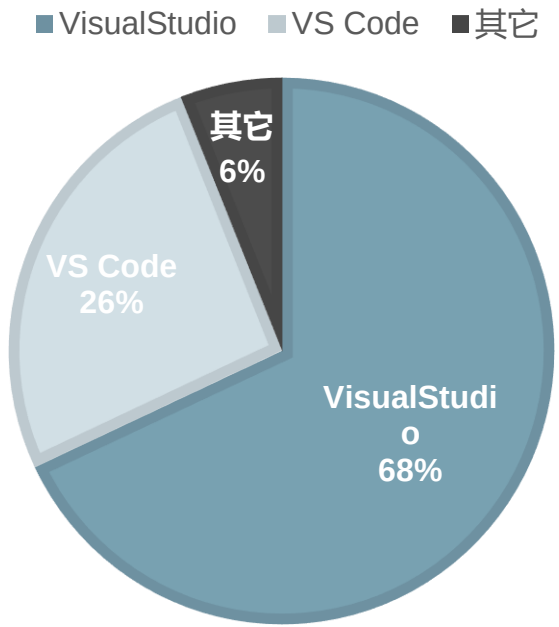
使用.NET CORE的开发者比例



使用.NET CORE的业务场景



使用.NET CORE的开发工具



数据来自张善友老师《2018 .NET开发者调查报告》

变革中需要重点考虑的几个主要问题

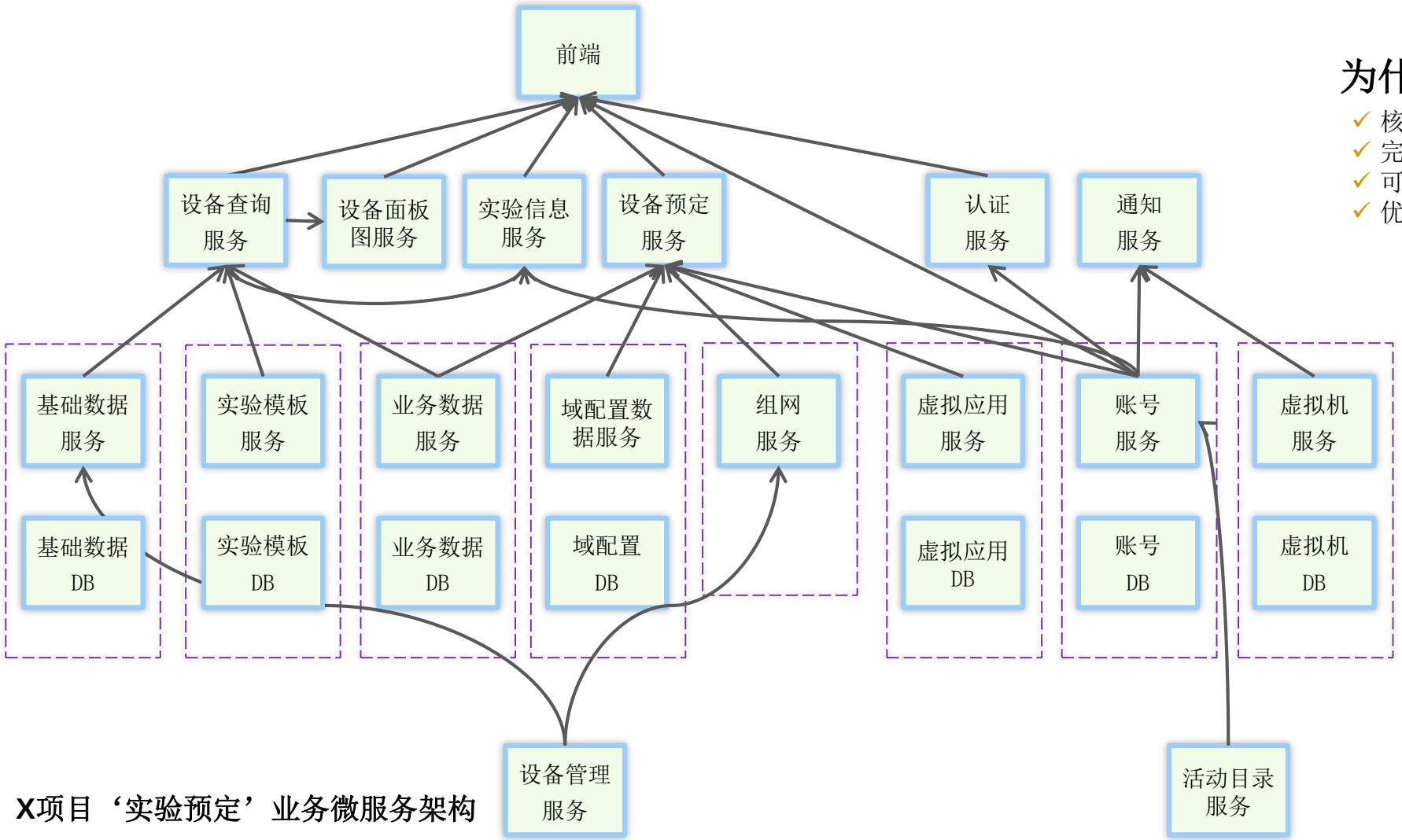
从核心业务还是周边业务开始

现有.NET都能迁移到.NET Core吗

怎么样利用好容器相关技术支撑变革

X项目微服务改造的切入点

对于X项目，微服务改造并非‘推倒重来’，而是在过去几年运营的积累上循序渐进完成。‘实验预订’被选为微服务改造的切入点。



X项目‘实验预定’业务微服务架构

为什么选择‘实验预订’业务？

- ✓ 核心的主线业务流程，覆盖30%以上业务；
- ✓ 完成‘实验预定’的微服务，其余部分迎刃而解；
- ✓ 可以与现有系统较好地过渡，用户无感知；
- ✓ 优先解决已成为瓶颈的技术组件，利于后期发展。

微服务拆分的原则是什么？

- ✓ 功能完整性、职责单一性；
- ✓ 促进可重用性；
- ✓ 逐步优化，迭代演进；
- ✓ 接口协议先行，保证兼容性。

.NET Framework to .NET Core

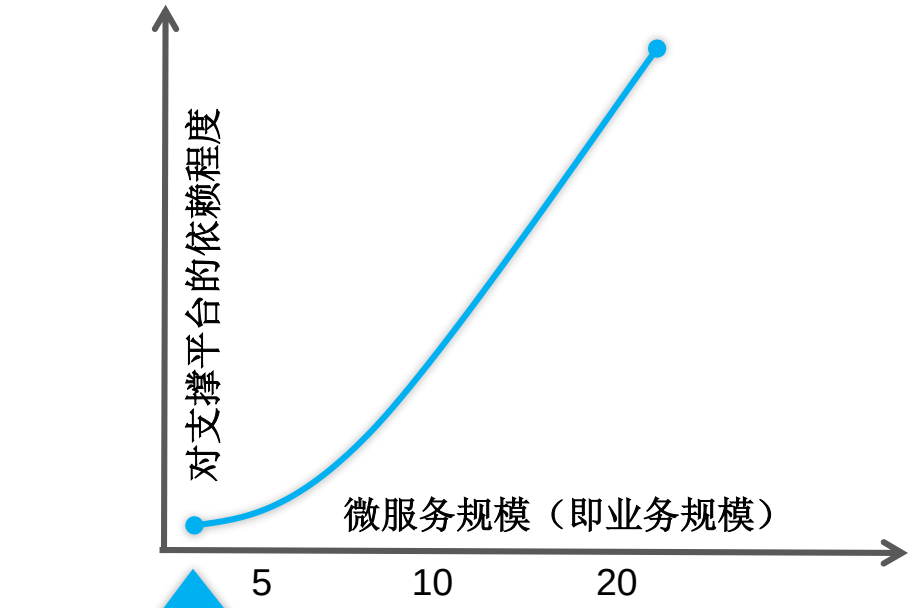
- ✓ 已有.NET代码和.NET Core的兼容性如何
- ✓ 哪些类型服务需要继续在.NET Framework上运行
- ✓ 数据库选型的考虑，依然SQL Server还是要切换到MySQL
- ✓ 面对新的语言、新的架构，开发人员知识转换难度
- ✓ 设立团队的一些考虑，对需求的准确理解是“重中之重”

处于技术变革中的应用，

.NET和.NET Core可能会并存一段时间

微服务支撑平台的必要性

向微服务架构的转变，仅仅在代码设计上做出改变是不够的，必须结合支撑微服务的平台能力才能让微服务落地、发挥出它的功效。



仅有一两个微服务时，似乎手工操作也可以解决，随着微服务规模扩大，发版频率几何增多，变更和运维压力大大增加。

	传统	微服务
运行环境	虚拟机	容器技术
构建方式	手动编译	持续集成
生产上线	人工部署	自动发版
业务响应	被动模式	自动伸缩
研发成果	互不透明	共享复用
代码管理	资源隔离	上收统管

诉求

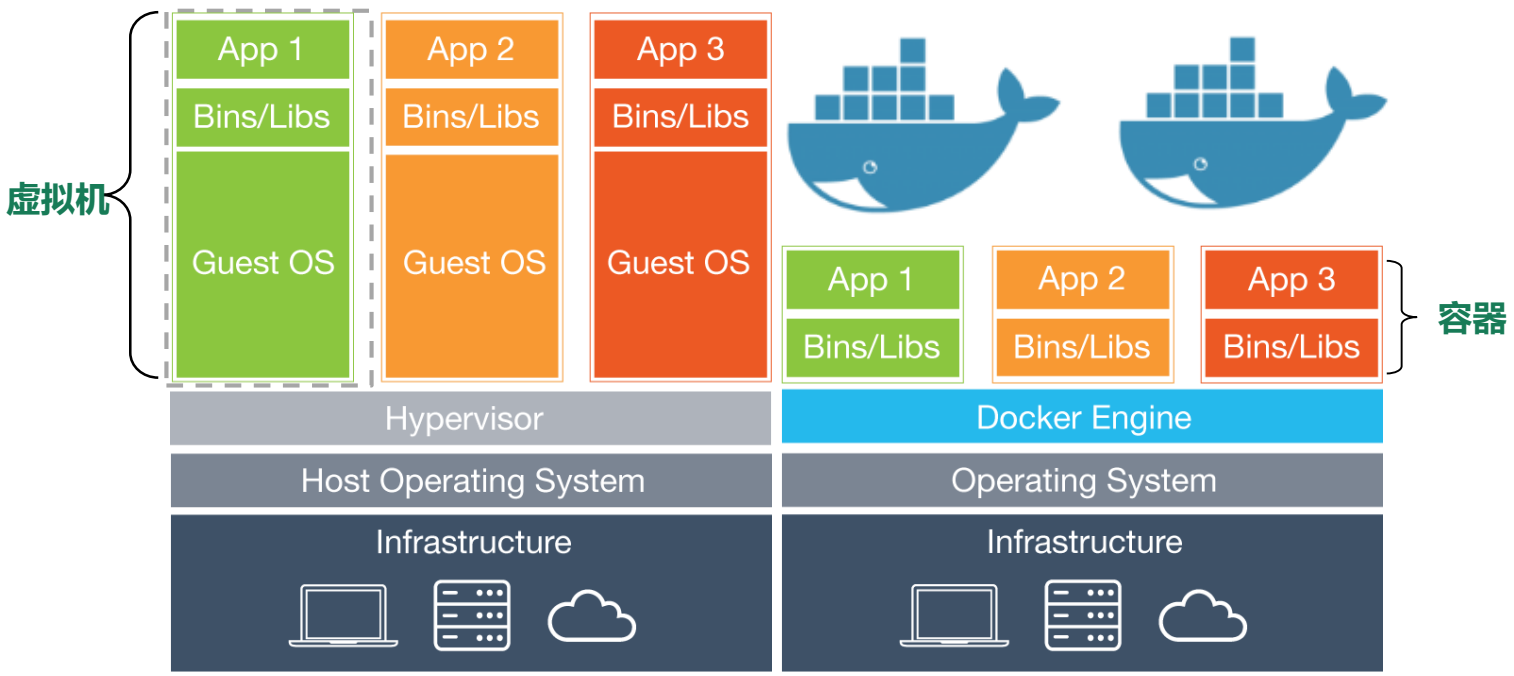
流程

工具

标准

支撑平台

容器——更轻量、更快速、更灵活



虚拟机　　容器

轻量

- 极低的开销
- 高效使用资源

快速

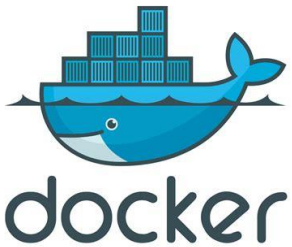
- 运行在系统内核上
- 业务秒速启动

灵活

- 易于迁移
- 易于管理

Microsoft ❤ Linux

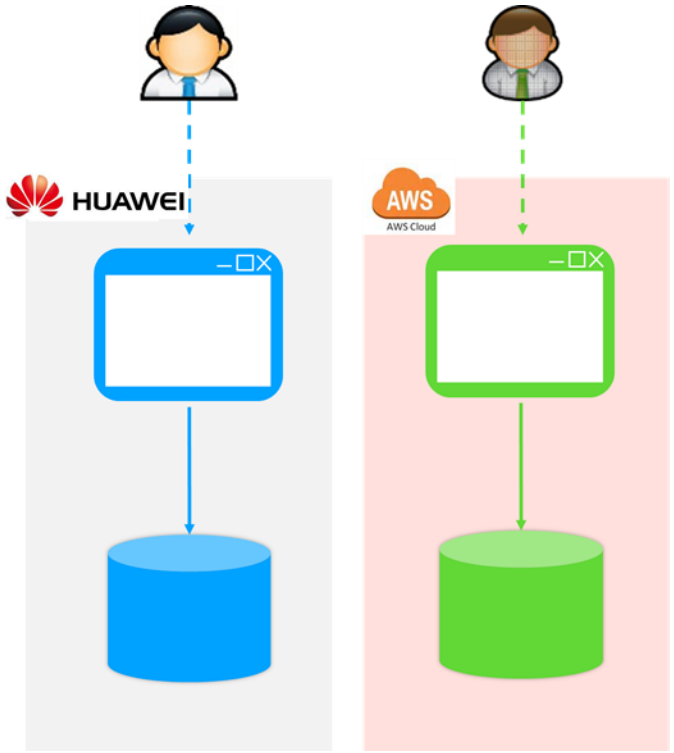
Microsoft ❤ Docker

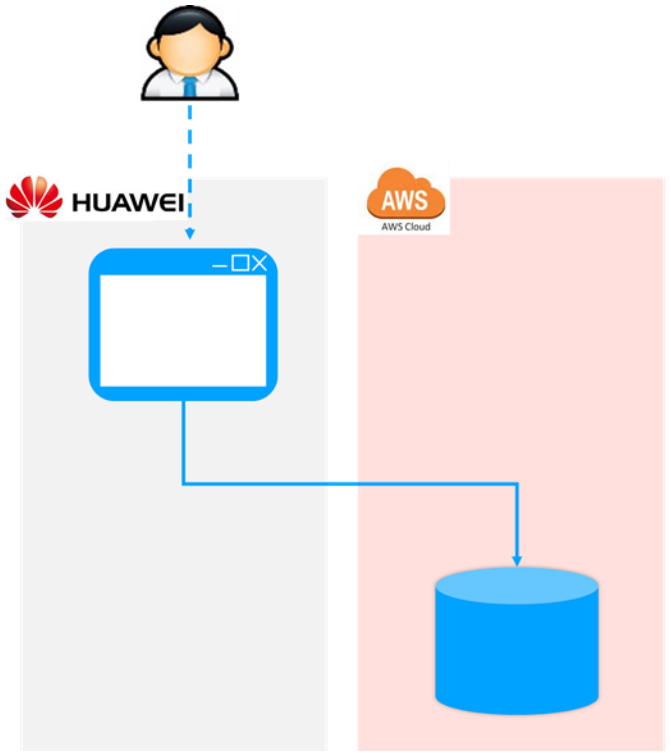


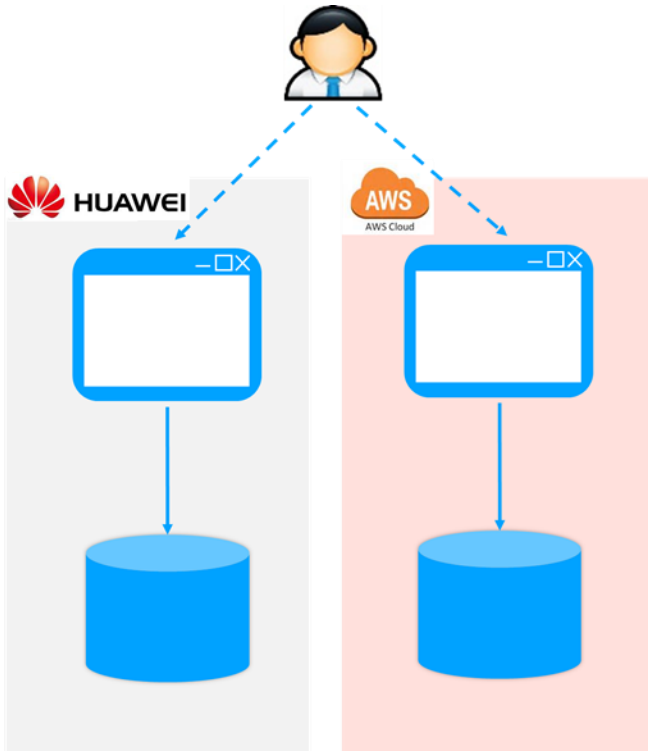
kubernetes

挑战：如何管理好多朵云和多操作系统下的容器环境?

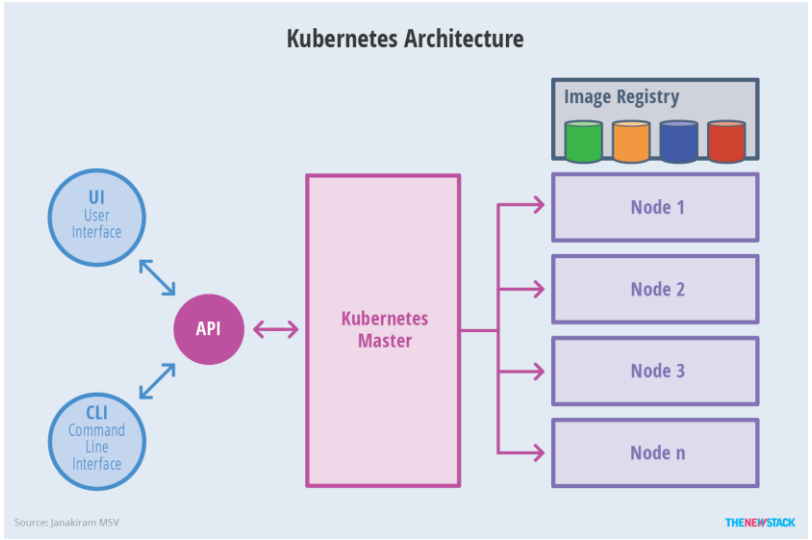
X项目中多云交付的三种典型场景

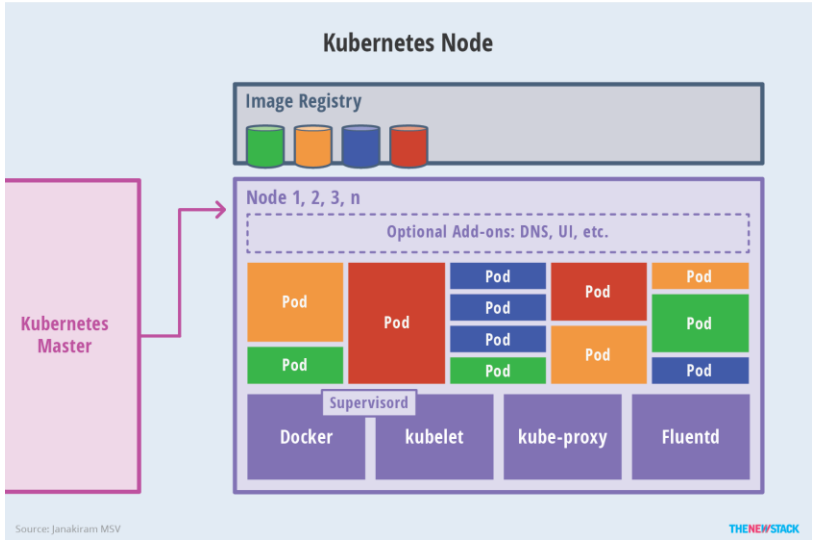






Kubernetes关注的是单一集群内容器管理

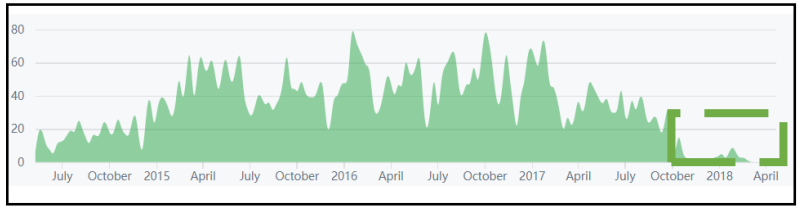




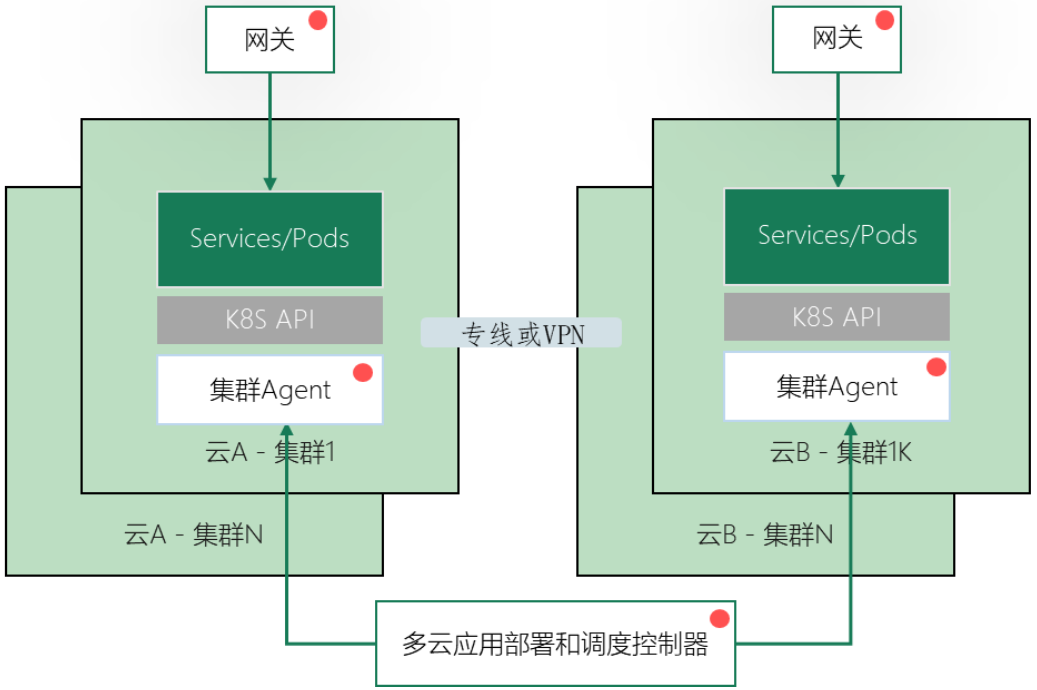
Federation (联邦) 项目为多集群而生，但还远未可用

- 14年开发至今依然Alpha
- 技术主张和实际需求偏差巨大
- 社区活跃度越来越低
- 对AWS and GKE/GCE严重依赖
- V1已经废弃，V2才刚刚开始

> *"Keep it simple, don't bake too much into Federation Api server...Migrating Azure to google is not a question for Federation"*
>
> *——Kelsey Hightower, Google*



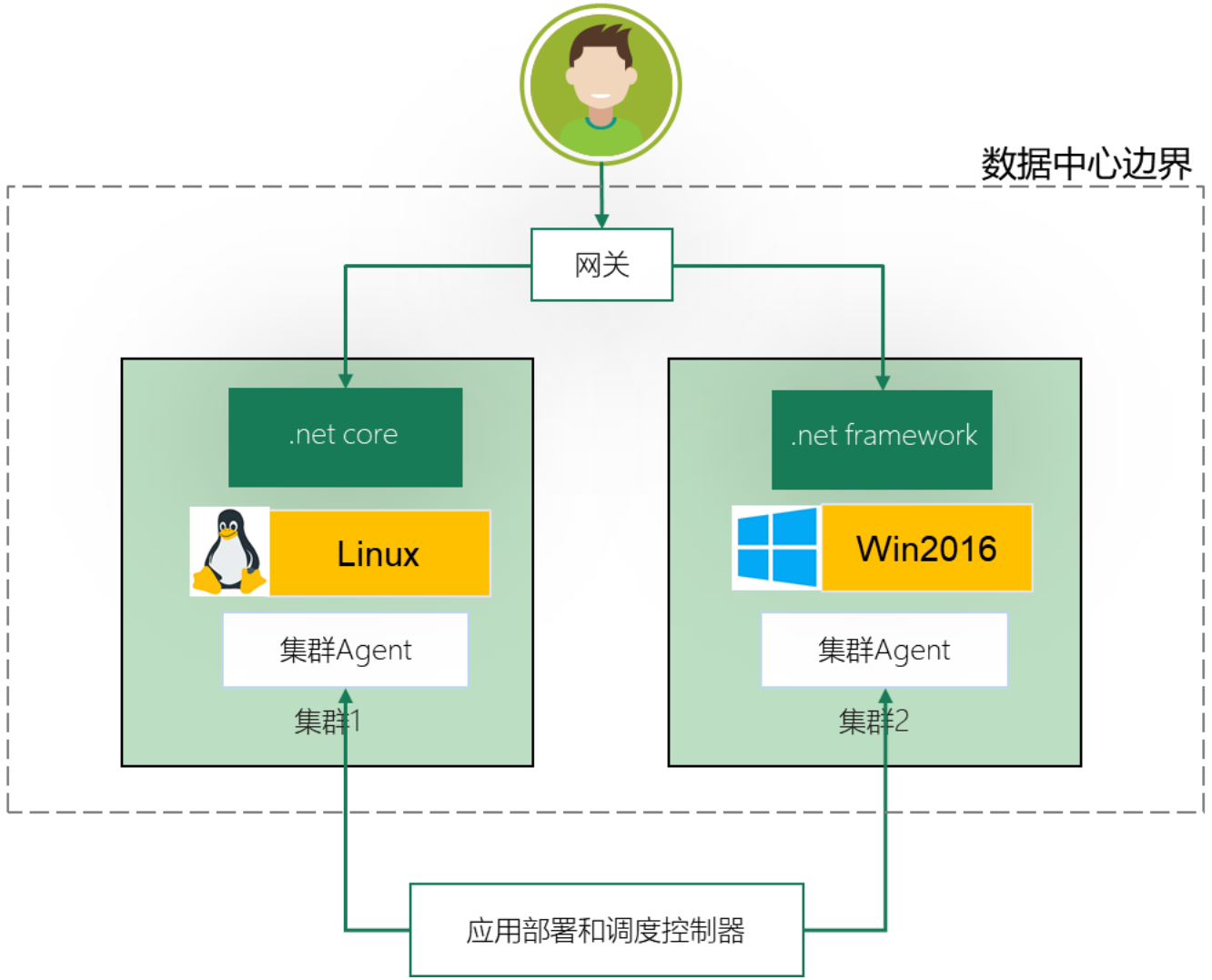
X项目采用的多容器集群技术架构



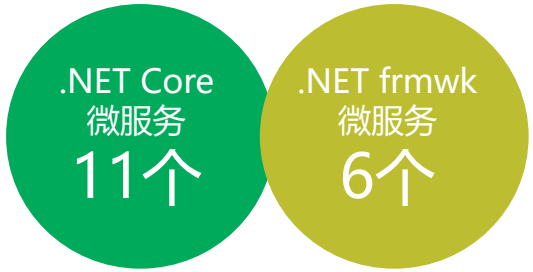
K8S之外的能力组件

- 业务接入网关
- 部署于集群的管理Agent
- 多集群应用部署和调度控制器

支持跨区域、跨云商、跨操作系统的容器管理和应用交付



X项目



X项目采用的DevOps和微服务支撑平台总体架构

为创新加速的**开发服务**

1 应用架构设计器

2 前端设计器

3 Web IDE

4 自动化测试

代码管理

5 云应用商店

以业务为中心的**PaaS平台**

6

基于Docker容器技术

采用K8S管理容器

多集群和多云应用交付

Windows, Linux容器统一管理

服务治理和安全保护

全面支持DevOps方法论

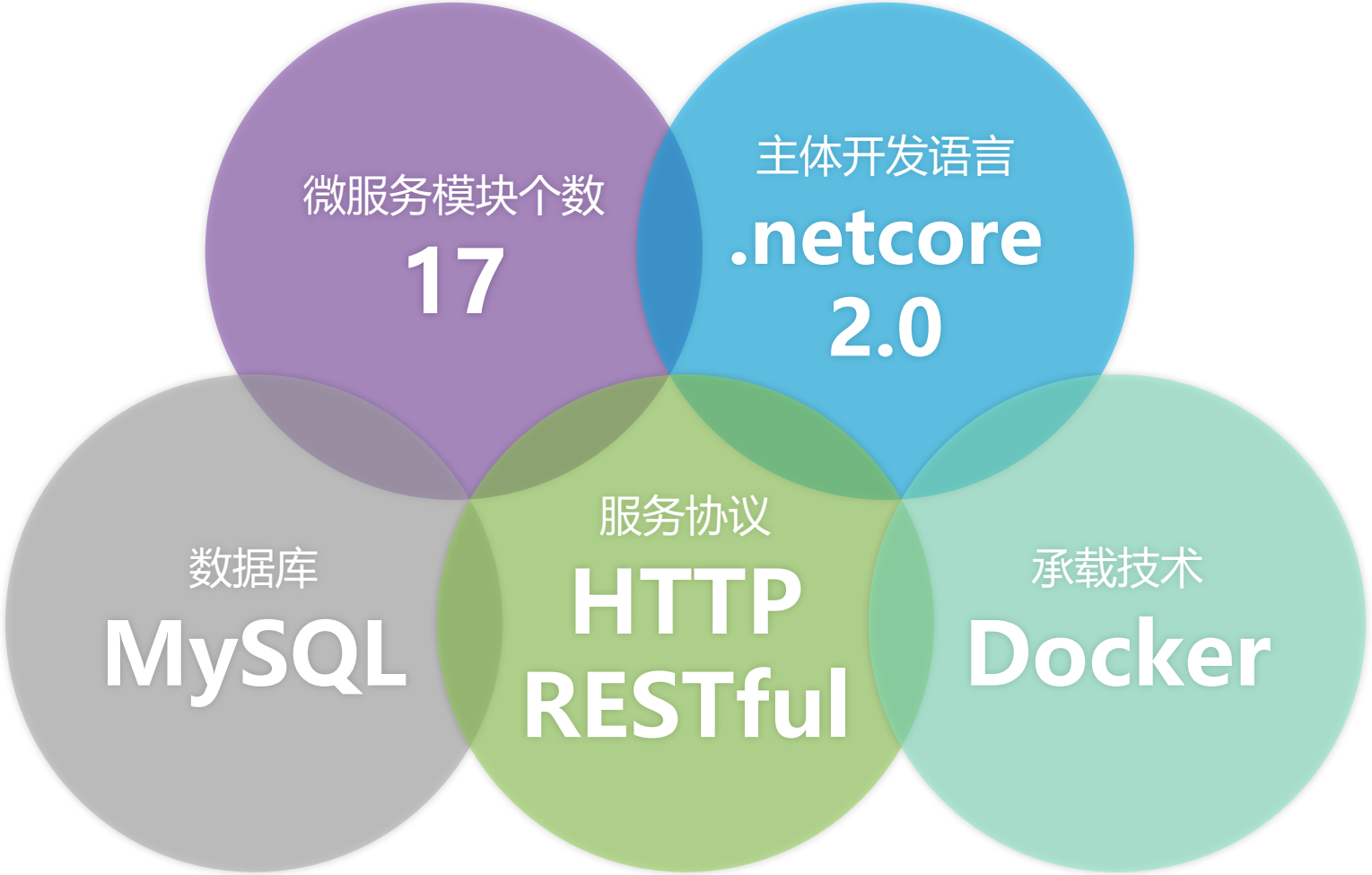
私有云

华为云服务

amazon web services

演示：混合管理Linux, Windows容器

X项目成果



X项目验证的收益

谢谢大家！

技术讨论微信：cloudtogo