
GOPS2018
Shenzhen
GOPS
全球运维大会 2018
2018.4.13-4.14
中国·广东·深圳·南山区 圣淘沙大酒店（翡翠店）

关于我

- 顾宇
- ThoughtWorks 高级咨询师
- 四年国内外微服务和 DevOps 咨询实施经验

当你听到微服务的感觉如何?

开心

疑虑

痛苦

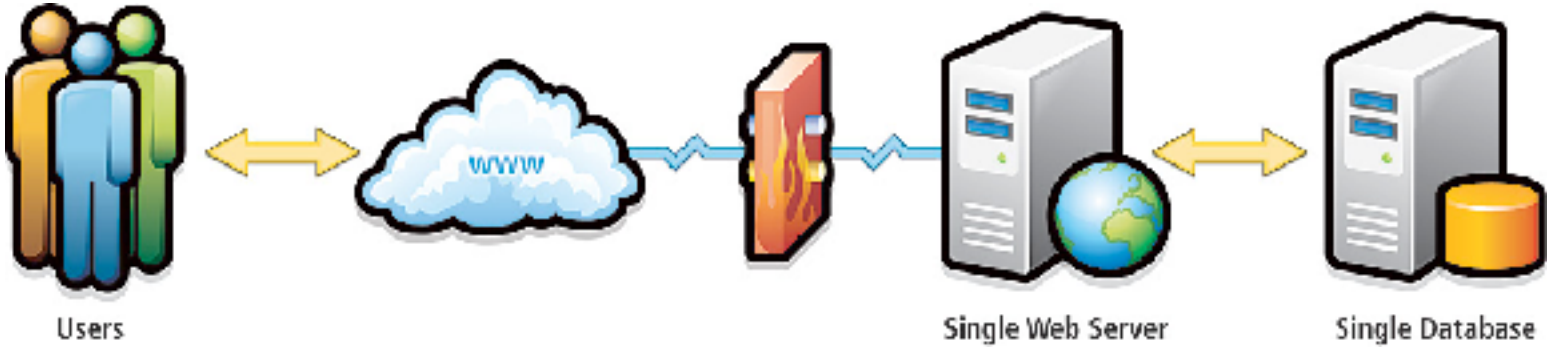
GOPS2018
Shenzhen

微服务落地反思以及高效落地

顾宇 ThoughtWorks 高级咨询师

目录

微服务项目一：Java 遗留系统解耦



面对的问题

- 如何合理的拆分应用
- 如何设计好 API
- 如何做到持续部署

微服务就是持续部署 Restful API

微服务项目二：某电信运营商

面对的问题

- 如何快速稳定的部署应用
- 如何提高团队的生产率，特别是自动化水平
- 如何解除流程瓶颈，跨部门协作

DevOps 做到极致就得到了微服务

微服务项目三：某大型 IT 集团云计算产品

面临的问题

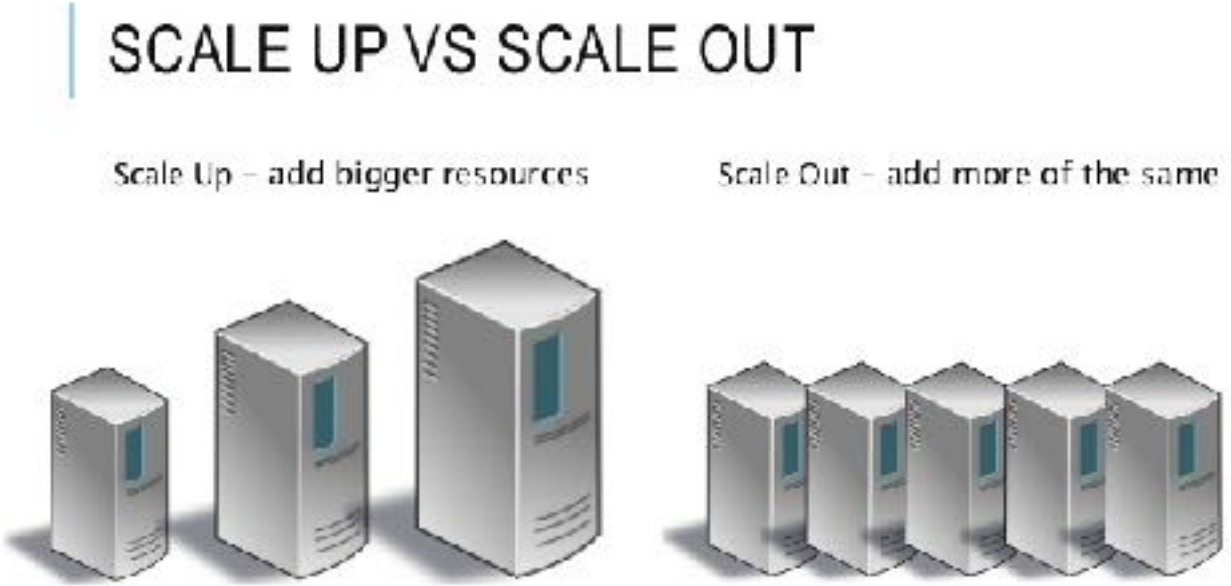
- 如何缩短发布时间
- 如何独立发布
- 微服务迟迟落不了地

从组织结构的调整做微服务是有效的

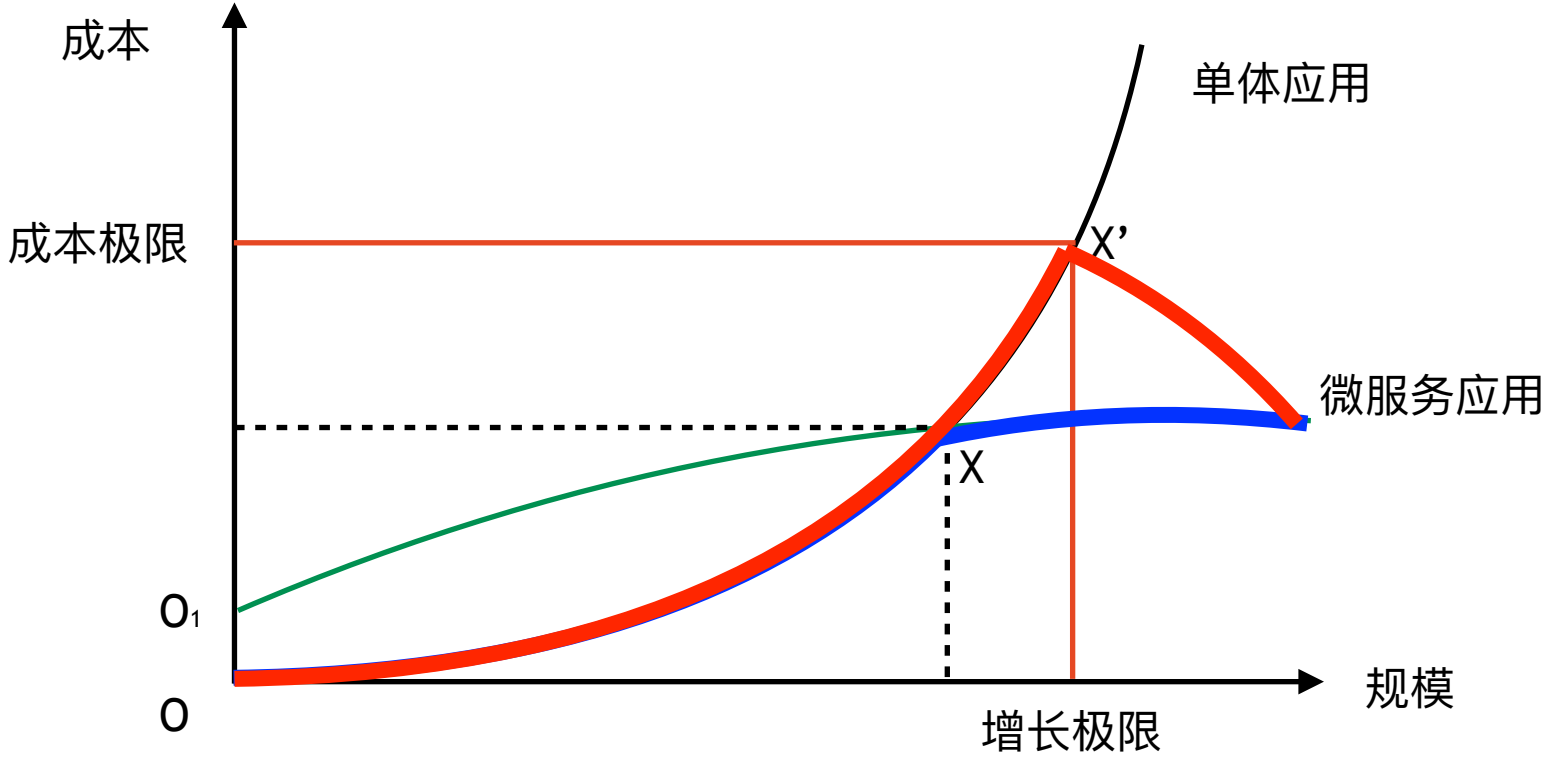
案例的共同点

- 应用和团队规模增长
- 实践持续交付/DevOps
- 在空间和时间上无冲突发布

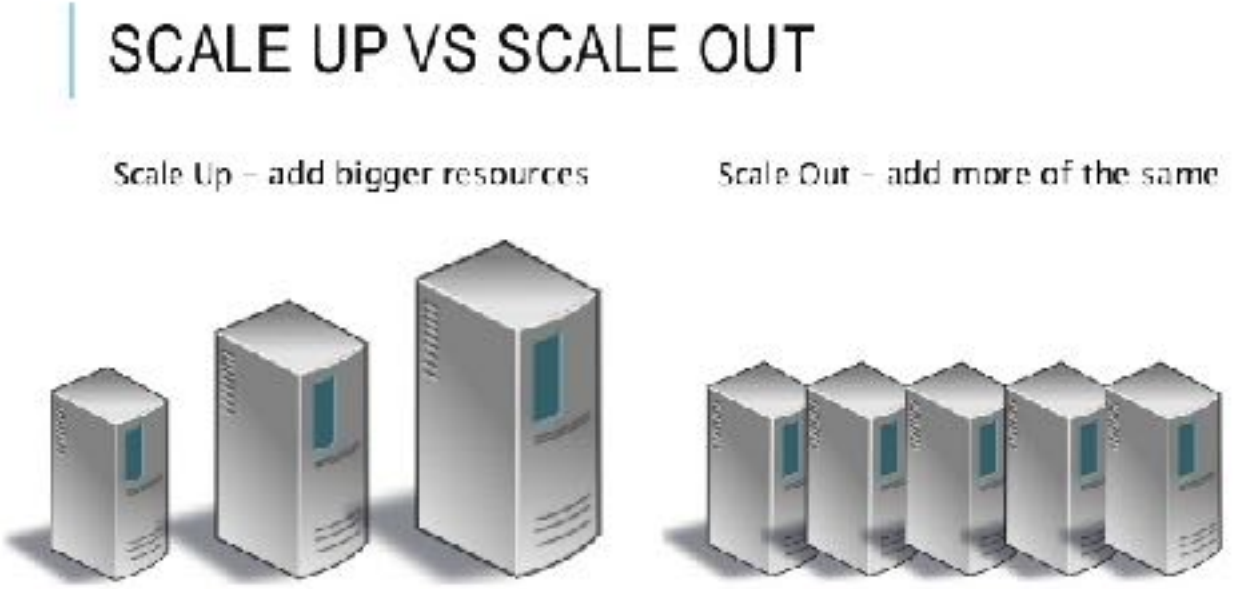
如何面对增长


SCALE UP VS SCALE OUT
Scale Up – add bigger resources
Scale Out – add more of the same

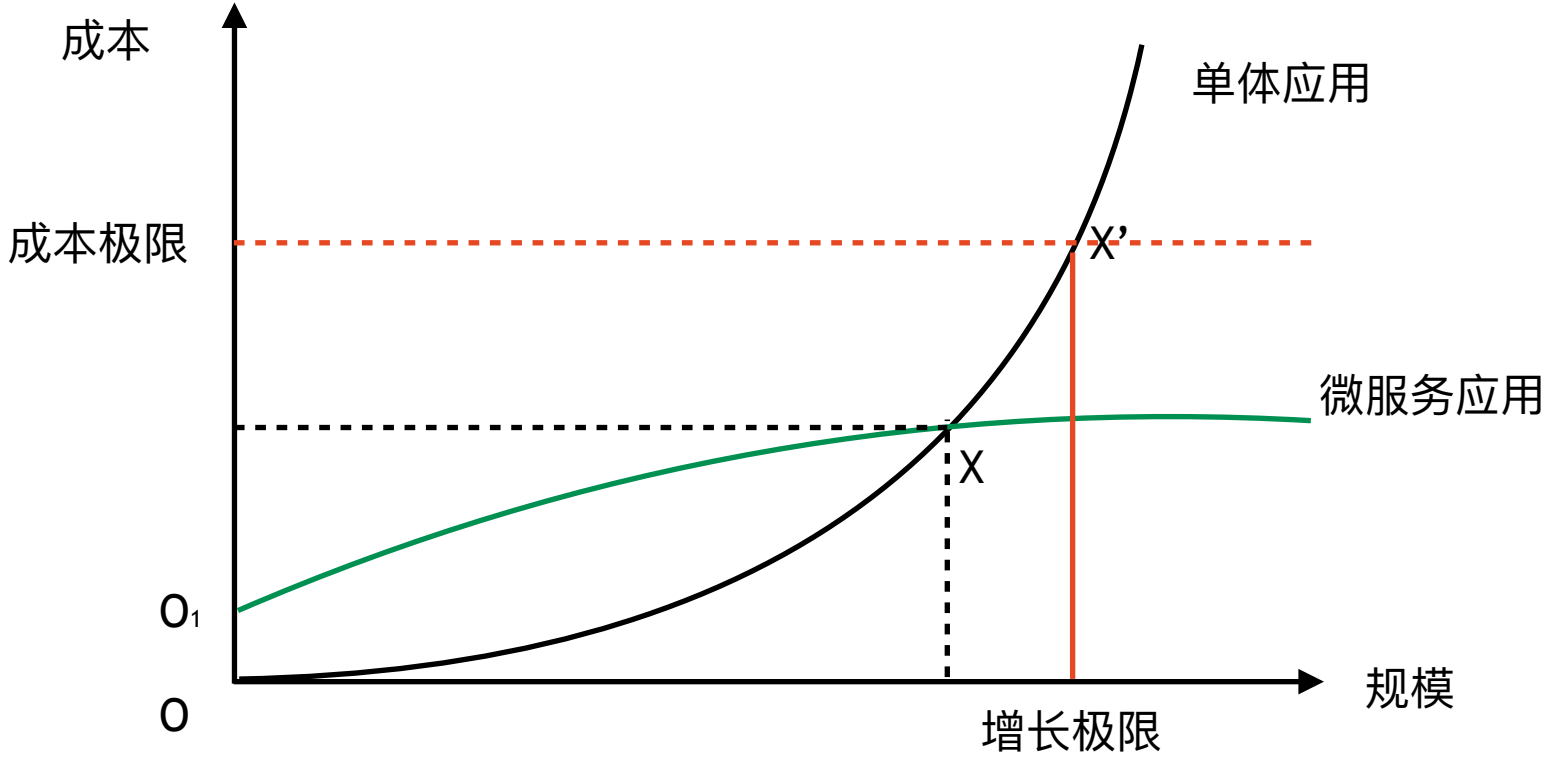
成本拐点 —— 增加单位代码行的净收益



复杂性在扩大你的管理成本



把微服务改造看做是对增长极限的投资



微服务本质上在解决什么问题?

- 处于团队和应用的高增长。
- 通过独立部署和独立发布提升整体交付效率。
- 在时间和空间上降低单一代码库的应用程序的冲突访问。

什么是微服务？ - Chris Richardson 定义

- 是一种架构风格。
- 将松散耦合的服务组成一个应用程序。
- 使之能在大型/复杂的应用上应用持续交付/部署。
- 能够使组织进化技术栈。

什么是微服务？ - Martin Fowler/ James Lewis 定义

- 应用程序作为一组可独立部署的服务套件。
- 没有精确定义，但有确定共同特征。
- 围绕组织，围绕业务
- 自动化部署
- 智能的端点（Endpoint）
- 去中心化控制的编程语言和数据。

什么是微服务？ - Sam Newman 定义

- 微服务就是一些协同工作的小而自治的服务。
- 专注：只做好一件事。
- 自治：可以独立运行的实体。

为什么用 Monolithic 和 Micro？

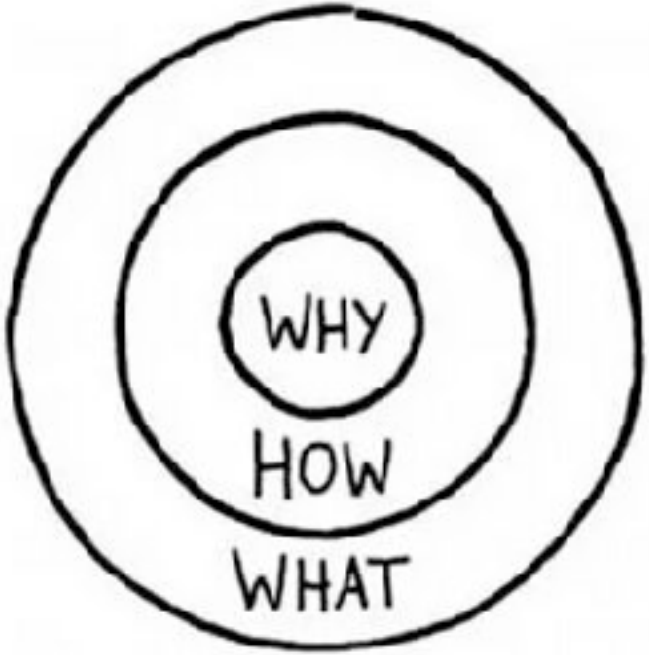
Linux

MINIX 3

小不是优点，简单才是

如果你做的微服务越来越复杂，一定是走错了路！

你为什么要用微服务?



目录

你有哪些微服务痛点?

- 不知道怎么拆?
- 架构落地困难?
- 管理越来越复杂?
- 技术栈不知道该怎么选?
- 团队不知道该怎么组织?
- 微服务应该怎么管理?

技术问题 vs 非技术问题

读《人件》的反思

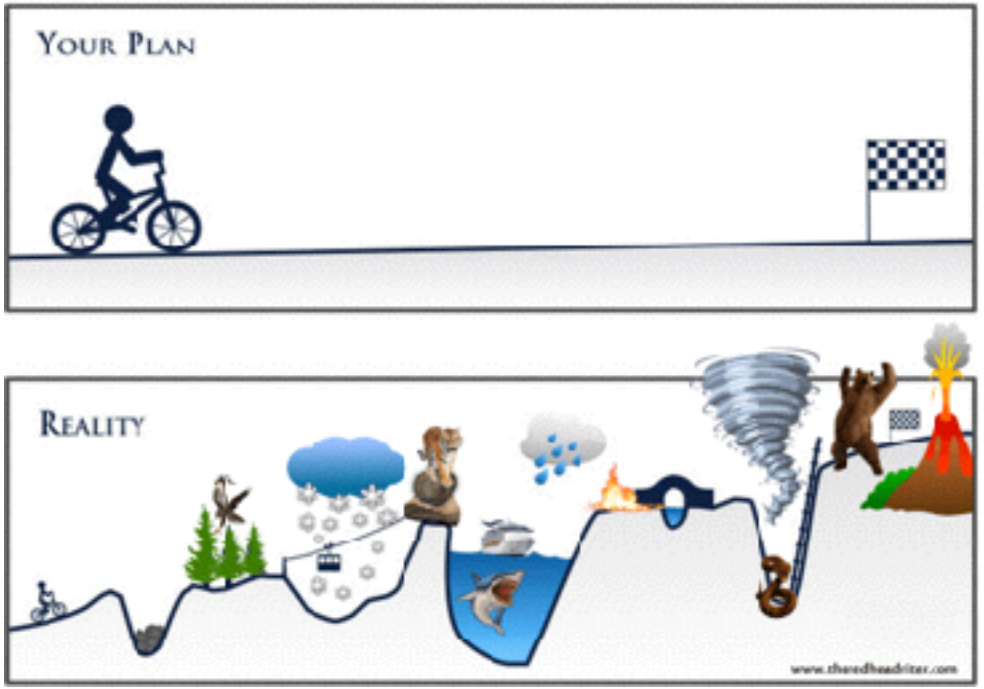
“也许 …… 软件系统的重要问题不在于技术，而在于社会性因素。”

“如果我们所面对的问题天生就属于社会学范畴，再好的技术可能也提供不了什么帮助。”

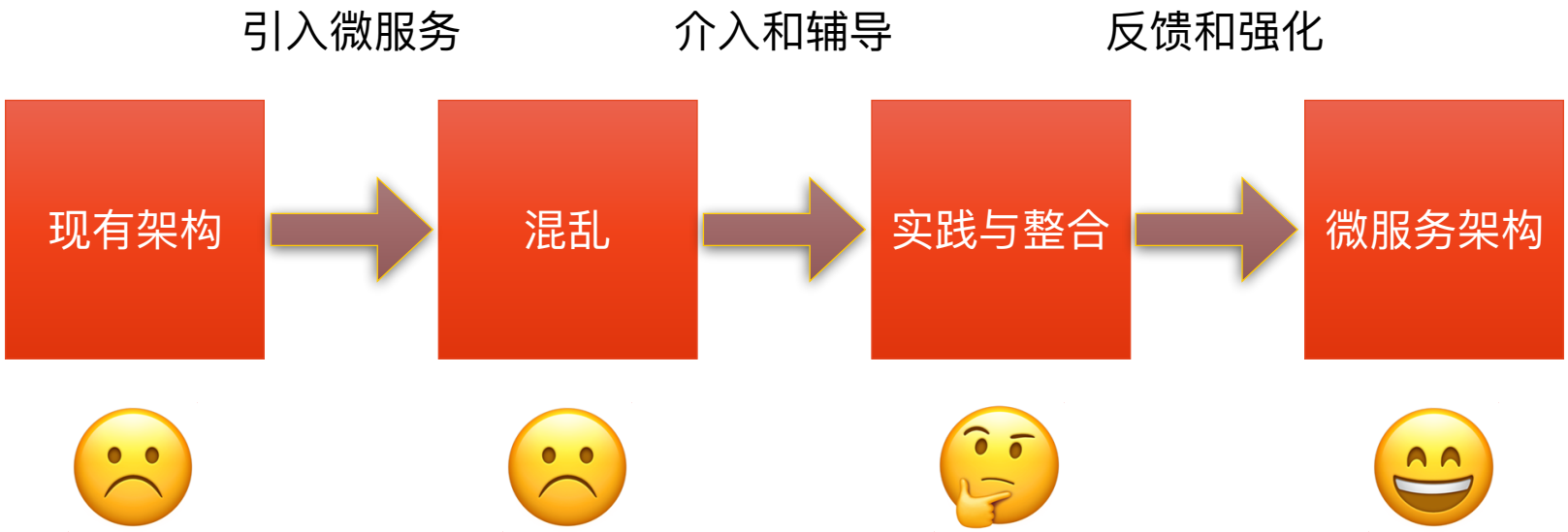
微服务是一个掩盖在技术问题之下的管理问题

没有意识到微服务是一个组织转型问题

现实并非一帆风顺



萨提亚改变模型



怎么才算高效?

- 从结果出发，找最短的距离。
- 最难的事情最优先解决。
- 低成本，低风险。
- 杜绝浪费。

目录

步骤1. 以终为始，组建团队

按照传统方式，只会重蹈覆辙

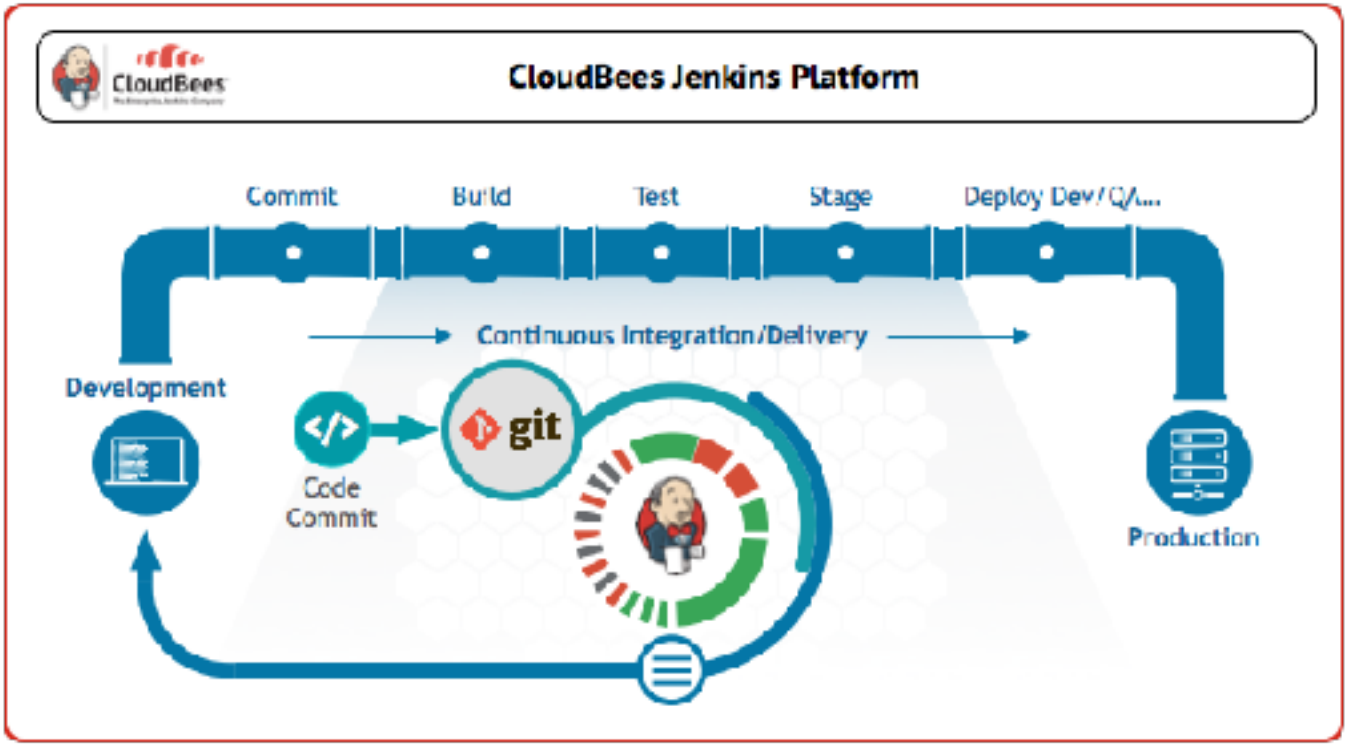
一个自治的全功能团队是什么样?

- 可以独立发布微服务的团队。
- 1 名微服务经理，解决流程依赖和干扰。
- 2-4名开发/测试，专注于微服务开发和测试。
- 1-2个运维，用来提供最快速的发布/部署。

独立的小团队!

- 避免依赖
- 提升技能
- 特事特办

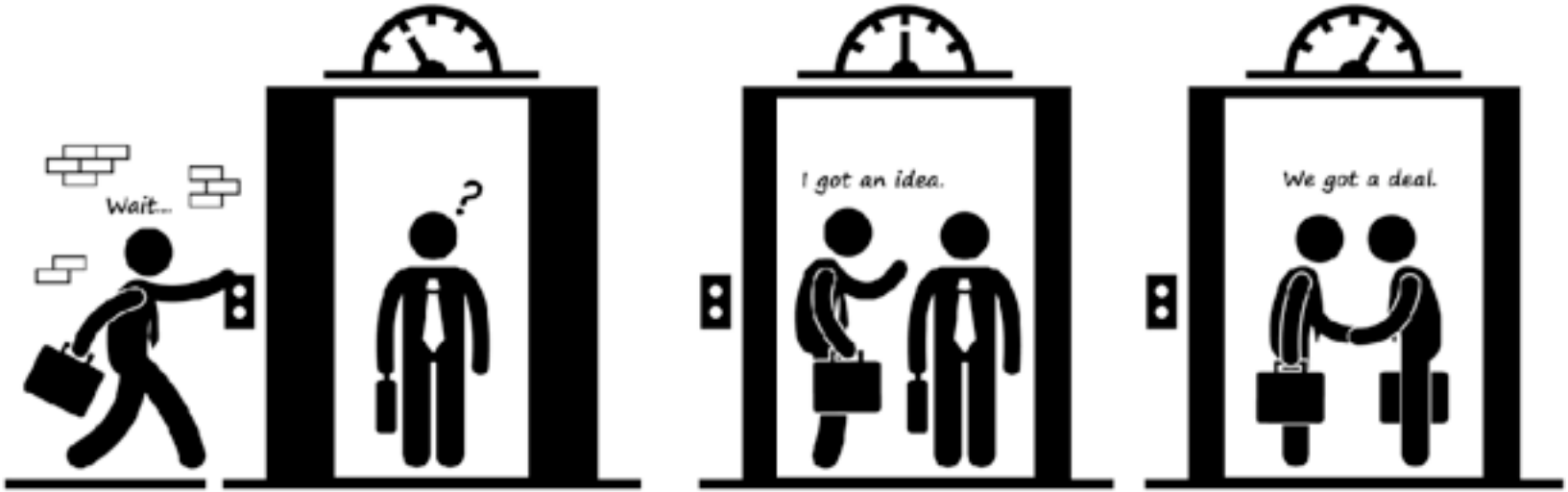
一个微服务，一个代码库，一条流水线。



步骤1：组建微服务团队

- 为微服务建立一个全新的代码库，而不要从原先的代码库上克隆或者复制，避免和原团队的开发依赖。
- 建设一个独立的持续交付流水线，最好是通过“流水线即代码技术”（例如 Jenkinsfile）来自动生成流水线。
- 一个微服务，一个代码库，一条流水线。

步骤 2：构建微服务电梯演讲



步骤2：构建微服务的“电梯演讲”

- （XX微服务）用来
- 在（出现痛点的场景）的情况下
- 解决了（解决现有的某个题）
- 从而（达到什么样的效果）
- 提升了（微服务的价值）

- （订单查询微服务）用来
- 在（订单查询数量快速）的情况下
- 解决了（访问数量迅速升高导致整体应用性能下降的问题）
- 从而（分离了订单查询请求）
- 提升了（提升了其他功能的性能）

不要超过 15 秒

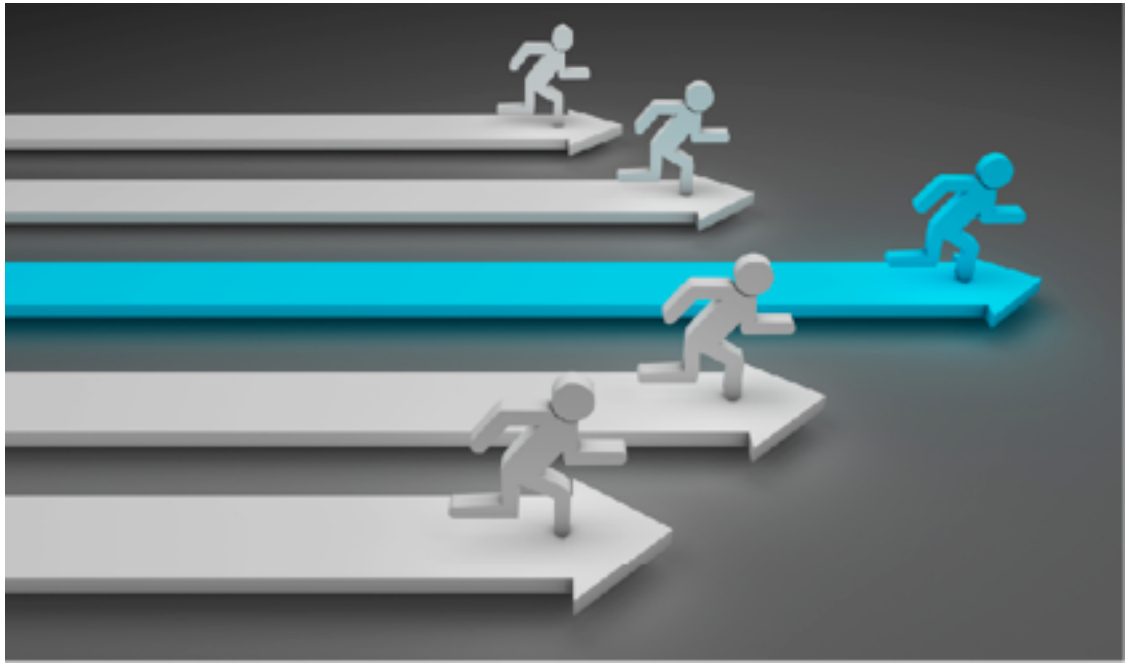
挂到墙上

形成共识

步骤2：构建微服务的“电梯演讲”

- 不要超过 15 秒
- 把微服务的电梯演讲打印出来挂到墙上，让团队成员铭记于心。这会强化组织对微服务的边界认识。
- 随着团队的反思和学习，电梯演讲有可能会变更，但一定要让团队形成共识好和一致的意见。
- 不要期望一次就能划分正确。划分是一个持续学习和研究的过程。

步骤 3：取得快速胜利



步骤3：取得快速胜利

- 以最小的代价发布第一个微服务
- 给团队提升士气
- 目标不宜太高
- ShowCase 驱动开发

用最小代价发布第一个微服务

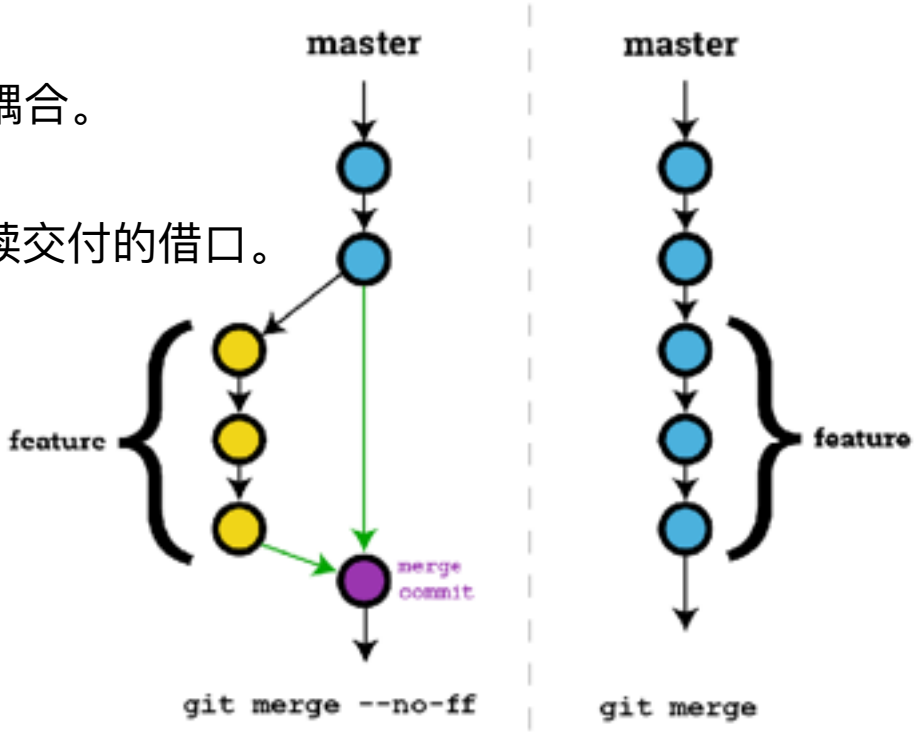
- 最小的代价：
  - 团队规模 （2~8人）
  - 时间（2~4周）
  - 选择代价较小的技术栈

特性开关（Feature Toggle ）

单主干开发

有分支，就造成了耦合。

给自己一个不做持续交付的借口。



步骤3：发布第一个微服务

- 采用动态特性开关（Feature Toggle），在发布后可以在生产环境动态的控制微服务的启用，降低失败风险。
- 如果采用了特性开关，一定要设立删除特性开关和对应旧代码的时间，一般不超过两个月。否则后面大量的特性开关会带来管理成本的提升和代码的凌乱。
- 由于团队比较小，功能比较单一，不建议采用分支来构建微服务，而应该采用单主干方式开发。

步骤4：取得快速胜利

- 要防止团队画大饼，完成好每日和每周的工作目标即可。微服务开发本身就没有很长周期。
- 除了让第一个微服务尽快发布到生产环境，其它的不要想太多。给团队继续前进最大的动力就是新服务快速投入生产。
- 完成了的发布，然后要考虑如何对发布流程进行改进。而不是上线业务。

步骤4：代码未动，DevOps 先行

工程师很容易陷到代码细节里

“不好弄”

“不能搞”

步骤4：代码未动，DevOps 先行

- 如果你的组织是 Dev 和 Ops 分离的组织，先咨询一下 Ops 工程师的意见。最好是能够给微服务团队里面配备一名 Ops 工程师。
- 如果不具备实施 DevOps 的条件，微服务架构就要从运维侧，而不是开发侧开始进行。

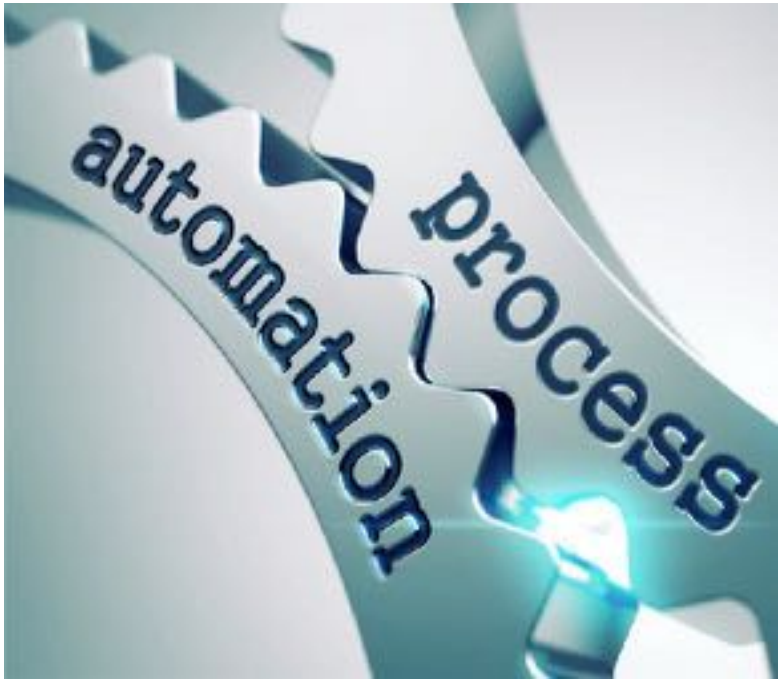
做 DevOps 就是做 CLAMS

GOPS2018
Shenzhen

- Culture
- Automation
- Lean
- Measurement
- Sharing

一定要有度量（Measure）

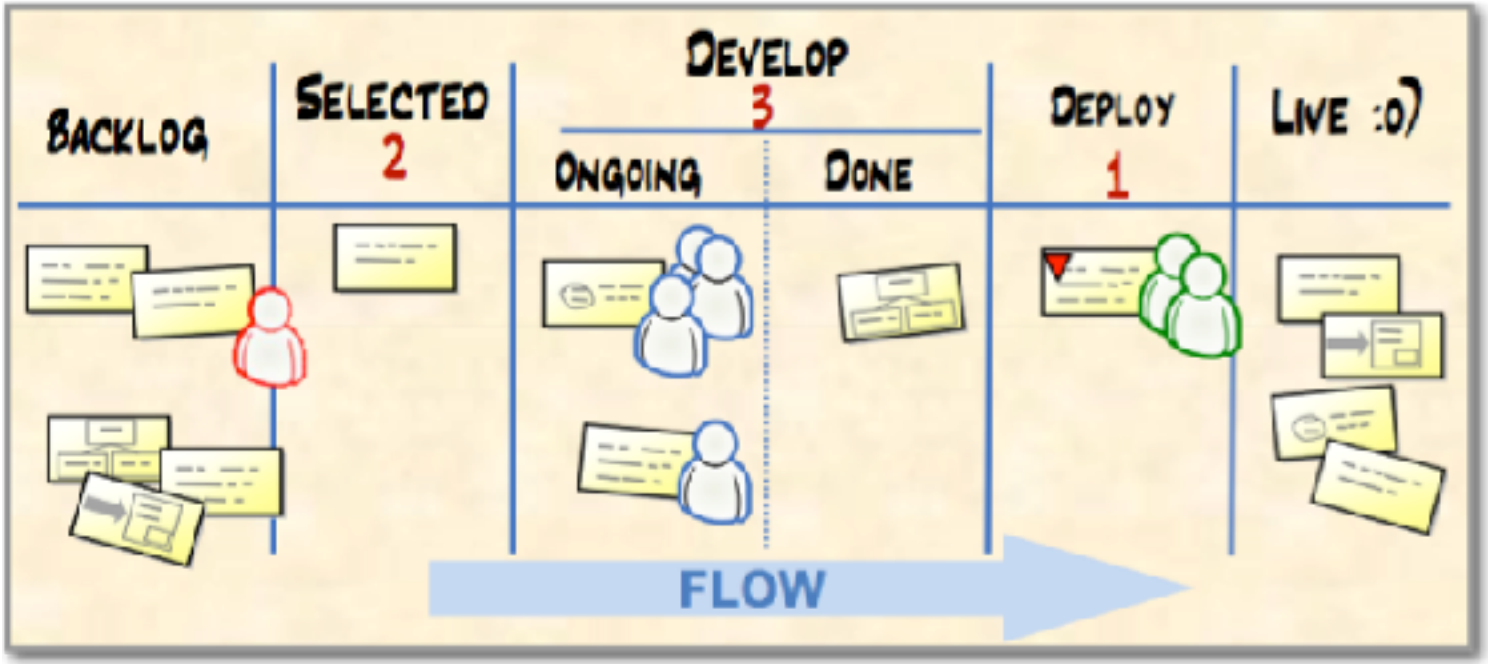
自动化-除了代码提交和发布，一切自动化



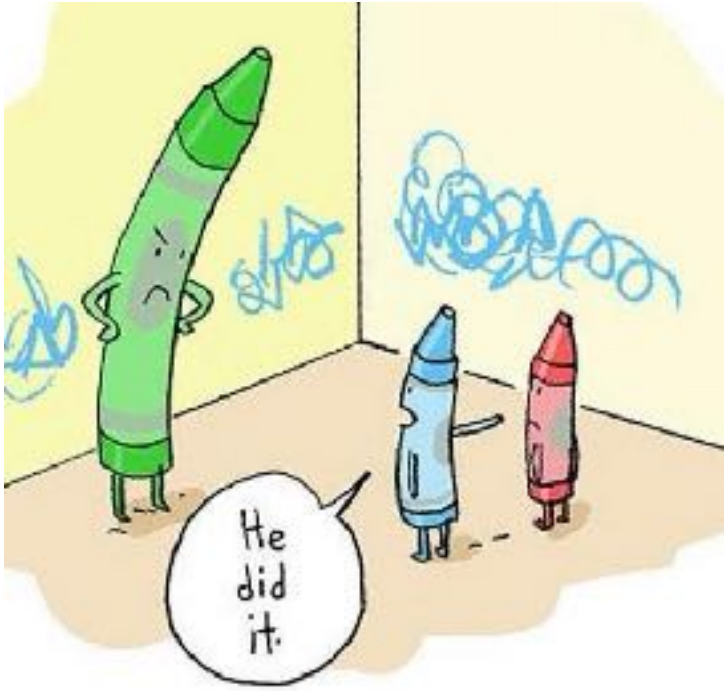
关键的自动化

- 自动化测试（功能/非功能）
- 自动化基础设施管理
- 自动化部署（不是发布）
- 自动化监控告警

精益 - 通过可视化来发现问题



构建分享和分担的理念



打造完整的 DevOps 文化

DevOps 核心实践

- 持续交付（蓝绿部署，灰度发布，金丝雀发布）
- 基础设施即代码（资源配置，资源编排）
- 基础设施对微服务是透明的

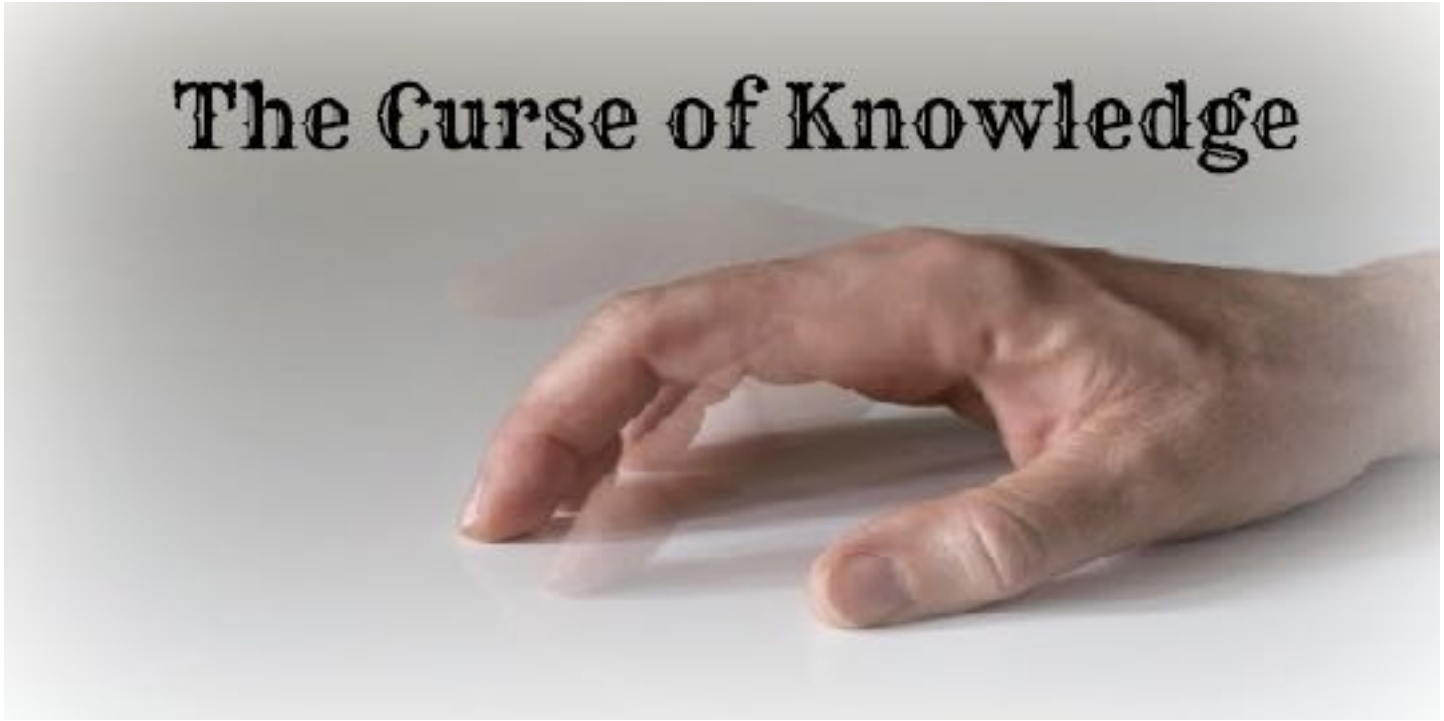
步骤6：自动化——管理提示

GOPS2018
Shenzhen

- 鼓励团队成员自发的进行自动化的改进，这会给未来微服务批量开发带来很多裨益。
- 不要一开始就追求全面的自动化，自动化需要花费一定时间。根据团队的进度视情况适度进行。
- 建立一个确定的系统会导致系统本身自愈能力的丧失。
- 风险管理不是要消除风险，是要当风险发生时有应对措施。

5. 持续改进，复制成功经验

知识的诅咒



步骤5：复制成功经验

- 需要总结出来的关键产出：
  - 微服务开发到发布的端到端流程规范。
  - 微服务开发的技术质量规范。
  - 团队合作中的坚持的最佳实践。
  - 常见技术问题总结。

不断改进

团队交叉轮换

步骤5：复制成功经验

GOPS2018
Shenzhen

- 刚开始的时候可以每周进行一个回顾会议，团队需要快速的反馈和调整
- 不要急于扩张团队，要在成功经验稳定并形成模式之后再快速扩充。
- 避免微服务良好的开发氛围被稀释，刚开始的时候扩充团队可以慢一点。新老成员的配比不要超过1:1。
- 虽然微服务平台趋于稳定，但在微服务没有上规模之前，不要让团队里缺少 Ops 成员。

高效落地微服务的 5 个步骤

1. 以终为始，构建微服务团队
2. 构建微服务的电梯演讲
3. 取得快速胜利，发布第一个微服务
4. 代码未动，DevOps 先行。
5. 复制成功经验

微服务，一本书就够了

Thanks

高效运维社区

开放运维联盟

荣誉出品

GOPS2018
Shenzhen

想第一时间看到高效运维社区的新动态吗?