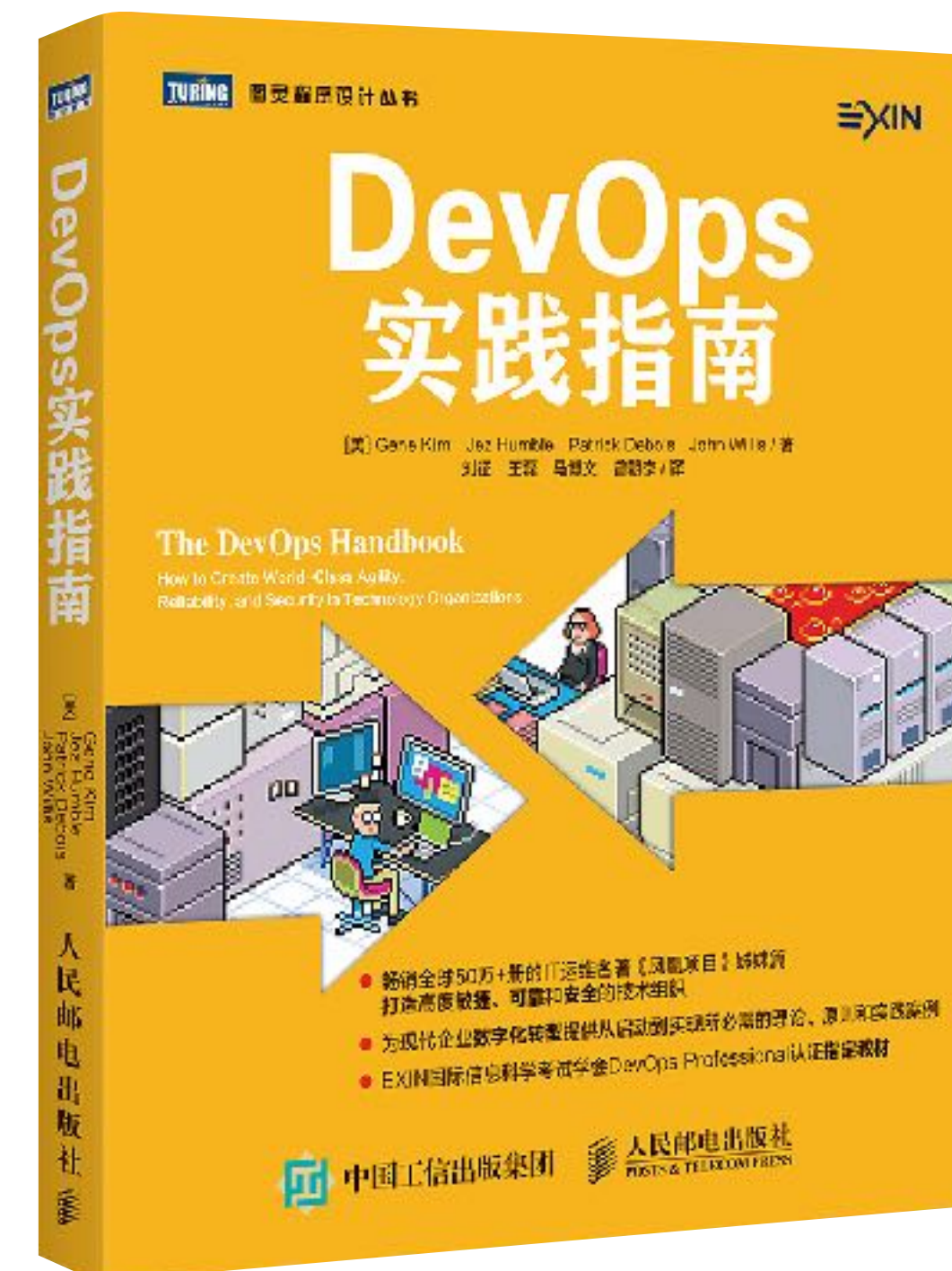
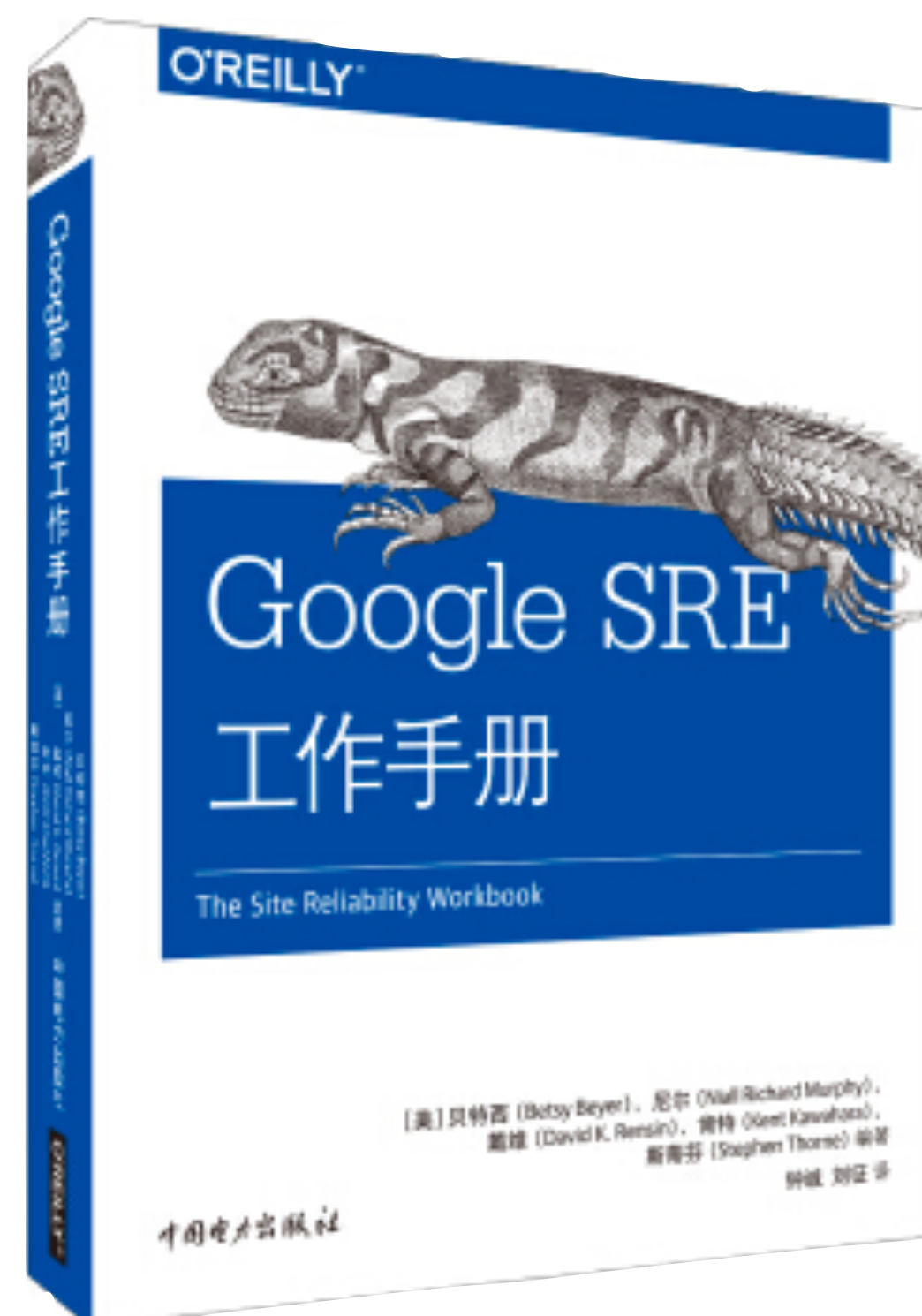




DevOps 能力建设和组织转型

软件工程持续渐进的精进

刘征 - 2021 年 3 月

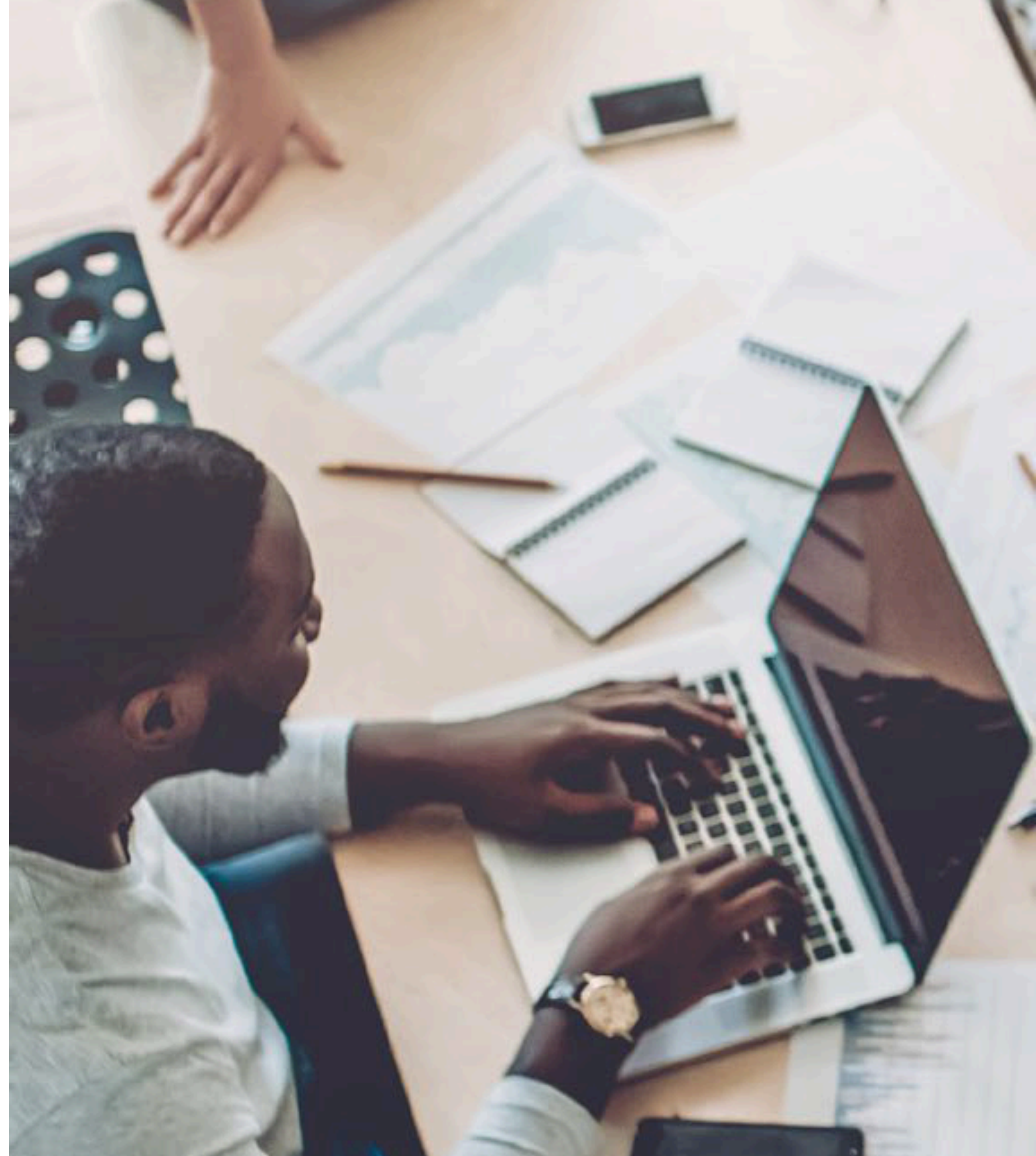


blog: <http://maritnliu.cn>



何谓DevOps

系统级产出



DevOps 是一项组织和文化运动，旨在加快软件交付速度，提高服务可靠性，并在软件的利益干系方之间建立共享的目标。

软件交付&运维效能

小调查（4 个问题）

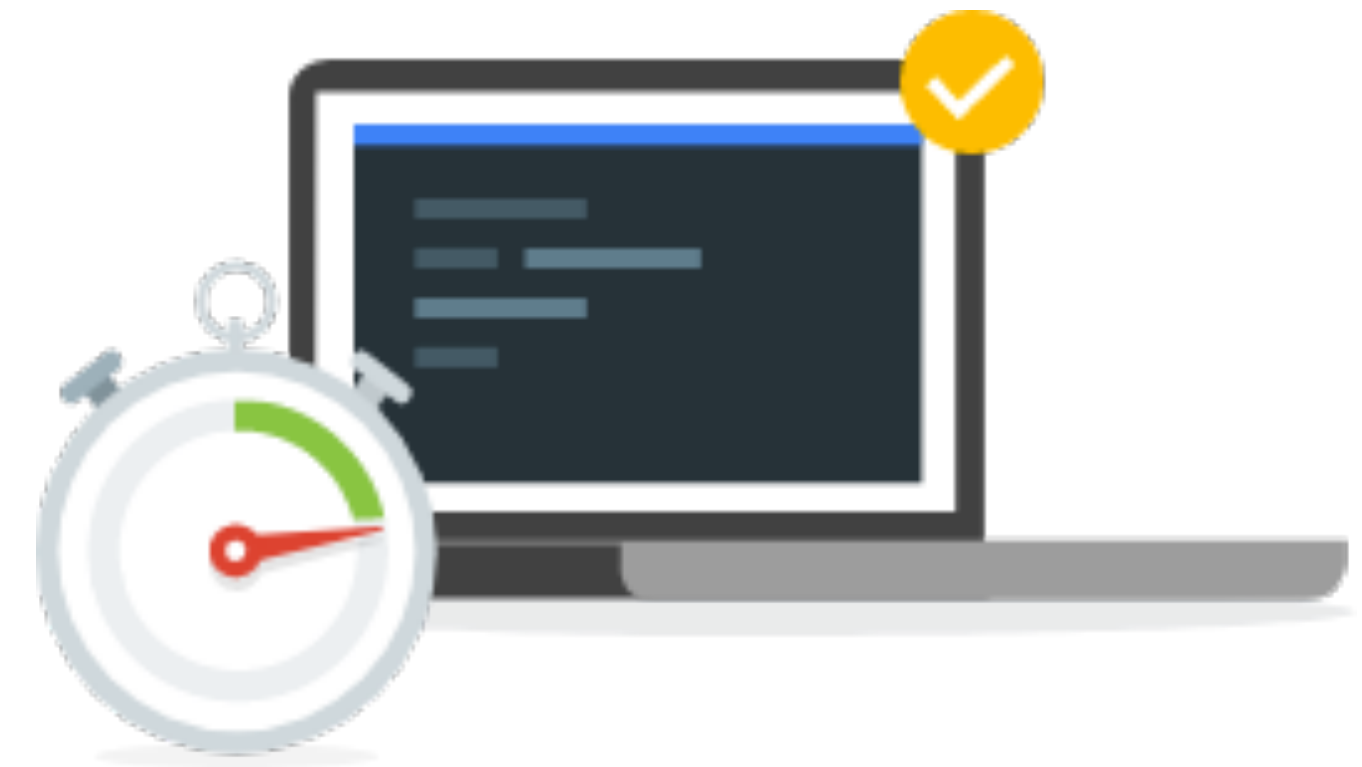
关于你所在的团队/部门



Key 1: lead time

前置时间

- 对于你工作的主要应用或服务，你的变更准备时间是多少（也就是说，从代码提交，到代码在生产环境中成功运行需要多长时间）？
 - 6 个月以上
 - 1 个月到 6 个月
 - 1 周到 1 个月
 - 1 天以内
 - 1 小时以内



Key2: Deploy frequency

部署频率

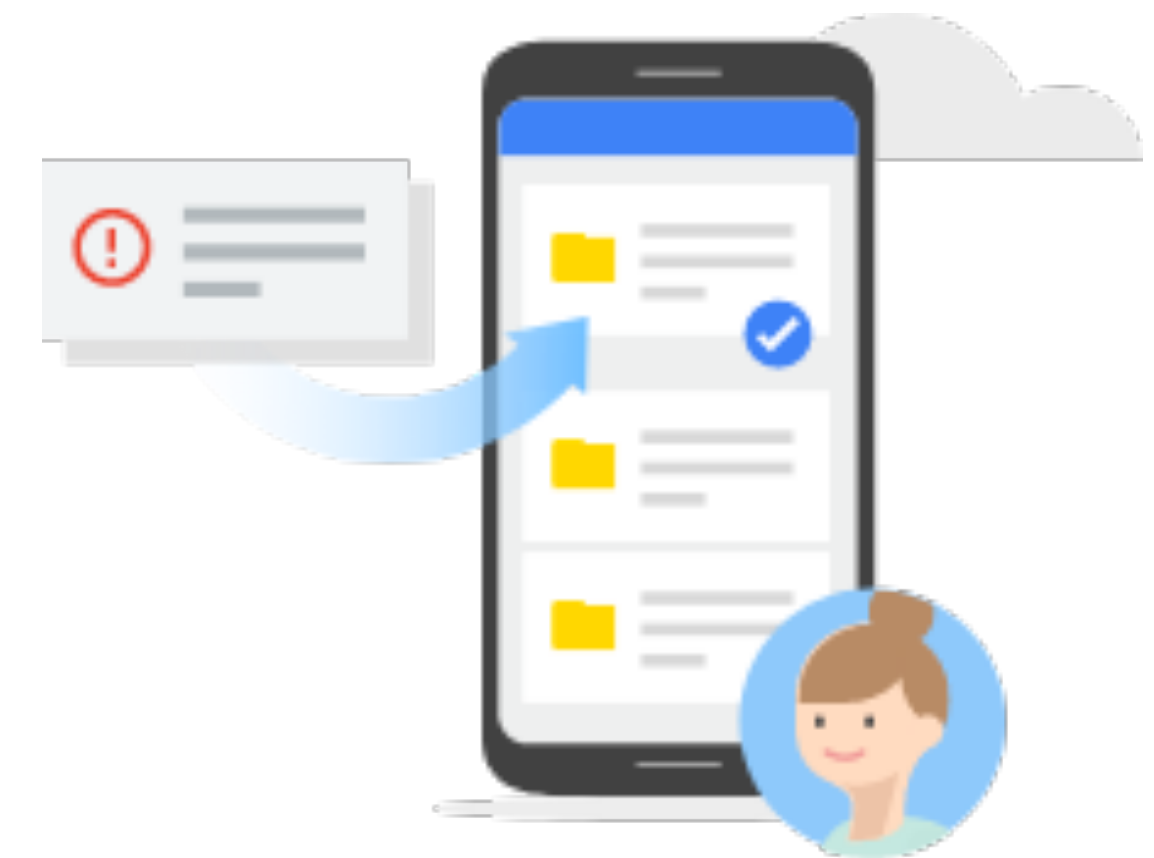
- 对于你工作的主要应用或服务，你的组织将代码部署到生产中或发布给最终用户的频率是怎样的？
 - 小于 6个月1 次
 - 在每 6 个月一次，到 1 个月一次之间
 - 在 1 周一次，到1 个月一次之间
 - 在 1 天一次，到1 周一次之间
 - 在 1 小时一次，到1 天一次之间
 - 按需部署（每天部署多次）



Key3: time to restore

恢复时间

- 对于你工作的主要应用或服务，当发生影响用户的服务事故或缺陷时，一般需要多长时间才能恢复服务/修复缺陷（如计划外中断、服务受损、客户宕机）？
 - 6 个月以上
 - 1 到 6 个月
 - 1 周到 1 个月
 - 1 天到 1 周
 - 小于 1 天
 - 小于 1 小时



Key4: change fail percentage

变更失败率

- 对于你工作的主要应用或服务，在对生产或向用户发布的变更中，由于变更而导致服务降级（如服务受损或中断），并随后需要修复（如需要热补丁、回滚、向前修复、补丁）的比例是多少？
 - 0-15%
 - 16-30%
 - 31-45%
 - 46-60%
 - 61-75%
 - 76-100%



Aspect of Software Delivery Performance*	Elite	High	Medium	Low
Deployment frequency For the primary application or service you work on, how often does your organization deploy code to production or release it to end users?	On-demand (multiple deploys per day)	Between once per day and once per week	Between once per week and once per month	Between once per month and once every six months
Lead time for changes For the primary application or service you work on, what is your lead time for changes (i.e., how long does it take to go from code committed to code successfully running in production)?	Less than one day	Between one day and one week	Between one week and one month	Between one month and six months
Time to restore service For the primary application or service you work on, how long does it generally take to restore service when a service incident or a defect that impacts users occurs (e.g., unplanned outage or service impairment)?	Less than one hour	Less than one day ^a	Less than one day ^a	Between one week and one month
Change failure rate For the primary application or service you work on, what percentage of changes to production or released to users result in degraded service (e.g., lead to service impairment or service outage) and subsequently require remediation (e.g., require a hotfix, rollback, fix forward, patch)?	0-15% ^{b,c}	0-15% ^{b,d}	0-15% ^{c,d}	46-60%

软件交付和运维效能

software delivery and operational (SDO) performance



软件开发



软件部署



软件运维

Lead Time

Change Fail

Availability

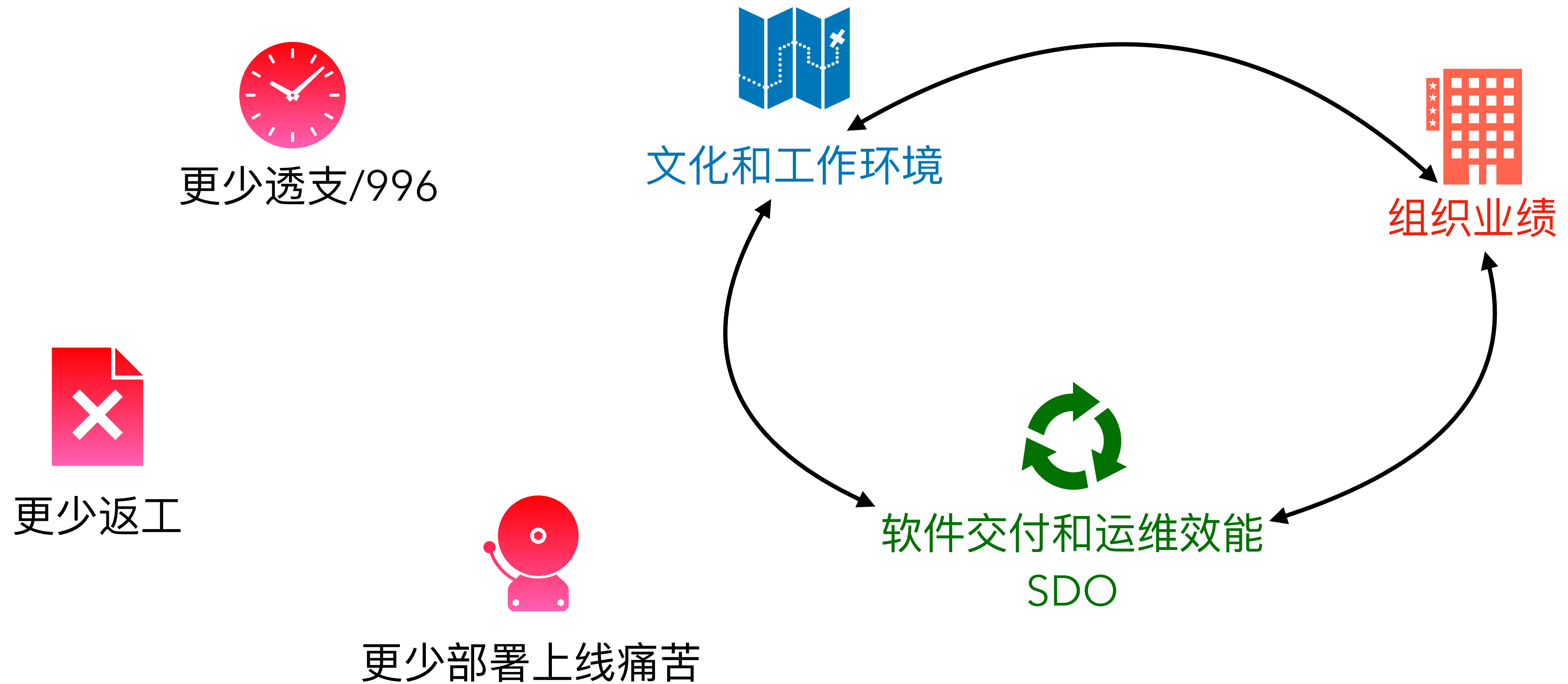
Deployment Frequency

Time to Restore

四大指标

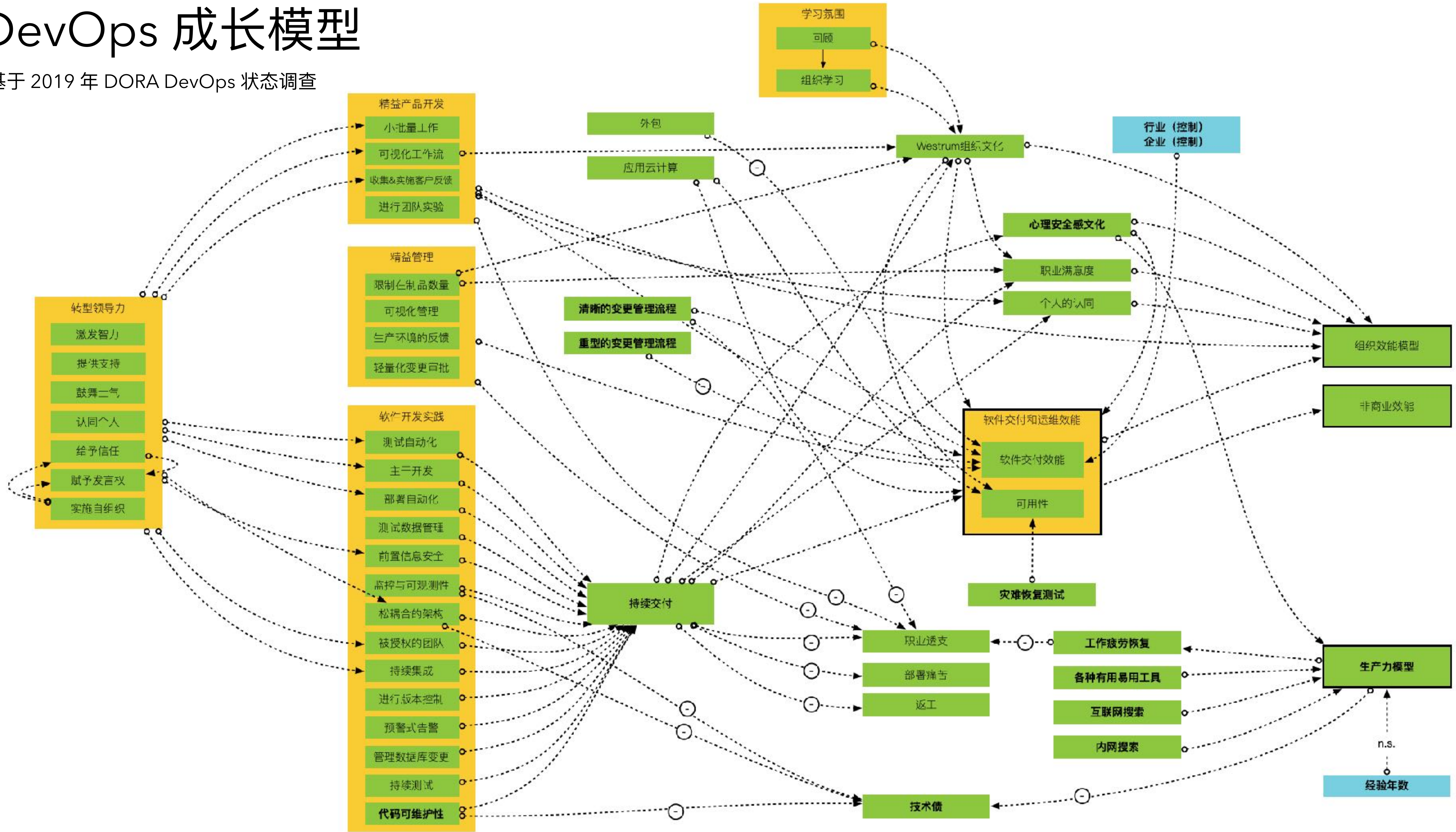
DevOps 的产出

互为因果和制约关系



DevOps 成长模型

基于 2019 年 DORA DevOps 状态调查



技术能力

- 版本控制
- 主干开发
- 持续集成
- 部署自动化
- 持续测试
- 持续交付
- 架构
- 赋权团队选择工具
- 测试数据管理
- 安全早期介入
- 数据库变更管理
- 云基础架构
- 代码可维护性

流程能力

- 团队实验
- 简化变更审核流程
- 客户反馈
- 价值流的可视化
- 小批量工作方式

度量能力

- 基于监控系统做出明智业务决策
- 监控和可观测性
- 主动式故障告警
- 限制在制品数量
- 可视化管理

文化能力

- 培养高度信任的生机文化
- 学习文化
- 工作满意度
- 变革领导力



如何进行转型？

原则和常见错误



组织级别 DevOps 转型

必须考虑到的问题

- 这种变革的方向是什么？
- 希望取得哪些系统级的成果？
- 组织能否更好地服务于客户，或获取更多客户，从而实现组织业绩？
- 能否为组织的业务模式和人员管理带来长期变革和可持续发展？



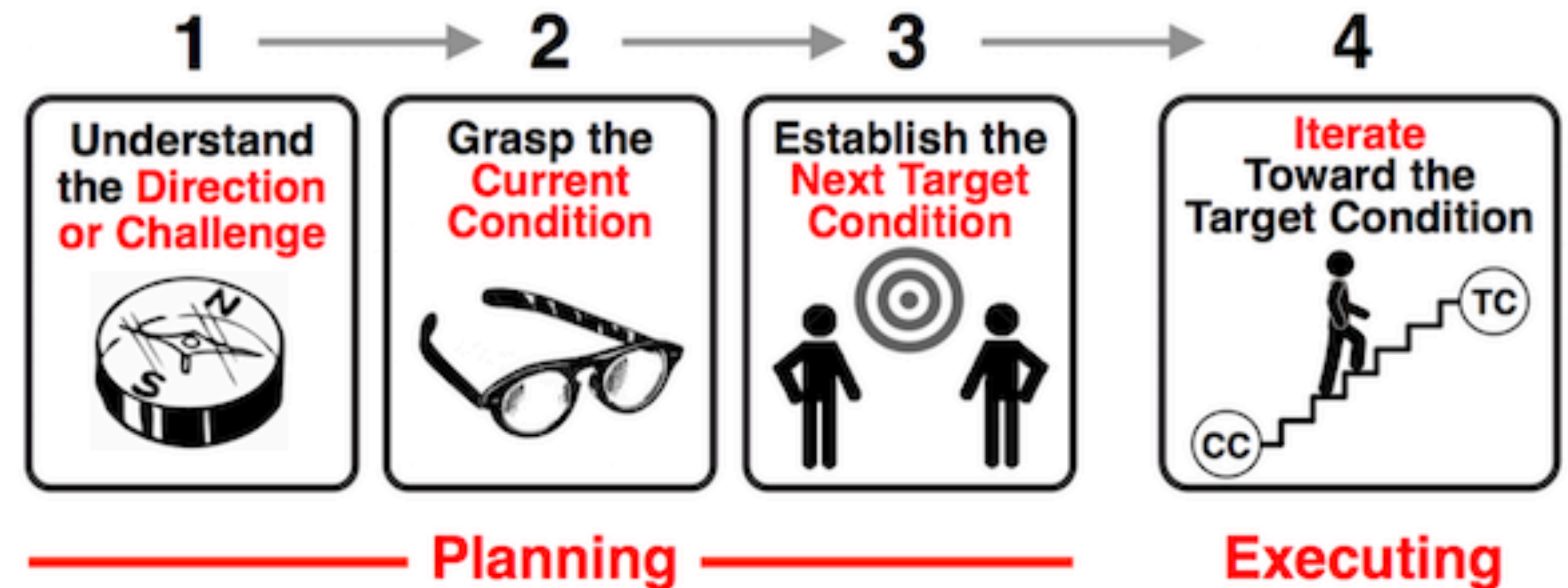
转型模式 1

设置目标并使能团队实验

- 团队必须设置自己的目标状况或 OKR
- 可以接受达不到目标；应努力达成约 80% 的目标
- 在努力实现目标的过程中学习和提升，迭代目标和度量指标
- 团队拥有必要的自主权、带宽、资源、管理领导的支持

THE FOUR STEPS OF THE IMPROVEMENT KATA MODEL

A systematic, scientific pattern of working



目标状况是什么？ 目前情况如何？ 有哪些障碍？ 下一步做什么？
正在解决什么障碍？ 期望的结果？



转型模式 2

推荐构建社区模式

- 应用在组织的基层和高层都创建社区结构的策略，可能会让社区结构在面临重组和产品变更时更具可持续性和弹性。
- 用社区传播实践+草根策略

☑️培训中心（有时也称为DOJO）

☑️卓越中心

☑️止步于概念证明

☑️用概念证明当模板

☑️用概念证明做种子

☑️实践社区

☑️大爆炸式

☑️自下而上或草根方式

☑️混搭型



有效的组织变更管理原则

循序渐进、上下一致

- 不断改进
- 领导者和团队会就可度量的成果达成一，致并加以传达，团队决定这些成果的实现方式
- 大规模的变化是以迭代方式逐步实现的

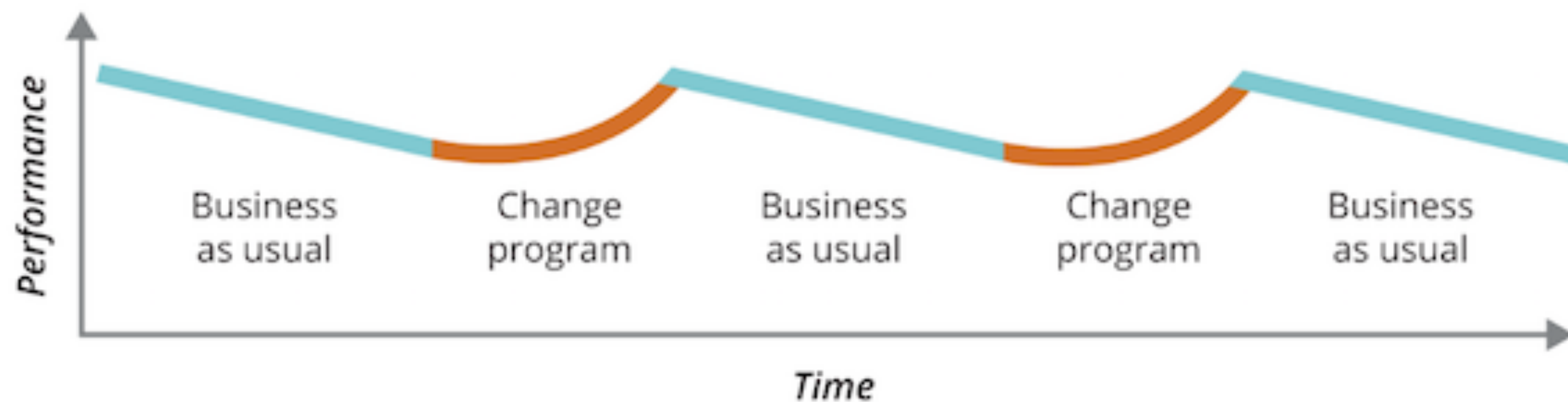




转型易犯的错误

三个大坑

1. 将 DevOps 转型视为一次性项目
2. 将转型视为自上向下的工作
3. 上下之间未能就预期成果达成一致，就向下层传达指令



小组讨论

什么转型方式适合我们？
为什么？



技术能力

- 版本控制
- 主干开发
- 持续集成
- 部署自动化
- 持续测试
- 持续交付
- 架构
- 赋权团队选择工具
- 测试数据管理
- 安全早期介入
- 数据库变更管理
- 云基础架构
- 代码可维护性

流程能力

- 团队实验
- 简化变更审核流程
- 客户反馈
- 价值流的可视化
- 小批量工作方式

度量能力

- 基于监控系统做出明智业务决策
- 监控和可观测性
- 主动式故障告警
- 限制在制品数量
- 可视化管理

文化能力

- 培养高度信任的生机文化
- 学习文化
- 工作满意度
- 变革领导力



工作的可视化管理

流程能力



如何实现可视化管理

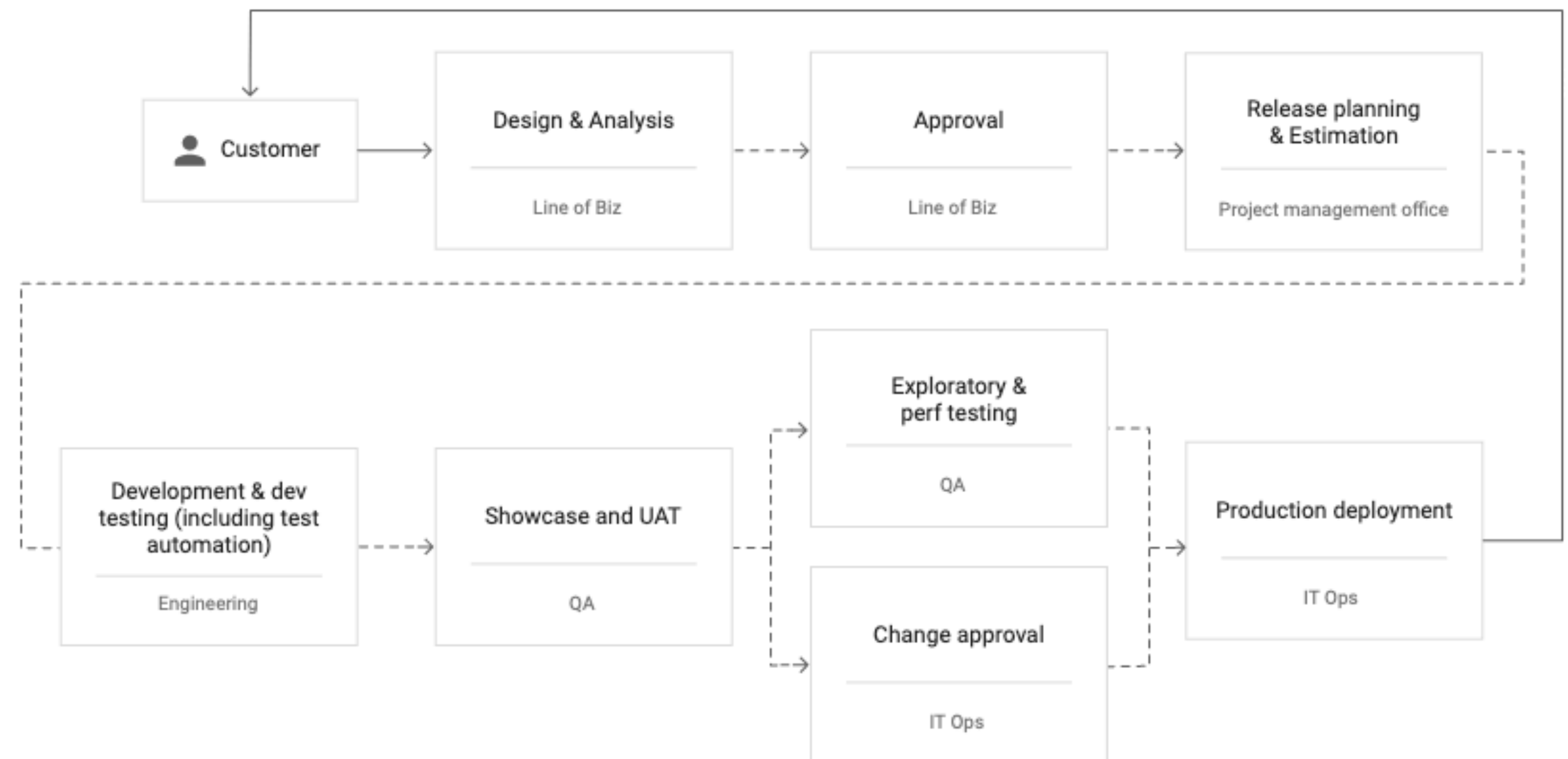
高手是这样的

- 团队了解工作如何从商业创意转变为客户之所需，包括产品或功能。
- 团队可以清晰了解此工作流程。
- 工作流程（包括其当前状态）显示在可视化显示设备或信息中心上。
- 整个价值流中产品开发工作流程的相关信息可随时使用。

使用价值流映射分析

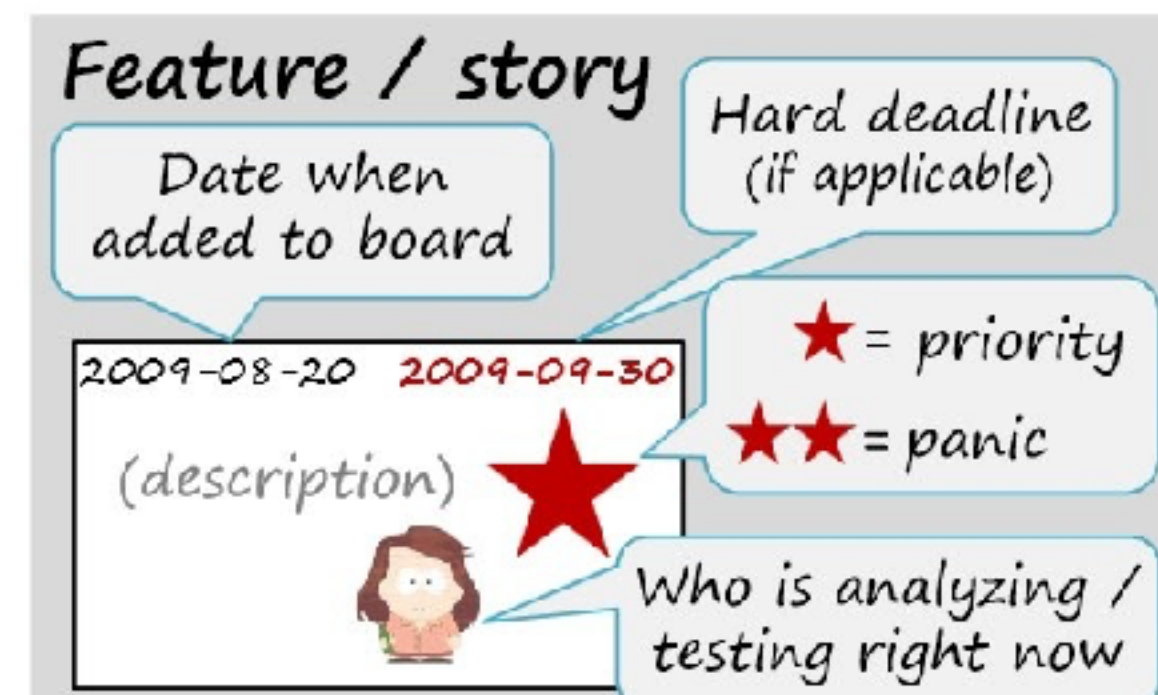
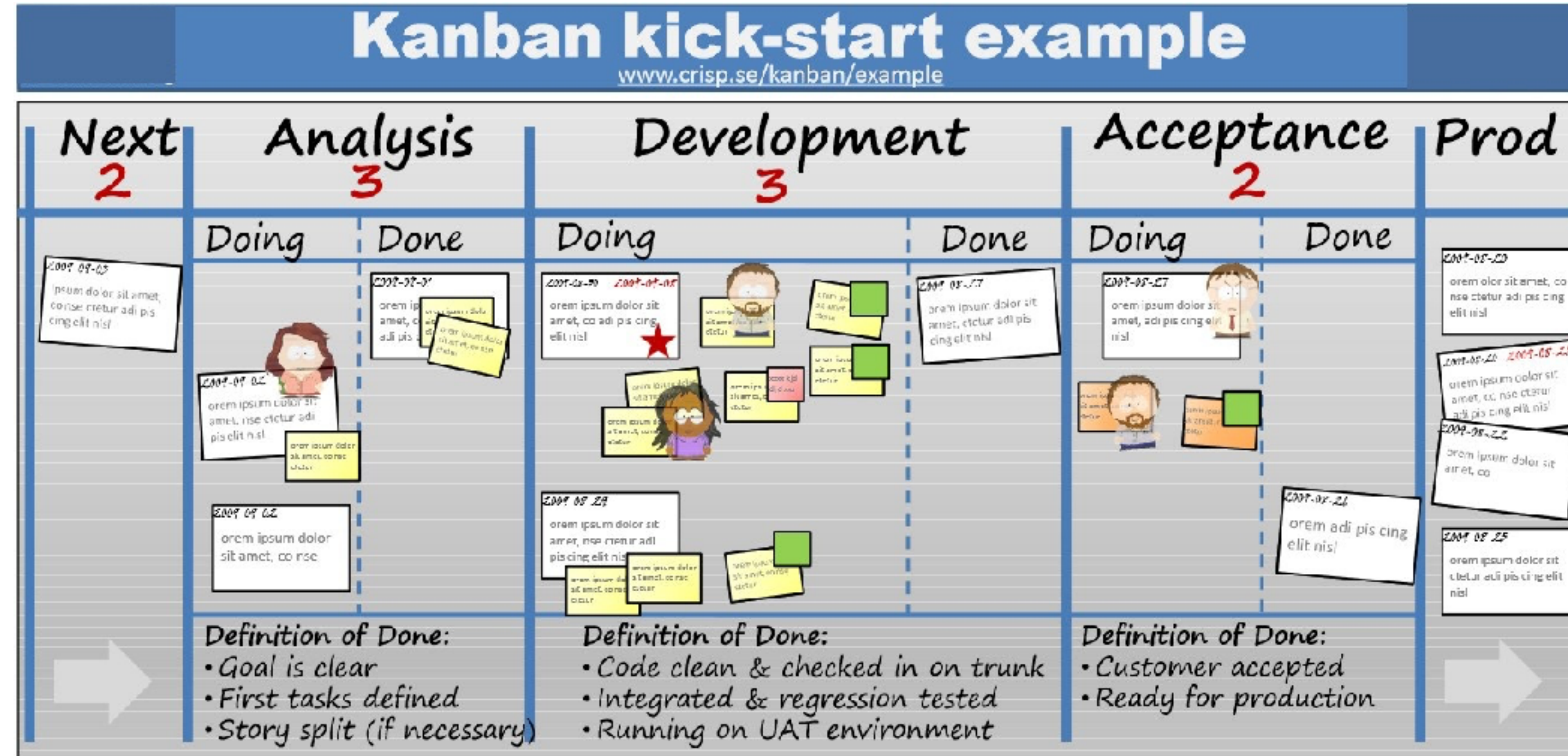
每个产品都做一年一次的 VSM 分析研讨会

- 参与方：业务线、设计、测试、质量保证、运维和支持
- 拆分和梳理：价值流拆分为 5 到 15 个流程模块
- 关键点：
 - 准备期
 - 处理器
 - C/A%



使用精益敏捷看板

开展和知道日常工作



工作可视化管理

常见困难和误区

- 高估组织知识的状态
- 无法映射整个价值流
- 专注于难以改进的领域
- 想做的改变，而得不到领导支持和授意
- 解决方法：
 - 收集任何可以证明焦油坑的证明，时间浪费，返工率。。。。
 - 用 VSM 探索那些鲜为人知的，隐藏的坑爹的数据
 - 用看板揭露大家无视的高并发导致大高冲突率

小批量的工作方式

流程能力



小批量工作方式

优势和福利

- 可让你更快速地获取有关变更的反馈，从而更轻松地对问题进行分类和修复
 - 可提升工作效率和动力
 - 可防止你的组织屈服于沉没成本谬论
-
- 产品/特性级别：最简可行产品（英文简称为 MVP）是指产品的原型
 - 团队每日工作级别：主干开发+持续集成，每一提交，少量功能分支并发，遇到了困难—重构项目/产品/代码架构，改善团队协作和沟通

小批量工作方式

遵循下列 INVEST 原则

- 独立性：尽量使特定批次的工作独立于其他批次，让团队可以按任意顺序处理这些特定工作，并独立于其他批次的工作进行部署和验证。
- 可协商性：每批次工作都是可迭代的，并且可以在收到反馈后重新进行协商。
- 有价值：每批次工作都有一定的作用，并且都会为利益相关者带来价值。
- 可估算性：各批次工作都可提供足够的相关信息，让你可以轻松估算范围。
- 规模小：在 Sprint 期间，你应该能够以较短的时间增量（即数小时到数天）完成各批次工作。
- 可测试性：每批次工作都可以按照用户期望的方式进行测试、监控和验证。

减小工作批次规模的方法

产品的计划和落地要配合

- 工作的分解方式支持团队更频繁地进行生产发布。
- 开发者在将工作分解为可在数小时（而非数天）内完成的小规模变更方面拥有丰富的经验。