

DOIS

DOIS | 2019 · 北京站  
DevOps 落地，从这里开始

# DevOps 国际峰会

暨 GNSEC 全球新一代软件工程大会

指导单位： 中国软件行业协会  
China Software Industry Association

主办单位： DevOps China

 高苏运维社区  
GiteeOps Community

时间：2019年7月5日-6日

地址：北京慈唐皇冠假日酒店

# 微服务产品团队规模化 DevOps 演进模式



顾宇

顾宇，埃森哲咨询经理。资深 DevOps 专家，架构专家，DevOps 标准、微服务标准核心编写专家。目前专注于研发质效提升、规模化 DevOps、云计算，微服务和规模化全功能敏捷团队的培训、设计、实践、落地以及经验推广。



PART

01



## DevOps 的模式语言

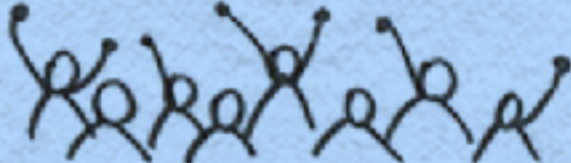
当我们在聊 DevOps 的时候，我们在聊什么？

# 去年的 DevOps 国际峰会



# John Willis 的启发



DEVOPS is a set  
of PRACTICES and   
PATTERNS that turn HUMAN CAPITAL  
into HIGH PERFORMINGANCE  
organizational capital.



# DevOps 国际峰会 2019 · 北京站

# 当我们在聊 DevOps，我们聊什么

## 第一类问题：

1. 风险
2. 收益
3. 时间

## 第二类问题：

1. 问题的现象
2. 问题的本质
3. 解决方案
4. 注意事项

模式语言就是一种结构化的表达

# DevOps 模式语言

1. 模式风险
2. 模式收益
3. 见效速度
4. 模式类别
5. 使用说明
6. 反模式



# DevOps 模式风险

- 低：不用担心，放心采用。该模式没有特殊场景和附加条件。
- 中：采用的时候需要注意场景和条件，否则会出现反模式。
- 高：请在实践过 DevOps 的专家指导下采用，轻易采用会带来严重后果。

# DevOps 模式价值

- 低： 采用该模式只会在初期产生收益，之后的重复边际效益递减。
- 中： 采用该模式产生固定的收益，每做一次就有一次收益。
- 高： 采用该模式一次就会对组织产生长久的收益。

# DevOps 模式见效时间

- 慢： 4 周内看不到显著改进。
- 普通： 2 - 4 周可看到显著改进。
- 快： 2 周内可看到显著改进。

# 五类 DevOps 模式

- 策略模式
- 组织模式
- 管理模式
- 文化模式
- 技术模式

# DevOps 反模式 - 常见的坑

1. 如果偏离了核心目标和原则，就会产生反作用。
2. 缺乏对错误的分析和分享就会让错误不断发生。

# 人件

(原书第3版)

第1版

Tom DeMarco  
Timothy Lister

著

机械工业出版社

机械工业出版社



第3版

软件行业影响最大、最具价值的著作之一。历时15年全面更新  
与《人月神话》共同被誉为软件管理领域必读的“双子星”  
近30年业界经典不衰

经典珍藏

## 人件

(原书第3版)

1000	1000	1000	1000	1000	1000	1000	1000	1000	1000
1000	1000	1000	1000	1000	1000	1000	1000	1000	1000
1000	1000	1000	1000	1000	1000	1000	1000	1000	1000
1000	1000	1000	1000	1000	1000	1000	1000	1000	1000
1000	1000	1000	1000	1000	1000	1000	1000	1000	1000
1000	1000	1000	1000	1000	1000	1000	1000	1000	1000
1000	1000	1000	1000	1000	1000	1000	1000	1000	1000
1000	1000	1000	1000	1000	1000	1000	1000	1000	1000
1000	1000	1000	1000	1000	1000	1000	1000	1000	1000
1000	1000	1000	1000	1000	1000	1000	1000	1000	1000



PRODUCTIVE PROJECTS AND TEAMS (Third Edition)



[美] Tom DeMarco Timothy Lister 著 冯晓 陈强 郭云 译



机械工业出版社  
China Machine Press

我们习惯性地不去管理的风险  
是我们自己的失败。

《人件》 P220



# DevOps 的目标

通过**组织改进**和**技术改进**

使得**端到端**的价值创造过程

能**更快的获得反馈并持续改进**



PART

02



## 规模化 DevOps 的模式和反模式

通过案例来看如何应用 DevOps 反模式

# 模式1：引入 DevOps 转型顾问

- 策略模式
- 风险 - 中
- 收益 - 中
- 见效 - 快
- 问题现象：缺乏 DevOps 能力
- 问题本质：岗位责任驱动而非团队目标驱动，不求有功但求无过。
- 关键措施：
  - 要有不同行业的案例讲解
  - 要同时具备管理实践和技术实践
  - 可以和团队“一起做”

# 反模式：招聘 DevOps 专家做转型

- 问题现象：DevOps 专家并没有带来实质上的改变。
- 问题本质：DevOps 专家仍然受制度的约束，无法突破制度。
- 关键措施：
  - 成立第三方专家团队。
  - 请外部顾问

## 模式2：“如何定义”和“如何度量”问题

- 文化模式
- 风险 - 无
- 收益 - 中
- 见效 - 快
- 问题现象：词汇不统一，理解不准确
- 问题本质：缺乏定量或者定性的定义和判断标准。
- 关键措施：
  - 明确所有的形容词的度量标准。
  - 建立统一的规范化词汇表，并作为单一可信数据来源。
  - 任何时候都要鼓励发问和讨论，而且要形成一致的认识。

## 模式3：定义 DevOps

- 策略模式
- 风险 - 中
- 收益 - 中
- 见效 - 快
- 问题现象：每个人都说出自己的 DevOps 定义
- 问题本质：DevOps 定义没有跨组织统一，每个人只看到了 DevOps 的需要。
- 关键措施：
  - 要统一收集跨组织的痛点。
  - DevOps 方案要按目标处理各方诉求。
  - 注意那些反对者。



# 模式4：DevOps 评估

- 策略模式
- 风险 - 中
- 收益 - 中
- 见效 - 快
- 问题现象：缺乏对 DevOps 的认识
- 问题本质：缺乏对关键问题的定义和可视化，岗位责任驱动而非团队目标驱动。缺乏对 DevOps 的定义
- 关键措施：
  - 参考成熟度模型要同时包括管理实践和技术实践。技术实践服务于管理实践。
  - 要能够明确定义和度量。
  - 要制定出下一步改进建议或方案。

# 反模式：低标准的 DevOps 定制化

- 问题现象：制定“适合的” DevOps
- 问题本质：难以承担风险和拒绝 DevOps 转型
- 关键措施：
  - 按高标准要求，但从实际情况出发。
  - 组织要对提高要求有统一和清晰的认识。

# 为什么质量差？

- 需求在传递的过程中交接太多，信息失真。
- User Story 不是 User Story ， 没有验收规格，各自瞎猜。
- 文档碎片化，不同的人看到片面的信息。
- 通过更长的流程来解决质量问题只会让质量反馈更慢。

## 模式5：最小可用流程

- 管理模式
- 风险 - 中
- 收益 - 高
- 见效 - 普通
- **问题现象：**端到端交付流程过长。
- **问题本质：**需求太大，没有切分到可以验证的最小颗粒度，通过加长流程增加不确定性。
- **关键措施：**
  - 合并流程中的活动，减少交接带来的失真。
  - 合并有效产出，避免碎片化。

# 反模式：增长的交付流程

- **问题现象：**很多交付环节和交付产出
- **问题本质：**需求质量规格不清晰就开始开发，妄图通过更多的活动来过滤风险，而不是尽早反馈给用户。
- **关键措施：**
  - 合并流程中的活动，减少交接带来的失真。
  - 合并有效产出，避免碎片化。

## 模式 6: Kanban

- 管理模式
- 风险 - 低
- 收益 - 高
- 见效 - 快
- 问题现象: 工作量工作效率没有可视化
- 问题本质: 缺乏度量的意识
- 关键措施:
  - 把所有的工作任务都记录下来。
  - 严格记录任务时长。
  - 采用累积流图改进资源分配。



# 模式 7：四类工作

- 管理模式
  - 风险 - 低
  - 收益 - 高
  - 见效 - 快
- **问题现象：** 有特别忙的团队和个人，任务经常超时或者进度延迟。
  - **问题本质：** 计划和风险没有隔离和区分对待，任务优先级失控。
  - **关键措施：**
    - 分析出-业务项目-内部IT项目-变更-计划外工作
    - 可视化所有任务并给任务分类。
    - 根据类别排优先级执行。
    - 根据不同的类别安排不同的团队。

## 模式 8：BAU 团队和 特性团队

- 组织模式
- 风险 - 低
- 收益 - 高
- 见效 - 普通
- 问题现象：团队因为很多计划外的工作而导致拖延。
- 问题本质：根据不同类型的工作分配资源。
- 关键措施：
  - 区分计划内工作和计划外工作。
  - 构建两个处理两类任务的团队。
  - 任务的处理流程需要严格规范化。
  - 建立轮岗机制。

# 模式 9：Scrum with Ops

- 管理模式
  - 风险 - 高
  - 收益 - 中
  - 见效 - 普通
- 问题现象：只有 Dev，没有 Ops
  - 问题本质：责任驱动而非目标驱动，人员按责任而不是任务区分。
  - 关键措施：
    - 3 - 3 - 5 - 5 要做到。
    - 任务要有度量，燃尽图 + 累计流图
    - 所有活动本身要有质量控制，不要只看产出。
    - 要通过 5 个价值观分享和反思。
    - 积累完善经验。

# 你们做 CI 了吗？

“大哥，做 CI 和做 CI 是不一样的！”

“我说的做 CI 是做齐十一个实践，你说的做 CI 是在 Jenkins 上跑构建。”

“测试驱不动，持续集不成”

## 3 种不同的 TDD

- 测试驱动开发
- 测试人员驱动开发人员
- 测试计划驱动开发计划



# 模式10：测试驱动开发

- 技术模式
- 风险 - 低
- 收益 - 高
- 见效 - 慢
- 问题现象：不知道如何 TDD
- 问题本质：缺乏对需求的规格化描述
- 关键措施：
  - 测试用例要明确。
  - 一定先写测试，再实现代码。代码只是为了修复测试
  - 将测试作为任务而不是职责。
  - 集成到 CI 里面。

# 模式11：测试人员驱动开发人员

- 管理模式
- 风险 - 低
- 收益 - 中
- 见效 - 慢
- 问题现象：测试期间爆发很多问题
- 问题本质：测试人员没有否决权，成为了“垃圾桶”
- 关键措施：
  - 减少测试人员，让测试避免成为一个角色。
  - 让测试人员参与需求的分析和规格的确定。
  - 让测试人员做少量的手工测试。
  - 让测试人员给团队的自动化测试赋能。

# 模式 12：测试计划驱动开发计划

- 策略模式
- 风险 - 中
- 收益 - 高
- 见效 - 快
- 问题现象：开发计划经常延期
- 问题本质：通过拉的方式而不是推的方式分散集中测试阶段的质量和进度风险。
- 关键措施：
  - 制定依赖倒置的发布计划和测试计划。
  - 根据测试计划的依赖顺序进行开发。
  - 将需求拆分的颗粒度匹配到测试计划的颗粒度。

“我都不知道要什么结果数据，怎么写测试？”

# 反模式：缺乏验收条件的用户故事

- **问题现象：** 开发周期很长，不断返工。
- **问题本质：** 没有和用户核对验收条件，过程中不验证。
- **关键措施：**
  - 在建立用户故事的时候以验收条件作为结束。
  - 验收条件要定义和度量。
  - 尽量要求验收条件自动化

那么？需求应该怎么分析？

## 模式 13：用户故事成熟度

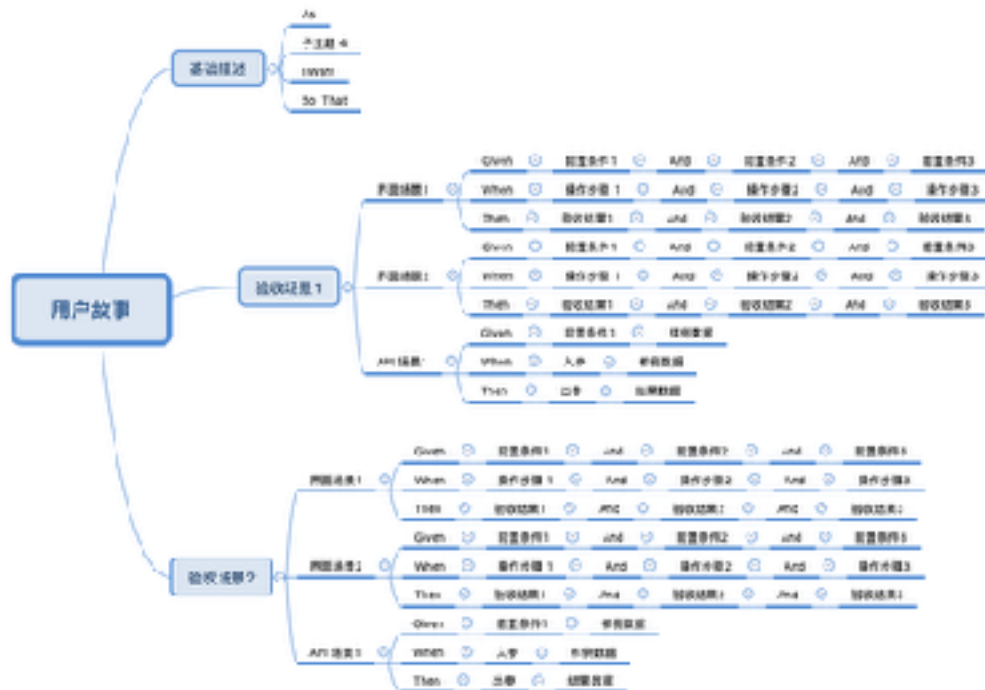
- 管理模式
- 风险 - 低
- 收益 - 高
- 见效 - 快
- 问题现象：用户故事有价值信息太少
- 问题本质：不关注用户故事活动的质量
- 关键措施：
  - 给用户故事建立 AC。
  - 根据 AC 分析测试场景和用例。
  - 和用户确认 AC

# 用户故事成熟度 5 级

- L1 - 没有AC
- L2 - 有 AC
- L3 - 有 BDD 风格的 AC 描述
- L4 - 有基于 AC 的 测试场景及用例分析
- L5 - 能够自动化验证用户故事



# 采用思维导图做用户故事地图



# DevOps 实现高频率高质量发布的核心

将需求拆分到短期可以低风险快速验证的颗粒度

# DevOps 试点成功经验的推广

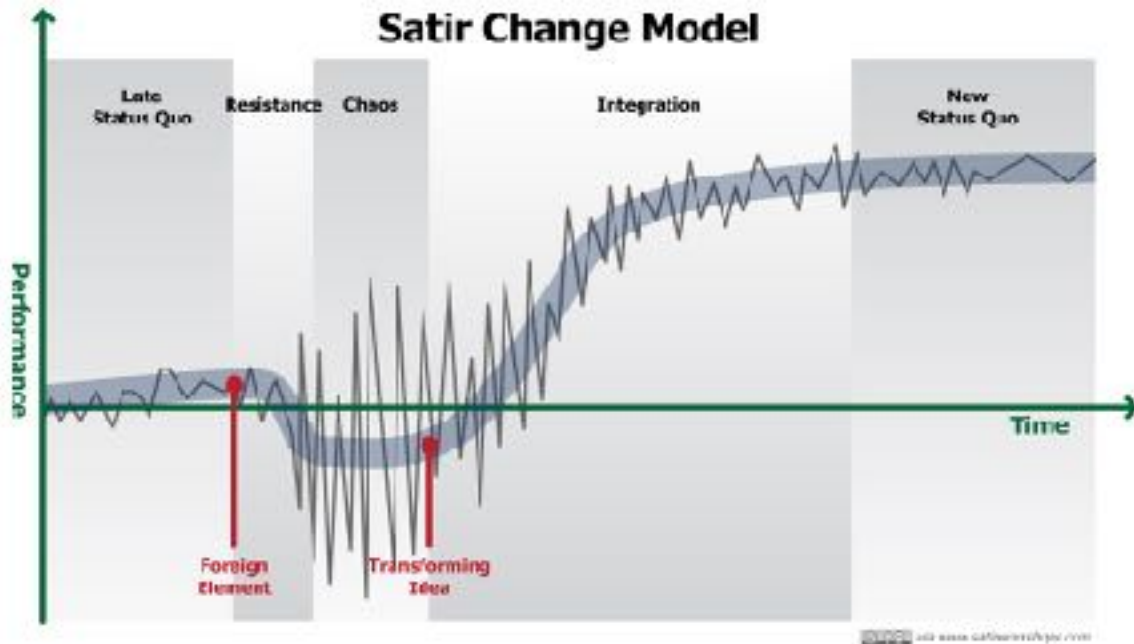
# 一次敏捷回顾会的反思

- 并不是所有人都欢迎 DevOps。
- DevOps 可能出现抵触。
- 团队成员在被动的做 DevOps。
- 团队交付压力大。

# 反模式：要我做 DevOps

- **问题现象：**大家对 DevOps 实施有反抗情绪。
- **问题本质：**从上而下的任务分解，而不是自我价值驱动的转型。
- **关键措施：**
  - 把问题抛给团队解决，而不是自己给出解决方案。
  - 鼓励成员承诺完成，而不是命令完成。

# 萨提亚改变模型



# 模式 14：DevOps 复制模式

- 策略模式
- 风险 - 高
- 收益 - 中
- 见效 - 快
- 问题现象：构建一个新榜样，按榜样复制
- 问题本质：忽略了转型过程中带来的负面情绪，追求快速见效。
- 关键措施：
  - 做资源和职责的隔离。
  - 建立全功能团队。
  - 总结完整的端到端的实践举措。
  - 及时复制，获得反馈。

# 模式 15: DevOps 改进模式

- 策略模式
- 风险 - 低
- 收益 - 中
- 见效 - 慢
- 问题现象: 按成熟实践推广
- 问题本质: 改变的引入集成需要一个过程。
- 关键措施:
  - 按实践总结, 并推广。
  - 建立全功能团队。
  - 及时复制, 获得反馈。



# 模式16： DevOps 试点团队

- 策略模式
- 风险 - 低
- 收益 - 中
- 见效 - 快
- 问题现象：限制 DevOps 带来转变的风险
- 问题本质：没有建立度量和评估体系，缺乏对 DevOps 转型带来影响的信心。
- 关键措施：
  - 项目要关键，团队要完整。
  - 需要有一名转型教练。
  - 团队要有一定的自主权。

# 模式 17: DevOps 推广团队

- 策略模式
- 风险 - 低
- 收益 - 中
- 见效 - 普通
- 问题现象: 限制 DevOps 带来转变的风险
- 问题本质: 没有建立度量和评估体系, 缺乏对 DevOps 转型带来影响的信心。
- 关键措施:
  - 试点团队需要建立实践总结和分享。
  - 推广团队由其它主要负责人担任。
  - 推广团队要定期评估和反馈实践落实情况以便及时调整。

# 模式 18: DevOps 排位赛

- 策略模式
- 风险 - 低
- 收益 - 高
- 见效 - 普通
- 问题现象: 组织对 DevOps 转型漠不关心
- 问题本质: 缺乏激励机制
- 关键措施:
  - 采用统一的度量标准
  - 只奖励, 不考核
  - 通过提升基线, 不断改进

# 模式 19: DevOps 规范

- 策略模式
- 风险 - 低
- 收益 - 高
- 见效 - 普通
- 问题现象: 缺乏统一的实践
- 问题本质: 缺乏严格定义的操作指导手册, 没有统一对实践的培训和赋能。
- 关键措施:
  - 文档化, 构建单一数据源。
  - 及时反馈, 及时更新。

## 模式 20：DevOps 平台

- 技术模式
- 风险 - 中
- 收益 - 高
- 见效 - 快
- 问题现象：构建自动化
- 问题本质：统一规范、操作度量并及时反馈。
- 关键措施：
  - CI/CD 规范化。
  - 做端到端的产品集成。

# 模式 21：DevOps 分享

- 文化模式
- 风险 - 中
- 收益 - 高
- 见效 - 快
- 问题现象：大家不愿意分享
- 问题本质：缺乏分享激励机制和引导。
- 关键措施：
  - 从简单的讲笑话开始。
  - 内部组织分享。
  - 参与到外部分享。
  - 积累，尝试，集成。

想了解更多的 DevOps 模式请关注我的公众号





# Thanks

DevOps 时代社区 荣誉出品



想第一时间看到高效运维社区  
的新动态吗?

