2017

# DevOps 现状调查报告

Presented by:

puppet + DORA
DEVOPS RESEARCH & ASSESSMENT

Sponsored by:

Hewlett Packard Enterprise
splunk>
amazon web services
Atlassian
Electric Cloud
Deloitte.
WAVEFRONT by vmware

# 目录



Back to Contents

# 目录

01

# 概述

今天， DevOps已经成为一套为人所熟知的实践方法集和文化价值观，它可以帮助任何规模的组织缩短软件发布周期，提升软件质量、安全以及快速获取产品开发反馈的能力。过去的六年里，超过27000份有效的调研反馈有力的证明了DevOps 实践可以带来更高的IT服务效能。高效能的IT组织实现了在生产力、盈利能力和市场份额方面的增长。

在今年的研究中，我们发现DevOps发挥的积极影响已经不仅仅局限于财务业绩，不论是商业组织，还是非盈利机构，无论他们的使命是什么，DevOps都能帮助他们实现自己的目标。

我们研究了有效的领导者如何通过影响技术实践和过程改进，提高IT和组织的效能。 我们深信自动化是带来组织效能差异的关键因素。我们深入研究了应用架构和组织结构如何影响组织的软件研发和交付能力。

我们希望这篇报告中宏观和实践型的调查结果能在您的DevOps旅程中助您前行。

# 主要亮点

## 变革型领导者有五大共同特征，这些特征对塑造组织的文化和实践，提高组织效能影响巨大。

变革型领导者的五大特征包括：愿景、鼓舞型沟通、智力激发、支持型领导、个人认同，这些与高效能组织休戚相关。报告显示，高效团队领导者普遍具有以上特征，并坚定实践着这些行为，低效能团队领导者则往往缺乏这些特质。报告显示，这些不具备变革能力的团队，工作效率仅仅是高效能团队的一半。

## 高效能团队在产品快速迭代和稳定性上可以兼得。

与去年相比，低效能团队改善了部署频率和变更前置时间，因此高低效能团队间的差距有所缩小。然而，值得关注的是低效能团队反馈的故障恢复时间更长，故障失败率也更高。我们认为，快速频繁的部署压力，导致低效能团队忽略了内建质量的重要性。

# 主要亮点（续）

## 自动化是组织的法宝。

与其他团队相比，高效能的团队的一个显著特征就是尽可能的自动化一切工作：配置管理、测试、部署和变更审批流程。结果是高效能团队在创新和快速反馈上能投入更多时间。

## DevOps适用于所有组织。

今年，我们考察了组织在财务和非财务方面的效能指标。我们发现，无论是财务还是非财务指标方面，高效能团队最终都可以达到预计目标的两倍。去年的研究表明，无论是商业成品软件（COTS）还是在云端部署微服务，DevOps实践都有助于获得更高效能。今年，我们会在DevOps世界中重新思考商业软件COTS的未来。

## 松散耦合架构和团队是持续交付最有力的预测指标。

如果您希望进一步提升IT效能，推荐使用松耦合服务架构，这些服务可以彼此独立开发和发布，同时组建松耦合团队，并授权这些团队对服务进行更新。为了适应这种改变，那些重流程管控的传统企业将花费更大的精力实现，确保从产品设计到生产的顺利开展。相应的松耦合的团队和服务将带来许多优势：更高的生产力、更高的质量、更可靠的稳定性。

## 精益产品管理推动组织效能提升。

精益产品管理实践帮助团队更频繁地交付客户实际需要的功能。更快的交付周期允许团队进行不断试错，建立与客户之间的反馈机制。结果呢？显然整个组织的盈利能力、生产力和市场份额均能受益。

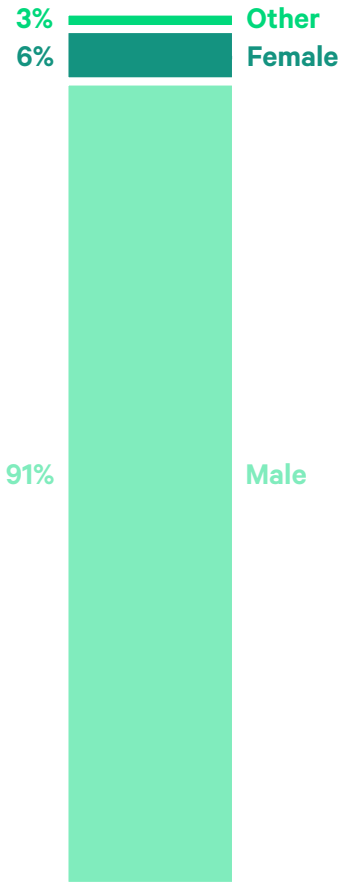

# 统计资料

我们分析了过去六年超过27000封调查反馈，这些反馈覆盖了IT领域的专家，开发者和高管，我们尽可能提供全面详尽客观的DevOps研究报告。今年又有全球范围内的3200人加入了我们的调查。

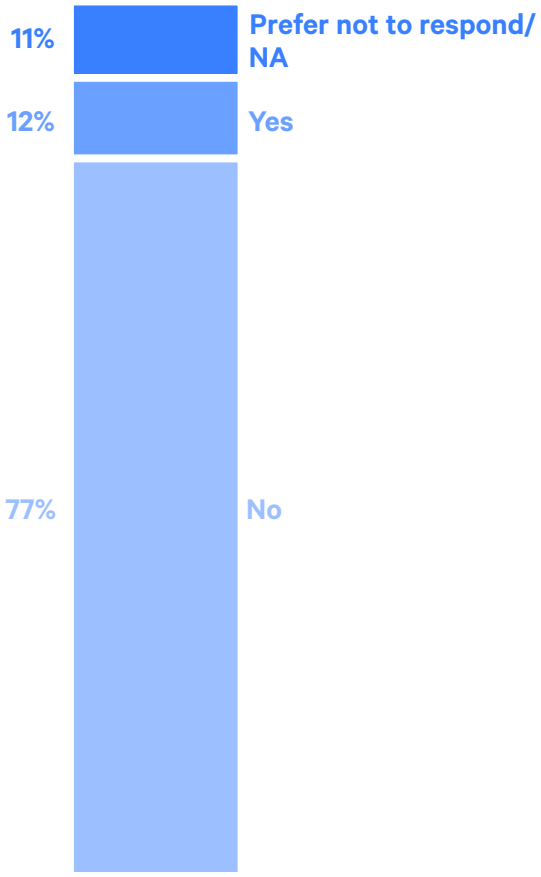
随着DevOps不断进化和扩展，我们注意到从事DevOps专业领域的人员比例逐年增长。在2014年，16%的受访者在DevOps型团队中工作，3年后的今天，这个比例达到了27%。我们认为这样一种客观数字的提升，一方面代表了DevOps可以带来实际效果的共识，另一方面也印证了越来越多的组织从传统工作方式到基于DevOps的新工作流程的转变成为一种趋势。

# 统计资料

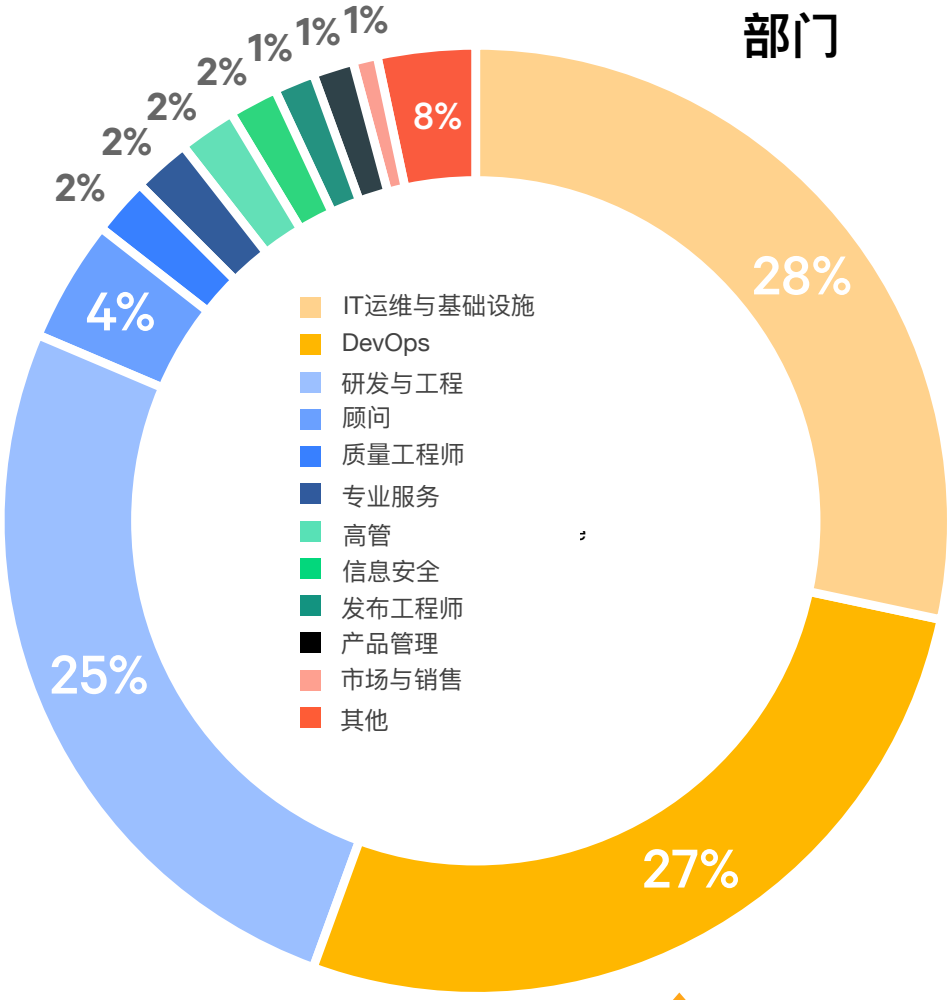
## 性别




## 弱势群体成员*




## 部门

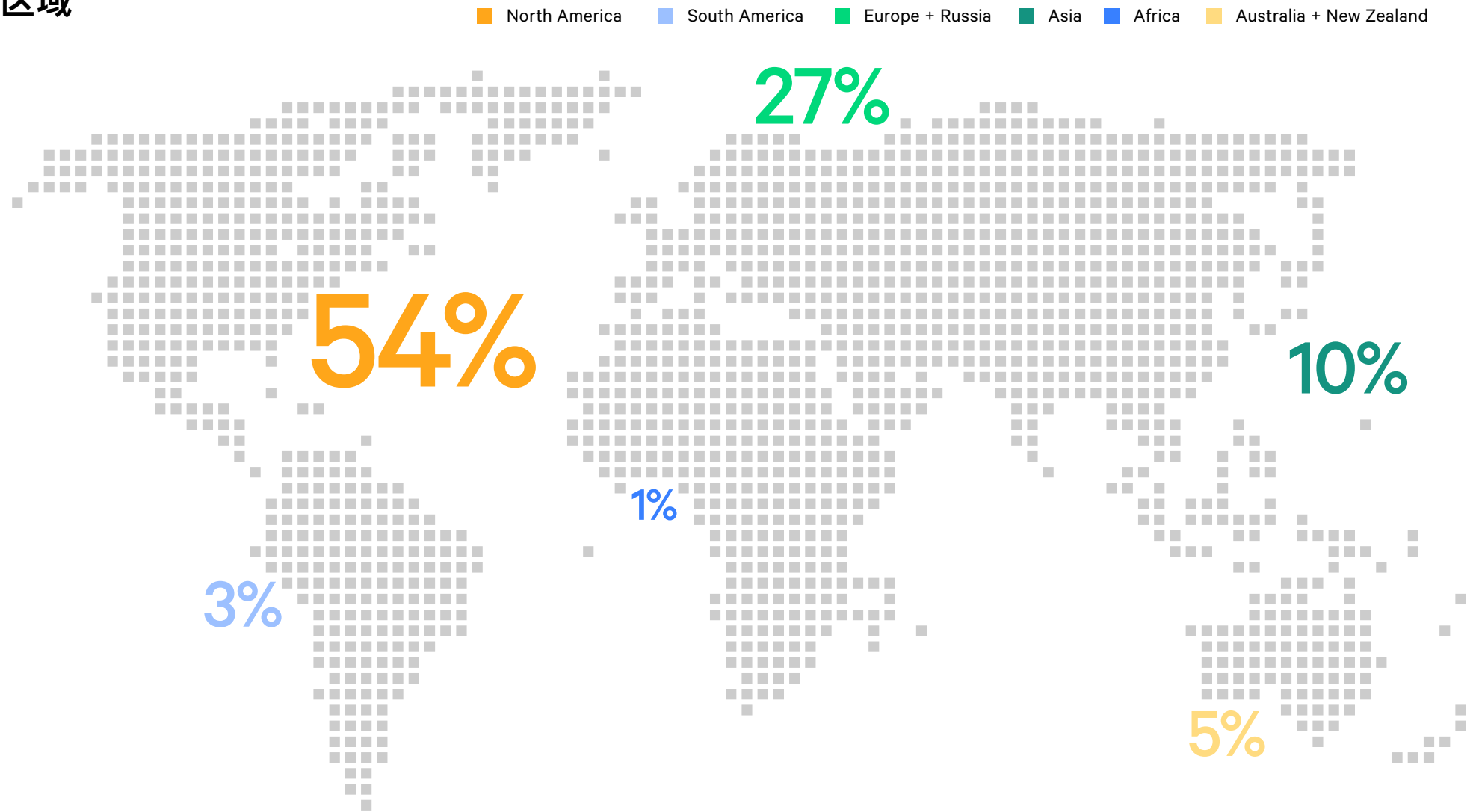



DevOps团队比例2014年16%，2015年19%，2016年22%，2017年27%

* 备注：在今年的调查中我们新增了一个问题，受访者是否认为他们是弱势群体中的一员，让受访者自行定义，这与他们的种族，性别以及其他人格特征等多方面因素有关

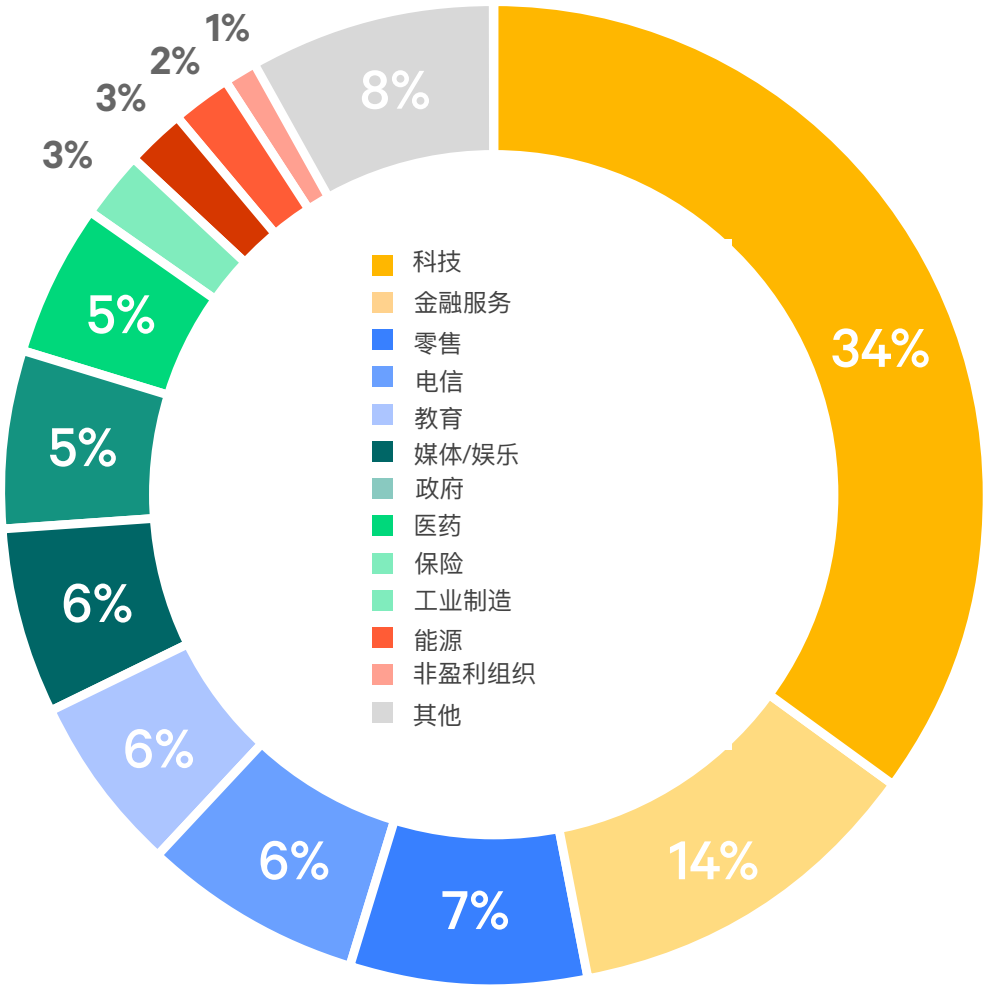
# 统计资料

## 区域

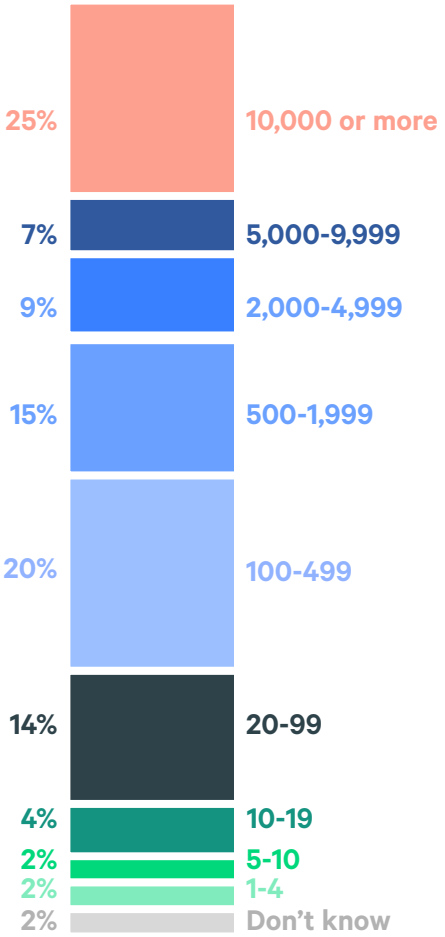

North America
South America
Europe + Russia
Asia
Africa
Australia + New Zealand
27%
54%
10%
1%
3%
5%

# 统计资料

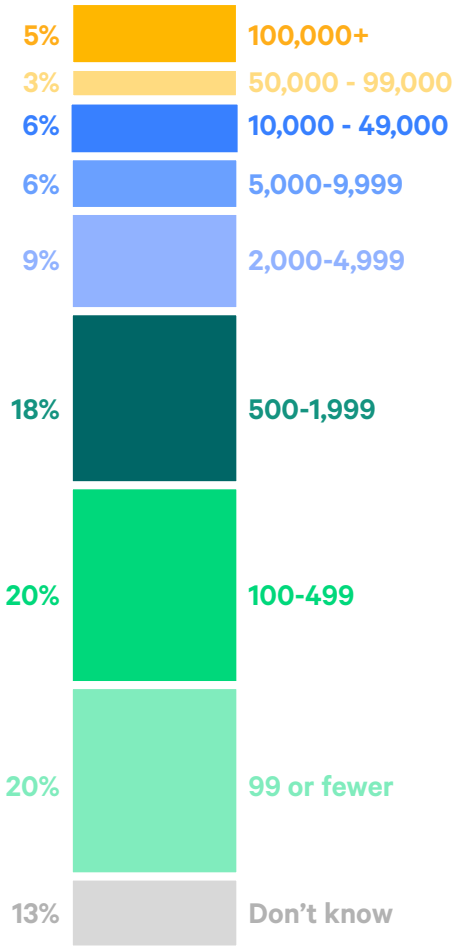
## 行业

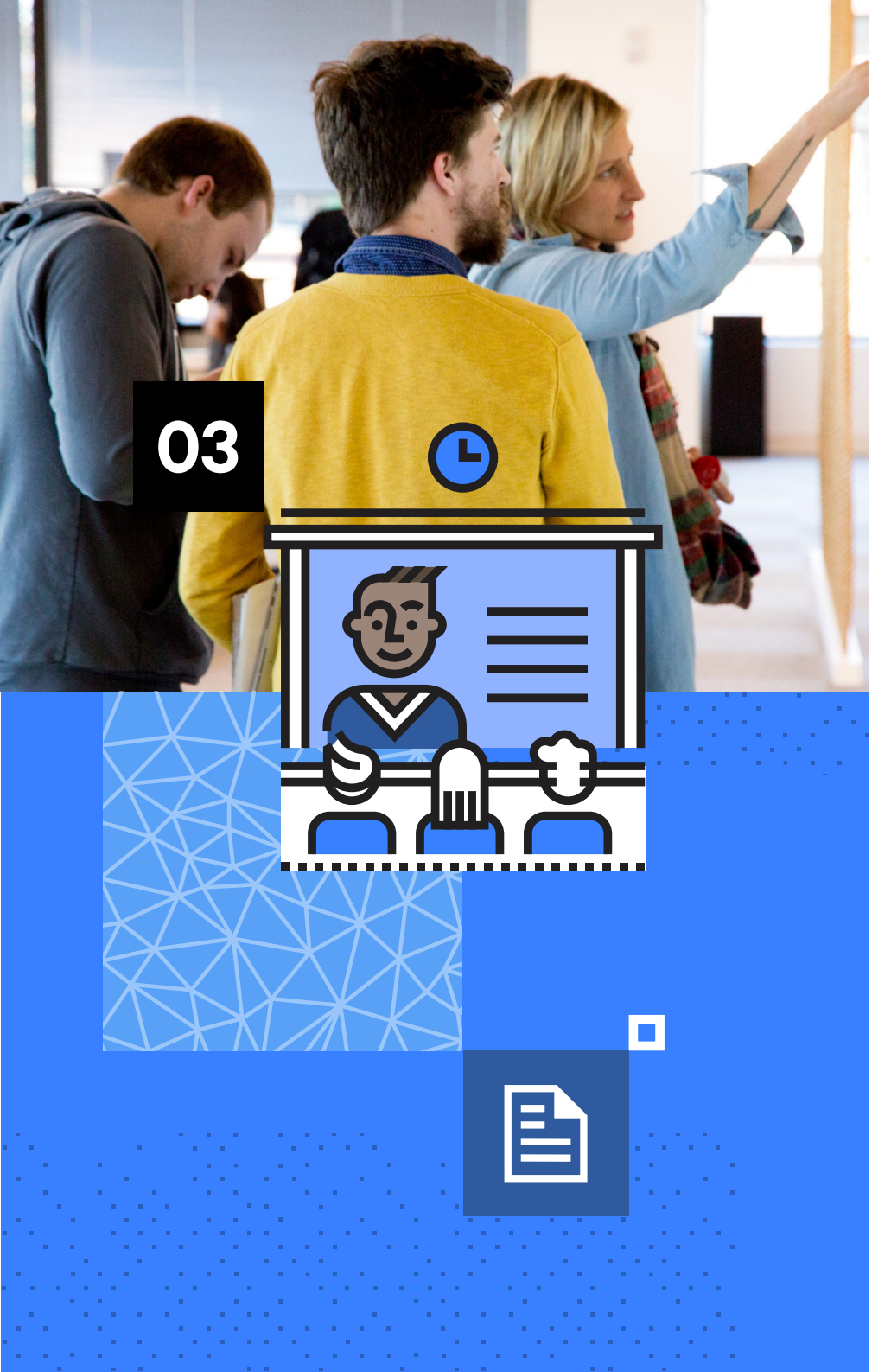



## 企业人员规模




## 服务器规模






# 变革领导力

还没有意识到IT领导力有么多重要？ 试想一下：到2020年，近半数还没有完成团队能力转型的CIO会从组织数字化领导小组中离开。[1]

毋容置疑领导力会对结果产生了显著的影响。 一个好的领导者会影响团队交付代码的能力，建立良好的系统架构，并将精益原则融入团队管理工作和产品开发过程之中。 组织的盈利能力，生产力和市场份额都会受到影响，并且这些影响都是可以度量的。 对于不考虑盈利的非营利机构和政府组织，可以从客户满意度，效率和实现组织目标的能力来看待领导力的影响。

---

[1]: Gartner预测 - IT基础架构与运营。 (link to infographic on gartner.com)

今年最令人兴奋的研究领域之一就是调查那些有助于推动高效能组织建设的领导特征。 在我们看来，这一直是DevOps中被习惯性忽视的话题之一，尽管事实上，变革领导力是必不可少的：

- 建立和支持高度互信的组织文化。

- 实施技术和流程，以提高开发人员的生产力，减少代码部署前置周期，并承载更可靠的基础架构。

- 支持团队实验和创新，更快更好的交付产品。

- 跨组织机构工作，实现战略协同。

在DevOps社区内，我们时常会吐槽领导的行为，例如，中层管理者或保守派的阻挠使我们无法推行IT和组织效能改进工作。

不过我们更常遇到的问题是“我们如何说服领导，帮助我们做出必要的改变”。 每个人都认识到领导的参与对于成功的DevOps转型至关重要。 领导有权力和预算来进行大规模的变革，在进行转型时提供正向的支持，改变整个工程师群体的激励措施，无论他们是在开发，质量保证，运营还是信息安全。 领导者是组织的基调，定义了基本的文化规范。

在今年的研究中，我们采用了包括五个维度在内的变革领导力（Rafferty和Griffin，2004）[2]。根据这一模式，变革型领导者的五个特点是：

- 愿景：对组织走向有明确的概念，五年后应该达到的目标。

- 鼓舞型沟通：采用一种鼓舞和激励的方式进行沟通，尤其是在一种不确定的环境中。

- 智力激发：鼓励员工以全新的角度思考问题。

- 支持型领导：设身处地地关注员工的个人需求和感受。

- 个体认同：表彰目标达成和工作质量改进，亲自祝贺那些做出了杰出贡献的同僚。

[2] Rafferty，A.E.，＆Griffin，M.A.（2004）。变革型领导的维度：概念和经验扩展。领导季刊，15(3),329-354。

## 什么是变革领导力？

变革领导力是领导者通过调动员工的价值观和使命感，来激发和鼓励追随者达成更高的效能，并促进广泛的组织变革。 这些领导者鼓励他们的团队通过他们的愿景，价值观，沟通，表率以及他们对追随者个人需求的明确关心，实现共同目标。

据观察，服务型领导与变革型领导层之间存在着相似之处，但领导者的重点却有差异。 服务型领导人侧重于追随者的发展和表现，而变革型领导者则致力于让追随者与组织产生共鸣并拥护组织的目标。

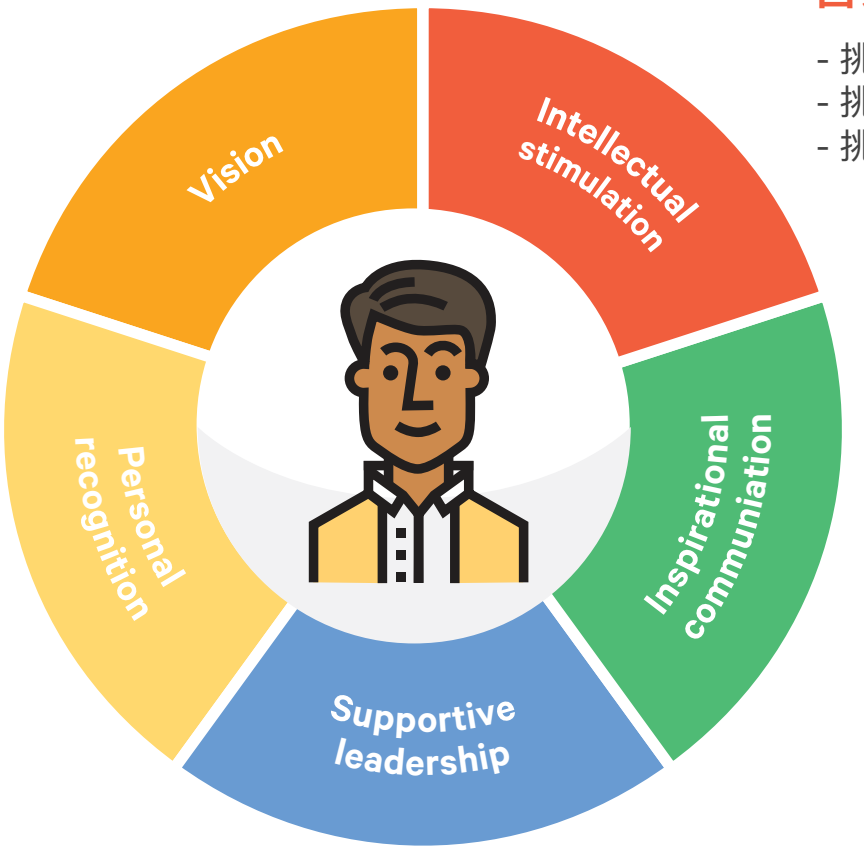
# Dimensions of transformational leadership

## 愿景

- 明晰组织方向。
- 明确团队方向。
- 预见5年后团队方向。

## 智力激发

- 挑战团队现状
- 挑战团队不断提出新问题。
- 挑战团队对工作的基本设想。




## 个体认同

- 赞扬高出平均水平的工作。
- 认可工作质量的提高。
- 亲自赞美个人的出色表现。

## 鼓舞型沟通

- 激发团队成员的自豪感。
- 宣传团队内部的正能量。
- 激发热情和积极性; 鼓励人们看到这种变化带来的机会。

## 支持型领导

- 在行动之前考虑他人的感受。
- 思考他人的个体需求。
- 关心个人的兴趣。

变革领导力的特征与IT效能休戚相关。事实上，在高，中，低效能IT团队中，领导者特征有显著差异。高效能团队中的领导者往往在各方面都坚定践行：愿景，鼓舞型沟通，智力激发，支持型领导和个体认同。相比之下，低效能团队的领导者则表现平平，这些发现都可以被统计数字来证明。

另外值得关注的是，那些不具备变革领导力的领导者的团队，在IT效能方面仅为高效能团队的一半。这也佐证了我们的普遍认知：虽然我们经常听到那些自下而上推动DevOps和技术转型的成功案例，但当领导者愿意提供有效支持的时候，这样的成功往往容易的多。

通过我们的分析发现，变革领导力与员工净推荐值(Net Promoter Score NPS) 高度相关。在一个让员工感到快乐，忠诚和敬业的团队背后经常存在着变革型领导者。综合近些年的研究成果，可以预见到在那些由变革型领导者带领的团队中，更加倾向于催生出生产力组织文化（借鉴了社会学家罗恩·韦斯特鲁姆的说法[3]），并且在工作中体现出价值，这一点正如我们在2016年的报告中提到的那样。

[3]: Westrum，R.组织文化的类型学
qualitysafety.bmj.com/content/13/suppl_2/ii22.

有意思的是，我们发现想要让DevOps的效果最大化，只有变革型领导者的存在是不够的。在那些反馈他们的领导者与变革领导力特征非常吻合的团队中，我们聚焦了其中的前10%，并假设这些团队理所应当会展现出最佳的效能。而实际情况并非如此，这些团队的效能水平参差不齐。这告诉我们，仅仅依靠变革领导力行为本身不足以驱动高IT效能。

为什么会产生这样的结果呢？因为领导者无法靠一己之力实现DevOps。DevOps的成功还取决于合适的架构设计，良好的技术实践，精益管理原则的善用以及多年来我们专注研究的其他所有因素。

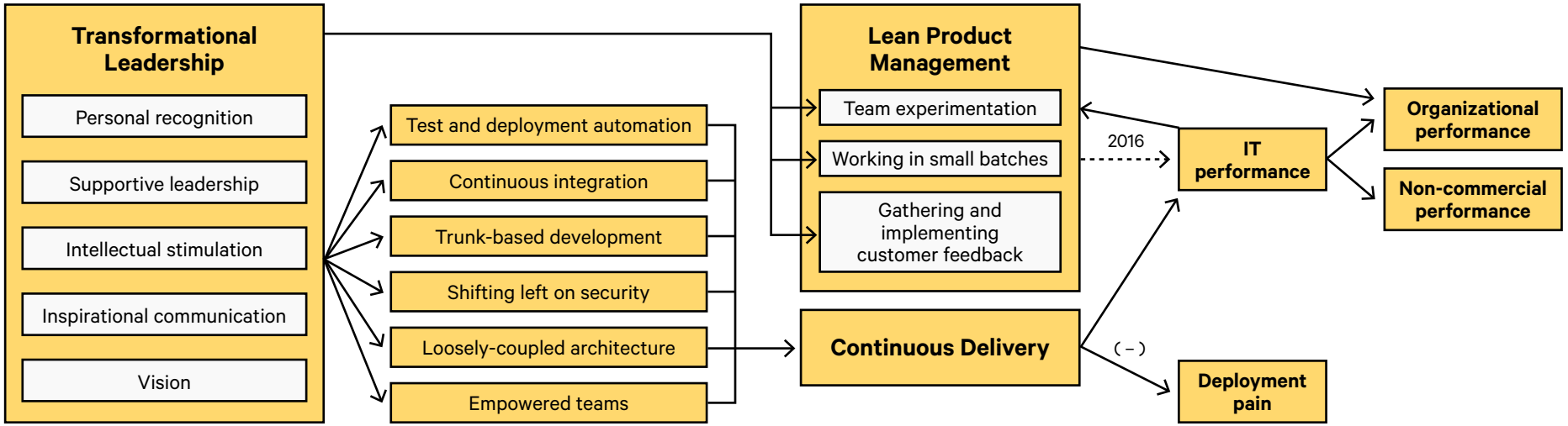

总而言之，我们发现优秀的领导者通过授权团队进行系统重构，实施持续交付和精益管理实践等多种途径，帮助建立卓越的团队，卓越的技术和卓越的组织。变革领导力能够实现通往高效能的必要实践，并且支持团队成员之间有效的沟通和协作，以追求组织目标达成。 这种领导奠定了团队文化基础，让连续试错和不断学习成为每个人日常工作的一部分。

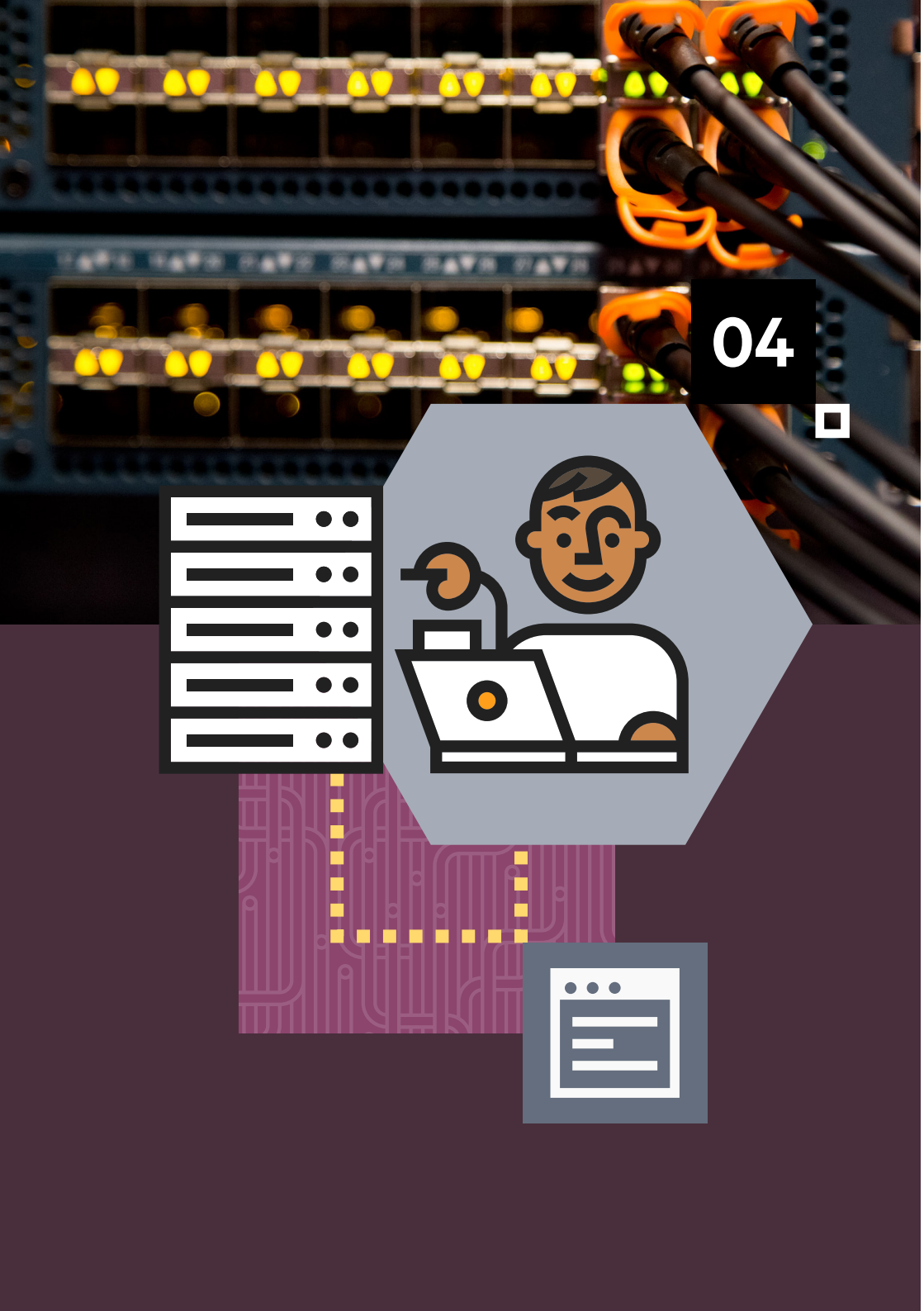
变革型领导者的行为让我们研究识别出来的价值观，过程和实践成为现实。变革领导力不能简单的理解为一个单独的行为或一套新的做法，相反作为一种综合能力，他推动了我们几年来持续关注的技术和组织实践的不断扩展。

下图展示了我们今天调查过程中反复验证的模型，它是一个结构化方程模型（structured equation model SEM），用于测试关系的预测。图中的每个框都代表了我们在研究中的衡量要素，每个箭头表示要素之间的关系。 为了更友好的解释模型，图中的所有箭头都可以使用 预测，影响，驱动或作用 等词汇来代替。

例如，IT效能可以预测组织效能。 如果您在其中一个箭头旁看到减号，则表示该关系为负值，比如持续交付会加剧部署困境。 模型中的所有箭头表示统计学上的明确关系。 精益产品管理与IT效能之间的关系在2016年的报告中有过详细论述，可在精益产品管理章节进一步讨论。

**Figure 1. Structured equation model showing relationships between constructs**






# IT效能与组织效能

如今，无论是企业还是公益组织，都依赖于软件和IT来实现组织目标。之所以组织都开始引入DevOps，是因为他们看到了越来越多的事实表明，DevOps实践能够帮助他们更快、更可靠、更高质量地交付软件。

我们从以下两方面度量IT效能：代码生产力和系统稳定性。生产力是通过团队交付代码的频率和从提交代码到部署上线的速度来衡量的。稳定性是通过系统故障修复速度和变更失败率来衡量的。

前几年里，我们发现高效能组织比低效能组织在生产力和稳定性方面有明显的优势，2017年，我们发现高绩效组织：

- 高出46倍的部署频率

- 快出440倍的前置时间

- 快出96倍的故障恢复时间

- 低出5倍的变更故障率

和2016年的结果相比，高效能组织和低效能组织在生产力（部署频率和周期时间）方面的差距在缩小，但是在稳定性（故障恢复时间和变更失败率）方面的差距在拉大。我们猜测是因为低绩效团队为了提高速度，在内建质量于流程上投入不够。其结果就是更大的故障，更多的服务恢复时间。高效能组织明白他们不能牺牲质量换取速度，反之亦然，他们通过内建质量来获得速度和质量的双赢。

**Table 1: Changes in IT performance of high performers, 2016 to 2017**

| IT performance metrics | 2016 | 2017 |
|---|---|---|
| **部署频率** | 200x more frequent | 46x more frequent |
| **变更前置时间** | 2,555x faster | 440x faster |
| **故障恢复时间（MTTR）** | 24x faster | 96x faster |
| **变更失败率** | 3x lower (1/3 as likely) | 5x lower (1/5 as likely) |

有人可能会好奇为什么同比差距有增有减。有一点很重要：度量指标是相对，这些指标用于比较高效能和低效能组织之间的差异。

在2016和2017年之间，代码部署频率差距缩小了：高效能组织依然在按业务需求交付代码，而低效能组织从原来2016年的1次/月 ~1次/6月提高到2017年的1次/周 ~1次/月。

2017年，低效能组织也缩短了变更的周期时间：从2016年的1个月~6个月到2017年的1周 ~1个月。这种变化并不是意味着高效能组织表现的不好了，这仅仅意味着低效能组织在生产力方面比以前做的更好了，我们也很高兴看到他们的提升。

相比之下，高效能组织在过去一年中，学会了从生产环境和基础设施宕机中恢复服务，并且优先防止故障，从而获得了更大的优势。 这可能会给了他们让客户满意的优势，因为他们有更多的机会提供新的价值，他们的发布质量也更高了。 结果就是高绩效组织能够更快的交付上市，提供更好的客户体验和拥有更快的市场想要速度。

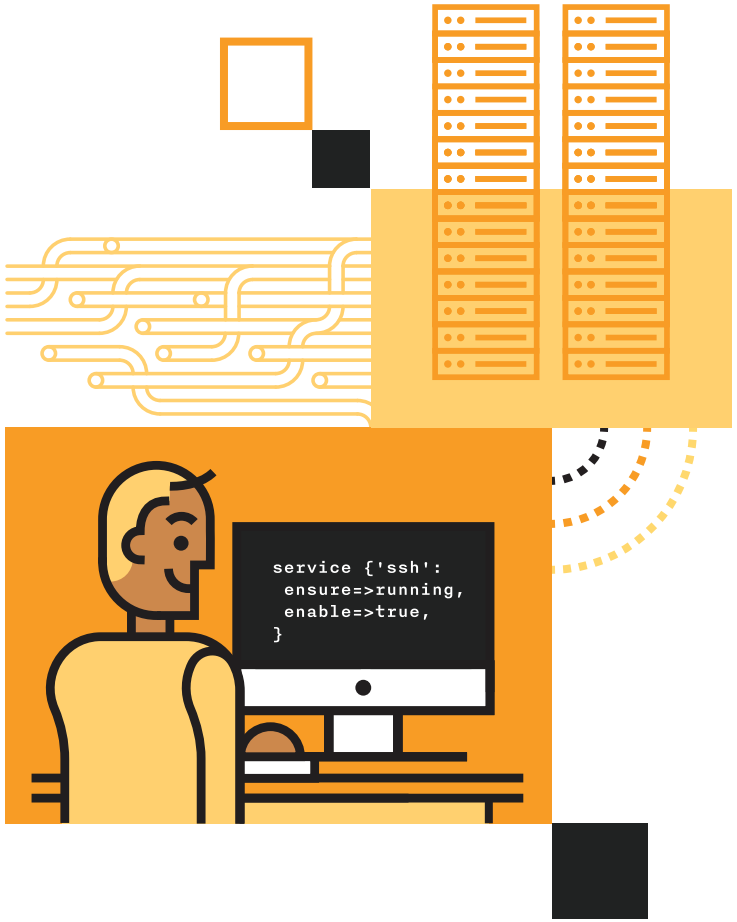


**Table 2: 2017 IT performance by cluster**

| Survey questions | High IT performers | Medium IT performers | Low IT performers |
|---|---|---|---|
| **部署频率**<br>针对主要应用和服务，你的组织多久部署一次代码? | On demand (multiple deploys per day) | Between once per week and once per month | Between once per week and once per month* |
| **变更前置时间**<br>针对主要应用和服务，变更的前置时间是多长（从代码提交到代码在生产环境中运行成功的时间）? | Less than one hour | Between one week and one month | Between one week and one month* |
| **故障恢复时间（MTTR）**<br>针对主要应用和服务，如果发生服务故障，一般多久能恢复服务（比如：计划外宕机，服务损害）? | Less than one hour | Less than one day | Between one day and one week |
| **变更失败率**<br>针对主要应用和服务，变更结果是回滚或随后修复的比例是多少（比如：导致服务损害，服务中断，需要紧急修复，回滚，向前修复，补丁）? | 0-15% | 0-15% | 31-45% |

* *Note: Low performers were lower on average (at a statistically significant level), but had the same median as the medium performers.*

For information on how we determined high, medium and low IT performance, please see the Methodology section.

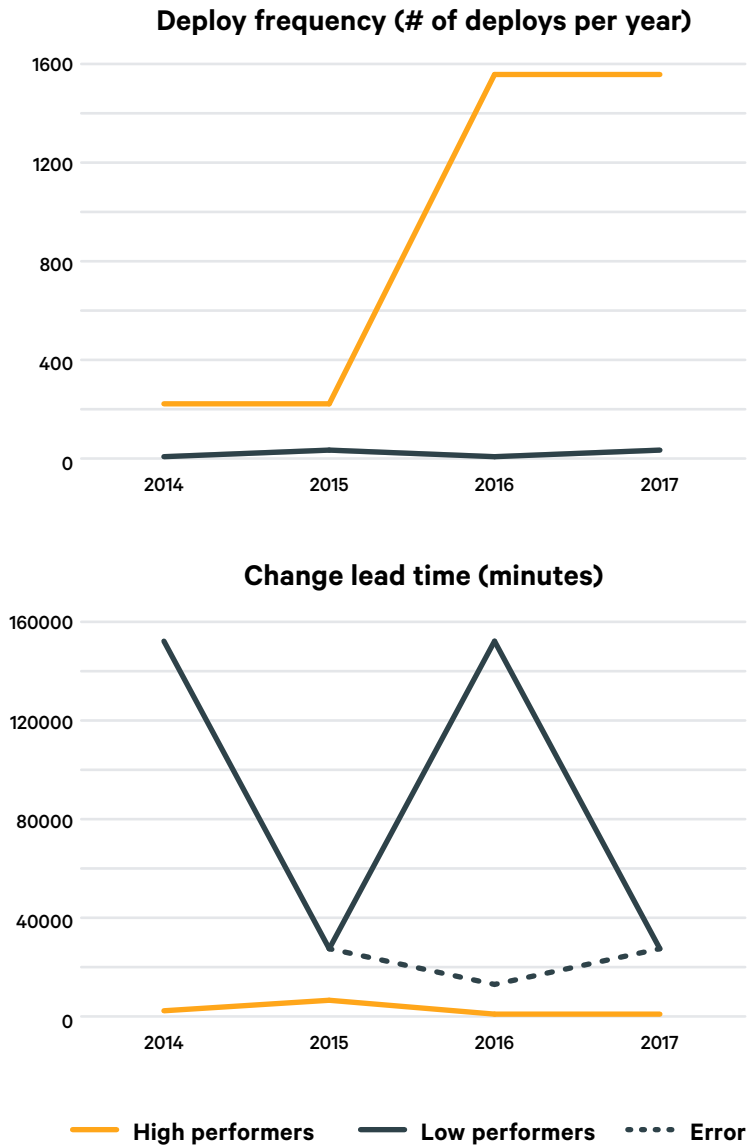
# 生产力

## 部署频率

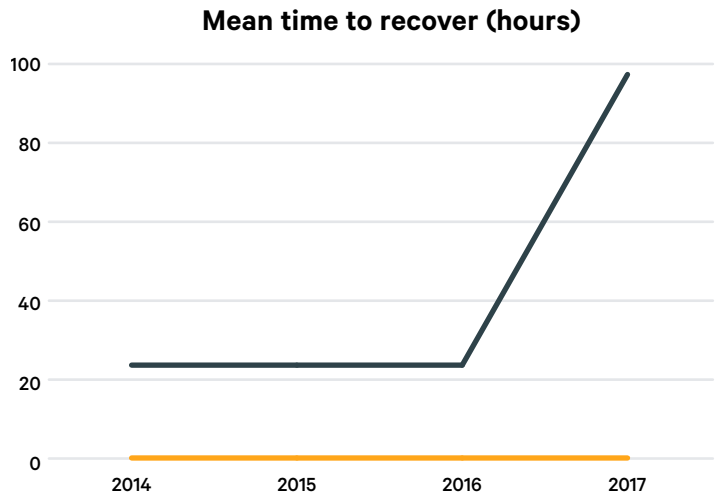
今年，高效能组织报告说他们日常都是按需部署，一天执行多次部署。而低效能组织一周或者一个月部署一次。这项数据比去年有所提升。针对高效能组织，我们统一该数据为1460次/年（4次/天 x 365天）。针对低效能组织，该数据为32次/年（取52次部署和12次部署的平均值）。这些统一后的数据显示，高效能组织的部署频率是低效能组织的46倍。一天部署4次相比于一天部署80次的Etsy，部署上千次的Amazon和Netflix而言（生产环境中成百上千个服务的总和）不值一提，还是过于保守。

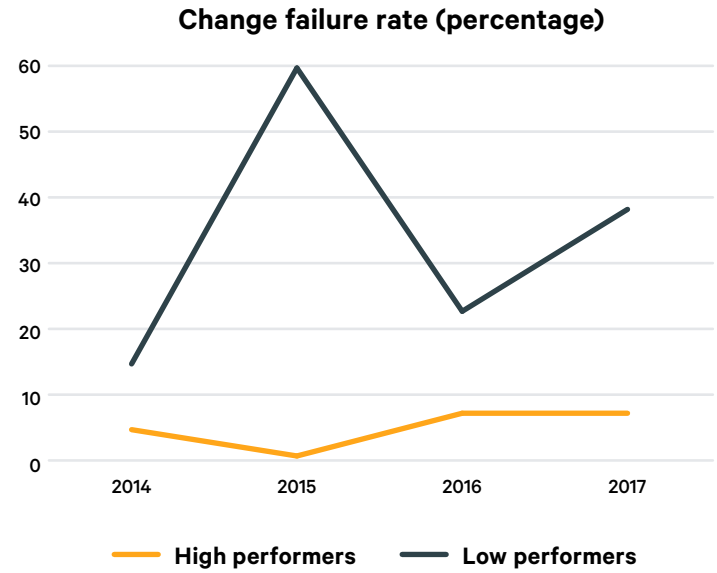
## 变更周期时间

类似的，高效能组织报告说他们的部署变更到生产环境的前置时间少于1小时，然后低效能组织需要一周到一个月的时间。因此，高效能组织比低效能组织的快速440倍。我们的计算方式，高效能组织算作60分钟，低效能组织算作26940分钟（每周10080分钟和每月43800分钟的平均值）。




* *Note: Last year, our graph incorrectly reported the lead time for low performers, as shown by the dotted line.*




# 稳定性

## 故障恢复时间(MTTR)

高效能组织报告他们的MTTR低于1小时，而低效能组织需要1天到1周的时间。换言之，高效能组织比低效能组织快出96倍，我们的计算方式是我们保守选择高效能组织是1小时，而低效能组织是24小时到168小时（一周）的平均值，前边已经提到，低效能组织的MTTR相比去年有所增加。

## 变更失败率

高效能组织报告他们的变更失败率在0~15%之间，然后低效能组织高达31~45%。我们取两个区间的平均值，即高效能组织为7.5%，低效能组织是38.5%，这意味着低效能组织的变更失败率是高效能组织的5倍，前文已经提到，低效能组织的变更失败率相比去年所有增加。

*Note: The increase in 2016 for high performers is due to a change in how we measured change fail rate. Prior to 2016, we allowed respondents to input a number. In 2016 and 2017, we changed the answers to ranges. We took the average of the range, 0-15 percent, which gives us 7.5 percent.*

# 自动化

去年，作为质量的代表，我们分析了团队在返工和计划外工作上花费的时间。 今年，除了关注返工外，我们还研究了我们的每个团队在如下实践中的手工的工作量：配置管理，测试，部署和变更审批。

我们惊人地发现：当我们将高效能组织与低效能组织进行比较时，高绩效者的手工作业明显更少，并已经自动化：

- 33%的配置管理。

- 27%的测试。

- 30%的部署。

- 27%的变更审批。

自动化是组织的法宝。将更多的工作自动化，高效能组织释放出技术员工去做为组织增加附加价值的创新工作。

惠普的Laser打印机转型项目就是一个很好的例子。固件部门在发布硬件的路上举步维艰。通过进行持续改进计划和投入自动化——包括在自动化测试方面的有效投入——HP LaserJet 最终将开发新功能的时间提高了7倍。

我们在和团队一起进行研究时还发现大家对估算自动化程度并不在行，但是大家对估算手工作业的比例却很擅长。这其实并不奇怪：手工作业很痛苦，大家有很清晰的意识。一旦工作自动化之后，就不再痛苦，大家也倾向渐渐不再关注。

4 *The Amazing DevOps Transformation of the HP LaserJet Firmware Team (Gary Gruver).* itrevolution.com/the-amazing-devops-transformation-of-the-hp-laserjet-firmware-team-gary-gruver

下面你可以看到我们针对手工作业的发现以及高效能组织到底自动化到什么程度。

显然，高效能IT组织报告在所有实践中的手工作业是最少的，在统计上显著说明是自动化程度更高（虽然你可能留意到低效能组织和中等效能组织有差异，但是差异不明显，因此被忽略）。

最让我们惊讶的是，中等效能组织在部署和变更审批流程方面竟然比低效能组织做了更多的手工作业。而且在统计上是明显的。这很奇怪，因为去年我们发现中等效能组织虽然部署频率很高，但是在返工上比低效能组织花费了更多的时间。怎么会这样呢?

**Table 3: Percentage of work that is done manually, by performance group.**

All percentages significantly different among High, Medium, and Low IT performers, except where otherwise noted.

| | High performers | Medium performers | Low performers |
|---|---|---|---|
| **Configuration management** | 28% | 47%[a] | 46%[a] |
| **Testing** | 35% | 51%[b] | 49%[b] |
| **Deployments** | 26% | 47% | 43% |
| **Change approval processes** | 48% | 67% | 59% |

[a, b] *Not significantly different*

根据我们对加速实现自动化的真正团队的了解，我们将手工作业和反馈作为临时阶段。在DevOps旅程的这个阶段，中等效能组织已经自动化了足够多的工作，但是他们也发现了技术债务的高山，并且技术债务在拖他们的后腿。因此中等效能组织利用增加自动化来降低阻碍他们前进的技术债务。其后果是处理这些技术债务会导致团队对变化引入更多的手工控制，增加最终会减缓他们步伐的流程。

虽然增加更多手动控制的想法是可以理解，但我们强烈建议组织抵制这种诱惑。我们对这个阶段团队的指导是将变更审查过程迁移到开发过程的早期，并依靠同行评审和自动化测试，而不是变更审查委员会。

最终，这将彻底消除变革审查委员会的存在，团队可以通过更快速（更可靠）的过程来审查变更。

一旦经过了降低技术债务的阶段，进一步的自动化变得触手可及，团队在DevOps旅程中也可以走向自动化的下一步。


# 衡量组织效能

多年来，一些人一直认为DevOps只能存在于那些独角兽公司里——像谷歌、亚马逊、Netflix及其他高科技企业能聘请成建制的工程师队伍来处理流程，而不仅仅是核心产品。我们已经在很大程度上纠正了这种偏见，目前主流企业的领导者已经认识到DevOps能够给予他们的竞争优势。但仍有一种观点认为相比于非营利组织或政府组织，DevOps对营利性企业更重要。

今年我们的调查结果显示，高效无误的开发和交付软件的能力是所有组织（营利性、非营利性、教育型和非政府组织）的关键差异点及价值驱动器。如果你想要交付价值，无论你怎样衡量它，DevOps是必经之路。

2014年，我们首先发布了关于如何通过IT效能推断组织效能的研究结果。我们发现高效能组织的实际表现超过了他们当初制定的盈利能力、市场份额和生产力目标的两倍。我们经常被问到的问题之一就是如何把它应用于非盈利组织，如军事服务和政府机构、大学和非营利组织。

所以今年，我们探索了IT效能如何影响组织能力，使组织达到更宽泛的目标 —— 也就是超越利润和收入的目标。因为无论你是否想赚取利润，今天的每一个组织都依赖于技术来实现它的使命，并为客户或投资人快速、可靠及安全地提供价值。无论使命如何，技术组织的能力对组织整体的效能都有重要影响。

5 Widener, S.K. (2007). An empirical analysis of the levers of control framework. Accounting, Organizations and Society, 32(7), 757-788.

我们发现高效能组织的实际表现在如下几个方面超过了当初目标的两倍左右：

- 产品或服务的数量

- 运营效率

- 客户满意度

- 产品或服务的质量

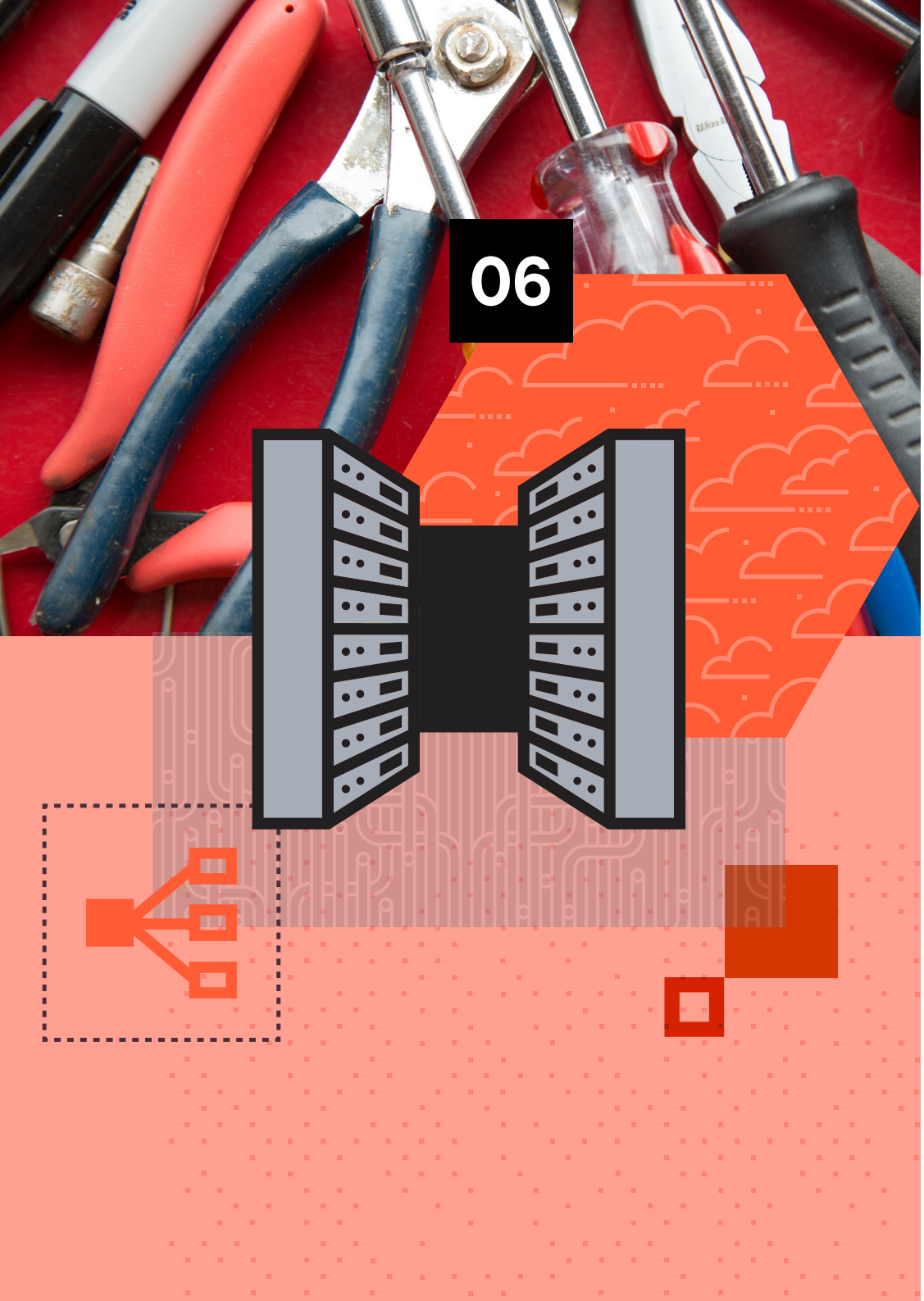
- 实现组织目标和使命目标

对外展示的程度，无论组织是否达到预期成果
非常有趣的是，盈利和非营利组织中的高效能者都是达到或超过原定目标的两倍。这表明DevOps在任何类型的组织、垂直行业中都能够促进使命达成。值得注意的是高效能盈利组织的利润优势——达到或超过目标的两倍——在过去四年一直保持不变。

## 监管领域里的DevOps

举一个在严格受监管的领域实施DevOps实践的例子，请阅读18F的开源平台及服务，cloud.gov. 通过在平台级别处理许多联邦信息系统合规性的要求，大大减少了政府服务的时间和成本。

18f.gsa.gov/2017/02/02/cloud-gov-is-now-fedramp-authorized




# 技术实践

现今，DevOps被视为一种更快交付软件、提高效率和提升竞争力的途径。有的公司希望从研究哪些工具值得购买，来开启他们的DevOps之旅，但是我们认为，相比具体的工具，技术实践更容易帮您走进DevOps。

我们深入研究了多个成功的DevOps组织，并提炼出他们的一些最佳实践——例如：版本控制、持续集成、主干开发，以及自动化。今年，我们将重点关注架构和组织结构，以及它们在开发和交付过程中呈现出的影响力。

# 持续交付

去年，我们发现那些构成持续交付的最佳实践可以作为重要的指标来预测部署错误、IT效能和变更失败率，这些实践包括自动化部署和自动化测试、持续集成和主干开发、对所有生产部件进行版本控制。同样，IT性能本身也能预测组织效能，就如同衡量生产力、市场份额和盈利能力一样。

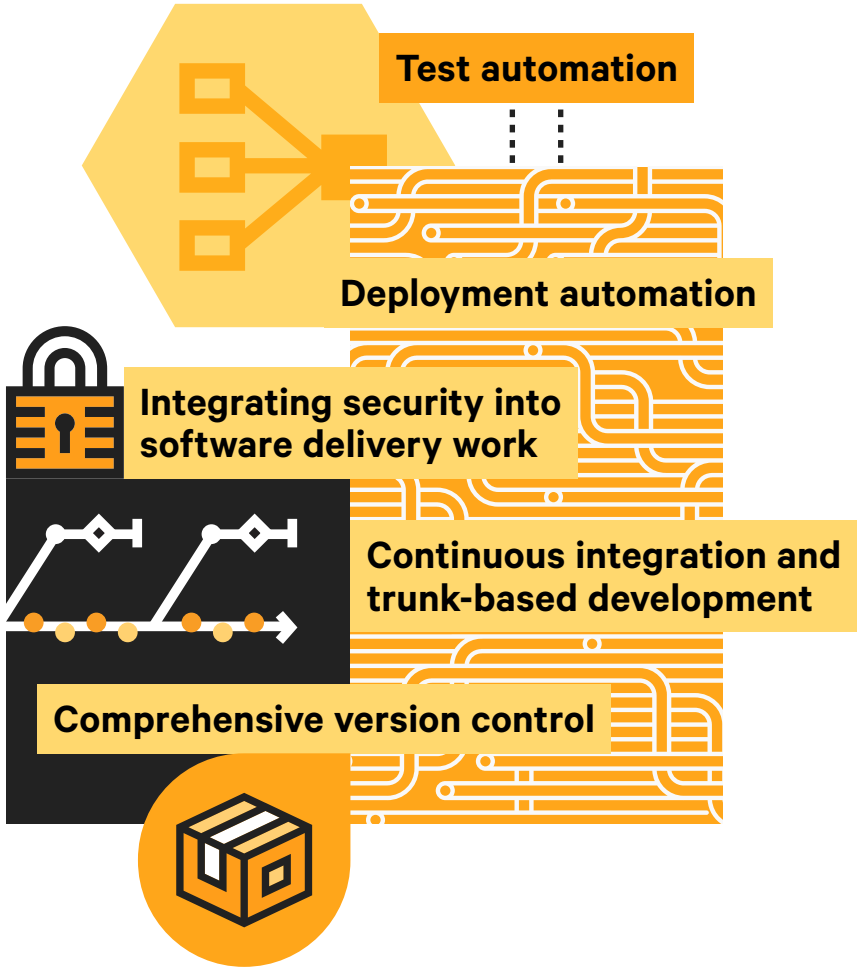
在过去几年中，我们将持续交付视为上述多种能力的集大成者。今年我们希望将视线聚焦于持续交付本身的产出（尽管我们早就意识到持续交付能够提升业绩，如更快上市、更高质量等）。在我们的模型中，我们根据一个团队达成的结果来衡量持续交付：

- 在整个软件交付生命周期中，团队可以按需部署到生产环境或终端用户。

- 将系统质量和部署问题快速反馈给团队中的每个人，并且确保大家对此类问题高度重视并做出反应。

这让我们可以做两个有趣的尝试。首先，我们可以观察做到以上这些会对IT效能和部署困境带来哪些影响。其次，我们可以看到哪些因素对达成这些成果贡献最大，同时考查对其他方面的影响，例如松耦合架构（参考架构章节）。

**Factors that positively contribute to continuous delivery:**




正如我们推测的那样，我们发现以下行为对持续交付有积极影响：全面使用版本控制；持续集成和主干开发；在软件交付中集成安全机制；导入自动化测试和自动化部署。其中，自动化测试所带来的正向影响最为显著。

另外一个有趣的发现是：由于今年我们衡量了持续交付本身的影响，我们认为它有助于明显降低部署错误并提升IT效能。

今年我们希望评估良好的架构实践对持续交付的影响，结果也的确证实了架构对实现高效能的重要性。

# 架构

在过去几年中，我们调查了架构如何与持续交付和IT性能相关联。今年我们想更深入地研究架构，并且实验一些关于有效架构的常规理念。我们发现，松耦合的架构和团队能够显著提高实施持续交付的能力。

我们通过以下两种方式评估服务和组件之间的耦合性：

- 受访者可以脱离集成环境进行测试。

- 应用程序可以独立于其所依赖的应用和服务进行部署或发布。

考虑到商用成品软件（COTS）的场景，我们还询问了受访者的环境是否包括COTS。

在松散耦合的架构中，在不依赖关联组件或服务的变更下修改独立的组件或服务是非常容易的。就组织而言，当团队不需要依赖于其他团队就能完成他们的工作时，就可以称之为松耦合团队

在2015年，我们发现，高效能团队比中低效能团队更倾向于采用松散耦合的架构。无论是他们正在开发的主要应用程序或服务，还是他们必须与其他团队进行交互的服务，都是如此。高效能团队在新系统和历史遗留系统均采用松耦合架构，包括商用成品软件（COTS）、嵌入式系统、用户安装系统和服务器端系统。

今年我们把研究方向基于两个新的假设进行拓展：

1. 具备自主工具决策和实施能力的团队有助于提高IT能力。

2. 松散耦合、良好封装的架构可以驱动IT能力。

对于第一个假设，我们发现与仅能使用集团授权工具的团队相比，可以自主决定使用哪些工具的团队在持续交付方面做得更加出色。他们能够根据自己的工作方式和具体执行的任务做出灵活选择。因为没有人比自己更了解哪种方式最为有效，所以自主工具选择有助于达成更好的结果并不奇怪。

我们的第二个假设也同样得到了证实，它验证并扩展了我们早期的研究。在具有强大的IT和组织能力的团队中，良好的系统架构设计使交付团队可以在不依赖于其他团队的额外工作（资源和审批）下进行测试、部署和更改系统，并减少了变更过程中的反复沟通。因此我们将架构和团队二者均称之为松耦合架构。

计算机程序员Melvin Conway最先提出了系统架构和通信带宽之间的联系，他说

“设计系统的组织，最终产生的设计等同于组织之内、之间的沟通结构。”

我们的研究支持所谓的“逆康威法则”，其中规定组织应围绕团队边界进行架构，以确保团队能够从设计到部署独立完成工作，而不需要团队之间的高带宽通信。

达成这一策略的架构方法包括使用限界上下文（bounded context）和API，它们作为解耦大型系统的一种实践，从而实现更小、更松散的单元。架构还应能够使用测试替身和虚拟化来隔离测试服务或组件。就像任何真正的微服务架构一样，面向服务的架构应该能够实现这些结果。但是，在实施这些架构时，您必须对这些结果非常严格。在现实生活中，许多所谓的面向服务的架构不允许彼此独立测试和部署服务，所以也不能使团队达成更好的效能。

除了自动化测试和自动化部署，以下实践在持续交付过程中尤为重要

- 研发团队可以自由的进行大规模系统设计变更

- 大规模系统设计变更可以独立于其他团队，不会给其他团队带来大量工作

- 无需同外部人员频繁的协调沟通

- 按需部署和发布产品或系统，同其他依赖系统解耦

- 内部完成大部分测试，无需强依赖集成测试环境

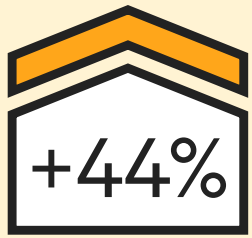
- 无需停机，随时完成部署

## 质量与安全

去年我们引入了质量的概念，来评估持续交付实践队软件质量带来的影响。由于质量难以度量，所以我们采用了计划外工作和重复工作作为指标。我们发现高效能组织在计划外工作和重复工作上花费的时间少22%，同时可以将29%的时间投入新功能和代码的开发之中。今年，我们很欣喜的发现这个比例进一步得到了提升，即21%的时间节省和44%的有效工作投入。

去年我们发现高效能组织在补救紧急安全缺陷方面可以减少50%的时间成本，同样的我们在今年继续跟踪这项工作的比例，毋容置疑的是，我们应该在更加早期频繁的引入安全质量方面的关注。


**less time spent on unplanned work and rework**




**more time spent on new work**

# 主干开发

去年我们调查了基于主干开发在持续交付过程中所起到的作用，根据我们的经验，在高效能研发团队中，相比长期存在的特性分支，基于主线的小批量研发分支更加受到欢迎，行业中的很多先驱者倾向于把工作置于分支上。去年的调查结果表明，以下开发实践可以显著帮助软件交付变得更加高效：

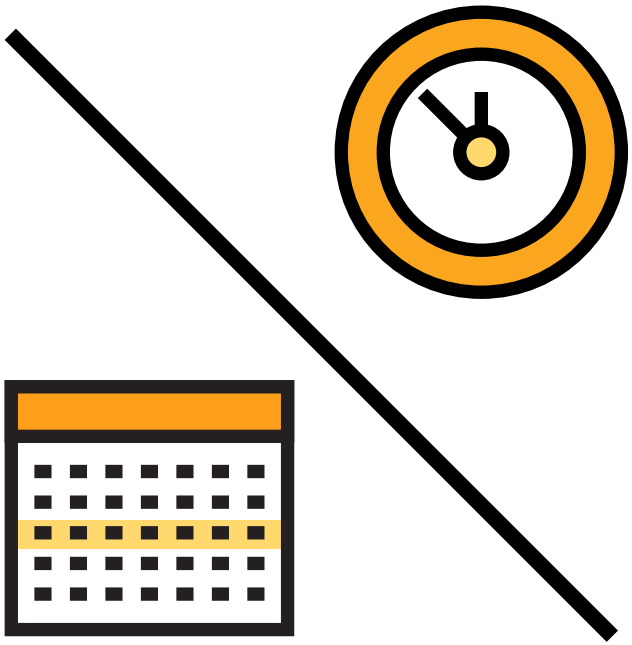
- 每天向主干合并一次代码

- 让分支生命周期尽量短（少于一天）

- 同一时间少于三条的活跃分支

我们同时发现没有代码锁定周期的研发团队有更好的软件交付效率（以上方法都可以支持没有代码锁定的工作方式）

不管有多么丰富的证据表明基于主干开发的实践有助于提升软件交付效率，很多已经适应了GitHub推荐的工作流的研发人员仍然处于观望状态。GitHub的模式推荐工作在分支上，并且周期性的合并主干。工作在短分支上并且至少每天向主干合并一次同当前已经普遍接受的持续集成思想一脉相承。有这样一种说法，只要研发团队不维护长分支，那么这个分支策略就足够有效，可问题是如何界定哪些属于长分支呢?



High performers typically experience branch life and integration lasting hours, versus days for low perfomers.

为了解决这个问题，今年我们引入了额外的分析想要来证实研发效率和主干开发模式之间的关系，以及他所带来的差异。我们深入调查了分支在合并回主线之前的存在周期，以及合并这些分支所需要的时间成本。我们希望能够找到高，中，低效能研发团队之间是否存在显著的差异，以下是我们发现的一些点：

- 高效能研发团队拥有最短的集成周期和分支存活时间，普遍持续若干小时

- 低效能研发团队拥有最长的集成周期和分支存活时间，普遍持续数日

- 这些差异从统计看来非常显著

这些结果印证了去年的结论，当研发团队使用短分支和短分支集成周期时，会显著提升软件交付相关的IT效能

通过这些发现，我们可以得出一个核心实践方法，即避免超过一天的分支存在，如果你需要花费更久的时间来完成分支合并，这就是一个明显的信号，是时候回过头来重新检视当前的方法和架构了。

07

# 精益产品管理

在去年的产品管理调研中，我们将精益产品管理模型划分为两方面的能力：

- 把大的工作分解为小批量工作，并在整个交付流程中可视化这些工作的流动

- 收集、传递和落实用户的反馈

今年，我们扩大了模型的研究范围，研究一个关键的敏捷原则对模型的影响：授权研发团队无需审批即可在开发过程中新增和修改需求规格。

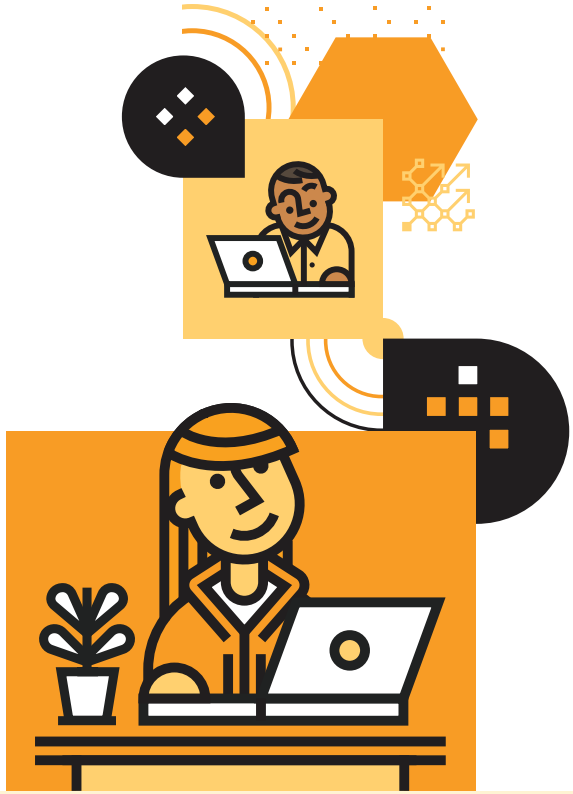
# 小步快跑和收集反馈

去年我们已经认识到一些精益产品管理的实践能够预见到更高的IT效能同时降低部署的困难。这些实践包括把大工作分解为小批量的工作；构建原型或最小化可行性产品（MVP）；并积极寻求客户反馈作为产品设计的输入。我们认为：要是你想在精益产品管理方面取得成功，那么你应该在软件交付方面有强大的基础。

今年，我们倒转了模型，发现IT效能也可以预见到精益产品管理实践。改进你的软件交付流水线可以提高你的小批量工作能力，同时也可以整合客户反馈的处理。如果我们把这两年的模型结合起来，就成为了相互促进的模型，或者通俗的说，是一个良性循环。

在软件开发的组织中，小步快跑的交付能力是特别重要的，因为它允许你使用如A/B测试等技术快速收集用户反馈。值得注意的是，采用实验方法做产品开发的能力高度依赖于持续交付技术实践。你可以在技术实践章节中了解这实践方法。

Technical practices section.



## 怎样培养精益的思想?

你在思考MVP时，思路对吗?
可以阅读 Henrik Kniberg的博客：

blog.crisp.se/2016/01/25/henrikkniberg/making-sense-of-mvp

# 给开发团队授权

许多开发团队在自称为敏捷的组织中都必须遵循别的团队定下的要求。这种限制会产生一些严重的问题，导致产品不吸引客户，不受客户喜爱，也不能达到预期的商业目标

敏捷开发的目的之一就是在整个开发过程中寻求客户参与，包括早期阶段。这样可以让开发团队收集重要信息并传递到开发的下一个阶段。如果开发团队未经允许不能根据他们的发现来变更需求，他们的创新能力定会急剧下降。

我们的分析显示，团队在开发过程中尝试新想法并新增或修改需求的能力（不需要团队外部人员批准）是预测组织效能的重要因素，包括盈利能力、生产力和市场占有率。

**我们并没有建议你让开发人员随便开发**

实现高效的前提，是赋予权利必须和我们在这里讨论的其它能力相结合：小步快跑的能力；可视化每个人交付工作流的能力；以及把客户反馈纳入到产品设计中的能力。这样可以确保你的团队已经能够很好地了解、理解、选择他们所做出的的设计、开发和交付的工作，并根据反馈来持续优化这些工作。这增加了他们的想法和产品的功能能够吸引客户的可能性，同时对整个组织也是十分有价值的。


# DevOps与商业成品软件

业界流传着这样一个说法：“我们的环境不适合DevOps，因为我们绝大多数使用的都是商业成品软件”。然而这一点我们在2015年的报告中就予以了澄清：

无论你的应用程序是全新开发的还是既存的，只要他们在架构设计时考虑到了可测试性和可部署性，那么高效能就并非遥不可及。对于持续交付，我们发现系统类型的差异并不是决定性因素，无论是参与型/记录型系统，通用/定制系统还是全新开发/既存系统，只要架构设计方向正确，持续交付都可以覆盖。

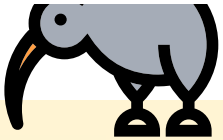
当然，说可以将 DevOps 的做法和原则应用于商业成品软件（COTS）要比实际这么去做简单的多。诚然很多关于如何开发软件的原则是不适用于 COTS 环境的。但是应用一些高阶的部署和测试实践带来的收益，绝对会加快整体组织速度，提升用户服务质量，并提高 IT 人员的生产力（同时也帮助他们学习新技术，这对于维持和追赶技术变革的组织能力非常重要）。


关键的第一步是了解各项目之间实用与战略的不同，并确保使用不同的方式对待它们。Martin Fowler 通过底层功能是否具有商业优势来区分实用和战略项目。他认为如果没有区别，最好使用现有产品而不是构建一个定制解决方案。许多人错误的将实用软件当做定制解决方案，花力气去定制软件，而不是调整软件的业务流程。

如果你把实用工具当做具有战略意义的工具一样对待，那么最终会两头吃亏。你将得到一些定制化的黑盒工具，维护和测试很昂贵，升级也很痛苦，并且还难以以迭代的方式进行改进。测试、部署和运维，各软件服务生命周期的自动化也几乎不可能实现 —— 而自动化正是 DevOps 实践的核心。

相比定制化你的商业成品软件以适应可能僵化了的业务流程，不如考虑更改业务流程，以帮助提升整个系统的最佳效能。在很多情况下，修改业务流程的成本要低的多，而且也可以在使用商业成品软件时更接近其本来的样子。在任何情况下，定期的重新审视你的过程都是有价值的。

应该允许违背技术组织的意愿直接拿一些其他人造好的东西来用。很多技术人员认为，“我们自己做的会更好。” 但是，如果可以谦虚的接受一些他人造好的轮子，您的组织就可以利用所有精力和才能来做些真正重要的事情：采用数字创新来推动您的商业进步，并赢得真正的竞争优势。



### TelstraClear 长达一年的升级

TelstraClear 是一家位于新西兰的电信运营商，拥有大量针对其服务平台的定制化软件，这使得他们的 IT 团队难以对其系统进行升级。

For more details, read Keep it simple, stupid: TelstraClear.

# 结论

过去6年，通过调查IT专业人士和编写DevOps报告，我们的团队取得了IT效能、技术实践、文化规范和组织效能之间关系的突破性的发现。我们解开了很多DevOps的秘密，我们非常惊喜的发现DevOps实践取得很多非凡的成果，甚至超出了我们的预期。同时我们通过数据验证我们有一些假设是错误，我们也学到了很多。

在我们的研究中，有一件事被证明一直是正确的：因为几乎每家公司都依赖于软件，IT效能成为每个组织的核心。并且IT效能会受到很多因素的影响，包括领导力，工具，自动化和持续学习与改进的文化。

我们希望这份报告能帮助你识别你可以改进自己IT、业务流程、工作文化、学习循环的领域，我们欢迎大家的反馈：devopssurvey@puppet.com

# 统计方法

Devops现状调查在过去六年中不断进步。我们当前严谨的研究方法提供了原始报告数据，并提供了IT绩效、组织绩效、技术实践、文化规范和管理之间的统计关系。在本节中，我们将描述我们的分析方法、如何征集受访者、以及如何设计问题、模型和结构。我们欢迎任何人对调查方法提出问题，请随时发邮件至：devopssurvey@puppet.com。

### 我们如何使用聚类分析确定高、中、低的IT绩效

这个成熟模型的根本性缺陷：他们概述了团队和组织可以设置固定的绩效水平，但是在“到达”这个水平的时候，它就可能因为你已经达到目标阻碍你改进工作，失去领导的支持。通过不断的基于真实数据重新评估团队的绩效水平，团队可以了解到行业的发展情况。

那么如何使用数据驱动的方法来确定我们的IT绩效组呢？我们使用聚类分析[7]，通过团队的部署频率、变更交付时间、平均修复时间、变更失败频率这些指标来分组。通过聚类分析，发现不同的分组和团队存在明显差异。高绩效团队是相似的，他们和中低端同行存在差异。低绩效IT团队相似，但又和高绩效IT团队存在差异等。连续第四年，证实了高、中、低绩效IT团队之间存在差异。（这也是另外一个成熟模型的缺陷：他们很少或者不进行分组）

聚类分析的好处是它允许我们在不知情的情况下对每个群体的价值进行分类，只需要对比的值存在显著差异。也就是说我们让数据告诉我们应该有多少组，以及如何如何使用平均值来评定他们。在表2我们对高、中、低绩效者的IT绩效和组织绩效做了数据报告。

---

7 "Cluster analysis," Wikipedia

## 目标人群和抽样方法

我们这次调查的目标人群是贴近或者从事IT行业的人和领导者，尤其是熟悉DevOps的人。 因为没有这些人的详细信息 - 我们可以描述他们，但不知道他们在哪里，如何找到他们以及他们有多少人 - 我们使用雪球抽样来获得受访者。 这意味着通过电子邮件列表，在线促销和社交媒体推广了调查问卷，并且还要求人们在他们的社交网络中分享调查，将样品像雪球一样增长。 我们的示例可能仅限于熟悉DevOps，或者类似的进行部分工作的的组织和团队。

## 创建潜在结构

一旦确定了目标人群和抽样方法，我们开始确定调查中要包括哪些问题的困难工作。 为此，我们首先要确定想要测试的假设。为了增加研究的严谨性，我们引用了现有的研究和理论。 制定了假设和结构，尽可能使用以前验证的结构。 当需要创建新的结构时，我们非常仔细地根据理论，定义和专家意见撰写它们。 然后，我们采取了进一步的措施来澄清意图和措辞，以确保从最终调查中收集的数据可靠性和有效性。8为了创建结构，我们使用了Likert-type9问题，它提供了一定的灰度，而不是 非黑即白，是或否，真假问题。 Likert型问题也可以进行更高级的分析。

## 统计分析方法

- 聚类分析。使用数据驱动的方法，进行分层聚类分析，推导出IT绩效概况。在这种方法中，同组中的那些在统计上彼此相似，并且与其他组中不同。对于IT绩效档案，它们基于吞吐量和稳定性的IT绩效行为类似（或不相似）： 部署频率，前置时间，平均恢复时间以及并变更失败率。
- 测量模型。在使用结构进行任何分析之前，包括相关性，回归和偏最小二乘法10（PLS） - 对这些结构的有效性和可靠性进行测试。结构通过检验收敛有效性，11判别有效性，12和可靠性，因此表现出良好的心理测量13属性。
- 回归分析。当引入预测或影响，并没有明确提及PLS时，使用简单的线性回归14。
- 结构方程模型。使用PLS分析测试结构化方程模型（SEM）15，这是基于相关性的SEM，非常适合于探索性分析。使用SmartPLS 3.2.6。图1所示的所有路径均为p <.001，IT绩效除外。精益和IT绩效？组织绩效，p <0.05。
- 研究设计。本研究采用横截面理论为基础的设计。

8 We used Churchill's methodology: Churchill Jr, G. A. "A paradigm for developing better measures of marketing constructs," Journal of Marketing Research 16:1, (1979), 64–73.
9 "Likert scale," Wikipedia
10 "Partial least squares regression," Wikipedia
11 "Convergent validity," Wikipedia
12 "Discriminant validity," Wikipedia
13 "Psychometrics," Wikipedia
14 "Linear regression," Wikipedia
15 "Structural equation modeling," Wikipedia

# 致谢

作者们在此想对几位外部评审专家表示感谢，他们对于这一年中新进展的度量项作出了投入与指导。

关于架构的度量项，我们向Neal Ford、Martin Fowler、Mik Kersten、Sam Newman和Rebecca Parsons表示感谢。

关于团队实验，我们要感谢Amy Jo Kim和Mary Poppendieck。

作者们还想对Aliza Earnshaw表示感谢，她在过去四年中在DevOps现状调查报告上作出了细致入微的编辑工作；没有她的细心工作，报告就不会具有一致的风格。

# 作者

Nicole Forsgren博士是DevOps研究与评估(DORA)协会的CEO和首席科学家。她是知识管理、IT落地与影响和DevOps等方向的作者与研究者，被广为人知的是，她也是迄今为止最大范围内DevOps领域研究的首席研究员。她曾经是教授、系统管理员、硬件性能分析师。Nicole曾经获得过各种公共的和私有的研究基金（投资者包括NASA和NSF），她的工作被各种不同媒体机构专门报道过，以及出现在同行评审的刊物和会议上。她拥有信息系统管理方向的博士学位和审计方向的硕士学位。

Jez Humble是《The DevOps Handbook》、《Lean Enterprise》以及获得过Jolt大奖的《Continuous Delivery》等书籍的作者之一。他的职业生涯中曾往返穿梭于三个大洲中的各种不同大小的公司，帮助它们改进代码、架构和产品开发，最近则在18楼为美国政府工作。他目前正在研究在其创业公司DevOps研究与评估协会中如何建立高效能团队，也在UC Berkely任教。

Gene Kim是一位多次获得大奖的CTO、研究者和作者，从1999年以来就在研究高效能科技组织。他曾经参与共同编写的书籍有《The Phoenix Project》与《The DevOps Handbook》。他是DevOps Enterprise Summit的主要组织者，也是DevOps研究与评估协会的共同创始人之一。

Alanna Brown是Puppet的产品营销总监，她在这里帮助营销业务和团队成长已超过五年时间。她于2012年酝酿发起了第一本DevOps现状调查报告，从那时起就开始负责相关的调查与报告。除了领导DevOps研究之外，Alanna也负责驱动DevOps、云和安全方面的合作伙伴的进入市场战略；培养客户关系以促进DevOps的落地；并且带领多个跨职能团队，让Puppet的产品走向市场。

Nigel Kersten是Puppet的首席技术战略官，他在这里曾经担任过多种角色，包括产品主管、CTO与CIO。他加入Puppet之前是来自Google位于加利福尼亚州山景城的总部，在那里他曾负责世界上最大的Puppet部署项目之一的设计与实现。Nigel曾深度参与Puppet的DevOps创始活动，并且经常就DevOps在企业与IT组织转型的落地等话题在全球各处进行演讲。



**About Puppet**

Puppet is driving the movement to a world of unconstrained software change. Its revolutionary platform is the industry standard for automating the delivery and operation of the software that powers everything around us. More than 37,000 companies — including more than 75 percent of the Fortune 100 — use Puppet's open source and commercial solutions to adopt DevOps practices, achieve situational awareness and drive software change with confidence. Headquartered in Portland, Oregon, Puppet is a privately held company with more than 530 employees around the world.

Learn more at puppet.com.

**About DevOps Research and Assessment**

DevOps Research and Assessment (DORA), founded by Dr. Nicole Forsgren, Jez Humble and Gene Kim, conducts research into understanding high performance in the context of software development, and the factors that predict it. DORA's research over four years and more than 25,000 data points serves as the basis for a set of evidence-based tools for evaluating and benchmarking technology organizations, and identifying the key capabilities to accelerate their technology transformation journey.

Learn more at devops-research.com.