利用自动化 APM 加速 CI/CD，提升应用性能

INSTANA
IBM 旗下公司

由此起步

目录

01
执行摘要

持续集成和持续交付 (CI/CD) 已成为大多数组织的目标。同时，Docker 和 Kubernetes 等现代技术迅速成长，已经在应用环境中得到广泛使用。这些趋势使得应用及其基础架构变得越来越动态，不断变化以满足更高的可伸缩性需求和快速发展的应用功能。

然而，典型的组织仍然依赖于在 CI/CD 成为应用交付模型之前设计的性能监测技术。这些解决方案需要 IT 人员的大量手动工作，导致组织难以有效地利用人员时间，控制 IT 支出，获得对应用和基础架构有意义的可见性。

只有采用自动化性能管理策略，组织才能突破应用挑战，充分实现 CI/CD 驱动的应用环境提供的机会。通过在每个性能管理层采用自动化，组织能够减少成本，同时改善成果。

02
变化成为新常态

说到我们生活在一个快速变化的世界，这可能听起来是老生常谈。然而，从 IT 组织的角度来看，如今的应用环境变化得极为快速。

我们与客户互动时，他们的要求明确偏向于速度。公司交付定制业务应用的速度越快，IT 交付的价值就越高。实际上，这使得 IT 的 ROI 增长了！

这一简单的现实正在推动新方法和新技术的采用，特别是 CI/CD，这种自动化操作能使交付速度更快，业务能力更强，并最终提高定制应用的质量。

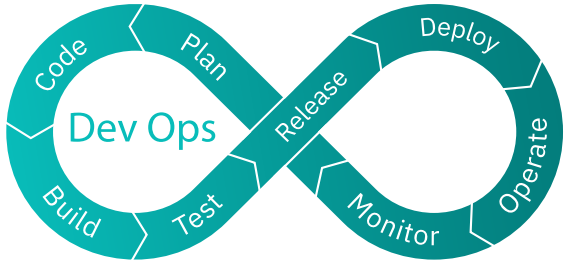


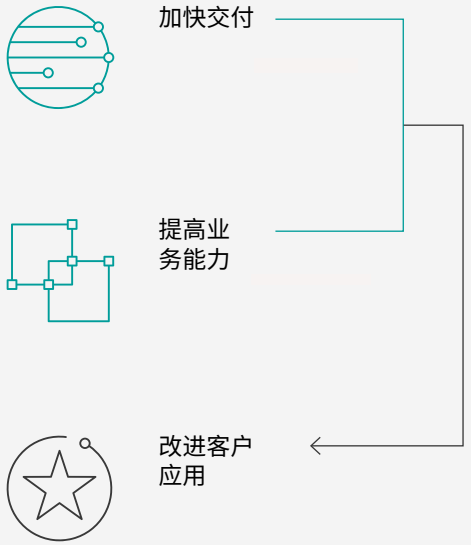
图 1. 持续集成和交付管道

CI/CD 循环中的最后一个阶段是监测。因此，结果就是，监测业务应用的自动化程度和无缝性越高，越容易完成 CI/CD 循环，并重新启动改进应用的循环。

在动态应用环境中，唯一不变的是变化。应用的结构在每一层持续变化。新主机联机后始终隐藏，而容器使得供应具备甚至更强的动态性。开发人员持续构建和供应全新的 API 和服务，而无需检查操作。即使是应用代码，也会因为随时进行改进或错误修订而变化。团队越接近 CI/CD，变化发生得就越频繁。

结果，性能管理配置、监测仪表板、依赖关系映射和警报规则必须能够自动发展，以跟上它们监测的环境。否则，IT 团队就会对他们管理的环境缺乏必要的准确了解，这将使组织面临失败的巨大风险，可能影响到用户。

CI/CD 自动化支持：



复杂依赖关系
这些持续变化会影响不同组件之间的实际依赖关系。任何特定服务都依赖于唯一的垂直软件堆栈以及来自其他服务的数据或处理。

为什么始终了解依赖关系很重要？故障诊断！在复杂的环境中，了解问题的根本原因是一种依赖关系分析活动。是什么造成了缓慢或错误的请求？请求横跨许多框架和基础架构上的许多服务，所以了解每个请求的结构依赖关系对回答该问题是非常重要的。但是，正如我们在上一段中详细说明的，依赖关系会不断变化！

试图手动解释此类依赖关系是根本行不通的，特别是当依赖关系因代码部署或基础架构缩放而快速变化时。即使人工操作员成功地在某一时刻手动映射了依赖关系，他们的映射也会很快变得过时。此外，手动依赖关系解释是巨大的资源消耗，需要最好的工程师来完成。

为什么始终了解依赖关系很重要？
故障诊断！

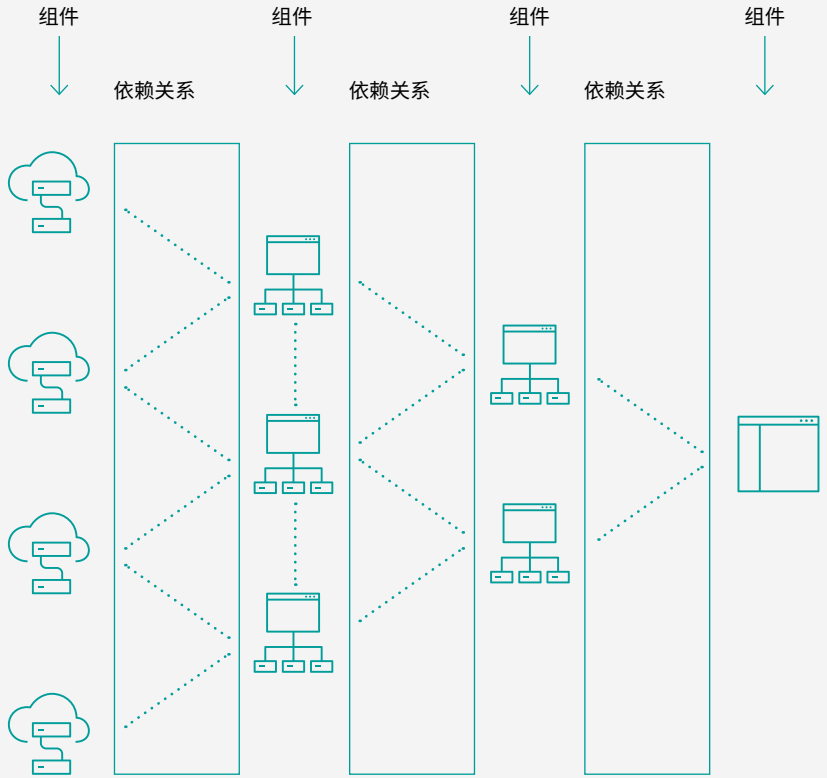


图 2. 任何特定服务（如硬件、软件或代码）都依赖于唯一的软件垂直堆栈，以及来自其他服务的数据或处理。变化会扩散蔓延，可能会加重整个环境中的问题。

动态应用的兴起

对于应用开发，对速度的需求推动了广泛采用相应技术来支持快速构造和交付新服务。

特别是，持续集成和交付管道被视为支持速度和质量的主要流程。此外，为了扩充这一流程，人们正在研究和采用新的体系结构，如容器、微服务、无服务器计算和 Kubernetes 编排。

随着越来越多的工作负载迁移到动态技术 - 一个月内 Docker Hub 上有 80 亿次拉取，自 2014 年启动主数据中心以来共有 1300 亿次拉取[1] - 这就是应用环境中的变化速度和规模的新现实。

IT 行业正在围绕 CI/CD 术语和方法进行整合。同时，公有云平台现在将 Kubernetes 管理的容器处理作为服务提供，可轻松集成到 CI/CD 交付管道中。显然，在应用构造和交付领域中，发生了重大变化。

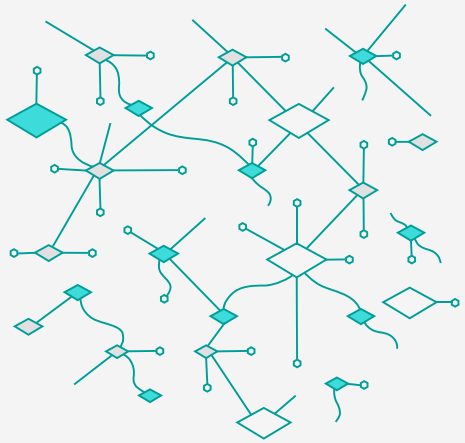


图 1. 变化会影响不同组件之间的实际依赖关系，可以在几秒钟内就使一个系统失势。

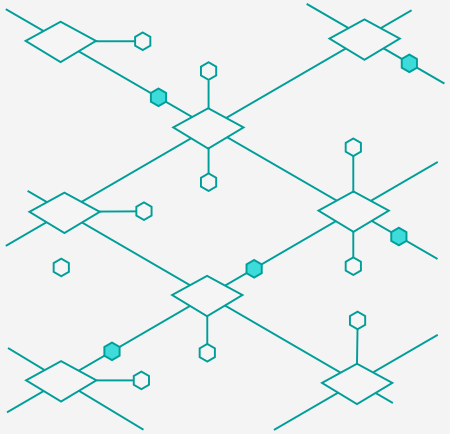


图 2. Instana 会自动摄取可观察性指标，追踪每个请求并对每个流程进行概要分析。它突破了环境复杂性，向您展示一切是如何在环境中相互配合的。

03
为何手动性能管理会失败

手动性能监测面临的挑战
因为传统的应用性能监测 (APM) 和监测工具不是为动态应用而设计的，所以出现了一组新的开放式源代码技术，帮助开发团队通过手动编码来手动设置自己的监测。无论是提供性能指标、跟踪应用的路径还是公开代码的其他详细信息，这些开放式源代码监测工具都会在生产性能监测方面带来一组独特的挑战。

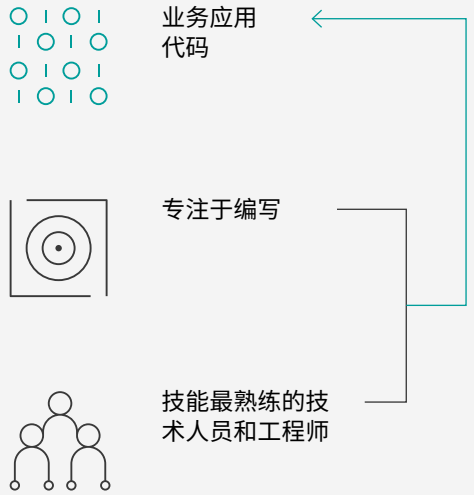
要手动监测应用的性能，工程师必须完成以下任务：

– 编写数据收集器
– 执行手动代码跟踪，以跟踪分发的请求
– 配置数据存储库
– 识别和指定依赖关系，通常通过反向工程来实现
– 选择要关联的数据
– 构建仪表板来可视化相关性
– 配置警报规则和阈值

最大的问题是，执行这些任务的人员通常是公司中技能最熟练、收入最高的技术人员和工程师。

简而言之，虽然开放式源代码意味着简化的性能管理，但需要太多的手动任务，这会导致部署缓慢，成本增加和额外的员工需求。而且这是您考虑编写监测代码（而不是业务应用代码）的宝贵开发人员的机会成本之前的情况。

目标是让公司中技能最熟练、收入最高的技术人员和工程师编写业务应用代码，而不是监测代码。



自行构建所需的真实工作

过去，手动设置监测系统并没有什么问题，因为无论是应用代码还是应用基础架构（中间件、应用服务器等）都不会经常变化。IT 将供应一个框架，设置其 IP 地址，装入一些软件，设置监测，然后多年不再管它。

基于传统技术构建的应用环境也不太可能大规模迅速波动；在任何给定时刻运行的应用实例数和主机服务器数通常都不会变化。因此，在静态环境中手动配置的技术堆栈不会影响到组织监测和管理性能的能力。

但是，当工作负载迁移到基于容器等技术和基于 CI/CD 等方法的动态环境中时，手动性能管理策略和自行构建解决方案就会瓦解。这是真实情况，原因有多种。

规则恶化

环境持续变化时，监测工具用于确定应用和服务是否正常运行的规则也需要持续变化。如果不变（您依赖于手动干预来更新规则时，很可能发生这种情况），规则将迅速恶化。例如，部署新服务时或编排器迁移工作负载时，只有更新规则之后，运行状况检查规则才能够准确解释环境依赖关系。

如果不手动更新规则，监测警报和洞察将基于过时的配置。这种恶化削弱了可视性，增大了发生基础架构或服务故障的风险。

手动监测阻碍了速度

编写跟踪和监测代码等任务太耗费时间。每当应用代码或环境体系结构发生变化时，手动解释监测信息并手动更新性能管理规则也同样很费时。

简而言之，如果没有自动化，人工无法支持快速变化的环境。试图手动管理性能将大大减缓应用发布周期。对于企业来说，这意味着未妥善使用 IT 员工所代表的昂贵专业知识。

对于寻求与快速变化的用户需求保持同步的组织来说，加速软件交付是最重要的。业务需求只会增加这种需求，所以展望未来，应用肯定会变得更加动态。

04

实施自动化 APM

既然手动监测是如此费力，那么这就提出了一个问题：为什么不是所有组织都已实现了自动监测？这实际上有三方面的原因：

– 直到最近，完全自动监测技术才出现。

– 软件工程师倾向于针对问题编写自己的解决方案代码，这导致监测即代码方法在短期内有效，但不可维护或不可伸缩。

– 团队尝试使用先前投资的现有监测工具，希望它们能在新的高速环境中工作，但遗憾的是，它们并不能。

考虑到这些原因，下面我们来看看应该成为完全自动化 APM 解决方案组成部分的功能是什么：

– 自动发现和监视完整基础架构和应用堆栈。这是 CI/CD 流程的一个关键元素，但如今缺少此元素，导致发布流程变慢。

– 实时复杂依赖关系映射，并针对任何变化自动更新。这对于执行根本原因分析来迅速解决性能问题至关重要。

– 自动、快速识别性能问题。基于自动规则配置和监视快速识别性能问题，是最大限度减少误报和避免应用交付延迟的唯一方法。

– 规则配置和警报，使用机器学习和 AI 来为正常的应用行为建立动态基线，然后根据这些基线来识别异常情况。这就无需人工执行的繁琐配置、监视和数据解释，同时最大限度地减少了误报警报。

– 自动监视设置 - 代理程序部署、代码检测、基础架构发现等。尽可能收集最多的数据，同时最大限度减少人工管理员收集数据所需的工作。

现有性能管理工具和现代开放式源代码监视工具都缺少这种自动化功能。虽然确切的工作不同，但两组工具都需要太多的手动配置、编程、设置和管理来优化监视。

05

最大限度地提高 CI/CD 敏捷性

CI/CD 的敏捷性是否达到预期？成功管理动态应用环境并非仅仅是自动监视。

IT 团队应该通过设法找到以下问题的答案，不断地评估已成功自动执行所有性能管理任务的程度：

- 在推出新发行版后，需要多长时间才能充分了解应用性能？
- 执行新应用或服务部署后，需要多长时间来更新监视规则？
- 开发人员花了多少时间和工作来编写跟踪代码？
- 每月或每季度错过了多少性能或可用性事件？
- 如何处理警报风暴，即短时间内快速出现的警报流？我们能否有效地响应每个警报，且游刃有余？我们能否快速跟踪警报以找到根本原因，以便多个警报源自同一个底层问题时能掌握这一情况？
- 监视和性能管理流程的自动化程度是否与应用交付管道的其余部分一样？如果不是，那么如何进一步自动化？

无论 IT 团队大小，自动化都是有效性能管理策略的关键所在，这样您就可以实现高速交付新的业务服务。

IBM Observability by Instana 是诞生于微服务、云计算和容器时代的自动化性能管理解决方案，可以帮助您真正基于 CI/CD 的承诺进行交付。

IBM Observability by Instana 专为管理动态 CI/CD 驱动的应用环境而设计，使用 AI 和自动化来交付全面、切实可行的的洞察，而无需手动工作。

06

关于 IBM Instana 公司

IBM Instana 公司为运行复杂、现代的云原生应用的企业提供具有**自动化应用性能监视**功能的**企业可观察性平台**，无论这些应用是在本地还是在公有云和私有云中，包括移动设备或 IBM Z® 大型计算机。

使用 Instana 的 AI 驱动的发现功能来发现混合应用中深层上下文依赖关系，控制现代混合应用。Instana 还提供对开发管道的可视性，以帮助实现闭环 DevOps 自动化。

这些功能提供客户所需的切实可行的反馈，帮助优化应用性能，支持创新并降低风险，进而帮助 DevOps 提高效率，增加软件交付管道的价值，同时满足服务和业务级别目标。

了解更多内容 →

国际商业机器中国有限公司
北京市朝阳区北四环中路27 号
盘古大观写字楼25层
邮编: 100101

美国出品
2021 年 4 月

1 Docker 将主数据中心统计信息汇总在一起，称拉取数超过 80 亿，DevClass，2020 年 2 月 5 日。