



中国软件企业 云上增长实战指南

第三卷：技术卓越

—

从云原生到持续现代化的创新路径

Contents.

《中国软件企业云上增长实战指南》第三卷-技术卓越
从云原生到持续现代化的创新路径

01

CHAPTER

技术栈现代化——摆脱历史遗留的枷锁

- P01 ----- 操作系统现代化：从Windows到Linux
- P01 ----- 数据库现代化：从商业数据库到云原生数据库
- P02 ----- 许可管理：优化现有投资

02

CHAPTER

架构现代化——拥抱事件驱动与无服务器

- P03 ----- 为什么选择事件驱动架构？
- P03 ----- 无服务器：事件驱动架构的完美搭档
- P04 ----- 核心服务：Amazon EventBridge
- P04 ----- 创新加速器：集成生成式AI

03

CHAPTER

部署现代化——应对混合云与边缘场景

- P05 ----- 混合云的驱动力
- P05 ----- 一致性的挑战与解决方案

Contents.

《中国软件企业云上增长实战指南》第三卷-技术卓越
从云原生到持续现代化的创新路径

04

CHAPTER

卓越运营——持续审查与优化

- P06 ----- 可观察性：理解您的系统
- P06 ----- 架构审查：Amazon Well-Architected 框架
- P07 ----- 持续学习与创新

ABSTRACT

摘要

您的SaaS业务已在云上稳健运行，但这是否意味着技术演进的终结？恰恰相反，这开启了一个持续现代化的新篇章。在竞争激烈的市场中，技术领先是保持产品竞争力和业务增长的关键。一个现代化的、解耦的、数据驱动的架构，是您快速响应市场变化、持续交付创新体验的基石。

本白皮书是“中国软件企业云上转型”系列的第三篇，专为成熟的SaaS企业决策者与架构师设计。我们将探讨在SaaS之旅的“Day 2”及以后，如何通过技术现代化，实现从“云原生”到“云卓越”的飞跃。

我们将深入剖析技术栈现代化的三大路径：摆脱商业软件许可束缚、拥抱事件驱动与无服务器架构、以及集成生成式AI。同时，我们也将探讨混合云这一现实场景，并提供一套完整的架构审查与持续优化框架。这不仅是一份技术路线图，更是一份创新催化剂，旨在帮助您构建一个面向未来的、卓越的SaaS业务。

NO.1

技术栈现代化——摆脱历史遗留的枷锁

许多SaaS应用在演进过程中，都或多或少地背负着历史遗留的技术债务，其中最常见的就是对特定商业操作系统和数据库的依赖。这些依赖不仅带来了高昂的许可成本，更限制了架构的灵活性和云原生能力的发挥。实现技术栈的现代化，首先要从摆脱这些枷锁开始。

1.1 操作系统现代化：从Windows到Linux

» 挑战 (Challenge)

对Windows Server的依赖限制了容器化和无服务器等现代云原生技术的选择，并带来了持续的许可费用。

» 路径 (Path)

对于.NET应用，利用Porting Assistant for .NET等工具，将传统的.NET Framework代码现代化为跨平台的.NET Core/.NET 6+。这使得应用可以无缝运行在成本更低、生态更开放的Linux容器环境中。

» 工具 (Tool)

Amazon App2Container可以帮助您自动化地将现有的Java和.NET应用容器化，简化向Linux环境的迁移。

1.2 数据库现代化：从商业数据库到云原生数据库

» 挑战 (Challenge)

传统的商业关系型数据库（如SQL Server, Oracle）在云原生环境下，面临扩展性、性能和成本的多重瓶颈。

» 路径 (Path)

拥抱云原生数据库。根据业务场景，选择最合适的数据库类型：

» NoSQL数据库 (NoSQL database)

对于需要极致扩展性和灵活性的场景，Amazon DynamoDB提供了无限的吞吐能力和个位数毫秒级的延迟。

» 迁移工具 (Migration Tool)

Amazon Database Migration Service (DMS)支持同构和异构数据库之间的在线迁移，最大限度减少停机时间。特别是Babelfish for Aurora PostgreSQL功能，可以让为SQL Server设计的应用，在几乎不修改代码的情况下，直接运行在Aurora PostgreSQL之上，极大加速了迁移进程。

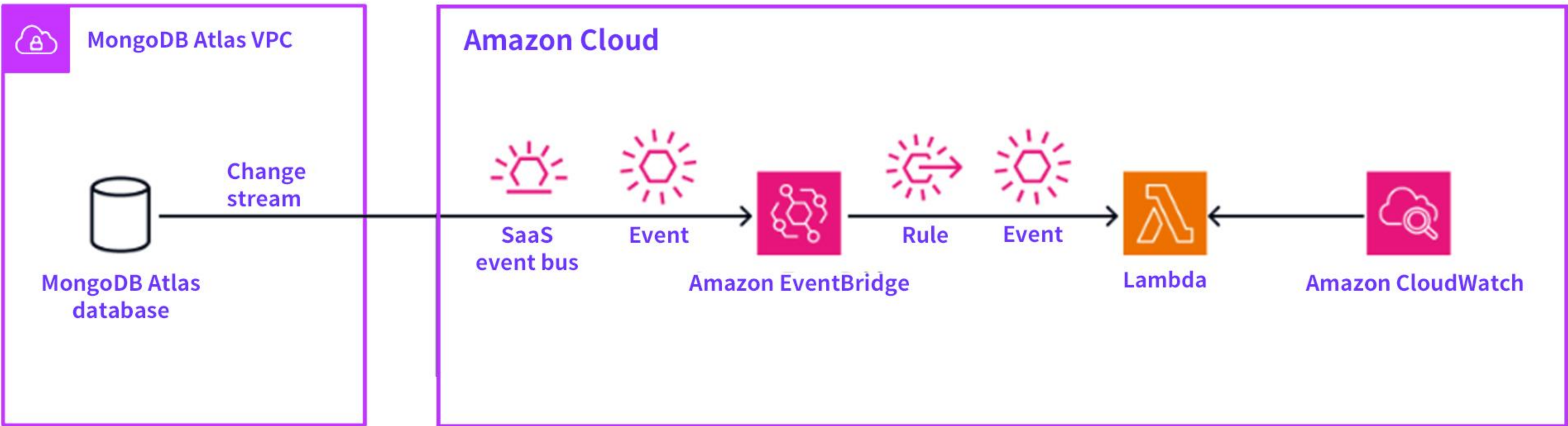
1.3 许可管理：优化现有投资

在彻底完成现代化之前，您仍然需要有效地管理现有的商业软件许可。Amazon License Manager可以帮助您集中跟踪和管理自有许可（BYOL），并通过与Amazon EC2 专用主机的结合，满足那些对CPU核心、物理隔离有严格要求的许可协议，从而在过渡期内最大化现有投资的回报。

NO.2

架构现代化——拥抱事件驱动与无服务器

当您的技术栈摆脱了底层束缚后，下一步就是对应用架构本身进行现代化改造。从紧耦合的请求/响应模式，转向松耦合、高弹性的事件驱动架构（Event-Driven Architecture, EDA），是构建现代化SaaS应用的核心。



附图：驱动架构示例图

2.1

为什么选择事件驱动架构？

事件驱动架构使用“事件”（即状态的变更，如“订单已创建”）来触发和驱动不同微服务之间的通信。它带来了三大核心优势：

解耦与弹性

»

服务之间不再直接调用，而是通过一个中央的“事件路由器”进行通信。这意味着单个服务的故障不会引发连锁反应，整个系统的韧性大大增强。

独立扩展与部署

»

每个服务都可以根据自己处理的事件量独立扩展，并可以独立于其他服务进行开发和部署，极大地提升了团队的敏捷性。

成本效益

»

基于事件的“推送”模式，意味着计算资源只有在有事件需要处理时才被激活，避免了传统轮询模式下的资源浪费。

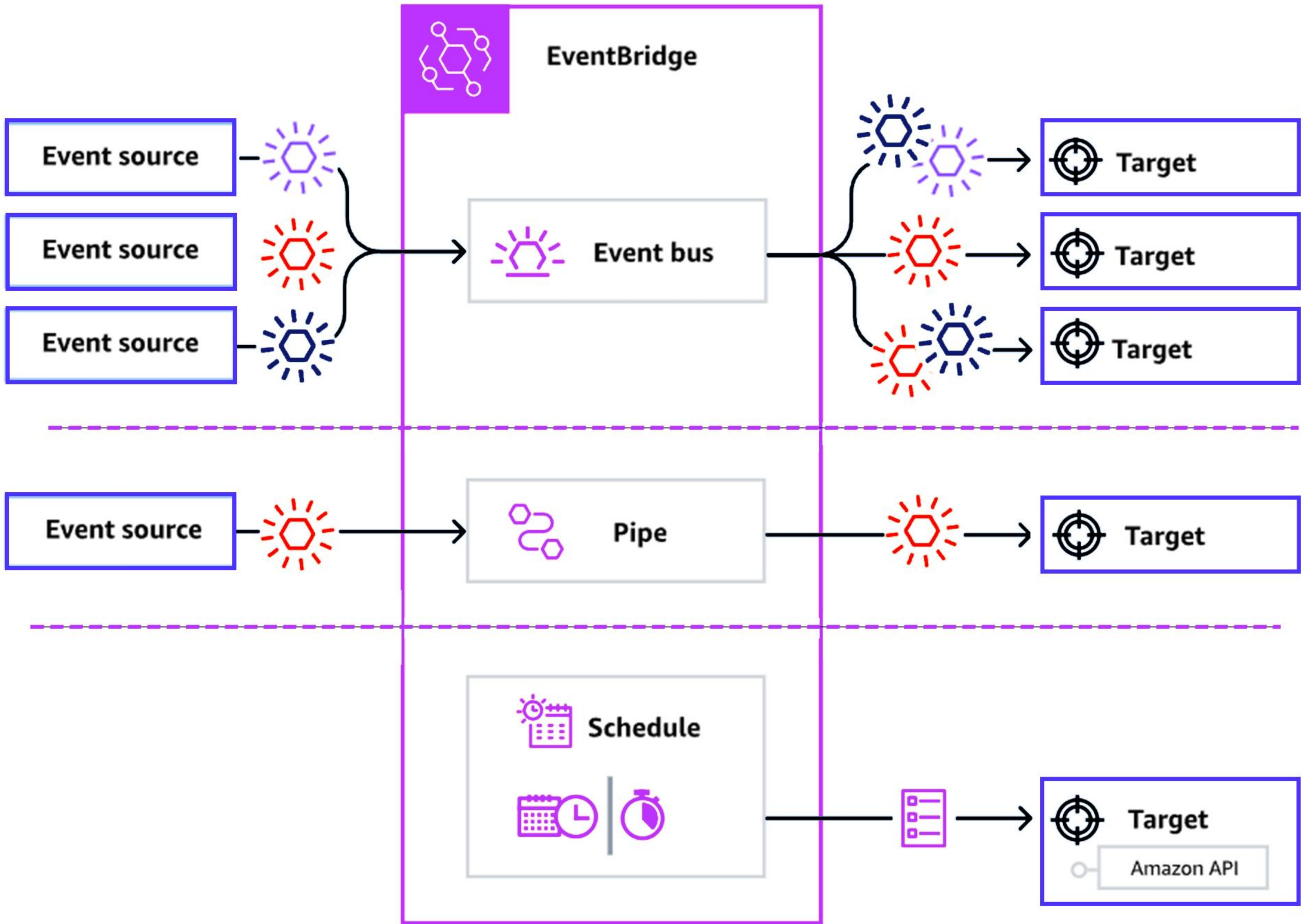
2.2

无服务器：事件驱动架构的完美搭档

Amazon Lambda等无服务器计算服务，是实现事件驱动架构的理想选择。您只需编写处理事件的核心业务逻辑代码，而无需关心服务器的配置、扩展和维护。Lambda的按执行付费模型与事件驱动的按需特性完美契合，能够将成本效益发挥到极致。

2.3 核心服务：Amazon EventBridge

Amazon EventBridge 是一个完全托管的、无服务器的事件总线服务，是构建事件驱动架构的中枢。它可以轻松连接来自您的应用程序、亚马逊科技服务以及第三方SaaS应用的数据，并根据您定义的规则，可靠地将事件路由到Amazon Lambda等目标，极大简化了EDA的构建和管理。



附图：EventBridge示意图

2.4 创新加速器：集成生成式AI

事件驱动架构也为集成生成式AI等前沿技术提供了绝佳的土壤。例如，当一个“新客户支持工单”事件产生时，可以触发一个Lambda函数，该函数调用Amazon Bedrock，利用强大的基础模型自动生成初步的回复建议。

Amazon Bedrock提供了一个统一的API来访问来自多家领先AI公司的基础模型，并内置了无服务器的运行模式，让您可以轻松地将生成式AI能力融入您的SaaS应用，而无需管理任何底层基础设施。

NO.3

部署现代化——应对混合云与边缘场景

并非所有的工作负载都能，或者都需要在云端运行。出于数据主权、超低延迟或现有投资等原因，部分应用组件可能需要保留在本地数据中心或部署在边缘设备上。一个现代化的SaaS架构，必须能够优雅地应对这种混合云（Hybrid Cloud）的现实。

3.1 混合云的驱动力

数据主权与合规	满足特定行业（如金融、医疗）或国家对于数据必须存储在本地的法规要求。
低延迟处理	在数据产生的源头（如工厂车间、零售门店）进行实时处理，减少网络延迟。
渐进式迁移	在漫长的迁移过程中，确保云上和本地的应用能够协同工作。
灾难恢复	将云作为本地数据中心的灾备站点。

3.2 一致性的挑战与解决方案

混合云管理的最大挑战，在于如何确保跨越云端和本地的“操作一致性”。您需要一套统一的工具和API来配置、管理和监控所有环境中的资源。亚马逊云科技提供了一系列服务，将云的体验延伸到您的本地数据中心：

容器管理	Amazon ECS Anywhere和Amazon EKS Anywhere允许您在本地服务器上运行与云端完全一致的容器编排环境，并通过亚马逊云科技控制台进行统一管理。
存储扩展	Amazon Storage Gateway可以在本地部署一个虚拟设备，让您的本地应用可以通过标准协议（NFS，SMB）无缝地访问云上无限扩展、高持久性的存储服务（如Amazon S3）。
统一运维	Amazon Systems Manager提供了一个单一的控制台，用于查看和管理您在云端和本地的所有资源，实现统一的配置管理、补丁管理和操作审计。

NO.4

卓越运营——持续审查与优化

技术演进永无止境。构建了一个现代化的架构之后，如何确保它能够持续保持卓越，并不断为业务创造价值？建立一个持续审查与优化的闭环流程至关重要。

4.1 可观察性：理解您的系统

“你无法优化你无法衡量的东西”。建立一个全面的可观察性（Observability）体系，是持续优化的基础。这需要从三个维度收集数据：

指标（Metrics）

系统性能和资源利用率的量化数据（如CPU使用率、应用响应时间）。Amazon CloudWatch提供了强大的指标收集和告警功能。

日志（Logs）

应用和系统产生的离散事件记录。CloudWatch Logs Insights允许您对海量日志进行高性能的交互式查询和分析。

追踪（Traces）

在一个分布式系统中，完整记录一个请求从开始到结束的完整路径。Amazon X-Ray可以帮助您分析和调试分布式应用，定位性能瓶颈。

4.2 架构审查：Amazon Well-Architected框架

如何系统性地评估您的架构是否“良好”？Amazon Well-Architected框架提供了一套基于六大支柱（卓越运营、安全性、可靠性、性能效率、成本优化、可持续性）的最佳实践和设计原则。

如何系统性地评估您的架构是否“良好”？Amazon Well-Architected框架提供了一套基于六大支柱（卓越运营、安全性、可靠性、性能效率、成本优化、可持续性）的最佳实践和设计原则。

更重要的是，该框架提供了一个专门面向SaaS的“透镜”——SaaS Lens。它在通用框架的基础上，增加了14个专门针对SaaS业务的审查问题，例如：

- 您是如何将租户上下文关联到每个微服务的？
- 您是如何隔离租户数据和资源的？
- 您是如何设计您的计费和计量系统的？

定期使用Amazon Well-Architected Tool（尤其是SaaS Lens）对您的架构进行“体检”，可以帮助您主动发现潜在风险和优化机会。

4.3 持续学习与创新

云技术日新月异。保持对新服务、新功能的关注，并勇于尝试和原型验证，是保持技术竞争力的不二法门。订阅亚马逊云科技官方博客、参加线上线下的技术峰会和社区活动，并与亚马逊云科技的解决方案架构师团队保持紧密沟通，将帮助您在SaaS的马拉松长跑中，始终保持领先。

