利用生成式 AI 加快软件开发生命周 AWS 期

AWS 规范性指导

AWS 规范性指导: 利用生成式 AI 加快软件开发生命周 AWS 期

Table of Contents

利用生成式 AI 加快软件开发生命周 AWS 期

Chetan Makvana，亚马逊 Web Services

2025 年 4 月（文档历史记录）

对高质量软件的需求不断增长，促使组织不断寻找加快其软件开发生命周期 (SDLC) 的方法。在各组织努力保持竞争力的同时，缩短上市时间的同时保持或提高产品质量至关重要。为了应对这些挑战，软件开发体验必须不断发展并使用尖端的技术、方法和实践，以简化流程，使软件开发团队能够提高工作效率和创造力。下一代开发体验的出现标志着软件的构思、构建、测试和部署方式发生了重大变化。它集成了各种功能，包括云原生开发、人工智能驱动的自动化、高级项目管理、协作工具等，共同提高了 SDLC DevSecOps 的效率和有效性。

这种变革的最前沿是生成式人工智能在软件工程中的兴起。根据 Gartner 的数据，到2027年，40%的平台工程团队将使用人工智能来增强SDLC的每个阶段，而2023年这一比例仅为5%。该报告还指出，软件工程领导者现在必须做好准备，在对开发过程至关重要的更广泛领域采用生成式人工智能。在另一份报告中，McKinsey研究表明，开发者速度指数较高的公司收入增长速度快4-5倍，股东回报率提高60%，创新性提高55%。通过采用生成式人工智能，而不仅仅是代码生成，组织可以在其软件开发工作流程中将效率、生产力和创新提升到一个新的水平。这可以减少手动工作，揭示洞察力，并增强人类的专业知识。

目标

本战略文档概述了框架、基础能力、用例、最佳实践和成功指标，可帮助您通过生成式 AI 加速 SDLC。它描述了如何在所有开发阶段有效地集成生成式人工智能，以提高产品质量和效率。

本战略文件可以帮助您和您的组织实现以下目标：

- 实施框架、基础能力、用例、最佳实践和成功指标，通过生成式 AI 加速 SDLC。
- 在所有开发阶段有效地集成生成式 AI，以提高产品质量、发布速度和开发效率。
- 通过整合尖端的人工智能技术、方法和实践来简化流程并增强开发团队的能力，从而适应下一代软件开发。

目标受众

本战略文件适用于希望通过将生成式人工智能应用于开发实践来加快软件开发生命周期的 IT 领导、工程经理、首席技术官和软件开发团队。

了解软件开发经验

软件开发体验包括开发团队在整个软件开发生命周期 (SDLC) 中使用的环境、工具和流程。它包括集成开发环境 (IDE)、协作平台、测试框架、知识管理系统、部署管道等。

精心设计的开发体验可以简化工作流程，减少手动工作，并使您的团队能够专注于高价值的任务，从而最终加快 SDLC 的速度。例如，与需要手动移交和上下文切换的分散工具链相比，通过无缝集成 IDE、版本控制系统和部署工具，开发人员能够以更快的速度和效率编写、测试和部署代码。同样，整合强大的知识管理框架可以帮助团队轻松访问和共享机构知识、最佳实践和文档。这提高了他们的整体生产力和解决问题的能力。

软件开发经验对软件开发团队的整体绩效和成功有直接影响。次优体验可能导致以下情况：

- 工作@@ 效率降低 — 工具效率低下、工作流程复杂和缺乏自动化会影响团队的工作效率，从而减慢功能和更新的交付。
- 技术债务增加 — 工具集成不良和临时流程可能导致技术债务，这使得随着时间的推移维护和扩展软件系统变得更具挑战性。
- 创新减弱 — 当被手动重复的任务所困扰时，您的团队探索新技术和推动创新的能力就会受到限制。
- 质量受损 — 分散的测试和部署过程增加了出现软件缺陷和漏洞的风险。这可能会对所交付软件的整体质量产生负面影响。

通过投资精心设计的软件开发体验，您可以获得显著的好处，例如更快的上市时间、更高的软件质量、更高的软件开发团队满意度以及更高的业务灵活性。

利用生成式 AI 为软件开发体验提供支持

将生成式人工智能集成到软件开发生命周期 (SDLC) 中，代表了整个软件开发团队构思、设计、实施和维护软件解决方案的方式的范式转变。生成式人工智能有可能彻底改变 SDLC 的每个阶段，包括项目管理、需求收集、设计、编码、测试、部署和维护。

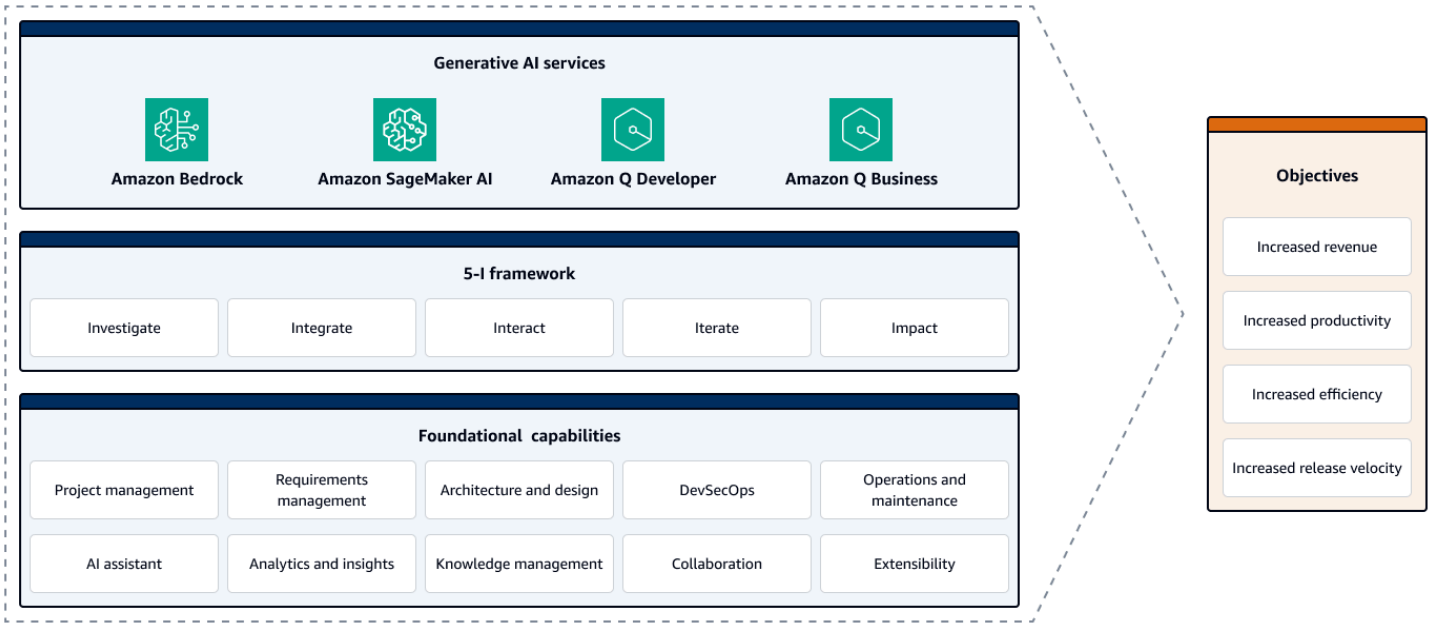
从本质上讲，由人工智能驱动的生成式开发体验可以充当整个软件开发团队的智能协作者，包括产品经理、设计师、解决方案架构师、开发人员、测试人员和运营人员。它提供情境感知帮助，生成工件（例如用户故事、设计模型、代码片段和测试用例），提供近乎实时的建议，甚至在潜在问题出现之前对其进行预测。这种人工智能增强方法可显著减轻团队成员的认知负担。这使他们能够专注于高层次的战略决策和复杂的问题解决，而生成式人工智能则可以处理更平凡、更重复的任务。

生成式人工智能还可用作知识放大器。它可以帮助团队成员从庞大的数据存储库中快速访问相关信息、最佳实践和模式。这可以有效地使整个组织的专业知识民主化。通过在整个开发工具链中无缝集成生成式 AI 功能，您可以为整个软件开发团队创建更直观、更高效、更高效的环境。这种增强的开发体验可加速 SDLC 并提高整体质量。它还可以减少错误并促进创新，因为团队成员可以更快地探索新的想法和方法。

要在组织中采用人工智能驱动的生成式开发体验，请考虑以下关键要素：

- 5-I 框架— 5-I 框架由五个维度组成，为驾驭现代软件开发过程提供了一种全面的方法。它提供了一种结构化的方法，可帮助您在SDLC的所有阶段系统地应用生成式人工智能。
- 基础能力— 要在现代软件开发的各个维度上充分利用生成式人工智能的力量，你需要建立一套强大的基础能力。这些功能构成了人工智能驱动的开发体验的支柱。这些功能可帮助您在整个 SDLC 中集成和使用生成式 AI。

5-I 框架和基础功能共同构成了重新构想软件开发体验的策略。这五个维度为应用生成式人工智能提供了战略框架，而基础能力则为您的组织做好了支持这种人工智能驱动方法的准备。 AWS 服务，例如 Amazon Bedrock、Amazon SageMaker AI、Amazon Q Developer 和 Amazon Q Bus iness，提供了生成人工智能功能和功能，您可以将其集成到软件开发体验中。



提供人工智能驱动的软件开发体验的 5-I 框架

5-I 框架为软件开发团队提供了一种结构化的方法，可以有效地将生成式 AI 整合到他们的开发实践中。它可以帮助您为在整个 SDLC 中使用生成式 AI 奠定坚实的基础。它还可以帮助您设置正确的开发实践、工作流程和思维方式，以充分利用生成式 AI 的潜力。

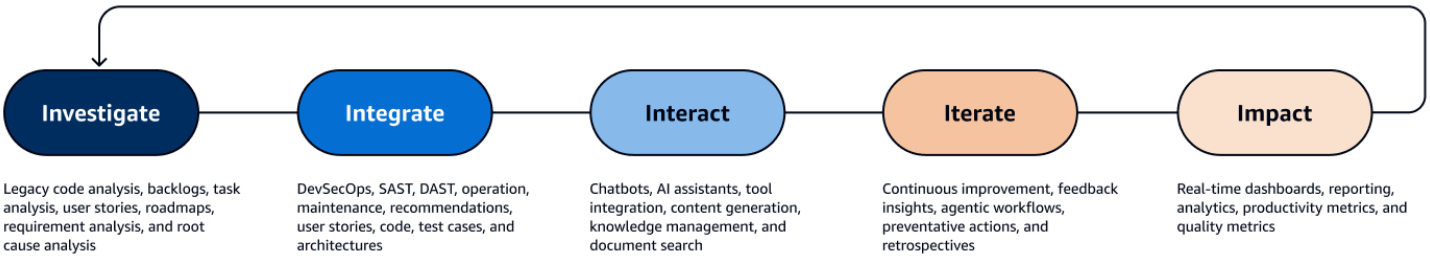
本节包含以下主题：

- 框架概述
- 与软件开发生命周期集成

框架概述

5-I 框架围绕五个关键维度构建：调查、整合、互动、迭代和影响。每个维度都代表着一个关键领域，在这个领域，生成式人工智能可以显著增强软件开发过程。通过战略性地整合这些维度的生成式人工智能，该框架满足了现代软件开发不断变化的需求。它可以减轻认知负荷并增强创造潜力。它认识到，理想的开发体验不仅仅是工具，而是要创造一个环境，让人工智能在每个阶段都能无缝增强人类能力。

下图显示了人工智能驱动的软件开发的五个维度。对于每个维度，它都显示了可以在哪里集成生成式人工智能，以提高效率和创新。



以下是框架中的五个维度：

- 调查 — 利用生成式 AI 增强软件开发过程中的每项分析任务。使用生成式 AI 来了解需求、处理大量数据、识别模式并生成可能超出人类能力或需要更长时间才能生成的见解。这些见解可以帮助您做出更明智的决策，快速发现改进机会，并更有效地交付高质量的软件。生成式人工智能可以成为整个 SDLC 分析过程的智能合作伙伴。通过利用生成式 AI，您可以对关键领域进行深入分析，例如需求收集、旧代码库检查和产品待办事项优化。例如，在创建用户故事之前，产品所有者可以使用生成式 AI 来分析用户旅程或需求。开发团队可以发现现有代码库中的低效率并发现优化机会。 DevOps 工程师可以应用根本原因分析来快速诊断性能问题或安全漏洞，从而提高可靠性。
- 集成 — 集成生成式 AI，实现整个 SDLC 中各种任务和流程的自动化。这包括自动生成代码片段、测试用例、架构设计、用户故事和部署管道。通过自动执行这些通常是手动的任务，团队可以专注于更具战略性和创新性的工作，从而缩短上市时间和高质量的应用程序。集成维度代表了软件开发的范式转变，在这种转变中，人工智能成为开发过程不可或缺的一部分。它与您的软件开发团队合作，以提高生产力、提高质量并推动创新。这样可以缩短上市时间。你的软件开发团队需要定期评估他们的流程和工作流程，每一步都要问：“这可以实现自动化吗？”
- 互动 — 使用人工智能驱动的生成式助手，为您的团队提供针对一系列任务和查询的即时情境支持。这些智能助手充当知识渊博的协作者，他们从庞大的信息库中汲取灵感。他们可以回答编码问题，提供设计建议，解释标准操作程序，并帮助解决复杂的问题。将这些 AI 助手集成到开发工作流程中可以提高工作效率，并营造一个更具协作性、更能解决问题的环境。
- 迭代 — 使用生成式 AI 在整个 SDLC 中实现快速的数据驱动调整。您可以持续分析来自客户反馈、使用模式、市场趋势和团队绩效指标等来源的数据，以便快速做出明智的决策。这种适应性使您的软件开发从静态的、预定义的流程转变为流畅、响应式的方法。它以多种方式表现出来，包括对待办事项进行动态优先排序、灵活的资源分配、自适应测试策略、不断演变的文档以及响应式部署流程。例如，产品经理可以使用人工智能生成的见解来重新排列待办事项，近乎实时地整合新的客户需求和市场趋势。 DevOps 工程师可以根据性能分析调整部署计划和基础架构配置，确保应用程序保持弹性和优化。开发团队可以将冲刺回顾中的反馈转化为可行的改进，供下一次迭代使用，从而推动持续改进流程的文化。
- 影响 — 应用生成式 AI 来评估软件开发过程的有效性和性能。通过使用 AI 支持的分析和指标，您可以更深入地了解开发效率、代码质量、用户参与度和整体应用程序性能。这种数据驱动的方法可以帮

助您做出明智的决策，优化开发工作流程，并持续改善应用程序的质量和用户体验。在评估软件团队的工作效率时，生成式 AI 会分析各种数据点，例如代码提交频率、问题解决时间、发布速度、功能交付率等。它还可以评估代码审查的质量、协作工具的有效性以及不同开发实践对团队整体产出的影响。通过将这些指标与项目结果相关联，人工智能可以识别人类分析师可能错过的模式和趋势，他们可以提供可行的见解，从而提高团队的工作效率。此外，生成式人工智能可以帮助您根据行业标准或历史数据对团队绩效进行基准测试，从而提供个性化的改进建议。它还可以预测开发过程中的潜在瓶颈或风险，以便您可以采取积极的措施。

与软件开发生命周期集成

SDLC 由多个阶段组成，可能因组织而异。通常，这些阶段包括以下几个阶段：需求和规划、设计和架构、实施、测试、部署以及运营和维护。

下表将 5-I 框架的维度映射到 SDLC 阶段，并提供了每个维度的集成级别。

框架维度	要求和规划	设计和建筑	实施	测试	部署	操作和维护
调查	高	低	低	低	低	中
整合	中	中	高	中	高	高
互动	高	高	高	中	中	高
迭代	中	低	低	低	低	中
Impact	高	中	高	低	高	高

整合程度从高到低不等。该映射显示了每个维度的关键重点领域。例如，“调查” 显示需求和计划阶段的强度很高。In tegrate在实施、部署以及运营和维护阶段表现出很高的强度。

通过使用此映射，您可以有效地确定工作的优先级。我们建议您先关注高、中、低。确保采用平衡且有影响力的方法，通过生成式 AI 增强软件开发体验。

人工智能驱动的软件开发体验的基础能力

要成功实现基于人工智能的生成式软件开发体验，你需要建立一套涵盖组织中多个角色的基础能力。这些功能代表了您在人工智能驱动的软件开发背景下有效部署资源、实施流程和实现预期结果的能力。通过培养这些能力，你可以打下坚实的基础，帮助你在 SDLC 的所有阶段无缝集成生成式 AI。

AWS 提供关键服务来帮助您实现这些功能。例如，Amazon Q D eveloper 通过充当人工智能驱动的助手来帮助加快软件开发。Amazon Q Bus iness 可帮助您快速获得紧迫问题的相关答案，解决问题并生成内容。它还可以通过集成与软件开发相关的工具来代表您行事。Amazon Bedrock 提供对基础模型的访问权限和广泛的功能，以自定义特定的开发工作流程和要求。

通过培养这些能力 AWS 服务，您可以打下坚实的基础，帮助您在SDLC的所有阶段无缝集成生成式人工智能。

以下是您应重点关注的基础功能：

- 项目管理
- 需求管理
- 建筑与设计
- 协作
- DevSecOps
- 操作和维护
- 人工智能助手
- 分析和见解
- 知识管理
- 可扩展性

每项基础功能都与 SDLC 的框架维度和 SDLC 的不同阶段相集成。这种集成可帮助您在整个软件开发过程中有效地使用 AI 功能。它提高了每一步的效率、质量和创新。这些基础功能、框架和 SDLC 阶段之间的协同作用为人工智能驱动的软件开发创造了一个全面的生态系统。这可以帮助您充分利用生成式人工智能的潜力，推动持续改进，加快开发周期，并交付高质量的软件产品。

下表显示了基础功能和子功能如何映射到框架维度和 SDLC 阶段。

能力：子能力	调查	集成	交互	迭代	影响
项目管理：问题管理	要求和规划	无	无	无	无
项目管理：Sprint 和任务管理	要求和规划	要求和规划	无	无	无

能力：子能力	调查	集成	交互	迭代	影响
项目管理：产品待办事项管理	要求和规划	无	无	要求和规划	无
项目管理：用户故事映射	要求和规划	无	无	无	无
项目管理：报告和分析	要求和规划	无	无	无	要求和规划
项目管理：产品路线图管理	要求和规划	无	要求和规划	无	无
项目管理：反馈循环	无	无	无	要求和规划	无
项目管理：回顾	无	无	无	要求和规划	无
需求管理	要求和规划	要求和规划	无	无	无
架构和设计：解决方案设计	设计和建筑	设计和建筑	无	无	无
协作：文档管理	所有 SDLC 阶段	无	所有 SDLC 阶段	无	无
协作：知识共享	所有 SDLC 阶段	无	所有 SDLC 阶段	无	无
协作：项目资产管理	无	所有 SDLC 阶段	所有 SDLC 阶段	无	无
DevSecOps: CI/ CD	测试、部署	实施、测试、部署	部署	无	无

能力：子能力	调查	集成	交互	迭代	影响
DevSecOps: DevOps 安全	实施	实施、测试、操作和维护	无	实施、测试、操作和维护	无
DevSecOps: 应用程序性能监控	无	操作和维护	无	无	无
DevSecOps: 日志聚合和分析	操作和维护	操作和维护	无	无	无
DevSecOps: AIOps	操作和维护	无	无	操作和维护	无
DevSecOps: 持续改进	无	无	无	操作和维护	无
DevSecOps: 仪表板监控	无	操作和维护	无	无	无
DevSecOps: 性能见解	操作和维护	无	无	操作和维护	无
操作和维护：事件管理	无	无	无	操作和维护	无
操作和维护：代码升级	无	操作和维护	无	无	无
操作和维护：代码优化	操作和维护	操作和维护	无	无	无
运营和维护：技术债务管理	无	操作和维护	操作和维护	无	无

能力：子能力	调查	集成	交互	迭代	影响
操作和维护：变更管理	无	实施、部署	无	无	无
操作和维护：逆向工程	操作和维护	无	无	无	无
操作和维护：代码现代化	无	实施	无	无	无
操作和维护：性能优化	无	操作和维护	无	操作和维护	无
分析和见解	无	要求和规划	无	无	所有 SDLC 阶段
人工智能助手	无	无	所有 SDLC 阶段	无	无
知识管理	无	无	所有 SDLC 阶段	无	无
可扩展性	无	部署	无	无	无

用于项目管理的生成式 AI 用例

有效的项目管理是成功开发软件的核心。在生成式人工智能的背景下，项目管理呈现出新的维度。它可以变得更具预测性、适应性和数据驱动性。人工智能驱动的项目管理工具可分析历史项目数据，以生成更准确的时间和资源估算。他们可以根据业务目标和团队能力自动确定任务的优先顺序，甚至可以在潜在的障碍出现之前对其进行预测。例如，项目经理可能会使用生成式人工智能，根据项目的要求和类似项目的历史数据来创建初步的项目计划。然后，人工智能可以根据技能、工作量和项目需求提出最佳的团队构成。在整个项目中，人工智能驱动的仪表板通过自动生成报告并突出显示需要关注的领域，提供对项目状态的近乎实时的见解。

这种人工智能增强的项目管理方法可以提高效率。它可以帮助项目经理专注于战略决策和团队领导，而不是陷入日常管理任务的困境。

下表显示了您可以使用生成式 AI 和负责这些用例的角色来增强的项目管理用例。

子功能：用例	女神异闻录
问题管理：创建和分配问题	项目经理
问题管理：在测试期间检测问题并记录下来	测试工程师
问题管理：根据严重程度确定问题的优先级并将其分配给开发人员	项目经理
问题管理：识别并合并重复的问题	项目经理
问题管理：跟踪并生成有关项目关键问题、指标和整体运行状况的报告	项目经理
冲刺和任务管理：估算任务的工作量并根据团队能力分配故事要点	混战大师
冲刺和任务管理：在团队成员之间分配任务，以平衡整个冲刺的工作量	混战大师
冲刺和任务管理：促进冲刺规划会议，使团队努力与冲刺目标保持一致	混战大师
产品待办事项管理：根据业务价值、紧急程度和用户反馈对待办事项重新排序	产品所有者
产品待办事项管理：将新的客户反馈和市场洞察整合到待办事项产品中，以实现近乎实时的优先级排序	产品所有者
产品待办事项管理：识别和管理待办事项之间的依赖关系，以简化开发	产品经理
用户故事映射：创建用户旅程地图，以识别所有必需的功能及其相应的用户故事	产品所有者
用户故事映射：识别用户流程中的差距或缺失步骤	业务分析师

子功能：用例	女神异闻录
用户故事映射：根据用户故事对商业价值的影响确定其优先级	产品经理
报告和分析：生成近乎实时的仪表板，以可视化关键项目指标，例如冲刺速度和问题解决率	项目经理
报告和分析：分析历史数据并预测未来的项目成果，例如潜在的延误或瓶颈	项目经理
报告和分析：创建为不同利益相关者量身定制的自定义报告，例如团队绩效或项目状态报告	项目经理
产品路线图管理：创建和维护概述主要里程碑和发布日期的产品路线图	项目经理
产品路线图管理：根据项目优先级或时间表的变化更新路线图	产品经理
产品路线图管理：与利益相关者共享路线图，以提供对产品方向的可见性	产品经理
反馈循环：每次冲刺结束后收集团队的反馈并确定需要改进的领域	混战大师
回顾：将反馈转化为下一次冲刺的可操作项目，推动持续改进	混战大师
回@@ 顾：跟踪先前回顾中实施的变更的影响，以衡量其有效性	混战大师

用于需求管理的生成式 AI 用例

需求管理是一个与项目管理密切相关的关键流程。想象一下，产品负责人使用 AI 工具来分析客户反馈、市场趋势和利益相关者的意见。人工智能工具可以生成一组全面的用户故事和需求，自动对其进行分类，检测潜在的冲突或差距，甚至可以根据业务价值和实施复杂性建议优先顺序。随着项目的进展和需求的演变，人工智能可以不断更新和完善需求，以确保它们与不断变化的业务需求和技术限制保持一

致。这种由人工智能驱动的动态需求管理方法有助于确保开发工作在整个项目生命周期中与用户需求和业务目标保持紧密一致。

下表显示了您可以通过生成式 AI 以及负责这些用例的角色来增强的需求管理用例。

使用案例	女神异闻录
创建业务需求	业务分析师
利用专题创作长篇故事	产品所有者
通过监控长篇故事的相关用户故事的完成情况来跟踪其进度	产品经理
创建用户故事	产品所有者
估算每个使用故事所需的工作量并分配故事积分	混战大师
为每个用户故事定义接受标准	产品所有者

建筑和设计的生成式 AI 用例

有了坚实的项目管理基础和明确定义的需求，下一个关键能力是架构和设计。在这里，生成式人工智能为创建强大、可扩展和高效的软件架构开辟了新的可能性。人工智能驱动的设计工具可以分析需求和限制，以建议最佳的架构模式和设计方法。它们生成了多种设计方案，并且每种方案都针对不同的优先级（例如性能、可扩展性或可维护性）进行了优化。例如，解决方案架构师可能会使用 AI 助手根据项目要求快速生成多个高级架构设计。这种增强人工智能的方法可以加快设计过程，并帮助建筑师做出更明智的决策。这会带来更强大、更经得起未来考验的软件设计。

下表显示了您可以通过生成式 AI 以及负责这些用例的角色来增强的架构和设计用例。

使用案例	女神异闻录
创建架构文档	解决方案架构师
创建详细的设计文档	技术领导
了解现有的架构和设计标准	解决方案架构师

使用案例	女神异闻录
开发用户界面的详细模型和原型	UX/UI 设计师

用于协作的生成式 AI 用例

软件开发本质上是一项协作工作。您可以使用生成式 AI 来增强软件开发团队的协作。人工智能驱动的协作工具不仅仅是简单的消息传递和文件共享。它们通过总结冗长的讨论话题，突出关键决策，甚至根据团队成员的日程安排和生产模式建议最佳会议时间，从而促进更有效的沟通。人工智能可以通过自动识别潜在问题、提出改进建议，甚至向审阅者解释复杂的更改来帮助进行代码审查。在头脑风暴会议中，人工智能可以充当促进者，提出想法，帮助组织思想，甚至调解讨论，以确保所有声音都能被听见。对于分布式团队来说，人工智能可以帮助弥合文化和语言障碍。它可以在聊天和视频通话中提供近乎实时的语言翻译，并提供文化背景以帮助防止误解。通过增强人类与人工智能的协作，该功能可以帮助团队更高效地工作，从而促进创新并改善整体项目成果。

下表显示了如何使用生成式 AI 来增强协作用例。

子功能：用例	女神异闻录
文件管理：创建和维护集中式文档存储库	技术作家
文档管理：允许多个团队成员实时协作处理文档	开发团队
知识共享：使用论坛作为开发人员提问、共享知识和协作解决问题的平台	开发团队
知识共享：使用讨论论坛记录和跟踪项目讨论期间做出的决策，确保捕捉关键决策背后的基本原理并可供将来参考	产品经理
项目资产管理：便于轻松共享项目相关资源	开发团队
项目资产管理：对共享内容实施版本控制，以便团队成员可以跟踪更改、恢复到以前的版本以及协作处理内容更新	开发团队

生成式 AI 用例 DevSecOps

人工智能驱动的 DevSecOps 工具可以自动执行软件交付管道的许多方面。例如，他们可以在开发人员编写代码时近乎实时地执行智能代码审查、检测潜在错误、检测安全漏洞并识别性能问题。AI 生成并运行全面的测试套件，并随着代码库的发展自动对其进行更新。这种增强人工智能的方法 DevSecOps 可以加快交付渠道，并显著增强所交付软件的安全性和可靠性。

下表显示了你可以通过生成式 AI 来增强的 DevSecOps 用例以及负责这些用例的角色。

子功能：用例	女神异闻录
DevOps 和持续交付：自动化整个部署管道	DevOps 工程师
DevOps 和持续交付：接收有关代码质量和潜在问题的近乎实时的反馈	软件开发人员
DevOps 并持续交付：接收近乎实时的安全问题和补救建议	软件开发人员
DevOps 持续交付：接收近乎实时的代码和最佳实践建议	软件开发人员
DevOps 和持续交付：自动执行重复任务并将命令集成到脚本中	DevOps 工程师
DevOps 和持续交付：每次提交代码后自动生成代码并生成工件	软件开发人员
DevOps 和持续交付：根据组织的标准和框架构建代码	软件开发人员
DevOps 和持续交付：在每次提交时自动运行单元测试，以便在开发过程的早期发现错误	软件开发人员
DevOps 和持续交付：分析单元测试的覆盖率，确保所有关键代码路径都经过测试	软件开发人员
DevOps 和持续交付：管理分支并合并更改	软件开发人员
DevOps 和持续交付：管理代码和工件版本控制	软件开发人员

子功能：用例	女神异闻录
DevOps 和持续交付：存储和管理构建工件和依赖关系	DevOps 工程师
DevOps 和持续交付：在构建过程中解析并获取依赖关系	软件开发人员
DevOps 和持续交付：生成并运行集成测试，确保组件按预期协同工作	测试工程师
DevOps 和持续交付：在集成测试期间使用模拟服务来模拟与外部系统的交互	测试工程师
DevOps 和持续交付：对不同负载下的应用程序性能进行基准测试	性能工程师
DevOps 和持续交付：模拟高流量场景以测试应用程序的可扩展性和响应时间	性能工程师
DevOps 和持续交付：测试系统从故障（例如服务器崩溃或网络中断）中恢复的能力	现场可靠性工程师
DevOps 和持续交付：执行混沌工程	现场可靠性工程师
DevOps 和持续交付：运行测试以验证应用程序是否满足业务需求	QA 工程师
DevOps 和持续交付：进行用户验收测试	产品所有者
DevOps 和持续交付：扫描依赖关系以查找漏洞和许可证合规性问题	安全工程师
DevOps 和持续交付：监控和管理开源依赖关系，确保它们是最新且安全的	安全工程师
DevOps 和持续交付：生成并维护软件物料清单 (SBOM)，以跟踪所有组件和依赖关系	安全工程师

子功能：用例	女神异闻录
DevOps 和持续交付：使用 SBOM 进行监管合规性审计	合规官员
DevOps 和持续交付：创建发行说明	发布管理器
DevOps 和持续交付：计划和协调发布	发布管理器
DevOps 和持续交付：实施回滚和发布管理的标准操作程序	发布管理器
DevOps 和持续交付：使用功能标志在生产环境中启用或禁用功能，无需部署新代码	产品经理
DevOps 和持续交付：使用功能标志运行 A/B 测试，以衡量不同功能对用户行为的影响	产品经理
DevOps 和持续交付：分析和监控管道故障	DevOps 工程师
DevOps 和持续交付：创建和管理基础架构资源	DevOps 工程师
DevOps 和安全：扫描代码存储库以获取硬编码的机密	DevOps 工程师
DevOps 和安全性：实施近乎实时的检测，以便在机密被提交到存储库时立即提醒开发人员	DevOps 工程师
DevOps 和安全：实施持续的代码质量监控	软件开发人员
DevOps 和安全：检测和标记代码中潜在安全漏洞的指标	软件开发人员
DevOps 和安全：对开放全球应用程序安全项目 (OWASP) 的十大安全风险实施自动测试，以确保应用程序符合行业标准的安全实践	安全工程师
DevOps 和安全：通过将检查整合到开发流程中，定期向开发人员更新 OWASP 风险并对其进行教育	安全工程师

子功能：用例	女神异闻录
DevOps 和安全：扫描第三方库和依赖项以查找已知的安全漏洞	DevOps 工程师
DevOps 和安全：扫描应用程序代码和基础设施以检测漏洞	DevOps 工程师
DevOps 和安全：在部署之前分析代码中是否存在漏洞	安全工程师
DevOps 和安全：通过防止存在严重漏洞的代码被合并来强制执行安全策略	安全工程师
DevOps 和安全：实施基于角色的访问控制 (RBAC)，以限制对敏感系统和数据的访问，并确保只有经过授权的人员才能访问关键资源	安全工程师
DevOps 和安全性：通过适应团队结构的变化，根据角色和职责调整访问控制	DevOps 工程师
DevOps 和安全：通过模拟对生产环境的攻击，近乎实时地测试正在运行的应用程序是否存在安全漏洞	安全工程师
DevOps 和安全：持续监控已部署的应用程序是否存在安全漏洞	DevOps 工程师
DevOps 和安全：安排在所有环境中定期进行漏洞扫描，以识别和解决安全漏洞	安全工程师
DevOps 和安全：根据漏洞扫描结果应用补丁和更新，以帮助维护系统的安全	DevOps 工程师
应用程序性能监控：近乎实时地持续监控应用程序性能，以便在性能问题影响用户之前对其进行检测和诊断	现场可靠性工程师

子功能：用例	女神异闻录
应用程序性能监控：检测性能异常，例如响应时间突然激增或错误率增加，并启动警报	DevOps 工程师
应用程序性能监控：在请求通过分布式系统传播时对其进行跟踪，以识别性能瓶颈和延迟问题	DevOps 工程师
应用程序性能监控：使用分布式跟踪来查明导致故障或性能下降的确切服务或组件	DevOps 工程师
日志聚合和分析：将来自多个来源的日志聚合到一个集中式系统中，便于搜索和分析，从而识别趋势和问题	现场可靠性工程师
日志聚合和分析：实施自动日志解析以提取相关信息并检测可能表明问题的模式或异常	DevOps 工程师
日志聚合和分析：收集和可视化关键性能指标	现场可靠性工程师
日志聚合和分析：根据预定义的服务级别协议监控指标 () SLAs	产品经理
AI 运营：无需人工干预即可检测事件、分析根本原因并启动纠正措施	DevOps 工程师
AI 运营：预测 future 的资源需求并优化容量规划以避免停机	现场可靠性工程师
持续改进：监控用户与应用程序的真实互动，以收集有关性能的见解并确定需要改进的领域	用户体验设计师
持续改进：跟踪不同地理区域的应用程序性能，以确保全球一致的用户体验	产品经理
仪表板监控：创建可自定义的仪表板，以近乎实时的方式可视化关键指标、日志和跟踪，从而全面了解系统运行状况	现场可靠性工程师

子功能：用例	女神异闻录
仪表板监控：为不同的团队（例如开发、运营和产品团队）创建仪表板，根据他们的重点领域提供相关的见解	DevOps 工程师
性能见解：对应用程序性能进行详细分析，以发现效率低下并优化代码或基础架构	软件开发人员
性能见解：随着时间的推移，使用性能见解以迭代方式提高应用程序性能并优化用户体验	产品经理

用于操作和维护的生成式 AI 用例

部署软件后，重点转移到操作和维护上。生成式 AI 可以通过提供更主动、更高效的系统管理来增强传统方法。人工智能驱动的操作工具可以持续监控系统性能，并在潜在问题影响用户之前对其进行预测。当问题发生时，他们会自动进行根本原因分析，从而大大缩短了解决问题的平均时间。AI 还可以近乎实时地优化系统性能。它会根据不断变化的负载模式和用户行为自动调整配置。例如，运营团队可能会使用 AI 助手来生成预测性维护计划，自动识别可能出现故障的组件，并提出先发制人的操作建议。人工智能还可以通过分析使用趋势和高精度预测未来的资源需求来帮助进行容量规划。

下表显示了您可以通过生成式 AI 以及负责这些用例的角色来增强的操作和维护用例。

子功能：用例	女神异闻录
事件管理：通过将监控工具与聊天平台集成，近乎实时地管理事件，以便团队可以直接在聊天环境中检测、讨论和解决问题	现场可靠性工程师
事件管理：允许团队直接从聊天界面启动部署、运行脚本和运行命令，从而简化操作	DevOps 工程师
代码升级：升级代码依赖项和库以减少手动工作，并确保代码库与最新版本保持同步	软件开发人员
代码优化：查看代码以寻找优化机会	软件开发人员

子功能：用例	女神异闻录
代码优化：识别代码中的瓶颈并重构或优化代码以提高性能	软件开发人员
技术债务管理：将技术债务记录为开发过程的一部分	产品经理
技术债务管理：根据影响、风险和成本对技术债务进行优先排序和解决，并将其整合到常规冲刺计划流程中	软件开发人员
技术债务管理：减少现有应用程序代码中的技术债务	软件开发人员
变更管理：实施变更批准流程，确保所有代码变更在部署之前都经过必要的利益相关者的审查、测试和批准	变更经理
变更管理：对提议的变更进行影响分析	DevOps 工程师
逆向工程：分析和了解遗留代码的结构和行为	解决方案架构师
逆向工程：解释现有代码并生成文档	软件开发人员
代码现代化：将代码从一种编程语言翻译成另一种编程语言	软件开发人员
代码现代化：将旧代码现代化为最新的编程语言	软件开发人员
性能优化：通过优化资源分配、负载平衡和重新配置应用程序，持续监控和调整系统性能	现场可靠性工程师
性能优化：识别并重构导致性能下降的代码，以提高速度和系统响应能力	软件开发人员

生成式 AI 助手在软件开发中的用例

人工智能助手功能是人工智能驱动的生成式开发体验的核心。这个智能的情境感知系统可充当整个 SDLC 中所有团队成员的虚拟协作者。想象一下，开发人员正在处理一段复杂的代码。他们只需向 AI 助手寻求帮助，它就可以提供相关的代码片段，解释复杂的算法，甚至可以根据当前背景和最佳实践提出优化建议。AI 助手可以帮助 ITOps 经理了解基于内部文档的标准操作程序。通过提供即时情境支持，AI 助手可以显著减轻团队成员的认知负担。这有助于他们专注于更高层次的问题解决和创造性任务。这种能力可以起到增强作用的作用，可以提高软件开发各个阶段的生产力和质量。

下表显示了您可以使用 AI 助手和受益角色增强的用例。

使用案例	女神异闻录
通过回答有关需求、架构和标准操作程序等问题，为开发团队提供即时帮助	软件开发组
搜索或检索大量文档的摘录，或者使用自然语言查询生成摘要	软件开发组
总结长篇技术文档，例如需求文档、架构设计文档和内部流程	软件开发组
维护一个提示库，供团队用于常见任务	软件开发组
将生成式 AI 无缝集成到现有工具和系统中	软件开发组
跨各种平台、工具和内部系统自动执行任务	软件开发组
创建集中的知识库，包括最佳实践、项目特定信息和团队知识，所有团队成员均可访问	软件开发组
根据任务的上下文从存储库中检索相关知识	软件开发组
执行自动代码审查、根本原因分析、提出改进建议、检测潜在错误并进行故障排除	软件开发人员、 DevOps 工程师和站点可靠性工程师
分析性能数据以确定趋势和模式，从而为性能优化决策提供依据	现场可靠性工程师
为提高效率、降低复杂性和增强安全性提供建议	软件开发人员

使用案例	女神异闻录
针对云资源使用提出优化建议，例如扩展建议或节省成本的策略	软件开发人员、 DevOps 工程师、站点可靠性工程师和解决方案架构师
生成新内容，例如基于代码、用户指南或产品功能版本的文档	软件开发组

用于分析和见解的生成式 AI 用例

分析和见解功能有助于将大量数据转化为可行的见解，从而推动决策和持续改进。通过使用生成式人工智能，该功能可以处理来自各种来源的数据，包括代码存储库、项目管理工具和团队协作平台，以提供开发过程和团队生产力的整体视图。生成式 AI 超越了传统指标，可以提供预测性和规范性分析。它可以预测潜在的问题并提出有针对性的改进建议。例如，它可以分析代码提交模式、错误解决率和功能交付速度，以识别高绩效团队、查明瓶颈并提出流程优化建议。此外，它还可以提供对团队动态和个人绩效的见解。这些见解可以帮助领导者就工作量分配、培训需求和团队组成做出以数据为依据的决策。通过交互式仪表板展示这些见解，该功能使各级利益相关者能够做出明智的决策，优化流程并持续提高团队生产力，从而更快地交付高质量的软件。

下表显示了您可以使用生成式 AI 来增强的分析用例以及负责这些用例的角色。

使用案例	女神异闻录
监控个人和团队的工作效率	开发经理
分析生产力趋势以发现潜在的倦怠，以便您可以采取积极措施来维持团队的健康和工作效率	开发经理
跟踪将代码更改部署到生产环境的频率，以衡量开发过程的速度和敏捷性	产品经理
分析部署频率数据，以确定可能表明流程效率低下或资源限制的低部署活动时段	产品经理
衡量代码提交到部署之间的时间，以确定简化开发和部署流程的机会	开发经理

使用案例	女神异闻录
跟踪导致需要立即修复的故障的部署百分比，以评估发布过程的可靠性	现场可靠性工程师
使用更改失败率指标来确定经常导致问题的代码区域，以指导有针对性的重构和测试工作	软件开发人员
监控中断或事故发生后恢复服务所需的时间，这样您就可以减少停机时间并提高系统的整体弹性	现场可靠性工程师
分析恢复时间趋势，以增强事件响应流程并加快从系统故障中恢复的速度	DevOps 工程师
创建自定义控制面板，汇总部署频率、交货时间和变更失败率等关键指标，以便全面了解开发和运营状况	产品经理
创建针对不同团队需求量身定制的仪表板，以便针对其特定职责领域（例如开发、运营或业务）提供有针对性的见解	产品经理
跟踪业务关键绩效指标 (KPIs)，例如收入影响、客户满意度和市场份额，以便使开发工作与更广泛的业务目标保持一致	产品经理
分析新功能对业务的影响， KPIs 以评估其成功与否，并指导未来的产品开发	业务分析师
监控代码质量指标，例如代码复杂性、测试覆盖率和错误密度，以确保代码库保持可维护和安全	软件开发人员
确定代码库中需要重构的区域，以推动长期可持续性并减少技术债务	解决方案架构师

知识管理的生成式 AI 用例

在任何软件开发组织中，知识都是至关重要的资产。由生成式 AI 支持的知识管理功能增强了该资产的捕获、组织和使用方式。传统的知识管理系统通常包含太多信息，包含过时的内容，或者难以搜索以快速找到相关信息。

生成式 AI 直面这些挑战。它会根据代码更改、对话和项目工件自动生成和更新文档。这样可以确保知识库保持最新状态，而无需团队成员手动操作。更重要的是，人工智能使这些知识能够以直观的方式获取。团队成员可以用自然语言提问，人工智能可以提供相关答案。人工智能可以从各种来源获取信息，例如官方文档、代码注释、讨论话题，甚至是外部资源。例如，试图理解特定组件的新团队成员可能会问 AI：“身份验证模块是如何工作的？” 然后，人工智能将提供简明的解释以及指向相关代码部分、架构图和最新更改的链接。它甚至可以根据团队成员的角色和专业知识水平定制这些信息。

此功能可加快入门速度，减少重复性问题，并促进整个组织内的知识共享。它有助于保存机构知识，使团队更容易随着时间的推移维护和发展复杂的系统。

下表显示了您可以通过生成式 AI 以及负责这些用例的角色来增强的知识管理用例。

使用案例	女神异闻录
创建一个统一的平台，便于访问所有与项目相关的知识	软件开发组
从各种开发活动中获取知识	软件开发组
提供高级搜索功能，以便在存储库中快速查找相关知识	软件开发组
为团队提供个性化的学习模块和途径	软件开发组

可扩展性的生成式 AI 用例

可扩展性可实现与现有工具和工作流程的无缝集成，同时允许组织根据其特定需求量身定制人工智能系统。该功能提供了强大且 APIs可自定义的界面，便于将 AI 功能集成到流行的开发和项目管理工具中。SDKs例如，组织可以通过人工智能驱动的功能来增强 Jira，这些功能用于自动工单优先排序、工作量估算和冲刺计划。您可以使用 AI 扩展 Jenkins 管道，以实现智能版本优化和预测性测试选择。

此外，可扩展性还允许与集成开发环境 (IDEs)、版本控制系统和代码审查平台进行深度集成。人工智能可以帮助编码、自动进行代码审查和生成上下文文档。

该功能还支持根据组织特定数据训练和微调 AI 模型。这有助于 AI 了解公司特定的编码模式、架构偏好和领域知识。结果是，所有集成工具都提供了更具相关性和情境感知能力的帮助。通过提供这种级别的灵活性和集成度，可扩展性可确保人工智能驱动的开发体验与组织一起发展。它可以适应不断变化的技术和业务需求，同时无缝增强现有的工具链和工作流程。

下表显示了可扩展性用例，您可以使用生成式 AI 以及负责这些用例的角色来增强这些用例。

使用案例	女神异闻录
将第三方工具集成到开发环境中	DevOps 工程师
创建针对团队独特开发流程量身定制的自定义自动化工作流程	DevOps 工程师
Connect 连接到 APIs 各种服务	DevOps 工程师
为跨平台工具创建连接器	DevOps 工程师

在软件开发中使用生成式 AI 的最佳实践

本节介绍将生成式 AI 集成到软件开发生命周期 (SDLC) 中的最佳实践。从实施无缝工具链和 DevSecOps 管道到促进协作和自动执行重复性任务，这些指南可帮助您利用人工智能的力量来增强开发流程和体验。通过遵循这些最佳实践，软件开发团队可以在工作中将效率、创新和质量提升到新的水平。

本节讨论以下最佳实践：

- 实现无缝 end-to-end集成的工具链
- 为以下各项实现 end-to-end CI/CD 管道 DevSecOps
- 采用协作工具和实践
- 自动执行重复性任务
- 定期审查和迭代开发经验
- 采用有效的项目管理实践
- 实施知识管理
- 提供可扩展性和自定义
- 针对运营进行优化
- 使用数据驱动的见解
- 采用基于平台的方法

实现无缝 end-to-end集成的工具链

实现无缝的 end-to-end集成工具链是创建人工智能驱动的生成式开发体验的基本最佳实践。核心思想是建立一个由工具和平台组成的有凝聚力的生态系统，供您的软件团队在整个 SDLC 中使用。该团队可以使用工具链来规划、构思、编码、构建、测试、部署和管理正在进行的操作。通过将生成式人工智能功能集成到该工具链中，可以确保在每个阶段都提供人工智能帮助。这种集成减少或消除了手动交接，减少了上下文切换，并帮助数据和工件在不同的开发阶段之间顺畅流动。例如，来自集成开发环境 (IDE) 的 AI 生成的代码片段可以无缝流入您的版本控制系统，而来自部署平台的 AI 驱动的分析可以为您的项目管理工具提供信息。这会创建一个持续的反馈循环，从而改善您的开发流程。

为以下各项实现 end-to-end CI/CD 管道 DevSecOps

要在此集成工具链的基础上进行构建，实现 end-to-end持续集成和持续部署（CI/CD) pipeline for DevSecOps. This AI-powered pipeline is a critical component that streamlines your software

delivery processes. It helps you release new applications and updates more quickly and reliably. By embedding security practices throughout the entire SDLC, you can identify and address vulnerabilities much earlier, which reduces the overall cost and risk. The pipeline should incorporate AI at every stage, from continuous integration and testing to security checks and deployment. For instance, you can use AI to analyze code commits in near real time so that you can predict potential integration issues before they occur. In the CI/CD管道），您还可以使用生成式 AI 根据最新的威胁情报自动更新安全策略。

采用协作工具和实践

在增强开发基础设施时，不要忘记人为因素。软件开发本质上是一项协作工作。它涉及由开发人员、设计师、产品经理、Scrum Masters、业务分析师和其他利益相关者组成的跨职能团队。这些人共同努力，将想法付诸实践。通过使用现代协作工具并培养开放式沟通和知识共享的文化，您可以显著提高软件开发团队的工作效率和效率。在你的 AI 驱动的软件开发体验中，这些工具呈现出新的维度。您可以将 AI 集成到协作平台中，以促进团队成员之间更有效的沟通和知识共享。AI 助手可以回答常见问题、总结讨论，甚至调解冲突。生成式 AI 可以通过自动提出改进建议或识别潜在问题来增强代码审查流程。此外，您可以使用 AI 来创建动态的、情境感知的文档，这些文档会随着项目的发展而近乎实时地更新，以便所有团队成员都能访问最新和最相关的信息。

自动执行重复性任务

通过使用生成式 AI 来处理耗时的例行活动，您可以让软件团队腾出时间专注于推动创新并带来业务影响的高价值、创造性的工作。重复任务的示例包括生成样板代码、创建测试数据、编写文档，甚至起草初始项目计划。通过将这些任务转移给 AI，团队成员可以专注于更具创造性和战略性的工作。例如，人工智能驱动的代码完成工具可以根据上下文和编码模式建议相关的代码片段，从而显著加快编码过程。同样，生成式 AI 可以在代码更改时自动创建和更新技术文档。这样可以使文档保持最新状态，并减少此任务通常所需的手动工作。在测试中，AI 可以根据需求和代码分析生成全面的测试用例，从而提高测试覆盖率并降低被忽视边缘案例的可能性。通过智能地自动执行这些重复性任务，生成式 AI 可以加快开发时间，提高一致性并减少人为错误。结果是软件输出质量更高。

定期审查和迭代开发经验

您的软件开发经验本身应被视为需要不断完善的产品。这包括建立一个系统的流程，用于定期审查和迭代开发生命周期、工具和实践的各个方面。定期评估整个工具链、工作流程和流程。收集来自不同角色的所有团队成员的反馈，包括产品经理、设计师、架构师、开发人员、测试人员和运营人员。让他们找出痛点、瓶颈和改进机会。例如，团队可以每季度对其 CI/CD 管道性能进行审查，并分析构建时间、部署频率和错误率等指标，以确定需要优化的领域。由于生成式人工智能能力持续快速发展，因此持续

评估新的人工智能驱动工具和功能至关重要，这些工具和功能可能会进一步简化工作流程或增强SDLC中所有角色的能力。

采用有效的项目管理实践

要有效地协调复杂的软件开发工作，请采用人工智能增强的项目管理实践。在这种情况下，有效的项目管理超越了传统方法。它采用人工智能增强方法，可增强整个 SDLC 的规划、执行和监控。敏捷框架可促进灵活性、协作和快速迭代，您可以使用生成式 AI 来优化这些流程。例如，生成式人工智能可以分析历史项目数据以获得更准确的估计，根据业务目标和客户反馈自动生成用户故事并确定其优先级，并提供对团队绩效的智能见解。人工智能驱动的项目管理工具可以预测潜在的障碍，并根据团队成员的技能和工作量建议最佳的任务分配。通过将人工智能驱动的功能整合到项目管理实践中，您可以提高可见性，更快地做出数据驱动的决策，并确保团队成员保持一致并高效地朝着共同的目标努力。

实施知识管理

随着人工智能驱动的软件开发经验日趋成熟，请实施强大的知识管理系统。强大的知识管理系统可帮助您捕获、整理和授予对宝贵见解、最佳实践和解决方案的访问权限。SDLC 的所有团队成员都应该可以轻松访问系统。使用生成式 AI 创建动态、智能的知识库，这些知识库会随着您的组织而发展。例如，AI 可以根据代码更改、对话和项目工件自动生成和更新文档，从而无需人工干预即可使信息保持最新状态。生成式 AI 还可以支持智能搜索功能，帮助团队成员使用自然语言查询快速找到相关信息，即使他们不知道确切的术语。此外，生成式人工智能可以根据团队成员当前的任务或挑战主动向他们显示相关信息。它充当虚拟导师，可以增强所有角色的决策和问题解决能力。通过实施人工智能驱动的知识管理系统，您可以打破孤岛，加快入职速度，减少冗余工作，并在整个软件开发团队中培养持续学习和创新的文化。

提供可扩展性和自定义

为了最大限度地发挥生成式 AI 在软件开发中的优势，请确保你的 AI 驱动的工具和平台是可扩展和可定制的。这可以帮助您根据自己的特定需求、工作流程和技术堆栈量身定制 AI 功能。例如，您可以根据自己的代码库和文档微调 AI 模型，为特定任务创建自定义 AI 驱动的工具，或者将 AI 功能集成到现有工具和流程中。这种可扩展性可以帮助您改进 AI 驱动的开发体验，以满足组织不断变化的需求。它还可以帮助您优化特定领域或项目类型的体验。

针对运营进行优化

生成式 AI 在优化软件运营和维护方面起着至关重要的作用。通过将 AI 功能集成到您的运营工具和流程中来优化运营。例如，使用生成式 AI 近乎实时地分析日志数据，预测潜在的系统故障，并自动执行日

常维护任务。生成式 AI 还可以通过关联复杂分布式系统中的事件来帮助进行根本原因分析。这提高了系统可靠性，减少了停机时间，并使您的运营团队能够腾出时间专注于更具战略性的计划。

使用数据驱动的见解

在整个 AI 驱动的开发旅程中使用数据驱动的见解。实施系统来收集、分析和处理来自 SDLC 各个阶段的数据。这包括代码指标、测试结果、部署数据、用户反馈和操作性能。使用生成式 AI 来发现人类观察者可能看不见的模式和见解。然后，将这些见解反馈到您的开发流程中，为从架构决策到功能优先级划分的所有内容提供信息。

采用基于平台的方法

要充分实现生成式人工智能在软件开发中的优势，请采用基于平台的方法。创建一个全面、集成的平台，将 SDLC 各个方面的人工智能功能融为一体。该平台应提供一致的用户体验、集中的管理和数据，以及不同工具和流程之间的无缝集成。这使整个组织都能统一使用 AI 优势，减少管理多个不同的 AI 工具的开销，并为持续改进和扩展 AI 能力奠定基础。

衡量生成式人工智能在软件开发中的成功程度

为了有效地衡量实施基于人工智能的生成式软件开发体验的效果，您需要建立一套涵盖软件开发生命周期 (SDLC) 各个维度的全面指标。这些指标应反映出效率和生产力的即时改进，还应反映软件质量、团队满意度和业务价值的长期提高。

要有效使用本节中推荐的指标，请执行以下操作：

1. 建立基准 — 在深入实施人工智能驱动的开发体验之前，请花点时间收集有关这些指标中当前绩效的全面数据。这提供了一个清晰的起点，可以帮助你以后进行有意义的比较。
2. 设定切合实际的目标 — 掌握基准后，为每个指标设定可实现的改进目标。雄心勃勃，但要现实一点。请记住，可持续的进步往往是渐进的。
3. 实施持续监控-使用自动化工具持续收集和分析环境中这些指标的数据。近乎实时的监控可帮助您监控进度并快速发现任何问题或机会。
4. 定期进行审查 — 安排季度或半年一次的评估会议，让你和你的团队彻底评估你在实现目标方面的进展。利用这些会议来确定需要进一步改进的领域并庆祝您的成功。
5. 迭代和调整 — 根据您获得的见解，不断完善您的生成式 AI 实现并根据需要调整目标。

本节描述了以下类别的指标：

- 部署速度
- 代码质量
- 运营效率
- 团队工作效率和满意度
- 业务影响

部署速度

考虑衡量以下部署速度指标。

指标	说明
是时候上市了	衡量从构思到生产部署的时间缩短情况

指标	说明
冲刺速度	追踪你的队伍每次冲刺完成的故事积分增加情况
代码提交频率	监控代码提交的增加，这表明开发周期正在加快
拉取请求解决时间	评估审查和合并仓库中代码变更所花费的时间缩短的情况
释放速度	衡量每个季度或每年的发布数量的增长情况

代码质量

考虑衡量以下代码质量指标。

指标	说明
缺陷密度	衡量软件错误的减少情况
代码覆盖率	跟踪整个代码库中测试覆盖率的增加情况
技术债务	监测已确定的技术债务随着时间的推移而减少的情况
静态代码分析分数	根据您的自动分析工具评估代码质量的改进

运营效率

考虑衡量以下运营效率指标。

指标	说明
部署频率	衡量成功部署数量的增加情况
平均恢复时间 (MTTR)	跟踪从系统故障中恢复所需的时间缩短情况
更改失败率	监控导致部署失败的变更百分比的下降情况

团队工作效率和满意度

考虑衡量以下团队工作效率和满意度指标。

指标	说明
提高生产力	监控每项任务的生产率提高百分比
满意度分数	定期进行调查，衡量团队士气和工作满意度的提高
知识共享效率	衡量您的团队在搜索信息或提出重复问题上花费的时间缩短程度
入职时间	跟踪新团队成员提高工作效率所需的时间缩短情况

业务影响

考虑衡量以下业务影响指标。

指标	说明
功能采用率	使用你发布的新功能衡量用户参与度的增长
客户满意度得分	跟踪用户反馈和评分的改进情况
收入影响（直接和间接）	评估因发布速度提高或生产率提高而带来的收入增长

结论

本战略文件概述了基于人工智能的生成式软件开发体验。它探讨了 5-I 框架中的五个维度——调查、集成、互动、迭代和影响。这些维度为在整个软件开发生命周期 (SDLC) 中集成生成式人工智能提供了战略路线图。它还描述了成功实施此框架所需的基础能力。这些功能涵盖项目管理 DevSecOps、AI 助手、知识管理等领域。它提供了集成生成式 AI 时需要考虑的最佳实践，它可以帮助您使用指标来衡量生成式 AI 对您的软件开发体验的影响。

将生成式人工智能集成到软件开发流程中代表着一种范式转变，有可能加速创新、提高质量和提高生产力。但是，重要的是要认识到，这不是一次性的实施。这是一个持续的演变，需要持续的努力和不断完善。

在您踏上这一旅程时，我们建议您首先对组织当前的能力和准备情况进行全面评估。AWS 评估工具是一款基于人工智能的软件开发评估工具，可以帮助您确定优先领域并创建量身定制的实施路线图。

资源

确定关键优先领域后，以下资源可以帮助您实施路线图：

AWS 文档

- 使用 Amazon Bedrock（AWS 规范性指南）实现 AWS 基础设施运营自动化
- Amazon Q Developer 在线生成和助手代码生成的最佳实践（AWS 规范性指南）
- 使用 Amazon Bedrock 代理和知识库（AWS 规范性指导）开发基于聊天的全自动助手
- 利用@@ 生成式 AI（AWS 规范性指导）转变应用程序开发和维护运营模式 AWS
- 使用 Amazon Q Developer 作为编码助手来提高工作效率（AWS 规范性指导）

AWS 博客文章和教程

- 亚马逊 Q 博客文章
- 使用 Amazon Q 加快您的软件开发生命周期（AWS 博客文章）
- 构建 AWS 解决方案架构师 AI 代理：利用 Amazon Bedrock 实现自动化架构和部署（AWS 视频）
- 生成式人工智能驱动的技术运营（AWS 博客文章）
- 使用 Amazon Q Developer 实现您的 Java 应用程序现代化（AWS 博客文章）
- 使用 Amazon Bedrock 生成、评估和理解软件开发管道中的代码（AWS 博客文章）

文档历史记录

下表介绍了本指南的一些重要更改。如果您希望收到有关未来更新的通知，可以订阅 RSS 源。

变更	说明	日期
初次发布	不适用	2025 年 4 月 18 日

AWS 规范性指导词汇表

以下是 AWS 规范性指导提供的策略、指南和模式中的常用术语。若要推荐词条，请使用术语表末尾的提供反馈链接。

数字

7 R

将应用程序迁移到云中的 7 种常见迁移策略。这些策略以 Gartner 于 2011 年确定的 5 R 为基础，包括以下内容：

- Refactor/re-architect — 充分利用云原生功能来提高敏捷性、性能和可扩展性，从而移动应用程序并修改其架构。这通常涉及到移植操作系统和数据库。示例：将您的本地 Oracle 数据库迁移到亚马逊 Aurora PostgreSQL-Compatible 版。
- 更换平台：将应用程序迁移到云中，并进行一定程度的优化，以利用云功能。示例：将本地 Oracle 数据库迁移到 AWS Cloud中的 Amazon Relational Database Service（Amazon RDS）for Oracle。
- 重新购买：转换到其他产品，通常是从传统许可转向 SaaS 模式。示例：将您的客户关系管理 (CRM) 系统迁移到 Salesforce.com。
- 重新托管（直接迁移）：将应用程序迁移到云，无需进行任何更改即可利用云功能。示例：将本地 Oracle 数据库迁移到 AWS Cloud中 EC2 实例上的 Oracle。
- 重新放置（虚拟机监控器级直接迁移）：将基础设施迁移到云中，无需购买新硬件、重写应用程序或修改现有操作。您将服务器从本地平台迁移到同一平台的云服务中。示例：将Microsoft Hyper-V应用程序迁移到 AWS。
- 保留（重访）：将应用程序保留在源环境中。其中可能包括需要进行重大重构的应用程序，并且您希望将工作推迟到以后，以及您希望保留的遗留应用程序，因为迁移它们没有商业上的理由。
- 停用：停用或删除源环境中不再需要的应用程序。

A

A2A () Agent-to-Agent

一种支持任务委托和状态转移的代理到代理协作的状态协议。

ABAC

请参阅基于属性的访问控制。

抽象服务

请参阅托管服务。

ACID

请参阅原子性、一致性、隔离性、持久性。

主动-主动迁移

一种数据库迁移方法，在这种方法中，源数据库和目标数据库保持同步（通过使用双向复制工具或双写操作），两个数据库都在迁移期间处理来自连接应用程序的事务。这种方法支持小批量、可控的迁移，而不需要一次性割接。它比主动-被动迁移更灵活，但工作量更大。

主动-被动迁移

一种数据库迁移方法，在这种方法中，源数据库和目标数据库保持同步，但在将数据复制到目标数据库时，只有源数据库处理来自连接应用程序的事务。目标数据库在迁移期间不接受任何事务。

座席

一种能够使用工具自主推理、计划和采取行动来实现目标的人工智能系统。

特工行动

在生产环境中大规模构建、测试、部署和运行 AI 代理的操作实践。

聚合函数

一种 SQL 函数，它对一组行进行操作并计算该组的单个返回值。聚合函数的示例包括 SUM 和 MAX。

AI

请参阅人工智能。

AIOps

请参阅人工智能运营。

匿名化

永久删除数据集中个人信息的过程。匿名化可以帮助保护个人隐私。匿名化数据不再被视为个人数据。

反模式

一种用于解决反复出现的问题的常用解决方案，而在这类问题中，此解决方案适得其反、无效或不如替代方案有效。

应用程序控制

一种安全方法，仅允许使用经批准的应用程序，以帮助保护系统免受恶意软件的侵害。

应用程序组合

有关组织使用的每个应用程序的详细信息的集合，包括构建和维护该应用程序的成本及其业务价值。这些信息是产品组合发现和分析过程的关键，有助于识别需要进行迁移、现代化和优化的应用程序并确定其优先级。

人工智能（AI）

计算机科学领域致力于使用计算技术执行通常与人类相关的认知功能，例如学习、解决问题和识别模式。有关更多信息，请参阅什么是人工智能？

人工智能运营（AIOps）

使用机器学习技术解决运营问题、减少运营事故和人为干预以及提高服务质量的过程。有关如何在 AWS 迁移策略中使用 AIOps 的更多信息，请参阅运营集成指南。

非对称加密

一种加密算法，使用一对密钥，一个公钥用于加密，一个私钥用于解密。您可以共享公钥，因为它不用于解密，但对私钥的访问应受到严格限制。

原子性、一致性、隔离性、持久性（ACID）

一组软件属性，即使在出现错误、电源故障或其他问题的情况下，也能保证数据库的数据有效性和操作可靠性。

基于属性的访问权限控制（ABAC）

根据用户属性（如部门、工作角色和团队名称）创建精细访问权限的做法。有关更多信息，请参阅 AWS Identity and Access Management (I AM) 文档 AWS中的 AB AC。

权威数据来源

存储主要数据版本的位置，被认为是最可靠的信息源。您可以将数据从权威数据来源复制到其他位置，以便处理或修改数据，例如对数据进行匿名化、编辑或假名化。

可用区

中的一个不同位置 AWS 区域 ，不受其他可用区域故障的影响，并向同一区域中的其他可用区提供低成本、低延迟的网络连接。

AWS 云采用框架 (AWS CAF)

该框架包含指导方针和最佳实践 AWS ，可帮助组织制定高效且有效的计划，以成功迁移到云端。AWS CAF将指导分为六个重点领域，称为视角：业务、人员、治理、平台、安全和运营。业务、人员和治理角度侧重于业务技能和流程；平台、安全和运营角度侧重于技术技能和流程。例如，人员角度针对的是负责人力资源（HR）、人员配置职能和人员管理的利益相关者。从这个角度来看，AWS CAF 为人员发展、培训和沟通提供了指导，以帮助组织为成功采用云做好准备。有关更多信息，请参阅 AWS CAF 网站和 AWS CAF 白皮书。

AWS 工作负载资格框架 (AWS WQF)

一种评估数据库迁移工作负载、推荐迁移策略和提供工作估算的工具。 AWS WQF 包含在 AWS Schema Conversion Tool (AWS SCT) 中。它用来分析数据库架构和代码对象、应用程序代码、依赖关系和性能特征，并提供评测报告。

B

恶意机器人

一种旨在扰乱或伤害个人或组织的机器人。

BCP

请参阅业务连续性计划。

行为图

一段时间内资源行为和交互的统一交互式视图。您可以使用 Amazon Detective 的行为图来检查失败的登录尝试、可疑的 API 调用和类似的操作。有关更多信息，请参阅 Detective 文档中的行为图中的数据。

大端序系统

一个先存储最高有效字节的系统。另请参阅字节顺序。

二进制分类

一种预测二进制结果（两个可能的类别之一）的过程。例如，您的 ML 模型可能需要预测诸如“该电子邮件是否为垃圾邮件？”或“这个产品是书还是汽车？”之类的问题

bloom 筛选条件

一种概率性、内存高效的数据结构，用于测试元素是否为集合的成员。

blue/green 部署

一种部署策略，您可以创建两个独立但完全相同的环境。在一个环境中运行当前应用程序版本（蓝色），在另一个环境中运行新应用程序版本（绿色）。此策略可帮助您在影响最小的情况下快速回滚。

自动程序

一种通过互联网运行自动任务并模拟人类活动或交互的软件应用程序。有些机器人是有用或有益的，例如在互联网上索引信息的 Web 爬网程序。还有一些被称为恶意机器人的机器人，其目的是扰乱或伤害个人或组织。

僵尸网络

被恶意软件感染并受单方（称为僵尸网络控制者或僵尸网络操作者）控制的僵尸网络。僵尸网络是最著名的扩展机器人及其影响力的机制。

分支

代码存储库的一个包含区域。在存储库中创建的第一个分支是主分支。您可以从现有分支创建新分支，然后在新分支中开发功能或修复错误。为构建功能而创建的分支通常称为功能分支。当功能可以发布时，将功能分支合并回主分支。有关更多信息，请参阅关于分支（GitHub 文档）。

紧急（break-glass）访问

在特殊情况下，通过批准的流程，用户 AWS 账户 可以快速访问他们通常没有访问权限的内容。有关更多信息，请参阅指南中的 “实施破碎玻璃程序” AWS Well-Architected 指示器。

棕地策略

您环境中的现有基础设施。在为系统架构采用棕地策略时，您需要围绕当前系统和基础设施的限制来设计架构。如果您正在扩展现有基础设施，则可以将棕地策略和全新策略混合。

缓冲区缓存

存储最常访问的数据的内存区域。

业务能力

企业如何创造价值（例如，销售、客户服务或营销）。微服务架构和开发决策可以由业务能力驱动。有关更多信息，请参阅在 AWS上运行容器化微服务白皮书中的围绕业务能力进行组织部分。

业务连续性计划（BCP）

一项计划，旨在应对大规模迁移等破坏性事件对运营的潜在影响，并使企业能够快速恢复运营。

C

CAF

请参阅 AWS 云采用框架。

金丝雀部署

缓慢而渐进地向最终用户发布版本。当您确信无误后，即可部署新版本，并完全替换当前版本。

CCoE

请参阅云卓越中心。

CDC

请参阅更改数据捕获。

更改数据捕获（CDC）

跟踪数据来源（如数据库表）的更改并记录有关更改的元数据的过程。您可以将 CDC 用于各种目的，例如审计或复制目标系统中的更改以保持同步。

混沌工程

故意引入故障或破坏性事件来测试系统的韧性。您可以使用 AWS Fault Injection Service (AWS FIS) 来执行实验，对您的 AWS 工作负载施加压力并评估其响应。

CI/CD

请参阅持续集成和持续交付。

分类

一种有助于生成预测的分类流程。分类问题的 ML 模型预测离散值。离散值始终彼此不同。例如，一个模型可能需要评估图像中是否有汽车。

公民开发者

使用无code/low代码平台创建 AI 应用程序但没有专业技术技能的企业用户。

客户端加密

在目标 AWS 服务 收到数据之前，对数据进行本地加密。

云卓越中心（CCoE）

一个多学科团队，负责推动整个组织的云采用工作，包括开发云最佳实践、调动资源、制定迁移时间表、领导组织完成大规模转型。有关更多信息，请参阅 AWS Cloud 企业战略博客上的 CCoE 帖子。

云计算

通常用于远程数据存储和 IoT 设备管理的云技术。云计算通常连接到边缘计算技术。

云运营模型

在 IT 组织中，一种用于构建、完善和优化一个或多个云环境的运营模型。有关更多信息，请参阅构建您的云运营模型。

云采用阶段

组织迁移到 AWS Cloud中时通常会经历四个阶段：

- 项目 - 出于概念验证和学习目的，开展一些与云相关的项目
- 基础 - 进行基础投资以扩大云采用率（例如，创建登录区、定义 CCoE、建立运营模型）
- 迁移 - 迁移单个应用程序
- Re-invention — 优化产品和服务，在云端进行创新

Stephen Orban在 AWS Cloud 企业战略博客的博客文章《走向之旅 Cloud-First 和采用阶段》中定义了这些阶段。有关它们与 AWS 迁移策略的关系的信息，请参阅迁移准备指南。

CMDB

请参阅配置管理数据库。

代码存储库

通过版本控制过程存储和更新源代码和其他资产（如文档、示例和脚本）的位置。常见的云存储库包括 GitHub 或 Bitbucket Cloud。每个版本的代码都称为一个分支。在微服务结构中，每个存储库都专门用于一个功能。单个 CI/CD管道可以使用多个存储库。

冷缓存

一种空的、填充不足或包含过时或不相关数据的缓冲区缓存。这会影响性能，因为数据库实例必须从主内存或磁盘读取，这比从缓冲区缓存读取要慢。

冷数据

很少访问的数据，且通常是历史数据。查询此类数据时，通常可以接受慢速查询。将这些数据转移到性能较低且成本更低的存储层或类别可以降低成本。

计算机视觉（CV）

一种 AI 领域，它使用机器学习来分析和提取数字图像和视频等视觉格式中的信息。例如，Amazon SageMaker AI 为 CV 提供了图像处理算法。

配置偏移

对于工作负载而言，一种偏离预期状态的配置更改。这可能会导致工作负载变得不合规，且通常是渐进的，不是故意的。

配置管理数据库（CMDB）

一种存储库，用于存储和管理有关数据库及其 IT 环境的信息，包括硬件和软件组件及其配置。您通常在迁移的产品组合发现和分析阶段使用来自 CMDB 的数据。

合规性包

一系列 AWS Config 规则和补救措施，您可以汇编这些规则和补救措施，以自定义合规性和安全性检查。您可以使用 YAML 模板将一致性包作为单个实体部署在 AWS 账户 和区域或整个组织中。有关更多信息，请参阅 AWS Config 文档中的一致性包。

持续集成和持续交付 (CI/CD)

自动执行软件发布过程的源代码、构建、测试、暂存和生产阶段的过程。 CI/CD 通常被描述为管道。 CI/CD 可以帮助您实现流程自动化、提高生产力、提高代码质量和更快地交付。有关更多信息，请参阅持续交付的优势。CD 也可以表示持续部署。有关更多信息，请参阅持续交付与持续部署。

CV

请参阅计算机视觉。

D

静态数据

网络中静止的数据，例如存储中的数据。

数据分类

根据网络中数据的关键性和敏感性对其进行识别和分类的过程。它是任何网络安全风险管理策略的关键组成部分，因为它可以帮助您确定对数据的适当保护和保留控制。数据分类是《 AWS Well-Architected 框架》中安全支柱的组成部分。有关详细信息，请参阅数据分类。

数据漂移

生产数据与用来训练机器学习模型的数据之间的有意义差异，或者输入数据随时间推移的有意义变化。数据漂移可能降低机器学习模型预测的整体质量、准确性和公平性。

传输中数据

在网络中主动移动的数据，例如在网络资源之间移动的数据。

数据网格

一种架构框架，可提供分布式、去中心化的数据所有权以及集中式管理和治理。

数据最少化

仅收集并处理绝对必要数据的原则。在中进行数据最小化 AWS Cloud 可以降低隐私风险、成本和分析碳足迹。

数据边界

AWS 环境中的一组预防性防护措施，可帮助确保只有可信身份才能访问来自预期网络的可信资源。有关更多信息，请参阅在上构建数据边界。 AWS

数据预处理

将原始数据转换为 ML 模型易于解析的格式。预处理数据可能意味着删除某些列或行，并处理缺失、不一致或重复的值。

数据溯源

在数据的整个生命周期跟踪其来源和历史的过程，例如数据如何生成、传输和存储。

数据主体

正在收集和处理其数据的个人。

数据仓库

一种支持商业智能（例如分析）的数据管理系统。数据仓库通常包含大量历史数据，通常用于查询和分析。

数据库定义语言（DDL）

在数据库中创建或修改表和对象结构的语句或命令。

数据库操作语言（DML）

在数据库中修改（插入、更新和删除）信息的语句或命令。

DDL

请参阅数据库定义语言。

深度融合

组合多个深度学习模型进行预测。您可以使用深度融合来获得更准确的预测或估算预测中的不确定性。

深度学习

一个 ML 子字段使用多层人工神经网络来识别输入数据和感兴趣的目标变量之间的映射。

深度防御

一种信息安全方法，经过深思熟虑，在整个计算机网络中分层实施一系列安全机制和控制措施，以保护网络及其中数据的机密性、完整性和可用性。当你采用这种策略时 AWS，你会在 AWS Organizations 结构的不同层面添加多个控件来帮助保护资源。例如，深度防御方法可能将多因素身份验证、网络分段和加密结合起来。

委派管理员

在中 AWS Organizations，兼容的服务可以注册 AWS 成员帐户来管理组织的帐户并管理该服务的权限。此账户被称为该服务的委托管理员。有关更多信息和兼容服务列表，请参阅 AWS Organizations 文档中使用 AWS Organizations的服务。

部署

使应用程序、新功能或代码修复在目标环境中可用的过程。部署涉及在代码库中实现更改，然后在应用程序的环境中构建和运行该代码库。

开发环境

请参阅环境。

侦测性控制

一种安全控制，在事件发生后进行检测、记录日志和发出提醒。这些控制是第二道防线，提醒您注意绕过现有预防性控制的安全事件。有关更多信息，请参阅在 AWS上实施安全控制中的侦测性控制。

开发价值流映射（DVSM）

用于识别对软件开发生命周期中的速度和质量产生不利影响的限制因素并确定其优先级的流程。DVSM 扩展了最初为精益生产实践设计的价值流映射流程。其重点关注在软件开发过程中创造和转移价值所需的步骤和团队。

数字孪生

真实世界系统的虚拟再现，如建筑物、工厂、工业设备或生产线。数字孪生支持预测性维护、远程监控和生产优化。

维度表

星型架构中的一种较小的表，其中包含事实表中定量数据的数据属性。维度表属性通常是文本字段或行为类似于文本的离散数字。这些属性通常用于查询约束、筛选和结果集标注。

灾难

阻止工作负载或系统在其主要部署位置实现其业务目标的事件。这些事件可能是自然灾害、技术故障或人为操作的结果，例如无意的配置错误或恶意软件攻击。

灾难恢复（DR）

您用来最大程度地减少由灾难造成的停机时间和数据丢失的策略和流程。有关更多信息，请参阅《工作负载灾难恢复 AWS： AWS Well-Architected 框架中的云端恢复》。

DML

请参阅数据库操作语言。

领域驱动设计

一种开发复杂软件系统的方法，通过将其组件连接到每个组件所服务的不断发展的领域或核心业务目标。埃里克·埃文斯（Eric Evans）在他的《Domain-Driven 设计：解决软件核心的复杂性》（波士顿： Addison-Wesley 专业版，2003年）一书中介绍了这个概念。有关如何使用带有 strangler fig 模式的域驱动设计的信息，请参阅使用容器和 Amazon API Gateway 逐步实现传统微软 ASP.NET (ASMX) 网络服务的现代化。

DR

请参阅灾难恢复。

偏差检测

跟踪与基准配置的偏差。例如，您可以使用 AWS CloudFormation 来检测系统资源中的偏差，也可以使用 AWS Control Tower 来检测着陆区中可能影响监管要求合规性的变化。

DVSM

请参阅开发价值流映射。

E

EDA

请参阅探索性数据分析。

EDI

请参阅电子数据交换。

边缘计算

该技术可提高位于 IoT 网络边缘的智能设备的计算能力。与云计算比较时，边缘计算可以减少通信延迟并缩短响应时间。

电子数据交换（EDI）

组织之间业务文件的自动交换。有关更多信息，请参阅什么是电子数据交换。

加密

一种将人类可读的纯文本数据转换为加密文字的计算流程。

加密密钥

由加密算法生成的随机位的加密字符串。密钥的长度可能有所不同，而且每个密钥都设计为不可预测且唯一。

字节顺序

字节在计算机内存中的存储顺序。 Big-endian 系统首先存储最重要的字节。 Little-endian 系统首先存储最低有效字节。

端点

请参阅服务端点。

端点服务

一种可以在虚拟私有云（VPC）中托管，与其他用户共享的服务。您可以使用其他 AWS 账户 或 AWS Identity and Access Management (IAM) 委托人创建终端节点服务， AWS PrivateLink 并向其授予权限。这些账户或主体可通过创建接口 VPC 端点来私密地连接到您的端点服务。有关更多信息，请参阅 Amazon Virtual Private Cloud（Amazon VPC）文档中的创建端点服务。

企业资源规划（ERP）

一种自动化和管理企业关键业务流程（例如会计、MES 和项目管理）的系统。

信封加密

用另一个加密密钥对加密密钥进行加密的过程。有关更多信息，请参阅 AWS Key Management Service (AWS KMS) 文档中的信封加密。

环境

正在运行的应用程序的实例。以下是云计算中常见的环境类型：

- 开发环境 — 正在运行的应用程序的实例，只有负责维护应用程序的核心团队才能使用。开发环境用于测试更改，然后再将其提升到上层环境。这类环境有时称为测试环境。
- 下层环境 — 应用程序的所有开发环境，比如用于初始构建和测试的环境。
- 生产环境 — 最终用户可以访问的正在运行的应用程序的实例。在 CI/CD 管道中，生产环境是最后一个部署环境。
- 上层环境 — 除核心开发团队以外的用户可以访问的所有环境。这可能包括生产环境、预生产环境和用户验收测试环境。

epic

在敏捷方法学中，有助于组织工作和确定优先级的功能类别。epics 提供了对需求和实施任务的总体描述。例如， AWS CAF 安全史诗包括身份和访问管理、侦探控制、基础设施安全、数据保护和事件响应。有关 AWS 迁移策略中 epics 的更多信息，请参阅计划实施指南。

ERP

请参阅企业资源规划。

探索性数据分析（EDA）

分析数据集以了解其主要特征的过程。您收集或汇总数据，并进行初步调查，以发现模式、检测异常并检查假定情况。EDA 通过计算汇总统计数据和创建数据可视化得以执行。

F

事实表

星型架构中的中心表。它存储有关业务运营的定量数据。通常，事实表包含两种类型的列：包含度量的列和包含维度表外键的列。

快速失效机制

一种使用频繁且增量式的测试来缩短开发生命周期的理念。这是敏捷方法的关键部分。

故障隔离边界

在中 AWS Cloud，诸如可用区 AWS 区域、控制平面或数据平面之类的边界，它限制了故障的影响并有助于提高工作负载的弹性。有关更多信息，请参阅 AWS 故障隔离边界。

功能分支

请参阅分支。

特征

您用来进行预测的输入数据。例如，在制造环境中，特征可能是定期从生产线捕获的图像。

特征重要性

特征对于模型预测的重要性。这通常表示为数值分数，可以通过各种技术进行计算，例如 Shapley 加法解释（SHAP）和积分梯度。有关更多信息，请参阅机器学习模型的可解释性 AWS。

功能转换

为 ML 流程优化数据，包括使用其他来源丰富数据、扩展值或从单个数据字段中提取多组信息。这使得 ML 模型能从数据中获益。例如，如果您将“2021-05-27 00:15:37”日期分解为“2021”、“五月”、“星期四”和“15”，则可以帮助学习与不同数据成分相关的算法学习精细模式。

少样本提示

在要求 LLM 执行类似任务之前，先向其提供少量示例，以演示任务和预期输出。这种技术是情境学习的应用，模型可以从提示中嵌入的示例（镜头）中学习。 Few-shot 对于需要特定格式、推理或领域知识的任务，提示可能非常有效。另请参阅零样本提示。

FGAC

请参阅精细访问控制。

精细访问控制（FGAC）

使用多个条件允许或拒绝访问请求。

快闪迁移

一种数据库迁移方法，通过更改数据捕获使用连续数据复制，在极短的时间内迁移数据，而非使用分阶段方法。目标是将停机时间降至最低。

FM

请参阅基础模型。

基础模型（FM）

一个大型深度学习神经网络，它已使用海量的通用和未标注数据集进行训练。FM 能够执行各种常规任务，例如理解语言、生成文本和图像以及使用自然语言进行对话。有关更多信息，请参阅什么是基础模型。

FM 网关

一种集中式中介，用于控制和规范对基础模型的访问。也称为 LLM 网关。

G

生成式人工智能

AI 模型的一个子集，这些模型已经过大量数据训练，可以使用简单的文本提示来创建新的内容和构件，例如图像、视频、文本和音频。有关更多信息，请参阅什么是生成式人工智能。

地理阻止

请参阅地理限制。

地理限制（地理阻止）

在 Amazon 中 CloudFront，一种阻止特定国家/地区的用户访问内容分发的选项。您可以使用允许列表或阻止列表来指定已批准和已禁止的国家/地区。有关更多信息，请参阅 CloudFront 文档中的限制内容的地理分布。

GitFlow 工作流程

一种方法，在这种方法中，下层和上层环境在源代码存储库中使用不同的分支。Gitflow 工作流程被认为是传统的工作流程，而基于中继的工作流程则是现代的、首选的方法。

黄金映像

系统或软件的快照，用作部署该系统或软件的新实例的模板。例如，在制造业中，黄金映像可用于在多个设备上预调配软件，并有助于提高设备制造操作的速度、可扩展性和生产效率。

全新策略

在新环境中缺少现有基础设施。在对系统架构采用全新策略时，您可以选择所有新技术，而不受对现有基础设施（也称为棕地）兼容性的限制。如果您正在扩展现有基础设施，则可以将棕地策略和全新策略混合。

防护机制

一种高级规则，用于跨组织单位（OU）管理资源、策略和合规性。预防性防护机制会执行策略以确保符合合规性标准。它们是使用服务控制策略和 IAM 权限边界实现的。侦测性护栏会检测策略违规和合规性问题，并生成提醒以进行修复。它们通过使用 AWS Config、、Amazon、 AWS Security Hub CSPM GuardDuty AWS Trusted Advisor、Amazon Inspector 和自定义 AWS Lambda 支票来实现。

护栏 (AI)

用于过滤、验证和限制代理输入和输出的安全机制，有助于确保负责任和安全的 AI 行为。

H

HA

请参阅高可用性。

异构数据库迁移

将源数据库迁移到使用不同数据库引擎的目标数据库（例如，从 Oracle 迁移到 Amazon Aurora）。异构迁移通常是重新架构工作的一部分，而转换架构可能是一项复杂的任务。AWS 提供了 AWS SCT 来帮助实现架构转换。

高可用性（HA）

在遇到挑战或灾难时，工作负载无需干预即可连续运行的能力。HA 系统旨在自动进行故障转移、持续提供良好性能，并以最小的性能影响处理不同负载和故障。

历史数据库现代化

一种用于实现运营技术（OT）系统现代化和升级以更好满足制造业需求的方法。历史数据库是一种用于收集和存储工厂中各种来源数据的数据库。

保留数据

从用于训练机器学习模型的数据集中保留的一部分标注的历史数据。通过将模型预测与保留数据进行比较，您可以使用保留数据来评估模型性能。

人机在圈 (HitL)

一种工作流程模式，其中代理执行在关键决策点暂停以供人工审查和批准。

同构数据库迁移

将源数据库迁移到共享同一数据库引擎的目标数据库（例如，从 Microsoft SQL Server 迁移到 Amazon RDS for SQL Server）。同构迁移通常是更换主机或更换平台工作的一部分。您可以使用本机数据库实用程序来迁移架构。

热数据

经常访问的数据，例如实时数据或近期的转化数据。这些数据通常需要高性能存储层或存储类别才能提供快速的查询响应。

修补程序

针对生产环境中关键问题的紧急修复。由于其紧迫性，修补程序通常是在典型的 DevOps 发布工作流程之外进行的。

hypercare 周期

割接之后，迁移团队立即管理和监控云中迁移的应用程序以解决任何问题的时间段。通常，这个周期持续 1-4 天。在 hypercare 周期结束时，迁移团队通常会将应用程序的责任移交给云运营团队。

我

IaC

请参阅基础设施即代码。

基于身份的策略

附加到一个或多个 IAM 委托人的策略，用于定义他们在 AWS Cloud 环境中的权限。

空闲应用程序

90 天内平均 CPU 和内存使用率在 5% 到 20% 之间的应用程序。在迁移项目中，通常会停用这些应用程序或将其保留在本地。

IIoT

请参阅工业物联网。

不可变基础设施

一种模型，可为生产工作负载部署新的基础设施，而不是更新、修补或修改现有基础设施。不可变基础设施本质上比可变基础设施更一致、更可靠、更可预测。有关更多信息，请参阅框架中的使用不可变基础架构部署最佳实践。 AWS Well-Architected

入站（入口）VPC

在 AWS 多账户架构中，一种接受、检查和路由来自应用程序外部的网络连接的 VPC。AWS 安全参考架构建议使用入站、出站和检查 VPC 设置网络账户，保护应用程序与广泛的互联网之间的双向接口。

增量迁移

一种割接策略，在这种策略中，您可以将应用程序分成小部分进行迁移，而不是一次性完整割接。例如，您最初可能只将几个微服务或用户迁移到新系统。在确认一切正常后，您可以逐步迁移其他微服务或用户，直到停用遗留系统。这种策略降低了大规模迁移带来的风险。

工业 4.0

该术语由克劳斯·施瓦布（Klaus Schwab）在2016年推出，指的是通过连接性、实时数据、自动化、分析和的进步实现制造流程的现代化。 AI/ML

基础设施

应用程序环境中包含的所有资源和资产。

基础设施即代码（IaC）

通过一组配置文件预调配和管理应用程序基础设施的过程。IaC 旨在帮助您集中管理基础设施、实现资源标准化和快速扩展，使新环境具有可重复性、可靠性和一致性。

工业物联网（IIoT）

在工业领域使用联网的传感器和设备，例如制造业、能源、汽车、医疗保健、生命科学和农业。有关更多信息，请参阅制定工业物联网（IIoT）数字化转型策略。

检查 VPC

在 AWS 多账户架构中，一种集中式 VPC，用于管理 VPC（相同或不同 AWS 区域）、互联网和本地网络之间的网络流量检查。AWS 安全参考架构建议使用入站、出站和检查 VPC 设置网络账户，保护应用程序与广泛的互联网之间的双向接口。

物联网（IoT）

由带有嵌入式传感器或处理器的连接物理对象组成的网络，这些传感器或处理器通过互联网或本地通信网络与其他设备和系统进行通信。有关更多信息，请参阅什么是 IoT？

可解释性

它是机器学习模型的一种特征，描述了人类可以理解模型的预测如何取决于其输入的程度。有关更多信息，请参阅机器学习模型的可解释性 AWS。

物联网

请参阅物联网。

IT 信息库（ITIL）

提供 IT 服务并使这些服务符合业务要求的一套最佳实践。ITIL 是 ITSM 的基础。

IT 服务管理（ITSM）

为组织设计、实施、管理和支持 IT 服务的相关活动。有关将云运营与 ITSM 工具集成的信息，请参阅运营集成指南。

ITIL

请参阅 IT 信息库。

ITSM

请参阅 IT 服务管理。

L

基于标签的访问控制（LBAC）

强制访问控制（MAC）的一种实施方式，其中明确为用户和数据本身分配了安全标签值。用户安全标签和数据安全标签之间的交集决定了用户可以看到哪些行和列。

登录区

landing zone 是一个架构精良的多账户 AWS 环境，具有可扩展性和安全性。这是一个起点，您的组织可以从这里放心地在安全和基础设施环境中快速启动和部署工作负载和应用程序。有关登录区的更多信息，请参阅设置安全且可扩展的多账户 AWS 环境。

大语言模型（LLM）

一种基于大量数据进行预训练的深度学习 AI 模型。LLM 可以执行多项任务，例如回答问题、总结文档、将文本翻译成其他语言以及完成句子。有关更多信息，请参阅什么是 LLM。

大规模迁移

迁移 300 台或更多服务器。

LBAC

请参阅基于标签的访问控制。

最低权限

授予执行任务所需的最低权限的最佳安全实践。有关更多信息，请参阅 IAM 文档中的应用最低权限许可。

直接迁移

请参阅 7 R。

小端序系统

一个先存储最低有效字节的系统。另请参阅字节顺序。

LLM

请参阅大型语言模型。

下层环境

请参阅环境。

M

机器学习（ML）

一种使用算法和技术进行模式识别和学习的人工智能。ML 对记录的数据（例如物联网（IoT）数据）进行分析和学习，以生成基于模式的统计模型。有关更多信息，请参阅机器学习。

主分支

请参阅分支。

恶意软件

旨在危害计算机安全或隐私的软件。恶意软件可能会破坏计算机系统、泄露敏感信息或获得未经授权的访问权限。恶意软件的示例包括病毒、蠕虫、勒索软件、木马、间谍软件和键盘记录器。

托管式服务

AWS 服务 它 AWS 运行基础设施层、操作系统和平台，您可以访问端点来存储和检索数据。Amazon Simple Storage Service（Amazon S3）和 Amazon DynamoDB 就是托管服务的示例。这些服务也称为抽象服务。

制造执行系统（MES）

一种软件系统，用于跟踪、监控、记录和控制在车间将原材料转化为成品的生产过程。

MAP

请参阅迁移加速计划。

MCP

参见模型上下文协议。

模型上下文协议（MCP）

一种用于代理与工具通信的无状态协议。

MCP 服务器

一种通过模型上下文协议公开一个或多个工具的服务。

机制

一个完整的过程，您可以在其中创建工具，推动工具的采用，然后检查结果以进行调整。机制是一种在运作过程中自我强化和改善的循环。有关更多信息，请参阅在 AWS Well-Architected 框架中构建机制。

成员账户

AWS 账户 除属于组织中的管理账户之外的所有账户 AWS Organizations。一个账户一次只能是一个组织的成员。

MES

请参阅制造执行系统。

消息队列遥测传输（MQTT）

一种基于publish/subscribe模式的轻量级机器对机器 (M2M) 通信协议，适用于资源受限的物联网设备。

微服务

一种小型独立服务，通过明确定义的 API 进行通信，通常由小型独立团队拥有。例如，保险系统可能包括映射到业务能力（如销售或营销）或子域（如购买、理赔或分析）的微服务。微服务的好处包括敏捷、灵活扩展、易于部署、可重复使用的代码和恢复能力。有关更多信息，请参阅使用 AWS 无服务器服务集成微服务。

微服务架构

一种使用独立组件构建应用程序的方法，这些组件将每个应用程序进程作为微服务运行。这些微服务使用轻量级 API 通过明确定义的接口进行通信。该架构中的每个微服务都可以更新、部署和扩展，以满足对应用程序特定功能的需求。有关更多信息，请参阅在上实现微服务。 AWS

迁移加速计划（MAP）

AWS 该计划提供咨询支持、培训和服务，以帮助组织为迁移到云奠定坚实的运营基础，并帮助抵消迁移的初始成本。MAP 提供了一种以系统的方式执行遗留迁移的迁移方法，以及一套用于自动执行和加速常见迁移场景的工具。

大规模迁移

将大部分应用程序组合分波迁移到云中的过程，在每一波中以更快的速度迁移更多应用程序。本阶段使用从早期阶段获得的最佳实践和经验教训，实施由团队、工具和流程组成的迁移工厂，通过自动化和敏捷交付简化工作负载的迁移。这是 AWS 迁移策略的第三阶段。

迁移工厂

Cross-functional 通过自动化、敏捷的方法简化工作负载迁移的团队。迁移工厂团队通常包括运营、业务分析师和所有者、迁移工程师、开发 DevOps 人员和冲刺专业人员。20% 到 50% 的企业应用程序组合由可通过工厂方法优化的重复模式组成。有关更多信息，请参阅本内容集中有关迁移工厂的讨论和云迁移工厂指南。

迁移元数据

有关完成迁移所需的应用程序和服务器的信息。每种迁移模式都需要一套不同的迁移元数据。迁移元数据的示例包括目标子网、安全组和 AWS 账户。

迁移模式

一种可重复的迁移任务，详细列出了迁移策略、迁移目标以及所使用的迁移应用程序或服务。示例：使用 AWS 应用程序迁移服务重新托管向 Amazon EC2 的迁移。

迁移组合评测（MPA）

一种在线工具，提供了用于验证迁移到 AWS Cloud的业务案例的信息。MPA 提供了详细的组合评测（服务器规模调整、定价、TCO 比较、迁移成本分析）以及迁移计划（应用程序数据分析和数据收集、应用程序分组、迁移优先级排序和波次规划）。所有 AWS 顾问和 APN 合作伙伴顾问均可免费使用 MPA 工具（需要登录）。

迁移准备情况评测（MRA）

使用 AWS CAF 深入了解组织的云就绪状态、确定优势和劣势以及制定行动计划以缩小已发现差距的过程。有关更多信息，请参阅迁移准备指南。MRA 是 AWS 迁移策略的第一阶段。

迁移策略

将工作负载迁移到 AWS Cloud的方法。有关更多信息，请参见术语表中的 7 R 词条，以及动员您的组织以加快大规模迁移。

ML

请参阅机器学习。

现代化

将过时的（原有的或单体）应用程序及其基础设施转变为云中敏捷、弹性和高度可用的系统，以降低成本、提高效率和利用创新。有关更多信息，请参阅在 AWS Cloud中实现应用程序现代化的策略。

现代化准备情况评估

一种评估方式，有助于确定组织应用程序的现代化准备情况；确定收益、风险和依赖关系；确定组织能够在多大程度上支持这些应用程序的未来状态。评估结果是目标架构的蓝图、详细说明现代化进程发展阶段和里程碑的路线图以及解决已发现差距的行动计划。有关更多信息，请参阅在 AWS Cloud中评估应用程序的现代化准备情况。

单体应用程序（单体式）

作为具有紧密耦合进程的单个服务运行的应用程序。单体应用程序有几个缺点。如果某个应用程序功能的需求激增，则必须扩展整个架构。随着代码库的增长，添加或改进单体应用程序的功能也会变得更加复杂。若要解决这些问题，可以使用微服务架构。有关更多信息，请参阅将单体分解为微服务。

MPA

请参阅迁移组合评测。

MQTT

请参阅消息队列遥测传输。

多分类器

一种帮助为多个类别生成预测（预测两个以上结果之一）的过程。例如，ML 模型可能会询问“这个产品是书、汽车还是手机？”或“此客户最感兴趣什么类别的产品？”

可变基础设施

一种用于更新和修改生产工作负载的现有基础设施的模型。为了提高一致性、可靠性和可预测性，该 AWS Well-Architected 框架建议使用不可变基础设施作为最佳实践。

O

OAC

请参阅来源访问控制。

OAI

请参阅来源访问身份。

OCM

请参阅组织变革管理。

离线迁移

一种迁移方法，在这种方法中，源工作负载会在迁移过程中停止运行。这种方法会延长停机时间，通常用于小型非关键工作负载。

OI

请参阅运营集成。

OLA

请参阅运营级别协议。

在线迁移

一种迁移方法，在这种方法中，源工作负载无需离线即可复制到目标系统。在迁移过程中，连接工作负载的应用程序可以继续运行。这种方法的停机时间为零或最短，通常用于关键生产工作负载。

OPC-UA

请参阅开放流程通信 – 统一架构。

开放流程通信-统一架构 (OPC-UA)

一种用于工业自动化的机器对机器 (M2M) 通信协议。 OPC-UA 提供了数据加密、身份验证和授权方案的互操作性标准。

运营级别协议（OLA）

一项协议，阐明了 IT 职能部门承诺相互交付的内容，以支持服务水平协议（SLA）。

运营准备情况审查（ORR）

一份问题核对清单和关联的最佳实践，可帮助您了解、评估、预防或缩小事件和可能的故障的范围。有关更多信息，请参阅 AWS Well-Architected 框架中的运营准备情况审查 (ORR)。

运营技术（OT）

与物理环境配合使用以控制工业运营、设备和基础设施的硬件和软件系统。在制造业中，OT 和信息技术（IT）系统的集成是工业 4.0 转型的关键重点。

运营整合（OI）

在云中实现运营现代化的过程，包括就绪计划、自动化和集成。有关更多信息，请参阅运营整合指南。

组织跟踪

由 AWS CloudTrail 此创建的跟踪记录组织 AWS 账户 中所有人的所有事件 AWS Organizations。该跟踪是在每个 AWS 账户 中创建的，属于组织的一部分，并跟踪每个账户的活动。有关更多信息，请参阅 CloudTrail文档中的为组织创建跟踪。

组织变革管理（OCM）

一个从人员、文化和领导力角度管理重大、颠覆性业务转型的框架。OCM 通过加快变革采用、解决过渡问题以及推动文化和组织变革，帮助组织为新系统和战略做好准备和过渡。在 AWS 迁移策略中，该框架被称为人员加速，因为云采用项目需要变更的速度。有关更多信息，请参阅 OCM 指南。

来源访问控制（OAC）

在中 CloudFront，一个增强的选项，用于限制访问以保护您的亚马逊简单存储服务 (Amazon S3) 内容。OAC 全部支持所有 S3 存储桶 AWS 区域、使用 AWS KMS (SSE-KMS) 进行服务器端加密，以及对 S3 存储桶的动态PUT和DELETE请求。

来源访问身份（OAI）

在中 CloudFront，一个用于限制访问权限以保护您的 Amazon S3 内容的选项。当您使用 OAI 时，CloudFront 会创建一个 Amazon S3 可以对其进行身份验证的委托人。经过身份验证的委托人只能通过特定 CloudFront 分配访问 S3 存储桶中的内容。另请参阅 OAC，其中提供了更精细和增强的访问控制。

ORR

请参阅运营准备情况审查。

OT

请参阅运营技术。

出站（出口）VPC

在 AWS 多账户架构中，一种处理从应用程序内部启动的网络连接的 VPC。AWS 安全参考架构建议使用入站、出站和检查 VPC 设置网络账户，保护应用程序与广泛的互联网之间的双向接口。

P

权限边界

附加到 IAM 主体的 IAM 管理策略，用于设置用户或角色可以拥有的最大权限。有关更多信息，请参阅 IAM 文档中的权限边界。

个人身份信息（PII）

直接查看其他相关数据或与之配对时可用于合理推断个人身份的信息。PII 的示例包括姓名、地址和联系信息。

PII

请参阅个人身份信息。

playbook

一套预定义的步骤，用于捕获与迁移相关的工作，例如在云中交付核心运营功能。playbook 可以采用脚本、自动化运行手册的形式，也可以是操作现代化环境所需的流程或步骤的摘要。

PLC

请参阅可编程逻辑控制器。

PLM

请参阅产品生命周期管理。

policy

一个对象，可以定义权限（请参阅基于身份的策略）、指定访问条件（请参阅基于资源的策略）或定义 AWS Organizations 的组织中所有账户的最大权限（请参阅服务控制策略）。

多语言持久性

根据数据访问模式和其他要求，独立选择微服务的数据存储技术。如果您的微服务采用相同的数据存储技术，它们可能会遇到实现难题或性能不佳。如果微服务使用最适合其需求的数据存储，则可以更轻松地实现微服务，并获得更好的性能和可扩展性。

组合评测

一个发现、分析和确定应用程序组合优先级以规划迁移的过程。有关更多信息，请参阅评估迁移准备情况。

谓词

返回 `true` 或 `false` 的查询条件，通常位于 `WHERE` 子句中。

谓词下推

一种数据库查询优化技术，可在传输之前筛选查询中的数据。这将减少从关系数据库检索和处理的数据量，并提高查询性能。

预防性控制

一种安全控制，旨在防止事件发生。这些控制是第一道防线，帮助防止未经授权的访问或对网络的意外更改。有关更多信息，请参阅在 AWS上实施安全控制中的预防性控制。

主体

中 AWS 可以执行操作和访问资源的实体。此实体通常是 IAM 角色的根用户或用户。 AWS 账户有关更多信息，请参阅 IAM 文档中角色术语和概念中的主体。

隐私设计

一种在整个开发过程中都考虑隐私的系统工程方法。

私有托管区

私有托管区就是一个容器，其中包含的信息说明您希望 Amazon Route 53 如何响应一个或多个 VPC 中的某个域及其子域的 DNS 查询。有关更多信息，请参阅 Route 53 文档中的私有托管区的使用。

主动控制

一种安全控制，旨在防止部署不合规资源。这些控制会在资源预置之前对其进行扫描。如果资源与控制不兼容，则不会预置它。有关更多信息，请参阅 AWS Control Tower 文档中的控制参考指南，并参见在上实施安全控制中的主动控制 AWS。

产品生命周期管理（PLM）

对产品在其整个生命周期内的数据和流程的管理，从设计、开发和发布，到增长和成熟，再到衰退和淘汰。

生产环境

请参阅环境。

可编程逻辑控制器（PLC）

在制造业中，一种高度可靠、适应性强的计算机，用于监控机器并实现制造过程自动化。

提示串接

使用一个 LLM 提示的输出作为下一个提示的输入，以生成更好的响应。该技术用于将复杂的任务分解为子任务，或者迭代地完善或扩展初步响应。它有助于提高模型响应的准确性和相关性，并允许获得更精细的个性化结果。

假名化

用占位符值替换数据集中个人标识符的过程。假名化可以帮助保护个人隐私。假名化数据仍被视为个人数据。

publish/subscribe (pub/sub)

一种支持微服务间异步通信的模式，可提高可扩展性和响应能力。例如，在基于微服务的 MES 中，微服务可以将事件消息发布到其他微服务可以订阅的频道。系统可以在不更改发布服务的情况下添加新的微服务。

Q

查询计划

一系列用于访问 SQL 关系数据库系统中的数据的步骤，类似于指令。

查询计划回归

当数据库服务优化程序选择的最佳计划不如数据库环境发生特定变化之前时。这可能是由统计数据、约束、环境设置、查询参数绑定更改和数据库引擎更新造成的。

R

RACI 矩阵

请参阅责任、问责、咨询和知情（RACI）。

RAG

请参阅检索增强生成。

勒索软件

一种恶意软件，旨在阻止对计算机系统或数据的访问，直到付款为止。

RASCI 矩阵

请参阅责任、问责、咨询和知情（RACI）。

RCAC

请参阅行列访问控制。

只读副本

用于只读目的的数据库副本。您可以将查询路由到只读副本，以减轻主数据库的负载。

重新架构

请参阅 7 R。

恢复点目标（RPO）

自上一个数据恢复点以来可接受的最长时间。这决定了从上一个恢复点到服务中断之间可接受的数据丢失情况。

恢复时间目标（RTO）

服务中断和服务恢复之间可接受的最大延迟。

重构

请参阅 7 R。

Region

地理区域内的 AWS 资源集合。每一个 AWS 区域 都相互隔离，彼此独立，以提供容错、稳定性和弹性。有关更多信息，请参阅指定您的账户可以使用的 AWS 区域。

回归

一种预测数值的 ML 技术。例如，要解决“这套房子的售价是多少？”的问题 ML 模型可以使用线性回归模型，根据房屋的已知事实（如建筑面积）来预测房屋的销售价格。

重新托管

请参阅 7 R。

版本

在部署过程中，推动生产环境变更的行为。

重新放置

请参阅 7 R。

更换平台

请参阅 7 R。

重新购买

请参阅 7 R。

韧性

应用程序抵御中断或从中断中恢复的能力。在 AWS Cloud中规划韧性时，高可用性和灾难恢复是常见的考虑因素。有关更多信息，请参阅 AWS Cloud 韧性。

基于资源的策略

一种附加到资源的策略，例如 AmazonS3 存储桶、端点或加密密钥。此类策略指定了允许哪些主体访问、支持的操作以及必须满足的任何其他条件。

责任、问责、咨询和知情（RACI）矩阵

定义参与迁移活动和云运营的所有各方的角色和责任的矩阵。矩阵名称源自矩阵中定义的责任类型：负责（R）、问责（A）、咨询（C）和知情（I）。支持（S）类型是可选的。如果包括支持，则该矩阵称为 RASCI 矩阵，如果将其排除在外，则称为 RACI 矩阵。

响应性控制

一种安全控制，旨在推动对不良事件或偏离安全基线的情况进行修复。有关更多信息，请参阅在AWS上实施安全控制中的响应性控制。

保留

请参阅 7 R。

停用

请参阅 7 R。

检索增强生成（RAG）

一种生成式人工智能技术，其中 LLM 在生成响应之前引用其训练数据来源之外的权威数据来源。例如，RAG 模型可以对组织的知识库或自定义数据执行语义搜索。有关更多信息，请参阅什么是 RAG。

轮换

定期更新密钥以使攻击者更难访问凭证的过程。

行列访问控制（RCAC）

使用已定义访问规则的基本、灵活的 SQL 表达式。RCAC 由行权限和列掩码组成。

RPO

请参阅恢复点目标。

RTO

请参阅恢复时间目标。

运行手册

执行特定任务所需的一套手动或自动程序。它们通常是为了简化重复性操作或高错误率的程序而设计的。

S

SAML 2.0

许多身份提供商 (IdPs) 使用的开放标准。此功能支持联合单点登录 (SSO)，因此用户无需在 IAM 中为组织中的所有人创建用户即可登录 AWS 管理控制台 或调用 AWS API 操作。有关基于 SAML 2.0 的联合身份验证的更多信息，请参阅 IAM 文档中的关于基于 SAML 2.0 的联合身份验证。

SCADA

请参阅监督控制和数据采集。

SCP

请参阅服务控制策略。

机密密钥

在中 AWS Secrets Manager，您以加密形式存储的机密或受限信息，例如密码或用户凭证。它由密钥值及其元数据组成。密钥值可以是二进制、单个字符串或多个字符串。有关更多信息，请参阅 Secrets Manager 文档中的什么是 Amazon Secrets Manager 密钥？。

安全设计

一种在整个开发过程中都考虑安全的系统工程方法。

安全控制

一种技术或管理防护机制，可防止、检测或降低威胁行为体利用安全漏洞的能力。安全控制有以下四种类型：预防性、检测性、响应性和主动性。

安全固化

缩小攻击面，使其更能抵御攻击的过程。这可能包括删除不再需要的资源、实施授予最低权限的最佳安全实践或停用配置文件中不必要的功能等操作。

安全信息和事件管理（SIEM）系统

结合了安全信息管理（SIM）和安全事件管理（SEM）系统的工具和服务。SIEM 系统会收集、监控和分析来自服务器、网络、设备和其他来源的数据，以检测威胁和安全漏洞，并生成警报。

安全响应自动化

一种预定义的程序化操作，旨在自动响应或修复安全事件。这些自动化可作为侦探或响应式安全控制措施，帮助您实施 AWS 安全最佳实践。自动响应操作的示例包括修改 VPC 安全组、修补 Amazon EC2 实例或轮换凭证。

服务器端加密

由接收数据的人在目的地对数据 AWS 服务 进行加密。

服务控制策略（SCP）

一种策略，用于集中控制 AWS Organizations的组织中所有账户的权限。SCP 为管理员可以委托给用户或角色的操作定义了防护机制或设定了限制。您可以将 SCP 用作允许列表或拒绝列表，指定允许或禁止哪些服务或操作。有关更多信息，请参阅 AWS Organizations 文档中的服务控制策略。

服务端点

的入口点的 URL AWS 服务。您可以使用端点，通过编程方式连接到目标服务。有关更多信息，请参阅 AWS 一般参考 中的 AWS 服务 端点。

服务水平协议（SLA）

一份协议，阐明了 IT 团队承诺向客户交付的内容，比如服务正常运行时间和性能。

服务水平指示器（SLI）

对服务性能方面的衡量，例如错误率、可用性或吞吐量。

服务水平目标（SLO）

代表服务运行状况的目标指标，由服务水平指示器衡量。

责任共担模式

描述您在云安全与合规方面共同承担 AWS 的责任的模型。 AWS 负责云的安全，而您则负责云中的安全。有关更多信息，请参阅责任共担模式。

暗影人工智能

在组织内受管控渠道之外构建或使用的未经授权的 AI 应用程序。

SIEM

请参阅安全信息和事件管理系统。

单点故障（SPOF）

应用程序的单个关键组件出现故障，可能会中断系统。

SLA

请参阅服务水平协议。

SLI

请参阅服务水平指示器。

SLO

请参阅服务水平目标。

split-and-seed 模式

一种扩展和加速现代化项目的模式。随着新功能和产品发布的定义，核心团队会拆分以创建新的产品团队。这有助于扩展组织的能力和服务，提高开发人员的工作效率，支持快速创新。有关更多信息，请参阅在 AWS Cloud中实现应用程序现代化的分阶段方法。

SPOF

请参阅单点故障。

星型架构

一种数据库组织结构，它使用一个大型事实表来存储事务数据或测量数据，并使用一个或多个较小的维度表来存储数据属性。此结构专为在数据仓库中使用或用于商业智能目的而设计。

strangler fig 模式

一种通过逐步重写和替换系统功能直至可以停用原有的系统来实现单体系统现代化的方法。这种模式用无花果藤作为类比，这种藤蔓成长为一棵树，最终战胜并取代了宿主。该模式是由 Martin

Fowler 提出的，作为重写单体系统时管理风险的一种方法。有关如何应用此模式的示例，请参阅使用容器和 Amazon API Gateway 逐步实现传统微软 ASP.NET (ASMX) 网络服务的现代化。

子网

您的 VPC 内的一个 IP 地址范围。子网必须位于单个可用区中。

监督控制和数据采集（SCADA）

在制造业中，一种使用硬件和软件来监控实物资产和生产操作的系统。

对称加密

一种加密算法，它使用相同的密钥来加密和解密数据。

综合测试

以模拟用户交互的方式测试系统，以检测潜在问题或监控性能。你可以使用 Amazon S CloudWatch ynthetics 来创建这些测试。

系统提示

一种为 LLM 提供上下文、说明或准则以指导其行为的技术。系统提示有助于设置上下文并制定与用户交互的规则。

T

标签

Key-value 对充当用于组织 AWS 资源的元数据。标签有助于您管理、识别、组织、搜索和筛选 资源。有关更多信息，请参阅标记您的 AWS 资源。

目标变量

您在监督式 ML 中尝试预测的值。这也被称为结果变量。例如，在制造环境中，目标变量可能是产品缺陷。

任务列表

一种通过运行手册用于跟踪进度的工具。任务列表包含运行手册的概述和要完成的常规任务列表。对于每项常规任务，它包括预计所需时间、所有者和进度。

测试环境

请参阅环境。

训练

为您的 ML 模型提供学习数据。训练数据必须包含正确答案。学习算法在训练数据中查找将输入数据属性映射到目标（您希望预测的答案）的模式。然后输出捕获这些模式的 ML 模型。然后，您可以使用 ML 模型对不知道目标的新数据进行预测。

工具

代理可以调用以在外部系统中执行操作的函数或 API。

中转网关

中转网关是网络中转中心，您可用它来互连 VPC 和本地网络。有关更多信息，请参阅 AWS Transit Gateway 文档中的什么是公交网关。

基于中继的工作流程

一种方法，开发人员在功能分支中本地构建和测试功能，然后将这些更改合并到主分支中。然后，按顺序将主分支构建到开发、预生产和生产环境。

可信访问权限

向您指定的服务授予权限，该服务可以代表您在其账户中执行任务。 AWS Organizations 当需要服务相关的角色时，受信任的服务会在每个账户中创建一个角色，为您执行管理任务。有关更多信息，请参阅 AWS Organizations 文档中的AWS Organizations 与其他 AWS 服务一起使用。

优化

更改训练过程的各个方面，以提高 ML 模型的准确性。例如，您可以通过生成标签集、添加标签，并在不同的设置下多次重复这些步骤来优化模型，从而训练 ML 模型。

双披萨团队

一个小 DevOps 团队，你可以用两个披萨来喂食。双披萨团队的规模可确保在软件开发过程中充分协作。

U

不确定性

这一概念指的是不精确、不完整或未知的信息，这些信息可能会破坏预测式 ML 模型的可靠性。不确定性有两种类型：认知不确定性是由有限的、不完整的数据造成的，而偶然不确定性是由数据中固有的噪声和随机性导致的。

无差别任务

也称为繁重工作，即创建和运行应用程序所必需的工作，但不能为最终用户提供直接价值或竞争优势。无差别任务的示例包括采购、维护和容量规划。

上层环境

请参阅环境。

V

vacuum 操作

一种数据库维护操作，包括在增量更新后进行清理，以回收存储空间并提高性能。

版本控制

跟踪更改的过程和工具，例如存储库中源代码的更改。

VPC 对等连接

两个 VPC 之间的连接，允许您使用私有 IP 地址路由流量。有关更多信息，请参阅 Amazon VPC 文档中的什么是 VPC 对等连接。

漏洞

损害系统安全的软件缺陷或硬件缺陷。

W

热缓存

一种包含经常访问的当前相关数据的缓冲区缓存。数据库实例可以从缓冲区缓存读取，这比从主内存或磁盘读取要快。

暖数据

不常访问的数据。查询此类数据时，通常可以接受中速查询。

窗口函数

一种对与当前记录有某种关联的一组行执行计算的 SQL 函数。窗口函数对于处理任务很有用，例如计算移动平均值或根据当前行的相对位置访问行的值。

工作负载

一系列资源和代码，它们可以提供商业价值，如面向客户的应用程序或后端过程。

工作流

迁移项目中负责一组特定任务的职能小组。每个工作流都是独立的，但支持项目中的其他工作流。例如，组合工作流负责确定应用程序的优先级、波次规划和收集迁移元数据。组合工作流将这些资产交付给迁移工作流，然后迁移服务器和应用程序。

WORM

请参阅一次写入多次读取。

WQF

请参阅 AWS 工作负载资格鉴定框架。

一次写入多次读取（WORM）

一种存储模型，可一次写入数据并防止数据被删除或修改。授权用户可以根据需要多次读取数据，但无法对其进行更改。此数据存储基础设施被认为不可变。

Z

零日漏洞利用

一种利用零日漏洞的攻击，通常为恶意软件。

零日漏洞

生产系统中不可避免的缺陷或漏洞。威胁主体可能利用这种类型的漏洞攻击系统。开发人员经常因攻击而意识到该漏洞。

零样本提示

为 LLM 提供执行任务的说明，但没有可以帮助指导的示例（样本）。LLM 必须使用预先训练的知识来处理任务。零样本提示的有效性取决于任务的复杂性和提示的质量。另请参阅少样本提示。

僵尸应用程序

平均 CPU 和内存使用率低于 5% 的应用程序。在迁移项目中，通常会停用这些应用程序。

本文属于机器翻译版本。若本译文内容与英语原文存在差异，则一律以英文原文为准。