Agentic AI 模式和工作流程已开启 AWS

AWS 规范性指导

AWS 规范性指导: Agentic AI 模式和工作流程已开启 AWS

Table of Contents

Agentic AI 模式和工作流程已开启 AWS

Aaron Sempf 和 Andrew Hooker，Amazon Web Services

2025 年 7 月（文档历史记录）

Organizations正在采用大型语言模型 (LLMs) 和软件代理，使用一种名为代理模式的新架构学科来解决动态的多域问题。代理模式是基础蓝图和模块化结构，用于在许多环境中设计和编排以目标为导向的 AI 代理。

目标受众

本指南适用于想要构建超越静态逻辑、符号逻辑和确定性自动化的智能应用程序的架构师、开发人员和产品负责人。

目标

本指南提供了 AI 代理系统的设计框架和实现方法，这些系统可以自主运行，同时保持可控性并与您的目标保持一致。它将事件驱动的架构模式与各种代理替代方案联系起来，演示了如何使用云原生架构构建生产级代理系统。本指南讨论了以下主题：

- 代理模式 — 代理模式是可重复使用的设计模板，用于描述各个代理的结构和行为。这包括推理代理、检索增强代理、编码代理、语音接口、工作流程协调器和协作式多代理系统。每种模式都说明了代理如何感知、推理、行为和学习，并映射到这些 AWS 服务模式中。
- 法学硕士工作流程 — 工作流程侧重于代理如何使用推理 LLMs 。他们探讨了提示策略和计划机制，并概述了如何 LLMs 不仅用于生成文本，还用于在代理循环中推动结构化、可解释和可靠的行为。
- 代理工作流程模式 — 工作流模式描述了多个代理、工具和环境如何相互作用以形成自治系统。这包括任务编排、子代理委派、基于事件的协调、可观察性和控制的模式。这些方面促进了可扩展、可组合和可审计的 AI 架构。

关于此内容系列

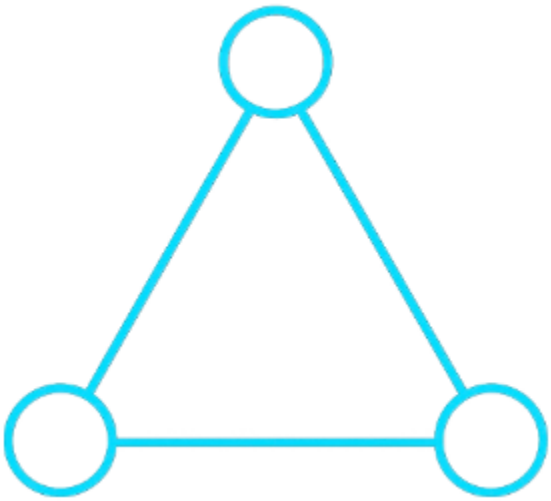
本指南是关于代理人工智能的系列文章的一部分。 AWS要了解更多信息并查看本系列中的其他指南，请参阅 AWS 规范性指导网站上的 Agentic AI。

代理模式

代理模式是可重复使用的可组合构造块，可以根据特定的领域、用例和复杂程度进行定制。但是，代理系统与传统应用不同。所有 AI 代理设计的核心是基于以下三个基本原则的概念模型：

- 异步 — 代理在松散耦合、事件丰富的环境中运行
- 自主权 — 代理独立行动，无需人为或外部控制
- 代理机构 — 代理代表用户或系统有目的地朝着特定目标行事

下图中的三角形表示软件代理的核心组成部分：感知、理性和行动。这使代理系统能够在其环境中进行观察、做出决策和采取行动。



通过设计，代理模式为构建 AI 系统提供了一种模块化设计语言，这意味着它们易于访问、可操作、可扩展并且可以投入生产。设计这些系统需要仔细注意以下三个相互关联的维度，本指南稍后将对此进行进一步讨论。

本节内容

- 工作流程编排代理
- 内存增强代理
- 仿真和测试平台代理
- 观察者和监视代理
- 多代理协作

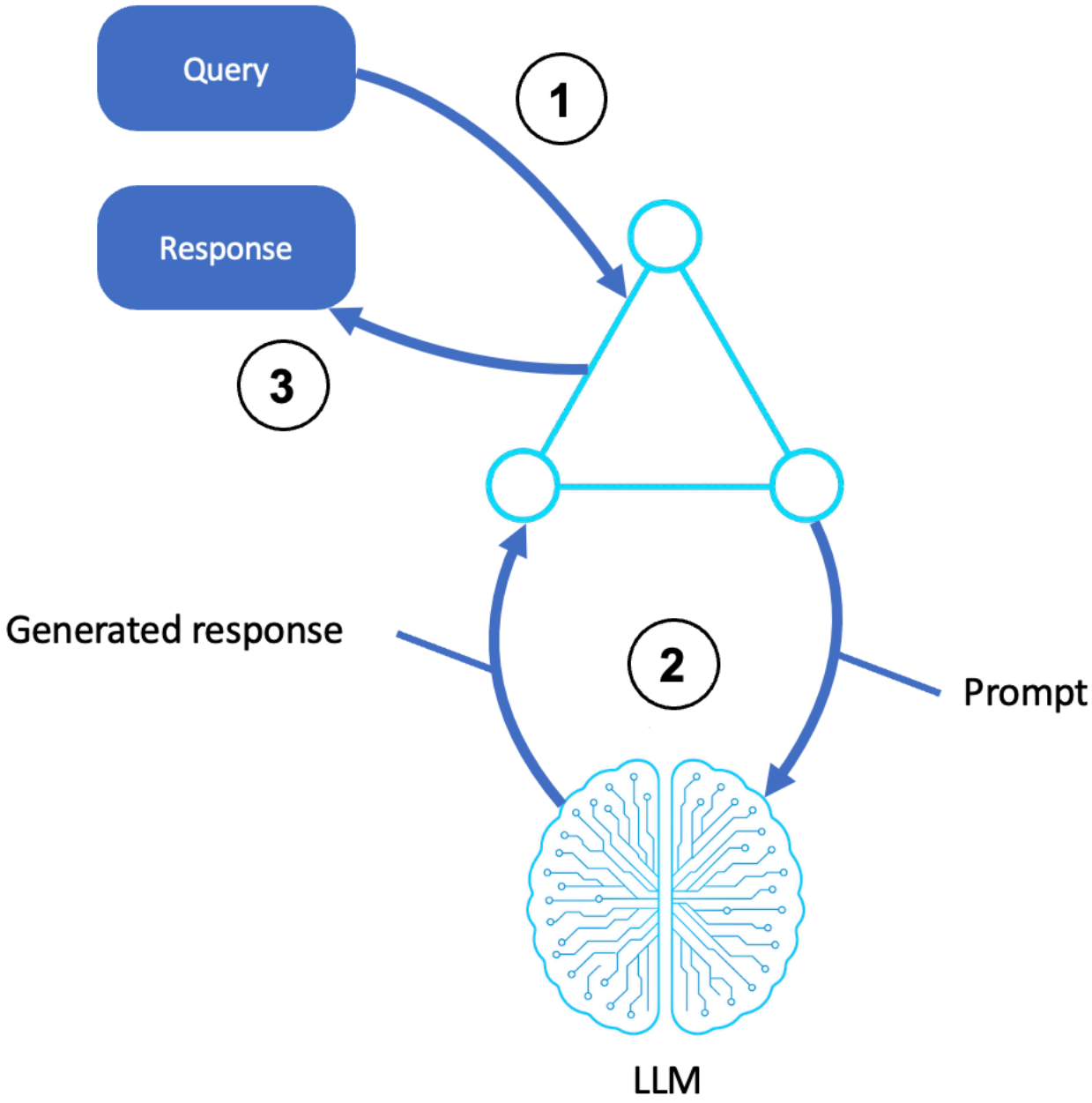
基本推理代理

基本推理代理是最简单的代理人工智能形式，它根据查询执行逻辑推理或决策。它接受用户或系统的输入，处理查询，并使用结构化提示生成响应。

这种模式对于需要根据给定上下文进行单步推理、分类或总结的任务很有用。它不使用内存、工具或状态管理，这使得它在大型工作流程中具有无状态、轻量级且高度可组合。

Architecture

基本推理代理的流程如下图所示：



描述

1. 接收输入
   - 用户、系统或上游代理提交查询或指令。
   - 输入将移交给代理 shell 或编排层。
   - 此步骤包括任何预处理、提示模板和目标识别。
2. 调用 LLM

- 代理将查询转换为结构化提示并将其发送给 LLM（例如，通过 Amazon Bedrock）。
- LLM 使用预先训练的知识和上下文根据提示生成响应。
- 生成的输出可能包括推理步骤 (chain-of-thought)、最终答案或排名选项。

3. 返回响应
   - 生成的输出将中继到代理的接口。
   - 这可能包括格式化、后处理或 API 响应。

功能

- 支持自然语言或结构化输入
- 使用提示工程来指导行为
- 无状态且可扩展
- 可以嵌入到 UI APIs、CLI 和管道中

限制

- 没有记忆力或历史意识
- 不与外部工具或数据源交互
- 仅限于法学硕士在推理时所知道的情况

常见使用案例

- 对话问题和答案
- 政策解释和摘要
- 决策指南
- 轻量级且自动化的聊天机器人流程
- 分类、标签和评分

实施指导

您可以使用以下工具和服务来创建基本的推理代理：

- 用于调用 LLM 的 Amazon Bedrock（Anthropic、、Meta） AI21

- Amazon API Gateway 或者 AWS Lambda 将其作为无状态微服务公开
- 提示模板存储在参数存储库中 AWS Secrets Manager，或存储为代码

Summary

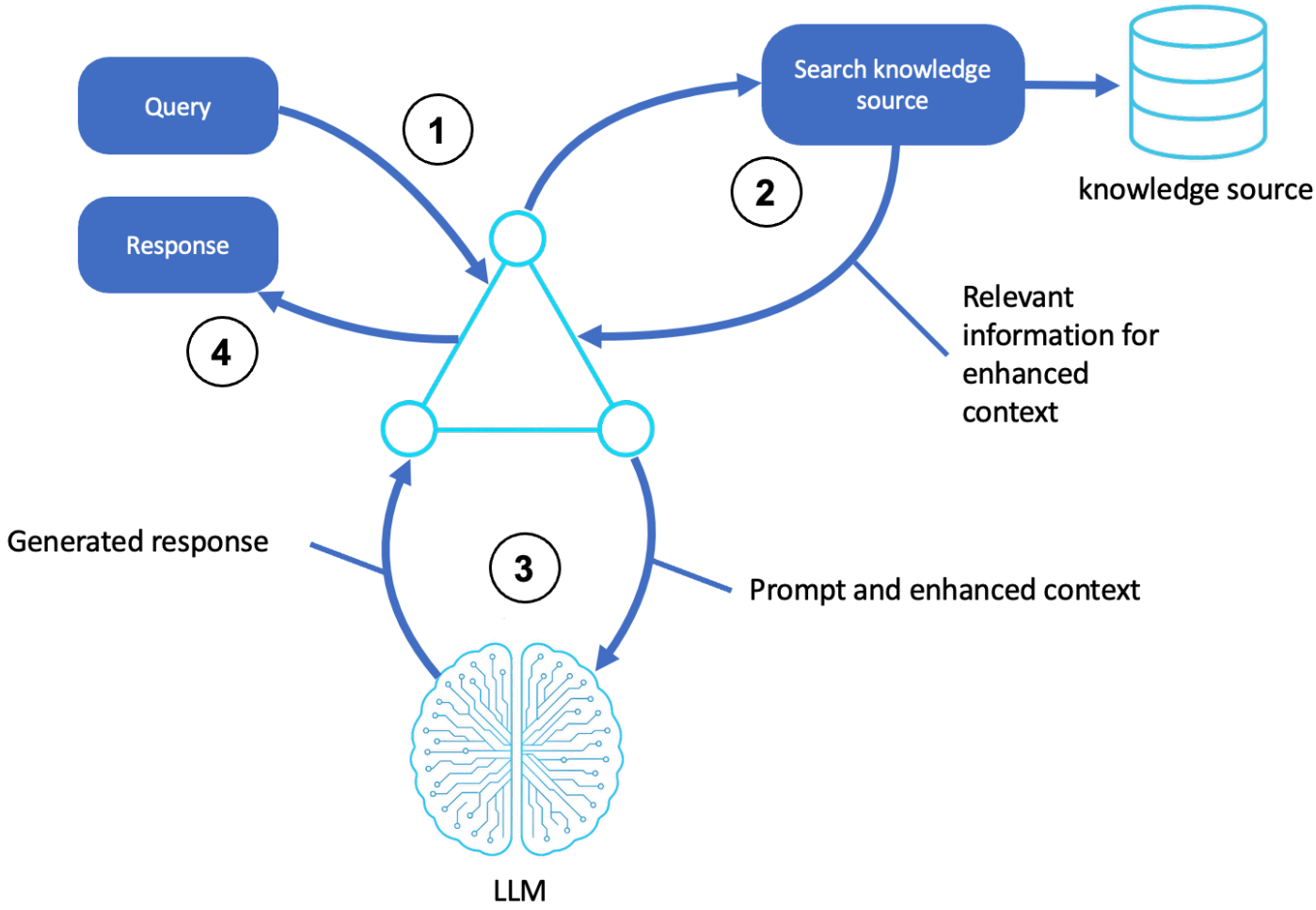
由于其结构简单，基本的推理代理是基础性的。它具有核心功能，可以将目标转化为推理路径，从而实现智能输出。这种模式通常是高级模式的起点，例如基于工具的代理和使用检索增强生成 (RAG) 的代理。它也是大型工作流程的可靠模块化组件。

RAG 探员

检索增强生成 (RAG) 是一种将信息检索与文本生成相结合的技术，可创建准确的上下文相关响应。RAG使代理能够在聘请法学硕士之前检索相关的外部信息。它通过将决策建立在事实或特定领域的信息基础上 up-to-date，从而扩展代理的有效记忆力和推理准确性。与仅依赖预训练权重 LLMs 的无状态相比，RAG 有一个外部知识搜索层，可以根据上下文动态增强提示。

Architecture

RAG 模式的逻辑如下图所示：



描述

1. 收到查询
   - 用户或上游系统向代理提交查询或目标。
   - 代理 shell 接受请求并将其格式化为推理提示。
2. 搜索外部来源
   - 代理从查询中识别概念和意图。
   - 它使用语义搜索或关键字匹配来查询知识源，例如矢量存储、数据库或文档索引。
   - 检索最相关的段落、文档或实体，以便在下一步中使用。
3. 生成上下文响应
   - 代理使用检索到的信息对提示进行扩充，从而为 LLM 形成上下文增强输入。
   - 法学硕士使用生成推理（例如 chain-of-thought或反射）处理任何输入，以产生准确的响应。
4. 返回最终输出

- 代理通过将输出封装在任何通信标头或所需格式中来准备输出，然后将其返回给用户或呼叫系统。
- （可选）检索到的文档和 LLM 输出可能会被记录、评分并存储在内存中以备将来查询。

功能

- 即使在长尾或企业特定领域也能以事实为依据的输出
- 无需微调模型即可扩展内存
- 基于每个查询和用户状态的动态上下文
- 与矢量数据库、语义索引和元数据筛选完全兼容

常见使用案例

- 企业知识助手
- 监管合规机器人
- 客户支持副驾驶
- 搜索增强型聊天机器人
- 开发者文档代理

实施指导

使用以下工具和服务创建使用 RAG 的代理：

- 用于调用 LLM 的 Amazon Bedrock
- 用于文档或结构化数据搜索的 Amazon Kendra 或 Amazon Aurora OpenSearch
- 用于存储文档的亚马逊简单存储服务 (Amazon S3) Service
- AWS Lambda 编排搜索、提示和 LLM 推理
- 与代理进行基于知识的集成（通过使用内存插件、语义检索器或 Amazon Bedrock）

Summary

Agent RAG 将静态模型推理与动态的现实世界智能联系起来。它使代理能够查找他们不知道的内容，从检索到的知识中综合答案，并生成高度可信、可审计的响应。

RAG 模式是构建无需再培训即可扩展知识访问权限的智能代理的基础。它通常是更复杂的编排模式的前身，涉及工具使用、计划和长期记忆。

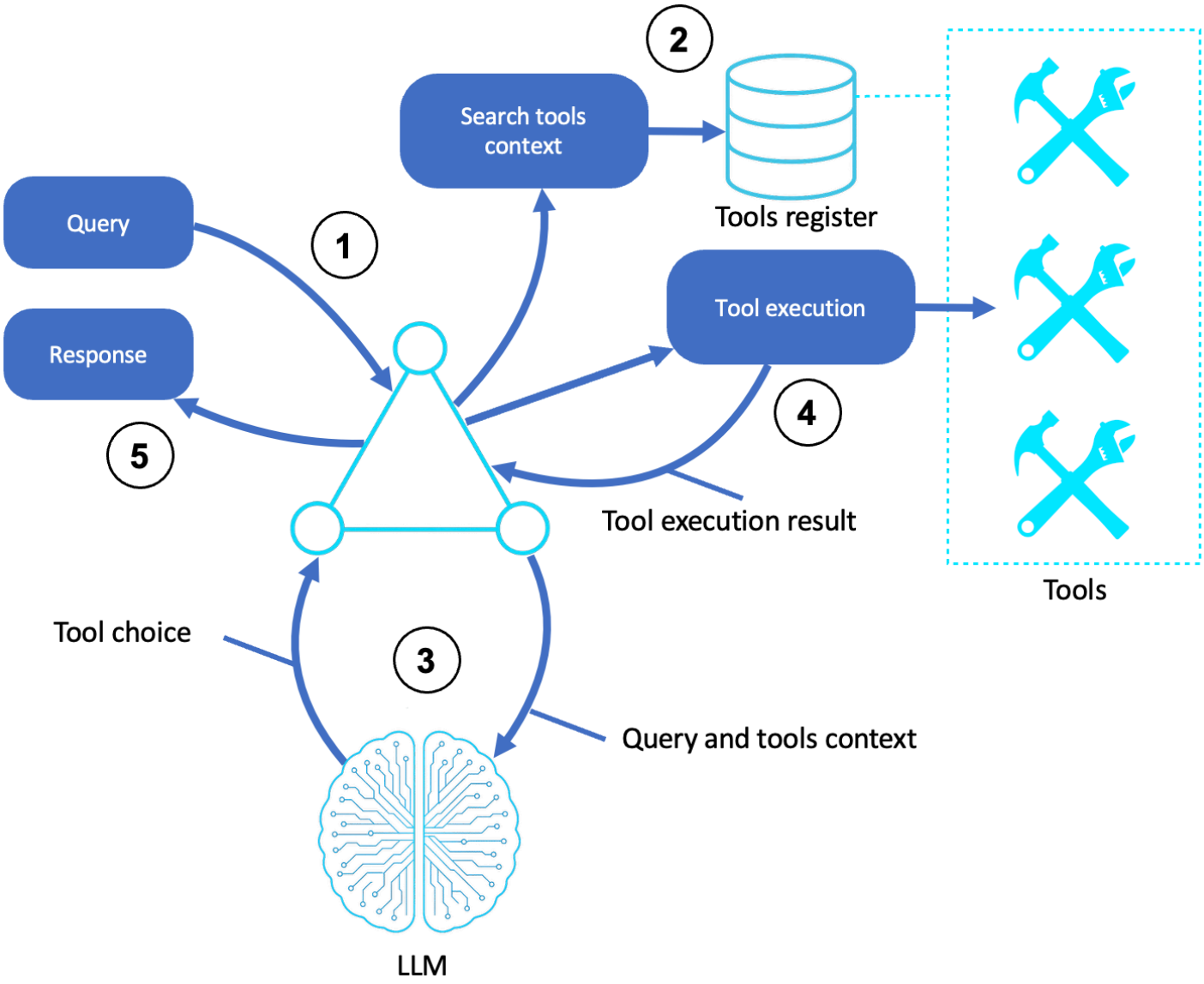
用于调用函数的基于工具的代理

基于工具的代理通过调用外部函数或完成超出纯语言推理范围的任务 APIs 来扩展推理代理的功能。此模式使用 LLM 来决定使用哪个工具，然后生成调用参数并将工具的输出合并到其推理循环中。

这种模式使代理能够采取行动，而不仅仅是提供响应。工具接口代表任何可调用的功能，从算术计算和数据库查找到外部 APIs 和云服务。

Architecture

下图显示了用于调用函数的基于工具的代理：



描述

1. 接收查询
   - 代理接收来自用户或呼叫系统的自然语言查询或任务。
2. 搜索工具
   - 代理使用内部元数据或工具注册表来搜索可用的工具、架构和相关功能。
3. 选择和调用工具
   - LLM 在其提示中接收查询和工具元数据（例如，函数名称、输入类型和描述）。
   - 它选择最相关的工具，构造输入参数，然后返回结构化函数调用。
4. 运行所选工具

- 代理 shell 或工具运行器执行选定的函数并返回结果（例如，API 输出、数据库值或计算）。

5. 返回响应
   - LLM 直接或作为更新提示的一部分将结果传递给代理。然后，它会返回自然语言结果。

功能

- 基于任务上下文的动态工具选择
- 基于架构的提示（OpenAPI、JSON 架构、函数接口） AWS
- 结果解释并将输出链接到推理中
- 无状态或会话感知操作

常见使用案例

- 具有外部数据访问权限的虚拟助手
- 财务计算器和估算器
- 基于 API 的知识工作者
- LLMs 调用 AWS Lambda Amazon SageMaker 终端节点和 SaaS 服务的方法

实施指导

使用以下命令创建用于调用函数的基于工具的代理：

- 支持函数调用的 Amazon Bedrock（Anthropic Claude）
- AWS Lambda 作为工具执行后端
- Amazon API Gateway 或 AWS Step Functions 用于工具编排
- 用于情境感知工具元数据的亚马逊 DynamoDB 或亚马逊关系数据库服务 (Amazon RDS)
- Amazon EventBridge 管道或 AWS Step Functions 将状态映射到路由输出的管道

Summary

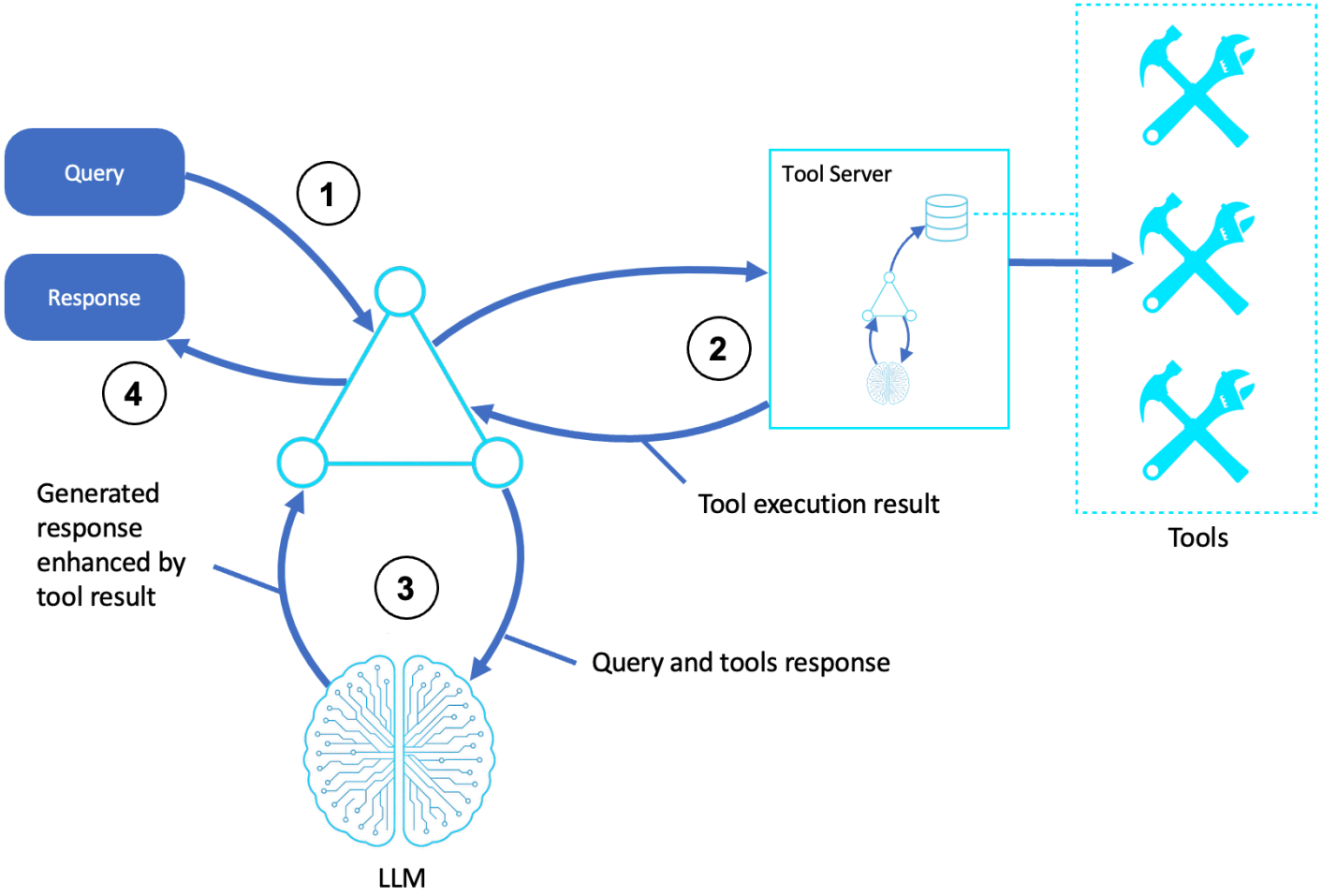
基于工具的函数调用代理代表了从理解语言到执行操作的转变。这些代理调用动态的情境感知工具，同时保持法学硕士推理，将被动助手转变为完成任务、访问服务和整合业务运营的系统。这种模式是企业环境中代理人工智能的重要组成部分，尤其是在与声明式架构、授权框架和多代理系统结合使用时。

基于工具的服务器代理

基于工具的服务器代理通过将工具执行委托给具有专门用于工具、脚本和复合代理的运行环境的外部服务器来增强函数调用代理。与代理循环选择和调用的内联函数调用不同，基于服务器的代理将逻辑和执行管道外包给其他代理或系统。这提供了高级功能，例如多工具链接、隔离执行和专业推理。工具服务器非常适合复杂、有状态或资源密集型操作，在这些操作中，工具本身可能涉及单独的 AI 模型、业务规则或环境。

Architecture

以下是基于工具的服务器代理的模式：



描述

1. 接收查询
   - 用户或系统向代理 shell 提交请求。

- 代理解释查询并准备将其发送到工具服务器。

2. 运行工具服务器进程
   - 代理将任务以及结构化参数发送到工具服务器。
   - 然后，工具服务器可以：
     - 在专用计算系统（例如，容器或 Amazon SageMaker）中运行脚本或逻辑 AWS Lambda
     - 使用自己的子代理和 LLM 推理来选择和运行工具
     - 管理依赖关系、重试或多步骤执行流程
     - 任务完成后将结果输出到主代理
3. 在工具输出中使用 LLM 推理
   - 代理调用 LLM，将原始查询和工具服务器结果作为提示的一部分传递。
   - 法学硕士综合了包含新获得信息的响应。
4. 返回响应
   - 代理向用户或呼叫系统返回自然语言或结构化响应。
   - （可选）结果可以存储在内存或审计日志中。

功能

- 工具是在主代理执行循环之外调用的
- 工具执行可能涉及 LLM 调用、逻辑链或子代理
- 代理充当控制器或调度员，而不仅仅是工具包装器
- 实现可组合性、可扩展性和逻辑隔离

常见使用案例

- 编排模型链（例如，通过组合 LLM、vision 和代码）
- AI 驱动的自动化管道
- DevOps 带有脚本运行器的助理代理
- 复杂的财务计算、模拟或优化代理
- 多模态工具（例如，通过组合音频、文档和操作）

实施指导

您可以使用以下方法构建此模式 AWS 服务：

- Amazon Bedrock（代理主机和 LLM 推理）
- AWS Lambda、Amazon ECS 或亚马逊 SageMaker 终端节点作为工具服务器运行时 AWS Fargate
- Amazon API Gateway 或者 AWS App Runner 用于公开工具服务器 APIs
- EventBridge 用于解耦 agent-to-tool消息的 Amazon
- AWS Step Functions 或者 AWS AppFabric 用于在工具服务器上编写多代理逻辑

Summary

使用服务器的基于工具的代理具有高度模块化和可扩展性。它们将决策逻辑与执行分离，从而允许主代理保持轻量级，同时将复杂或敏感的操作转移到其他系统。这对于企业级代理人工智能非常重要，尤其是在需要治理、可观察性、隔离、动态组合或其任意组合的环境中。

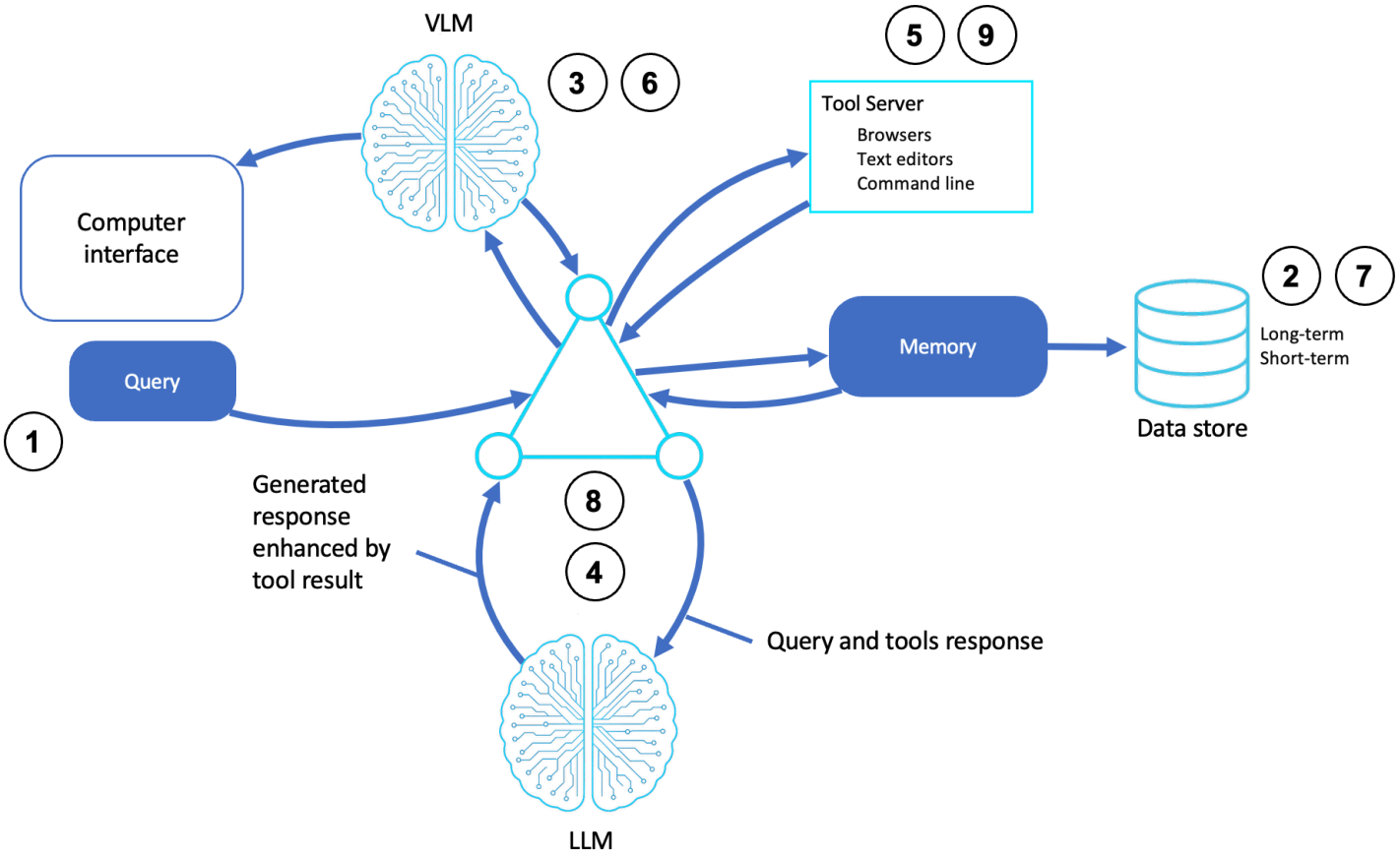
计算机用代理

计算机用代理可以模拟或控制数字环境，例如浏览器、终端、文件系统和应用程序。这些代理通过组合 LLM 推理、视觉语言模型 (VLMs) 和执行命令或模拟输入事件的工具服务器来解释用户意图，与视觉和文本界面交互，并执行以目标为导向的操作。

这种模式对于实际的人工智能自动化非常重要，在这种自动化中，代理不仅可以充当助手，还可以充当代理，像人类一样执行操作，通常使用相同的工具和环境。

Architecture

计算机使用的代理模式如下图所示：



描述

1. 收到查询
   - 任务或请求通过 UI、API 或自然语言界面提供。
2. 访问内存
   - 代理会检索短期和长期记忆，以回忆过去的命令、目标和系统状态。
3. 分析视觉背景
   - VLM 观察计算机屏幕、系统状态或用户界面元素，以了解给定的上下文并识别可操作的项目。
4. 法学硕士学位的理由
   - LLM 将查询、内存状态、工具和服务器响应结合起来，以确定下一个操作。
5. 与工具服务器交互
   - 代理调用服务器上托管的工具，其中可能包括以下内容：
     - 浏览器（例如，无头 Chrome）和外壳环境
     - 文本和代码编辑器
     - 自定义脚本接口

6. 更新视觉输入
   - 如果系统用户界面发生变化或需要进一步观察，VLM 可能会重新分析屏幕状态或文本缓冲区。
7. 更新内存
   - 新的见解、系统状态或用户反馈会写入短期和长期记忆。
8. 制定最终决定和解释
   - LLM 根据查询和工具输出综合结果或建议操作。
9. 返回响应
   - 代理将结果返回到界面（例如，已完成的任务、确认或生成的内容）。

功能

- 带有视觉和文本输入的多模态推理
- 通过模拟或 API 驱动的输入控制应用程序
- 永久状态的内存管理
- 序列执行中的自主权（多步流程）

常见使用案例

- 在其中编写和运行代码的 AI 开发人员 IDEs
- 用于重复数字化工作流程的计算机使用代理
- 用于软件测试和质量保证的模拟用户
- 辅助功能代理，用于 UIs 浏览语音或高级指令
- 通过推理增强的智能机器人流程自动化 (RPA)

实施指导

- 您可以使用以下方法构建此模式 AWS 服务：
- Amazon Bedrock 用于基于 LLM 的计划和推理
- Amazon Elastic Compute Cloud (Amazon EC2) AWS Lambda，或亚马逊 SageMaker 笔记本电脑，用于在模拟 UI 环境中运行工具服务器
- 用于内存持久性的亚马逊简单存储服务 (Amazon S3) Service 或 Amazon DynamoDB
- Amazon Rekognition（或自定义模型）用于混合场景中的用户界面图像分析

- Amaz CloudWatch on Logs 或者 AWS X-Ray 用于可观察性和审计跟踪

Summary

计算机使用代理充当自主的数字运营商，弥合了人机交互和人工智能驱动的操作之间的差距。通过整合内存、工具编排和 VLMs，这些代理可以与专为人类设计的系统进行自适应交互、执行操作、更新文件、浏览菜单和生成响应。

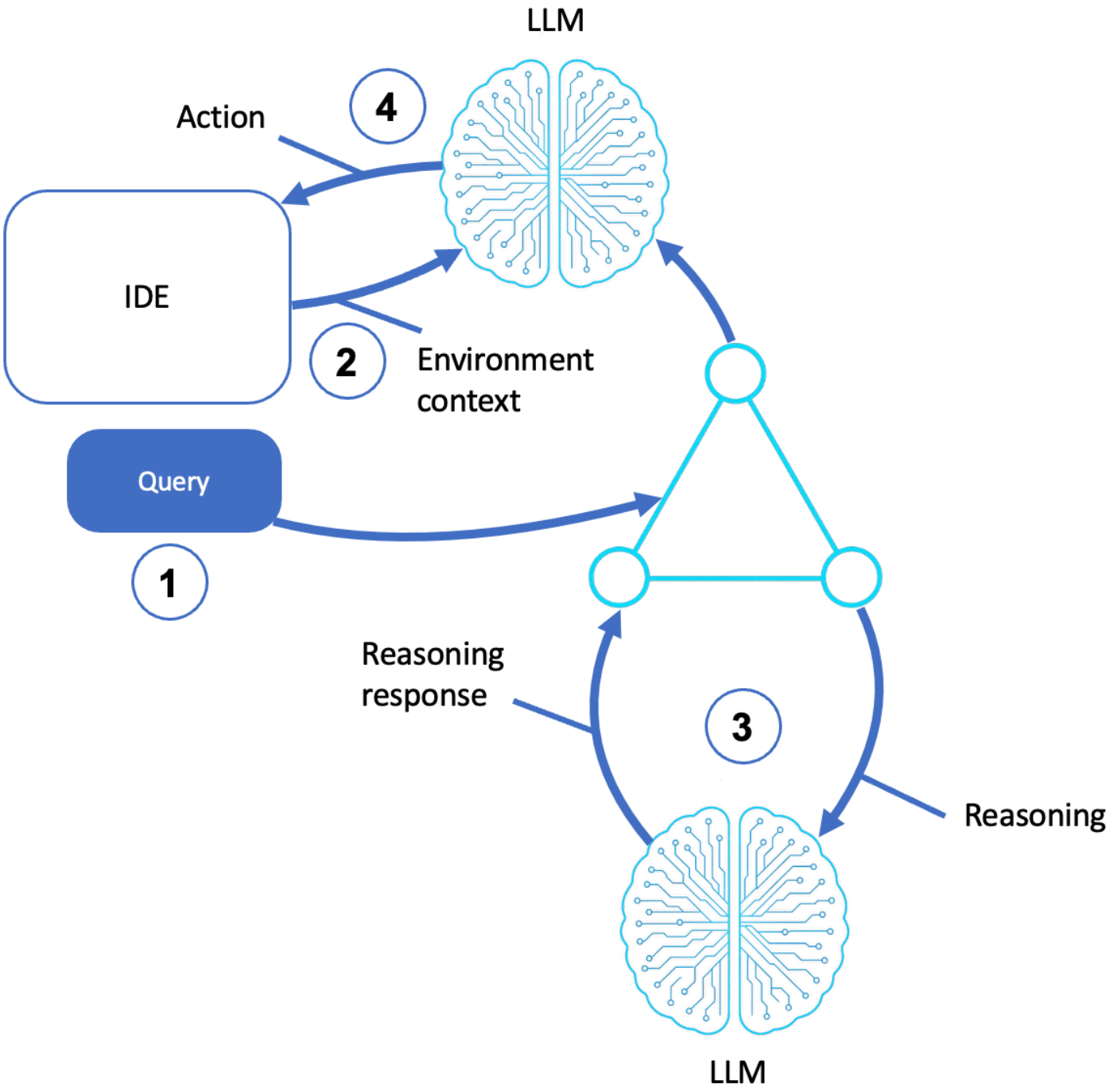
编码代理

编码代理可以推理编程任务、生成或修改代码，以及与开发者环境（例如 IDEs 和）进行交互 CLIs。这些代理将自然语言理解与结构化推理相结合，以协助、增强和自动化软件开发，从函数生成到错误修复和测试创作。

与自动完成工具不同，编码代理会主动解释用户目标，查询开发环境以获取上下文（例如，它打开文件并跟踪错误），识别需求，然后提出和执行操作。

Architecture

编码代理模式如下图所示：



描述

1. 接收查询
   - 用户通过命令面板、聊天窗口或 CLI（例如，“向此功能添加日志记录”或“重构以提高可读性”）提供自然语言指令。
2. 提取环境上下文

- 代理从 IDE 收集上下文，包括活动文件、光标位置、代码片段和符号表。
- 它输出错误消息、测试结果和其他代理的输出。

3. 法学硕士推理
   - 代理向 LLM 发送提示，包括查询和环境上下文。
     - 法学硕士通过推理来确定以下内容：
     - 需要改变什么
     - 如何生成解决方案
     - 任何重构、重写或编码步骤
4. 执行动作
   - LLM 将输出返回到代理并将其导入到 IDE 或运行时环境中。
   - 这可能包括插入或修改代码、生成注释或文档，以及触发下游构建、测试和 linting 任务。

功能

- 高度上下文感知能力（例如，IDE 状态、光标和语法树）
- 目标和反馈的迭代推理
- 可选的代码规划和操作分离（例如，先理由，然后再行动）
- 适用于同步或异步开发者工作流程

常见使用案例

- 根据任务描述生成代码
- 代码重构和优化
- 测试用例生成和验证
- 错误解释和调试
- 文档助手
- 配对编程副驾驶

实施指导

- 您可以使用以下工具和以下工具来构建此模式 AWS 服务：

- Amazon Bedrock 用于 LLM 驱动的生成和推理
- Amazon Q 开发者提供编码建议和完成信息
- AWS Lambda 或者用于运行和测试沙盒环境的亚马逊弹性容器服务 (Amazon ECS) Container Service
- AWS Cloud9、VS Code 扩展或用于托管和评估上下文的自定义 IDE 集成
- Amazon Simple Storage Service (Amazon S3) 用于存储中间提示、响应和修订历史记录

Summary

编码代理是基于人工智能的新开发工具，能够解释自然语言、分析上下文、生成多步骤代码更改以及与软件开发生命周期集成。

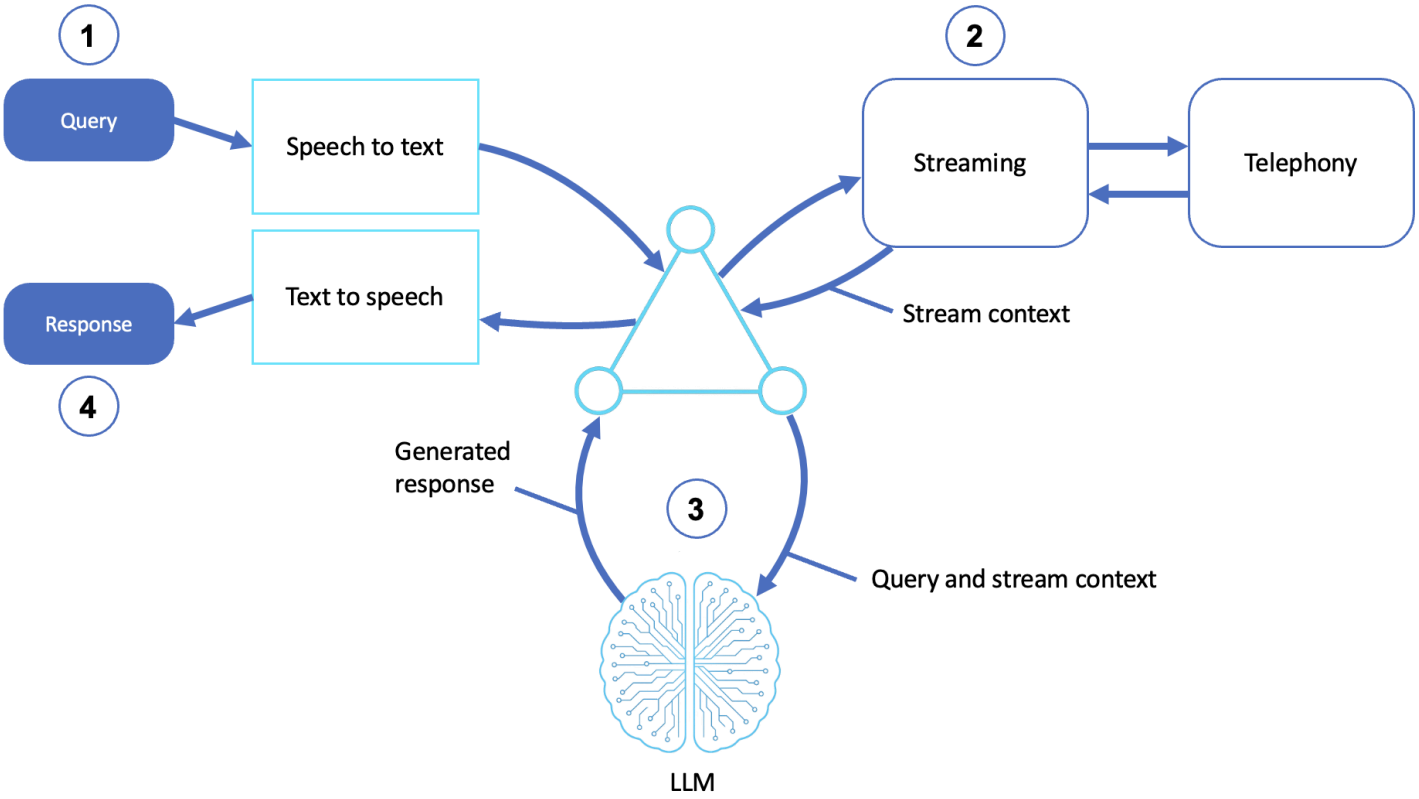
语音和语音代理

语音和语音代理通过语音对话与用户互动。这些代理集成了语音识别、自然语言理解和语音合成，可在电话、移动、网络和嵌入式平台上实现对话式 AI。

语音代理在免提、实时或可访问性驱动的环境中特别有效。通过将流媒体接口与LLM支持的推理相结合，它们促进了用户感觉自然的丰富、动态的互动。

Architecture

语音和语音代理如下图所示：



描述

1. 收到语音查询
   - 用户向手机、麦克风或嵌入式系统发出请求。
   - speech-to-text(STT) 模块将音频转换为文本。
2. 集成了流媒体和电话环境
   - 代理使用流媒体接口实时管理音频 I/O 。
   - 如果部署在联络中心或电信环境中，则电话集成会处理会话路由、双音多频 (DTMF) 输入和媒体传输。

注意：DTMF 是指按下电话键盘上的按钮时产生的音调。在语音代理内部的流媒体和电话环境集成中，DTMF 被用作电话呼叫期间的信号输入机制，尤其是在交互式语音应答 (IVR) 系统中。DTMF 输入使代理能够：

- 识别菜单选项（例如，“按 1 进行计费。 按 2 获得支持。”）
- 收集数字输入（例如，账号和确认号码） PINs
- 在呼叫流中触发工作流程或状态转换

- 必要时从语音恢复为按键音

1. 通过 LLM 直播上下文了解原因
   - 查询被发送到代理，代理会将其与任何会话元数据（例如，呼叫者 ID、之前的上下文）一起传递给 LLM。
   - LLM 生成响应，如果交互正在进行中，则可能使用 chain-of-thought策略或多圈记忆。
2. 返回语音响应
   - 代理使用 text-to-speech (TTS) 将其响应转换为语音。
   - 它通过语音通道将音频返回给用户。

功能

- 实时语音理解和生成
- 支持 ST I/O T 和 TTS 的多语言版本
- 与电话或流媒体集成 APIs
- 会话感知和回合之间的记忆切换

常见使用案例

- 对话式 IVR 系统
- 虚拟接待员和预约安排员
- 语音驱动的帮助台代理
- 可穿戴语音助手
- 智能家居的语音接口和无障碍工具

实施指导

您可以使用以下工具和以下工具来构建此模式 AWS 服务：

- 适用于 STT 的 Amazon Lex V2 或 Amazon Transcribe
- 适用于 TTS 的 Amazon Polly
- 用于直播和电话的 Amazon Chime SDK、Amazon Connect 或亚马逊互动视频服务 (亚马逊 IVS) Interactive Service

- Amazon Bedrock 用于使用 Anthropic 或其他基础 AI21模型进行推理
- AWS Lambda 连接 STT、LLM、TTS 和会话上下文

（可选）其他增强功能可能包括以下内容：

- 亚马逊 Kendra 或者 OpenSearch 用于情境感知型 RAG
- 用于会话内存的 Amazon DynamoDB
- Amazon CloudWatch 日志和可 AWS X-Ray 追溯性

Summary

语音和语音代理是通过自然对话进行交互的智能系统。通过将语音接口与 LLM 推理和实时流媒体基础设施集成，语音代理可实现无缝、可访问和可扩展的交互。

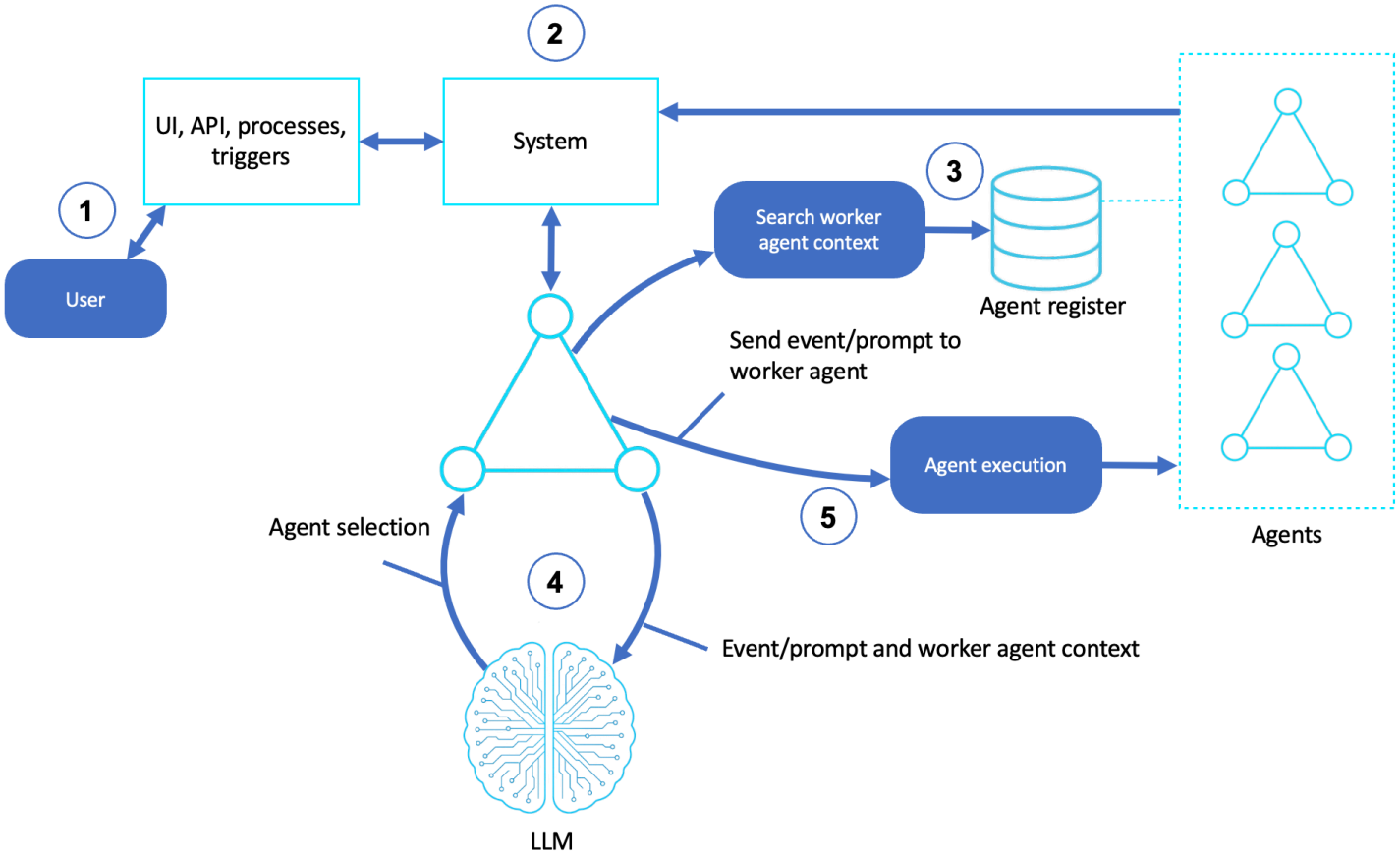
工作流程编排代理

工作流协调代理管理和协调分布式系统中的多步骤任务、流程和服务。这些代理不是孤立地推理和行动，而是将工作委托给子代理或其他系统，维护执行环境，并根据中间结果进行调整。

这些代理是自动化流程的基本组成部分。它们在处理长时间运行的任务、多代理组合和跨域集成时特别有用，在这种情况下，必须按顺序或有条件地调用各种代理和工具。

Architecture

下图显示了工作流程编排代理：



描述

1. 接收用户输入
   - 用户（或外部触发器）通过 UI、API 或系统事件启动任务。
2. 处理系统事件
   - 系统组件接收请求并发出需要编排的事件或命令。
3. 检索上下文
   - 工作流代理会查询知识库和代理注册表，根据元数据、域名和先前的成功率为任务找到合适的工作代理。
4. 选择法学硕士代理
   - 法学硕士通过分析任务描述和可用选项来帮助选择最佳代理或工作流程计划。
   - 它还可以制定特定于任务的提示以发送给选定的代理。
5. 委托和执行人
   - 选定的工作器代理收到事件或提示并开始运行命令。
   - 它可以跟踪执行状态，在失败时重试，并将中间结果传递给序列中的下一个代理。

功能

- 代理构成（例如，主管、协作者代理和工具）
- 事件驱动或计划执行
- 随着时间的推移进行内存和状态跟踪
- 分层或并行任务编排（同步与异步工作流程相比）
- 动态代理选择和链接

常见使用案例

- 多步骤自动化（例如，数据摄取和报告）
- 客户服务路线和上报（例如， agent-as-coordinator）
- AI 代理在同一个循环内与人类和机器人进行协调
- 使用 LLM 支持的逻辑实现企业流程自动化
- 混合系统结合了 AI 代理和传统编排工具

实施指导

您可以使用以下工具和以下工具来构建此模式 AWS 服务：

- Amazon Bedrock 用于推理和代理选择
- AWS Step Functions 或 Amazon EventBridge 用于工作流程构成
- AWS Lambda 作为执行单元或任务运行器
- 亚马逊 DynamoDB、亚马逊简单存储服务 (Amazon S3) Simple Storage S3 或 Amazon RDS 用于跟踪状态和结果
- AWS AppFabric 或者 AppFlow 用于跨系统协调的 Amazon
- （可选）使用 Amazon SageMaker 运行代理托管特定域的工作代理

Summary

工作流代理在多代理环境中协调、调整和调整目标。这意味着 AI 代理可以通过模块化、可解释的工作流程进行协作、适应运行条件并交付复杂的结果。

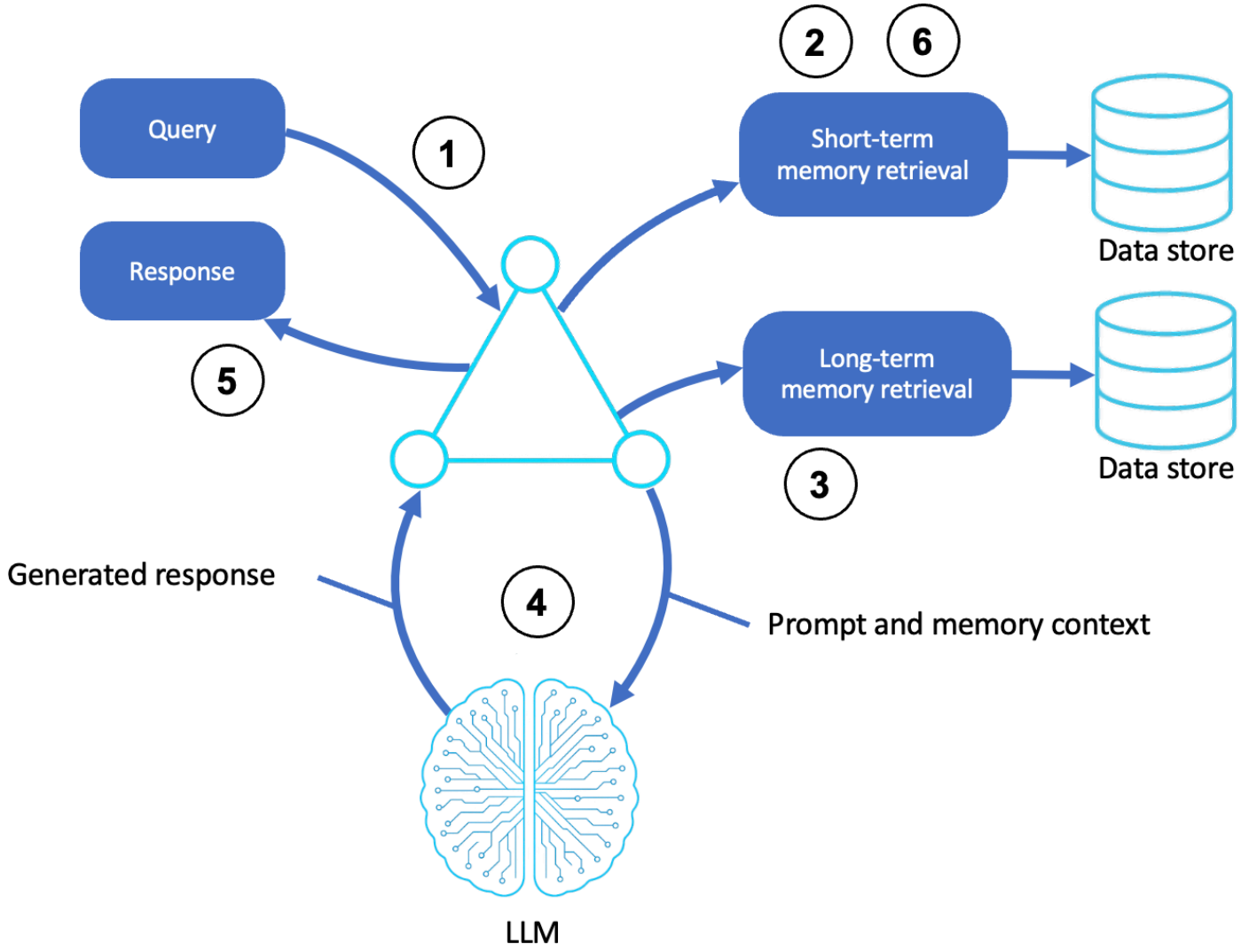
内存增强代理

通过使用短期和长期记忆进行存储、检索和推理的能力，增强了内存增强型代理。这使他们能够在多个任务、会话和互动中保持背景信息，从而产生更连贯、更个性化和更具战略性的响应。

与无状态代理不同，内存增强型代理通过引用历史数据进行调整，从先前的结果中学习，并做出符合用户目标、偏好和环境的决策。

Architecture

内存增强代理如下图所示：



描述

1. 接收输入或事件

- 代理接收用户查询或系统事件。这可能是文本、API 触发器或环境变化。

2. 检索短期记忆
   - 代理检索与会话或工作流程相关的近期对话历史记录、任务上下文或系统状态。
3. 检索长期记忆
   - 代理查询长期内存（例如，矢量数据库和键值存储）以获取历史见解，例如：
     - 用户偏好
     - 过去的决定和结果
     - 学到的概念、总结或经验
4. 通过法学硕士学位考试的原因
   - 记忆上下文嵌入到 LLM 提示符中，允许代理根据当前输入和先验知识进行推理。
5. 生成输出
   - 代理会根据任务历史记录和用户输入生成情境感知的响应、计划或操作，并对其进行个性化设置。
6. 更新内存
   - 存储了新的信息，例如更新的目标、成功和失败的信号以及结构化的响应，以备将来的任务之用。

功能

- 跨对话或事件的会话连续性
- 目标随着时间的推移而保持不变
- 基于不断变化的状态的情境感知
- 以往的成功和失败为依据的适应性
- 根据用户偏好和历史记录进行个性化设置

常见使用案例

- 记住用户偏好的对话副驾驶
- 跟踪代码库变化的编码代理
- 可根据任务历史进行调整的工作流代理
- 从系统知识演变而来的数字双胞胎

- 避免重复检索的研究代理

实现内存增强代理

AWS 服务 对于内存增强型代理，请使用以下工具：

内存层	AWS 服务	目的
短期	亚马逊 DynamoDB、Redis、亚马逊 Bedrock 上下文	快速检索最近的交互状态
长期（结构化）	亚马逊 Aurora、亚马逊 DynamoDB、亚马逊 Neptune	事实、关系和日志
长期（语义）	OpenSearch、Postgre SQL、Pinecone	基于嵌入的检索（即 RAG）
存储	Amazon S3	存储笔录、结构化记忆和文件
编排	AWS Lambda 或者 AWS Step Functions	管理内存注入和更新生命周期
Reasoning	Amazon Bedrock	带有记忆提示的 Anthropic Claude 或 Mi

实现注入内存的提示

要将记忆整合到代理推理中，请结合使用结构化状态和检索增强上下文注入：

- 在为语言模型构建提示时，将最新的代理状态和最近的对话历史记录作为结构化输入，这样它就可以在完整的上下文中进行推理。
- 使用检索增强生成 (RAG) 从长期记忆中提取相关文档或事实。
- 总结之前的计划、背景和互动，以实现压缩和相关性。
- 在推理过程中注入外部存储器模块，例如向量存储或结构化日志，以指导决策。

Summary

增强内存的代理通过从经验中学习和记住用户情境来保持思维的连续性。这些代理通过使用长期协作、个性化和战略推理，超越了被动情报。就代理人工智能而言，内存使代理的行为更像自适应数字对应物，而不像无状态工具。

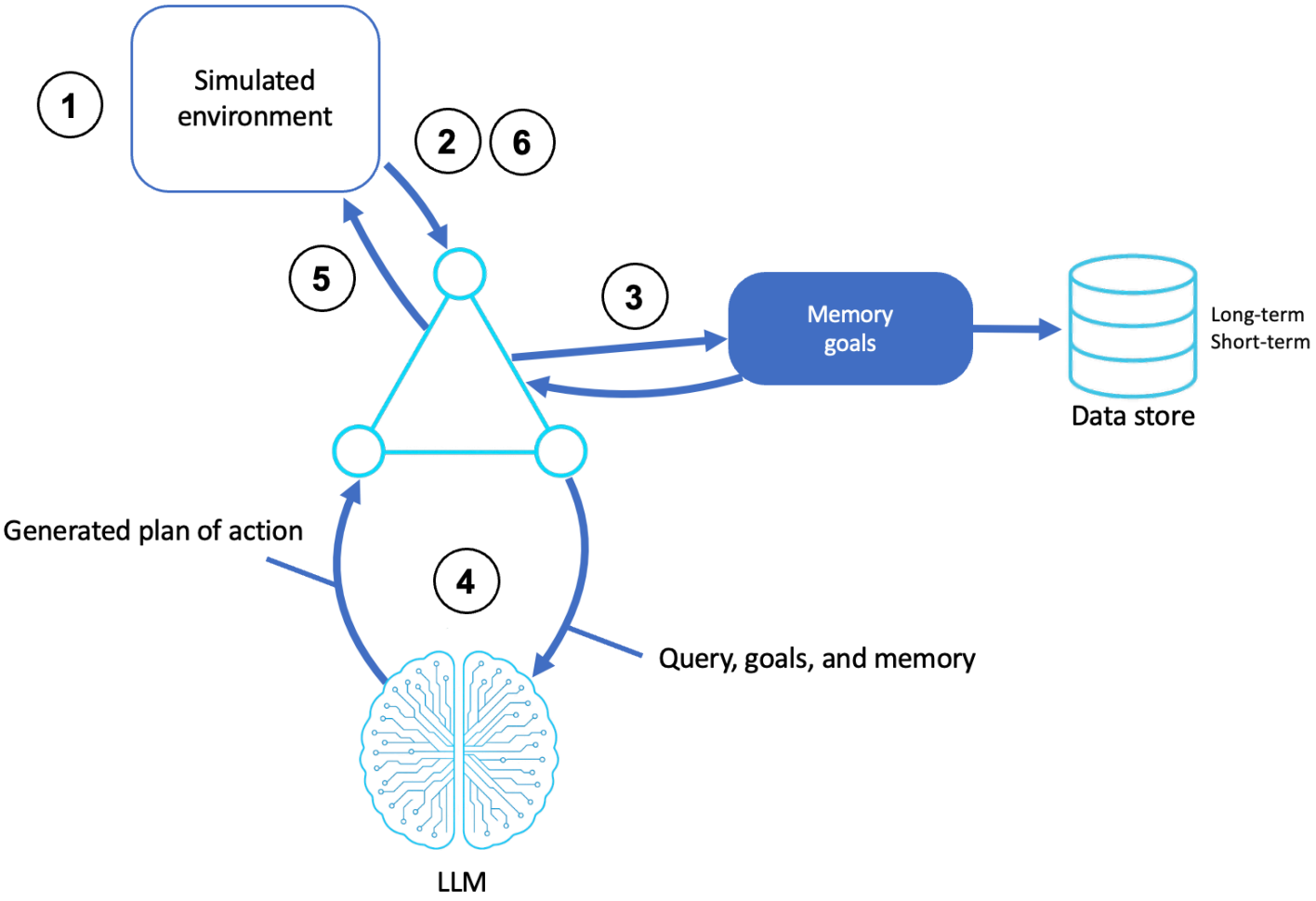
仿真和测试平台代理

仿真和测试平台代理在虚拟化或受控环境中运行，在那里他们可以推理、行动和学习。这些代理在将策略应用于现实环境之前，先在可重复的环境中模拟行为、建模结果并训练策略。

这种模式对于迭代开发、强化学习 (RL)、自主决策评估和紧急行为测试非常有用。仿真代理通常在闭环中运行，接收来自其环境的反馈并相应地调整其行为，这使得它们对于涉及空间推理、实时控制或复杂系统动力学的任务至关重要。

Architecture

下图显示了仿真或测试平台代理：



描述

1. 启动环境
   - 代理启动模拟环境（例如，3D 世界、物理引擎、CLI 沙盒或合成数据流）。
   - 将代理加载到环境中，并附有初始任务、目标或策略。
2. 感知特工
   - 代理通过仿真遥测（例如，传感器仿真、虚拟摄像机和结构化日志）感知当前状态。
3. 检索目标和记忆
   - 代理检索其分配的目标、场景说明或上下文目标。
   - 它还可以检索以前的内存，包括以下内容：
     - 长期战略或政策
     - 环境地图或已知限制
     - 类似模拟中过去的成功或失败

4. 原因和计划
    - 法学硕士解释模拟状态、任务目标和所学知识。
    - 它生成行动计划或控制命令。
5. 执行模拟动作
    - 代理执行计划、修改状态、在空间中导航或与虚拟实体交互。
6. 学习
    - 代理评估行动结果
    - 根据代理的配置，它可能会执行以下操作：
        - 执行 RL
        - 记录结果以备将来微调
        - 实时调整策略

功能

- 可在合成环境或虚拟环境中运行
- 支持 trial-and-error学习、策略完善和系统建模
- 针对行为、故障处理和边缘案例的低风险测试
- 支持在多智能体设置中进行紧急代理行为分析
- 支持闭环控制和探索 human-in-the-loop

常见使用案例

- 机器人、无人机和游戏的强化学习
- 在虚拟道路上进行自动驾驶车辆训练
- 模拟 UIs 或 CLIs 针对 DevOps 和试验台场景
- 社交模拟中的紧急行为实验
- 在生产前对决策逻辑进行安全验证

实施指导

您可以使用以下工具和以下工具构建仿真和测试平台代理： AWS 服务

组件	AWS 服务	目的
环境	亚马逊 ECS EC2、Amazon 或亚马逊 SageMaker工作室实验室中的自定义模拟器	运行虚拟世界（凉亭、Unity、虚幻）或沙盒 CLIs
代理逻辑	亚马逊 Bedrock SageMaker 、Amazon 或 AWS Lambda	基于 LLM 的策划者或 RL 代理商
反馈回路	亚马逊 SageMaker 强化学习 CloudWatch、Amazon 或自定义日志	奖励跟踪、结果评分和行为记录
记忆和重播	亚马逊 S3、亚马逊 DynamoDB 或亚马逊 RDS	永久状态、剧集历史记录或场景数据
可视化	亚马逊 CloudWatch 控制面板或亚马逊 SageMaker 笔记本	观察政策变化、结果和培训指标

以下是其他应用程序：

- AWS SimSpace Weaver用于大规模空间模拟
- AWS IoT Core用于测试影子设备
- 用于代理评估和基准测试的 Amazon SageMaker 实验

Summary

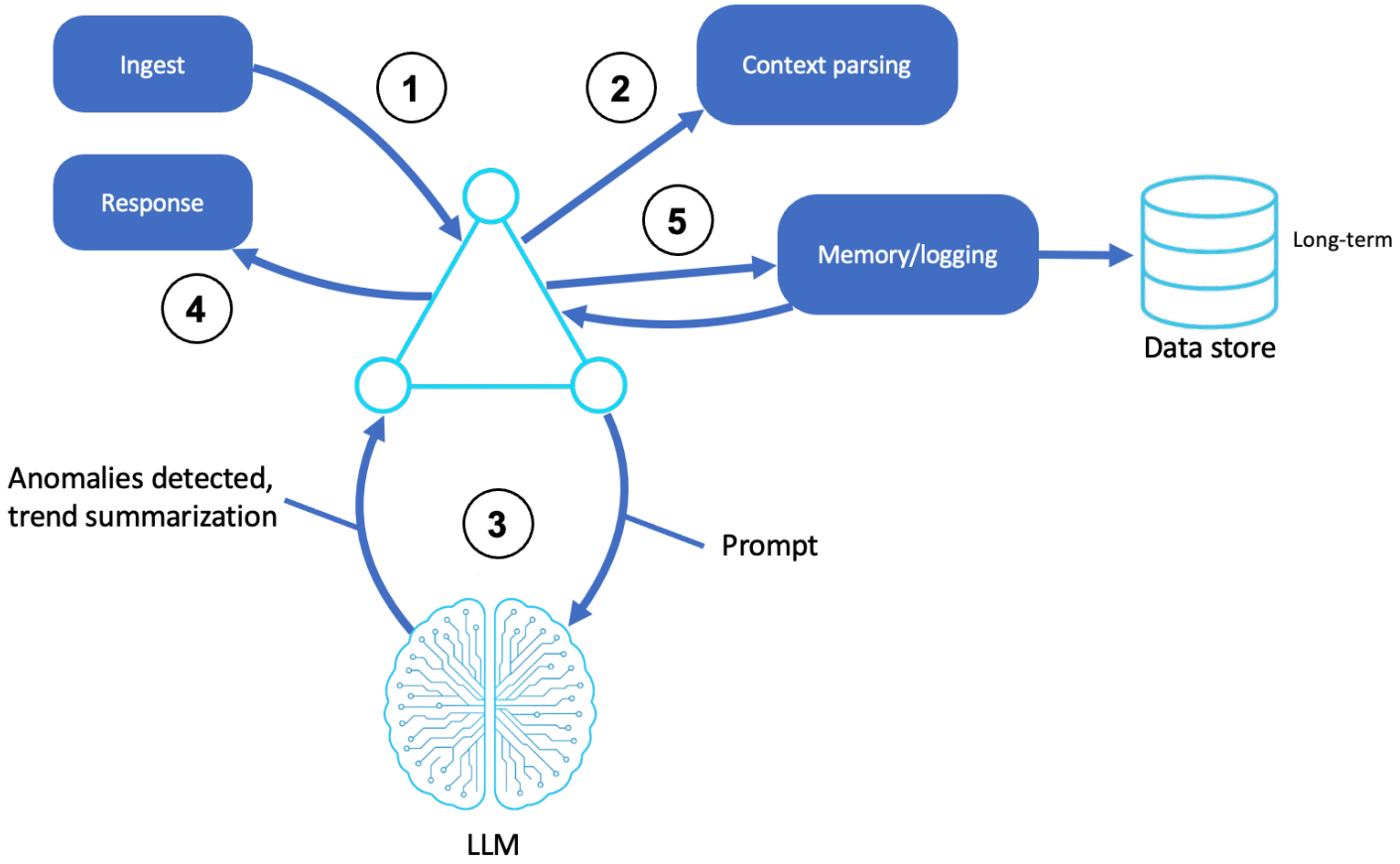
仿真和测试平台代理用于在部署到生产系统之前进行结构化探索。使用这些代理来训练自主导航策略，在合成环境中测试业务流程，并评估群体的协调模式。

观察者和监视代理

观察者和监视代理被动观察系统、环境和交互，以检测模式、生成见解并触发操作。作为智能观察者，他们无需直接启动行为即可增强警报、诊断和审计。

这些代理在传统监控缺乏适应性或推理性方面表现出色，尤其是在 AI-in-the-loop监控、异常检测、合规监督和安全情报方面。观察者代理是持续监控系统遥测和用户交互的事件侦听器。代理取决于感知、解释以及有条件的上报或报告。

Architecture



描述

1. 摄取遥测数据
   - 代理接收来自一个或多个系统源的输入，例如：
     - 日志（应用程序、基础架构、安全）
     - 指标（性能、延迟、使用情况）
     - 事件（API 调用、用户操作、传感器数据）
2. 解析上下文
   - 原始输入经过解析、结构化并丰富元数据，例如时间戳、参与者身份、系统状态和跟踪 ID。
3. 使用法学硕士学位的原因

- 代理使用 LLM 或逻辑模块通过识别异常、汇总趋势以及跨分布式跟踪或时间窗口关联来解释已解析的输入。

4. 分类或提醒
   - 代理确定所观察到的行为是否符合以下条件：
     - 警报或升级
     - 报告或控制面板更新
     - 响应触发器（例如，自动补救和策略执行）
5. 记录内存或反馈回路
   - 代理存储事件和决策，以供其他代理长期学习、审计或将来参考。

功能

- 被动和非侵入性（代理不直接起作用）
- 高度可扩展和异步
- 人工智能驱动的噪声或分布式信号之间的关联
- 支持审计、合规和实时洞察
- 可以为下游代理或人工工作流程提供信息

常见使用案例

- 微服务的 AI 增强可观测性和 APIs
- 监控模型偏差、策略违规 out-of-band行为或行为
- 客户活动分析或互动摘要
- 监视提交或部署的代码审查代理
- 使用 LLM 推理进行安全或合规日志监控

实施指导

您可以使用以下工具和以下工具来构建观察者和监视代理 AWS 服务：

组件	AWS 服务	目的

事件摄取	亚马逊 EventBridge、亚马逊 CloudWatch 日志、亚马逊 Kinesis、亚马逊 S3	摄取结构化和非结构化遥测
预处理	AWS Lambda, AWS Glue, AWS Step Functions	将原始数据转换为结构化提示
推理引擎	亚马逊 Bedrock、亚马逊 SageMaker、 AWS Lambda	分析事件、对行为进行分类、生成见解
存储和内存	亚马逊 S3、亚马逊 DynamoDB、 OpenSearch	持续的观察、总结和输出
警报和升级	亚马逊 SNS、、 AWS AppFabric亚马逊 EventBridge	触发下游系统或代理

以下是其他应用程序：

- AWS Security Hub CSPM用于安全日志监控
- 用于可视化代理输出的 Amazon Quick Suite

Summary

观察者和监视代理可以实时跟踪系统和行为。他们通过识别人类或规则可能忽视的模式来检测异常、审计安全并收集运营情报。此功能有助于创建能够适应不断变化的条件并基于全面的数据分析做出决策的系统。

多代理协作

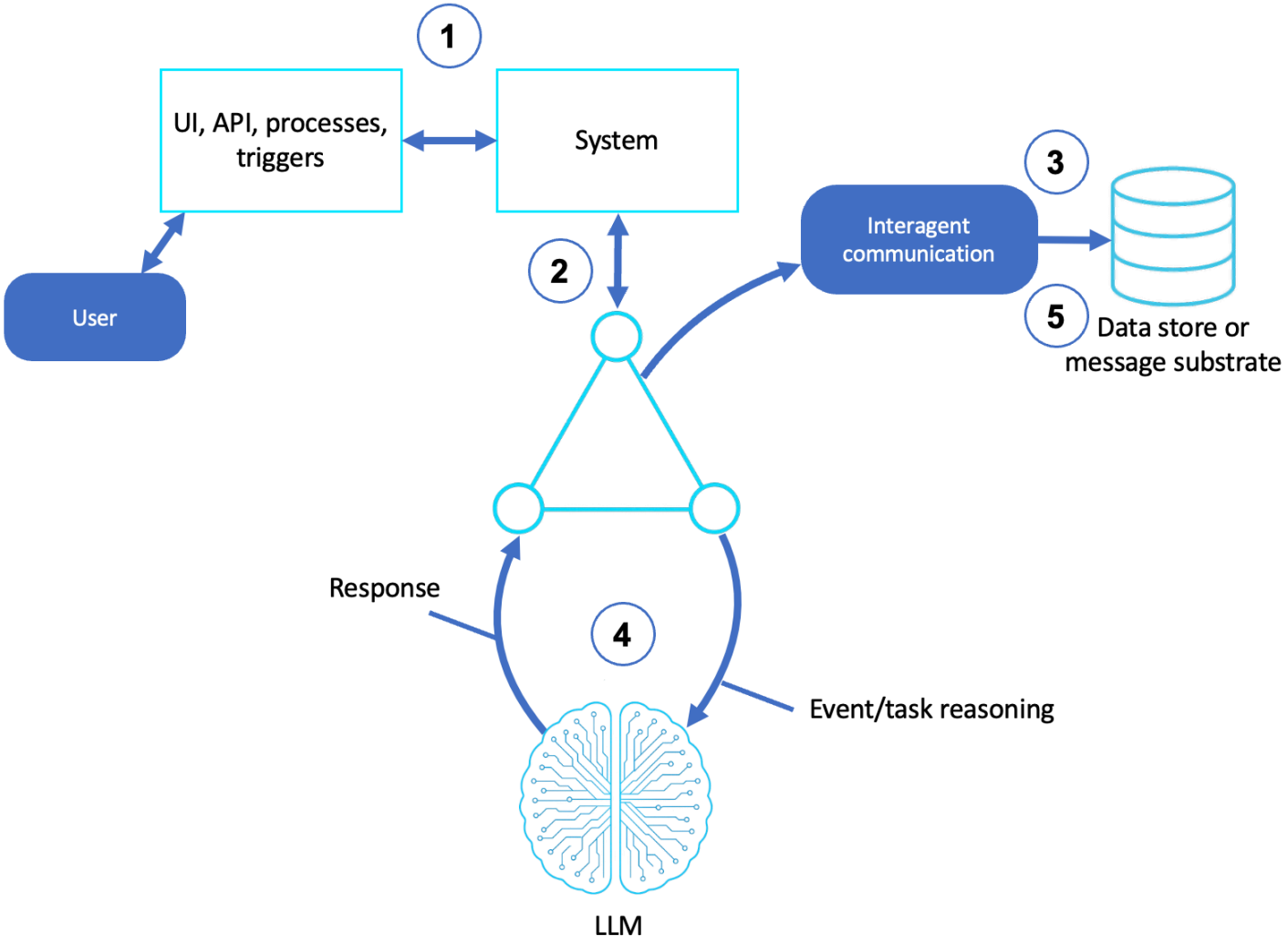
多代理协作是指一种模式，在这种模式中，多个自主代理通过协商解决复杂的任务，每个代理都有不同的角色、专业或目标。这些代理人可以通过共享信息、划分责任和共同推理目标来独立运作，也可以与其他代理人一起行动。

这种模式不同于工作流代理，后者在结构化流程中集中协调任务并将其委派给下属代理。相比之下，多主体协作通过实现适应性、平行性和认知分裂来强调 peer-to-peer紧急协调。下表比较了多代理与工作流代理的协作：

功能	工作流程代理	目的
控件	集中式协调员	去中心化、分布式或基于角色的 Peer 节点
交互	一个代理委托并跟踪执行情况	多个代理进行协商、共享和调整
设计	预定义的任务顺序	快速、灵活的任务分配
协调	程序编排	合作或竞争互动
使用案例	企业流程自动化	复杂的推理、探索和应急策略

Architecture

下图显示了多代理协作：



描述

1. 启动任务
   - 用户或系统发出高级目标或问题。
   - “经理” 代理或启动上下文定义了目标。
2. 分配或发现角色
   - 代理人自行分配（符号逻辑或推理）或被委托（事件经纪人）到其他角色，例如策划者、研究员、执行者、批评者或解释者。
3. 与其他代理沟通
   - 代理通过共享内存、消息队列或提示链进行通信。
   - 他们可能会互相辩论、询问或提出子任务。
4. 使用专门的推理

- 每个代理都使用自己的模型或域逻辑来解决自己的问题部分。
- 代理可以 LLMs 与角色特定的提示和内存一起使用。

5. 协调产出或目标
   - 代理人将贡献综合成最终答案、计划或行动。
   - （可选）监督代理可以验证或汇总合成后的输出。

功能

- 具有特殊角色或技能的同行级代理
- 通过沟通或谈判出现的紧急行为
- 并行处理复杂或多方面的问题
- 支持深思熟虑、自我纠正和反思性迭代
- 为社交动态、科学合作或企业团队角色建模

常见使用案例

- 自主研究团队（搜索代理、摘要器和验证器）
- 软件开发（规划员、编码员和测试员）
- 业务场景建模（财务、策略和合规）
- 谈判、竞标或多方推理
- 多模式任务（图像、文本和逻辑）

实施指导

您可以使用以下工具和 AWS 服务以下工具构建多代理系统：

组件	AWS 服务	目的
代理托管	亚马逊 Bedrock、亚马逊 SageMaker、 AWS Lambda	托管个人 LLM 驱动的代理
通信层	亚马逊 SQS、亚马逊、EventBridge AWS AppFabric	代理之间的消息传递和协调

共享内存	亚马逊 DynamoDB、亚马逊 S3 或 OpenSearch	多代理存储器或黑板系统
编排层	AWS Step Functions， AWS Lambda 管道	启动、超时、回退和重试逻辑
代理识别	亚马逊 Bedrock 代理（角色定义）和 AWS AppConfig 亚马逊 Bedrock converse API（亚马逊 Bedrock 以外的代理）	基于角色的工具或代理调用和边界强制执行
紧急互动	Amazon EventBridge 渠道或代理注册表	启用动态任务路由或升级

Summary

多代理协作将问题解决任务分配给模块化、角色驱动的代理。与工作流程协调不同，协作模式使用的是反映人类解决问题的新兴智能、弹性和可扩展性。对于开放式领域、创造性任务、多模态推理和受益于不同视角的环境来说，它尤其有价值。

结论

前面讨论的模式说明了在现实世界中实现代理人工智能的基本方法。从基本推理到记忆增强智能，每种模式都针对基于自主性、异步性和代理性的感知、认知和行动进行了独特的配置。

这些模式共享用于构建智能、以目标为导向的系统的词汇和技术蓝图。无论模式是嵌入在用户界面中，还是通过云服务进行编排，还是跨代理团队进行协调，每种模式都具有适应性和模块化性。

外卖

- 代理模式是可组合的 — 大多数现实世界中的代理会混合两种或更多模式（例如，具有基于工具的推理和记忆的语音代理）。
- 代理设计是基于情境的 — 根据交互表面、任务复杂性、延迟容限和特定领域的限制来选择模式。
- AWS原生实现是可以实现的 — 借助 Amazon Bedrock SageMaker AWS Lambda、、 AWS Step Functions、、和事件驱动架构，每种代理模式都可以大规模交付。

法学硕士工作流程

在代理模式中，我们探索了常见的 AI 代理模式，每种模式都围绕一组模块化功能构建：感知、行动、学习和认知。在许多代理模式中，认知模块的核心是能够进行推理、计划和决策的大型语言模型 (LLM)。但是，仅调用法学硕士学位不足以产生明智的、以目标为导向的行为。

为了可靠地执行复杂的任务，代理必须将 LLM 嵌入结构化工作流程中，通过工具、内存、计划循环和协调逻辑增强模型的功能。这些 LLM 工作流程允许代理分解目标、路由子任务、呼叫外部服务、反思结果以及与其他代理进行协调。

本章介绍构建强大、可扩展和智能 LLM 驱动的认知模块的核心设计模式，这些模块围绕可重复使用的工作流程进行组织。

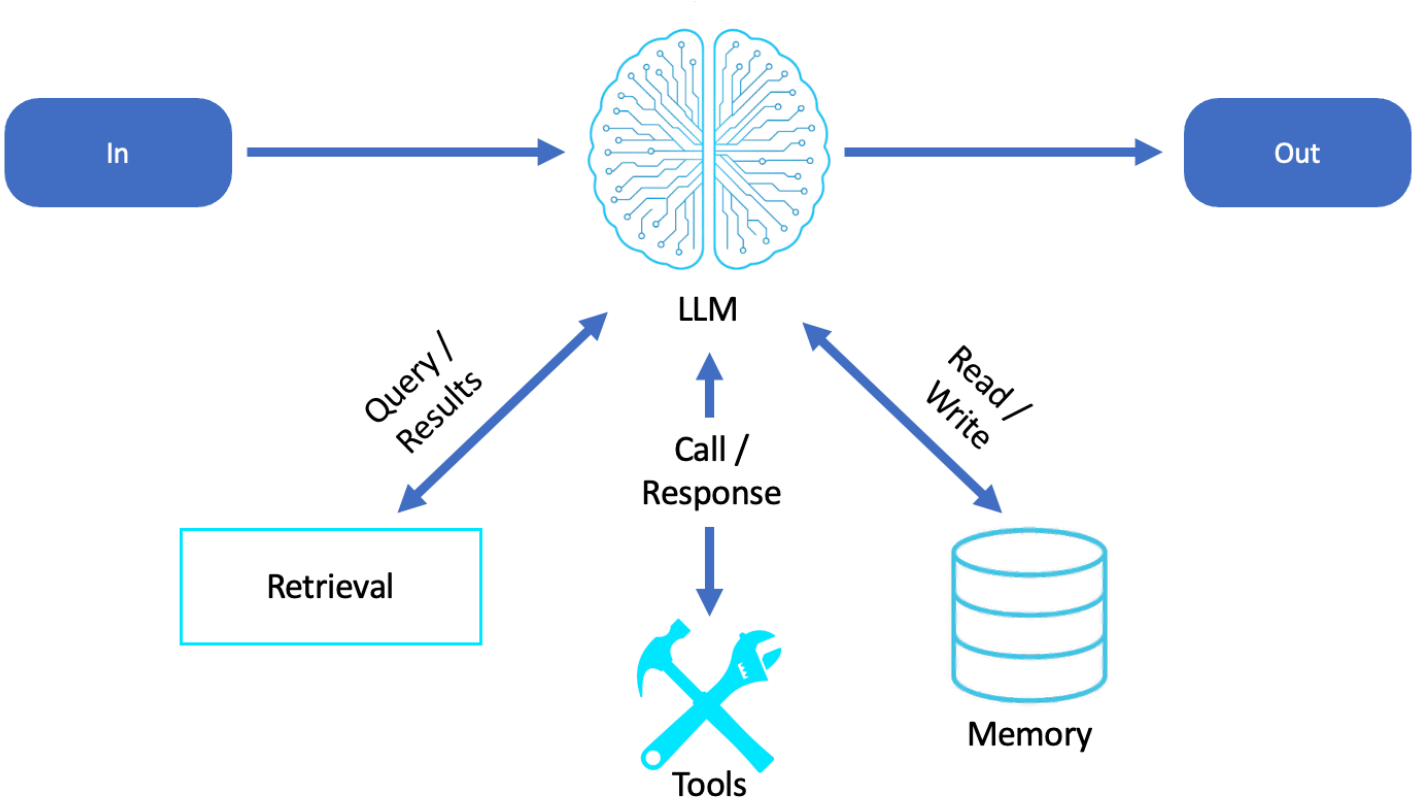
本节内容

法学硕士增强认知概述

从本质上讲，软件代理的认知模块可以看作是包含增强功能的法学硕士。代理可以使用以下构造块在其环境中进行有效的推理：

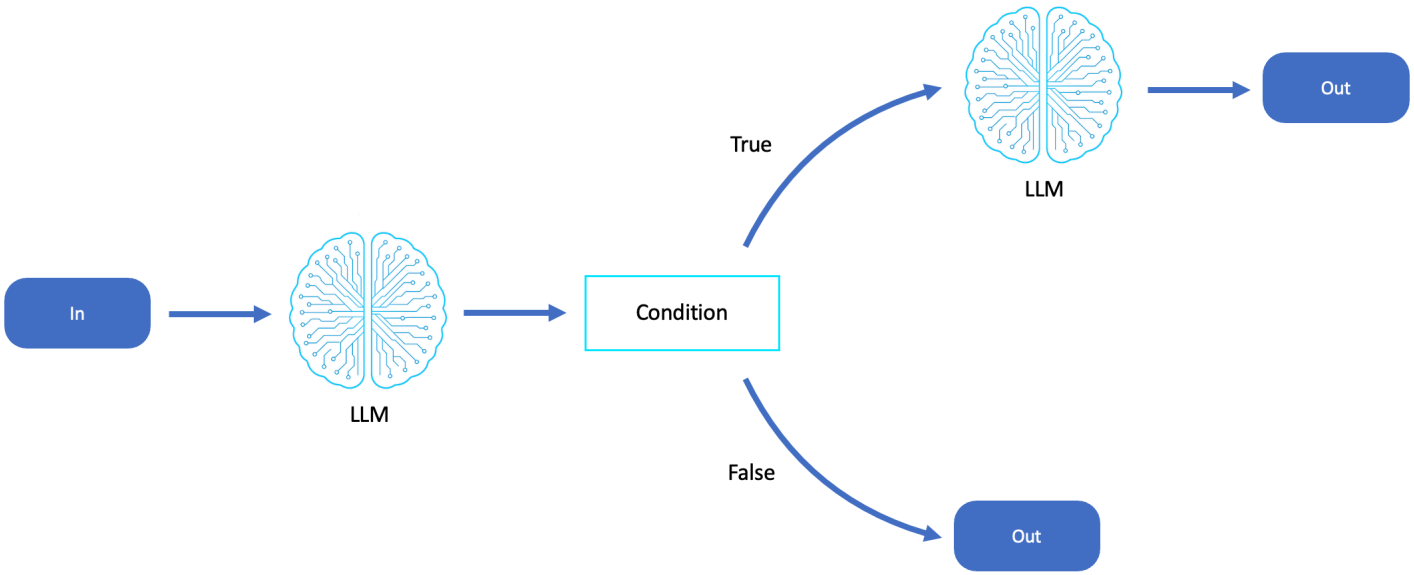
- 提示 — 使用上下文、指令、示例和内存来构图输入
- 检索 — 通过向量搜索 up-to-date或语义记忆（例如，通过检索增强生成 (RAG)）向 LLM 提示符提供或特定领域的知识
- 工具使用-允许 LLM 调用 APIs 或调用函数来检索或处理信息
- 记忆 — 通过使用结构化数据库或情境摘要，将持续状态或基于会话的状态整合到推理循环中

这些增强功能由工作流程组成，这些工作流定义了如何随着时间的推移和跨任务使用 LLM，将其从无状态引擎转变为动态推理代理。



提示链接的工作流程

Prompt chaining 将复杂的任务分解为一系列步骤，其中每个步骤都是一个离散的 LLM 调用，用于处理或建立在前一个任务的输出之上。



提示链接工作流程适用于可以在逻辑上将任务划分为顺序推理步骤以及中间输出为下一阶段提供信息的场景。它在需要结构化思维、渐进式转换或分层分析的工作流程中表现出色，例如文档审查、代码生成、知识提取和内容完善。

说明

- 任务的复杂性超过了单个 LLM 调用的上下文窗口或推理深度。
- 一个步骤（例如分析、总结或规划）的输出将成为后续决策或生成阶段的输入。
- 您需要跨推理阶段的透明度和控制力（例如，可审计的中间结果）。
- 您想在步骤之间插入外部验证、筛选或扩充逻辑。
- 它非常适合在管道式推理循环中操作的代理，例如研究代理、编辑助理、计划系统和多阶段副驾驶。

功能

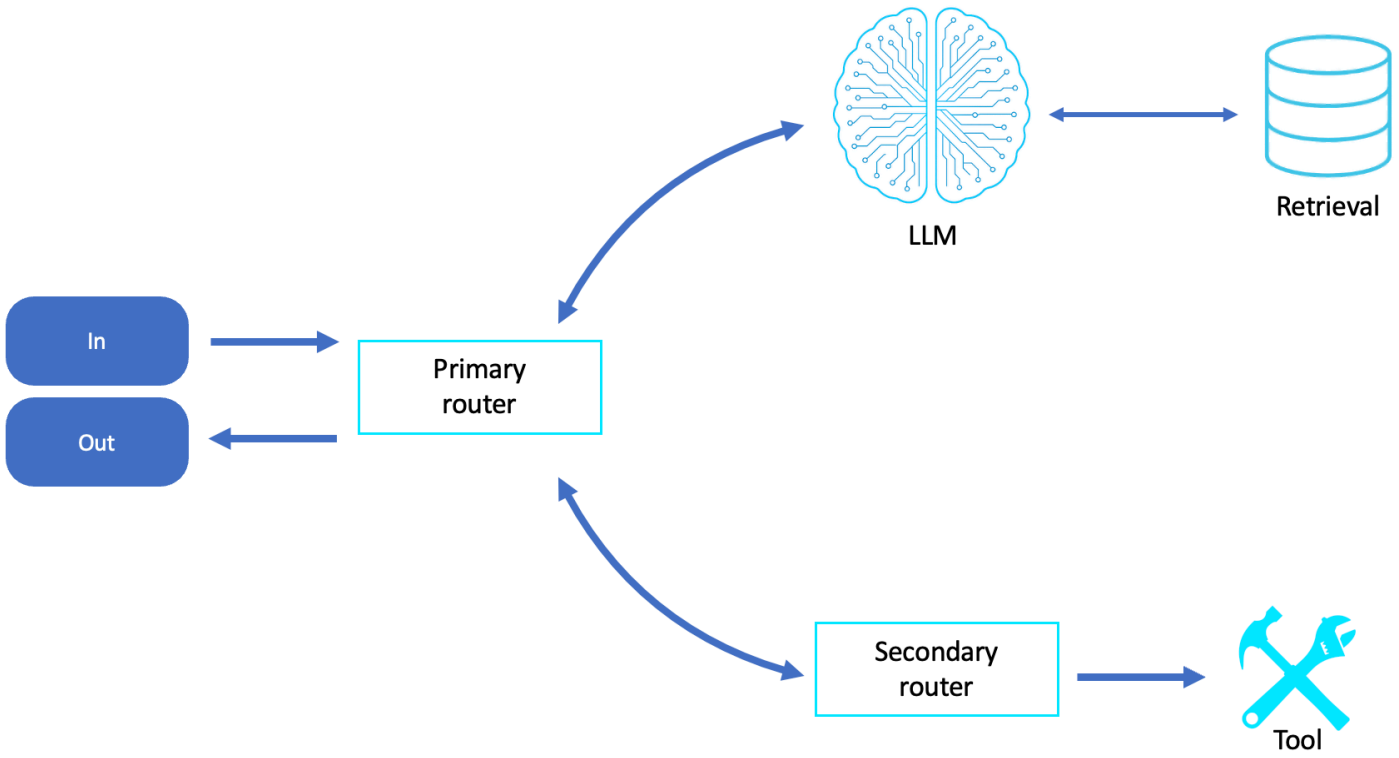
- LLM 调用的线性链或分支链
- 中间结果作为结构化输入传递或嵌入到后续提示中
- 可以与、或代理特定的运行 AWS Step Functions器 AWS Lambda一起编排

常见使用案例

- 多步推理任务（例如，“总结批评重写”）
- 研究助理合成分层输出（例如，“搜索摘录事实回答问题”）
- 代码生成管道（“生成计划编写代码测试代码解释输出”）

路由工作流程

在路由模式中，分类器或路由器代理使用 LLM 来解释查询的意图或类别，然后将输入路由到专门的下游任务或代理。



路由工作流用于代理必须快速对输入意图、任务类型或域进行分类，然后将请求委托给专门的子代理、工具或工作流的场景。它在能力代理中特别有用，例如充当一般助理、企业功能的前门或跨域的面向用户的 AI 接口。

在以下情况下，路由特别有效：

- 对各种任务（例如搜索、汇总、预订、计算）的请求进行分类。
- 在进入更专业的工作流程之前，必须对输入进行预处理或标准化。
- 不同的输入类型（例如，图像与文本、结构化查询与非结构化查询）需要自定义处理。
- 代理充当对话总机，将任务委托给专业代理或微服务。
- 这种工作流程在特定领域的副驾驶、客户支持机器人、企业服务路由器和多模式代理中很常见，在这些代理中，智能调度决定了代理行为的质量和效率。

功能

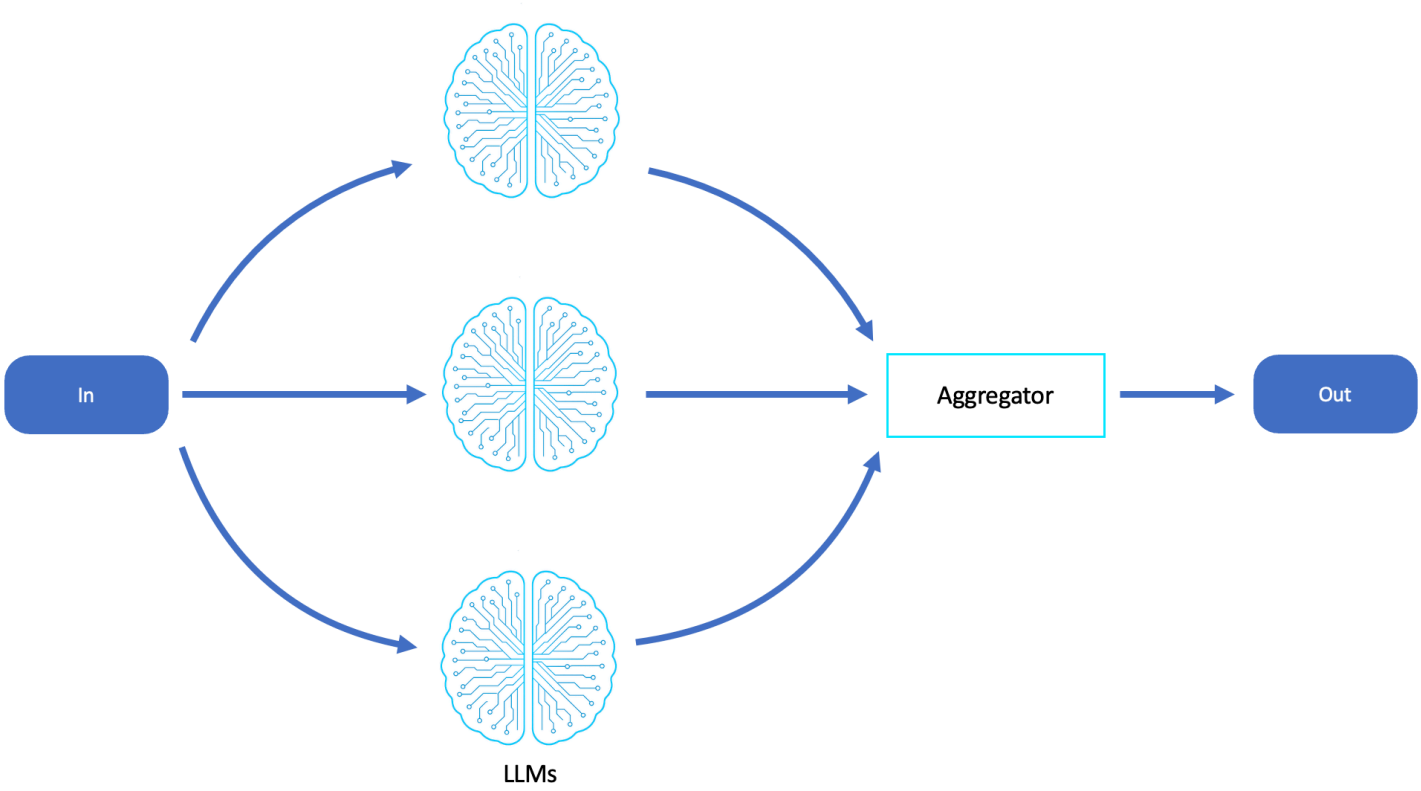
- 初级法学硕士充当调度员
- 路由可以调用不同的工作流程，甚至可以调用其他代理模式
- 支持功能的模块化扩展

常见使用案例

- 多域助手（“这是法律、医疗还是财务问题？”）
- 通过法学硕士推理增强决策树
- 动态工具选择（例如，搜索与代码生成）

并行化工作流程

此工作流程包括将任务分解为独立的子任务，这些子任务可以由多个 LLM 呼叫或代理同时处理。然后以编程方式汇总输出并合成结果。



Parallelization 工作流程用于将任务划分为可以同时处理的独立、非顺序的子任务，从而显著提高效率、吞吐量和可扩展性。它在数据密集、批处理导向或多视角问题空间中特别强大，在这些空间中，代理必须通过多个输入分析或生成内容。

在以下情况下，并行化特别有效：

- 子任务不依赖彼此的中间结果，允许它们在没有协调的情况下并行运行。
- 一项任务涉及对许多项目重复相同的推理过程（例如，总结多个文档或评估选项列表）。

- 并行探索多种假设或观点，以促进多样性、创造力或稳健性。
- 您需要通过并发 LLM 执行来减少大容量或高频请求的延迟。
- 此工作流程通常用于文档处理代理、调查或比较引擎、批量汇总器、多代理头脑风暴以及可扩展的分类或标签任务，尤其是在快速、并行推理具有性能优势的情况下。

功能

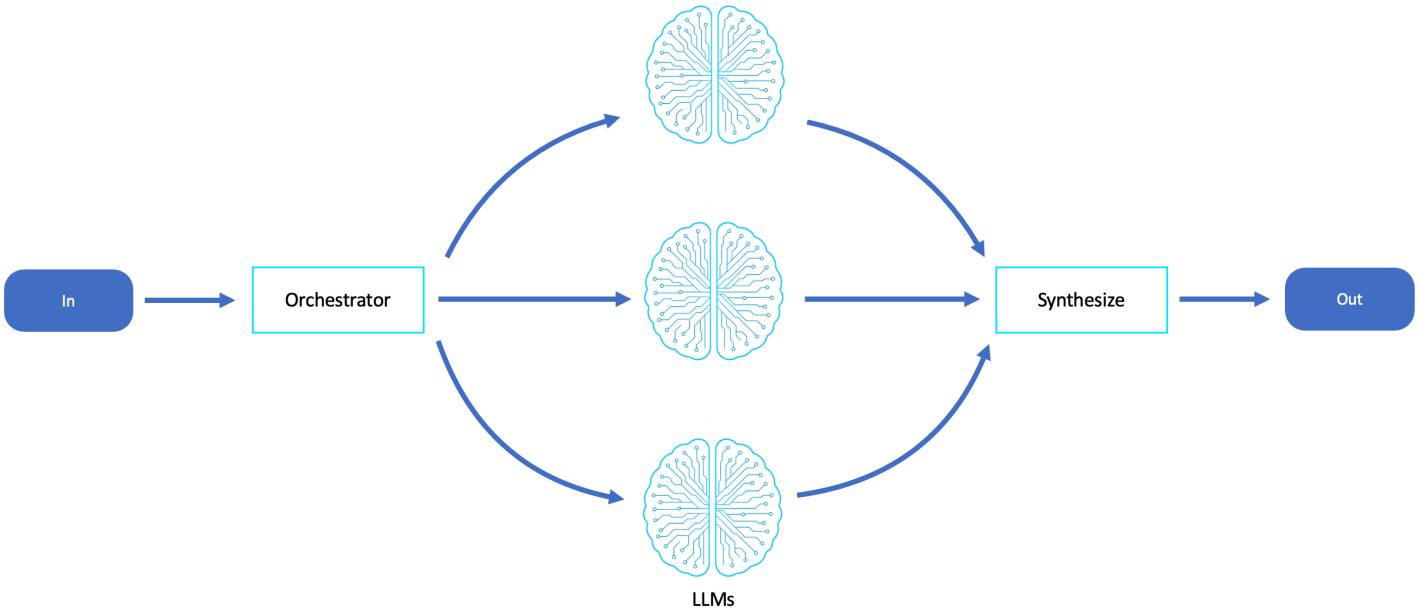
- 并行执行 LLM 任务（通过使用 AWS Lambda AWS Fargate、或 AWS Step Functions 映射状态）
- 需要在综合阶段对结果进行校准、验证或重复数据删除
- 非常适合无状态代理循环

常见使用案例

- 并行分析多个文档或视角
- 生成不同的草稿、摘要或计划
- 加快批处理作业的吞吐量

编排工作流程

中央协调器代理使用 LLM 来规划、分解子任务，并将子任务委托给专门的工作代理或模型，每个代理或模型都有特定的角色或领域专业知识。这反映了人类团队的结构，并支持多个代理的紧急行为。



编排工作流程非常适合复杂、分层或多学科的场景，需要结构化分解和专业执行。它特别适合需要分工的任务，在这种情况下，任务的不同子组件最好由具有不同能力、知识或工具集的代理来处理。

在以下情况下，此工作流程特别有效：

- 任务可以分为范围、类型或推理各不相同的子任务（例如，计划、研究、实施和测试）。
- 法学硕士或元代理必须协调其他代理，监控进度并综合结果。
- 您想对代理职责进行模块化，从而实现可扩展性、重复使用和专业调整。
- 该系统要求基于角色的行为，模仿人类团队（例如项目经理、开发人员和审阅者）的协作运作方式。

Orchestration 非常适合多回合计划代理、软件开发副驾驶、企业流程代理和自主项目执行者。在实现需要集中式任务分解但需要分布式执行逻辑的多代理系统时，它特别有用，可以跨代理层实现可扩展性和更易于解释的行为。

功能

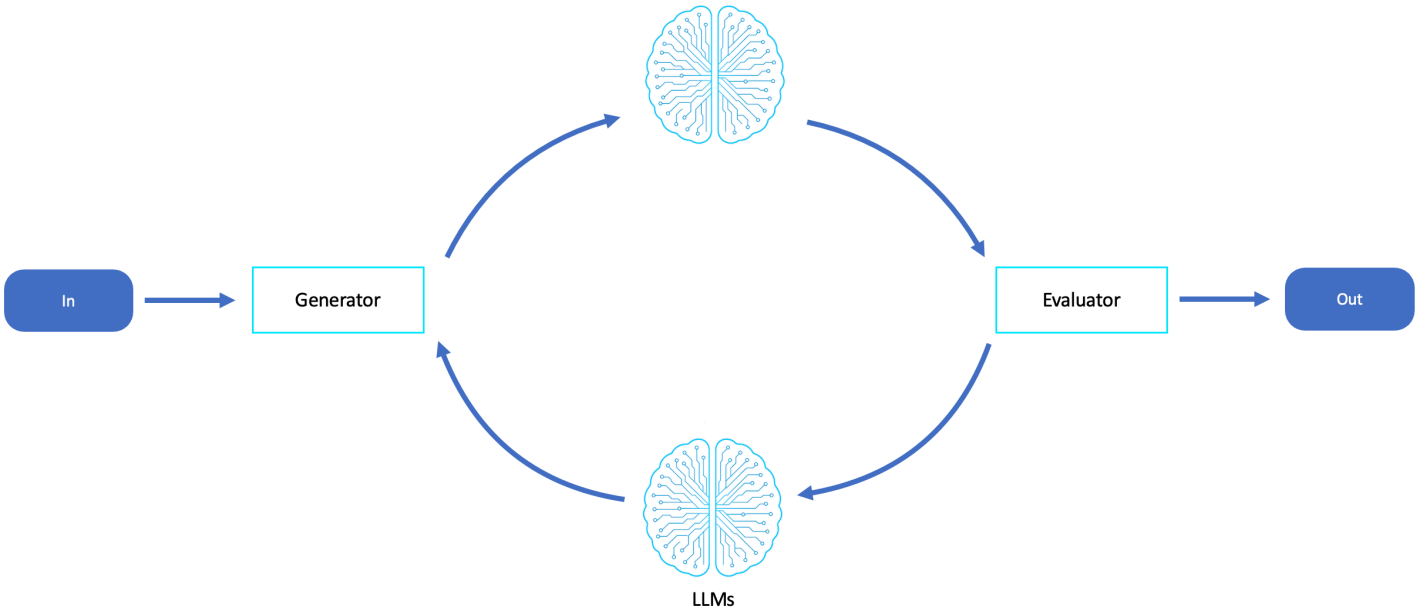
- Orchestrator 执行目标元推理
- Worker 代理可能包括工具访问权限、内存或特定域的提示
- 可以是分层的（即多级任务委派）

常见使用案例

- 项目经理、协调研究人员、作家和质量保证人员
- 将计划、执行和测试结合在一起的编程副驾驶
- 监督工具链或 API 访问模式的代理

赋值器的工作流程和反射优化循环

该工作流程提供了一个反馈循环，其中一个 LLM 生成结果，另一个法学硕士评估或批评结果。这促进了自我反思、优化和迭代改进。



评估器工作流程非常适合输出质量、准确性和对齐性很重要，以及单通道生成不可靠或不足的场景。当工程师必须自我批评、迭代和完善其输出时，这种工作流程非常出色，要么是为了达到更高的正确性标准，要么是为了根据反馈探索改进的替代方案。

在以下情况下，此工作流程特别有效：

- 输出涉及主观质量指标（例如风格、语气和可读性）或客观标准（例如正确性、安全性和性能）。
- 代理必须权衡推理、评估约束条件或针对目标进行优化。
- 您需要内置的冗余和质量保证，尤其是在受管制、面向客户或创意领域。
- Human-in-the-loop 审核费用昂贵或不可用，需要自主验证。

此工作流程用于内容生成、代码合成和审查、策略执行、对齐检查、指令调整和 RAG 后处理。它对自我完善的代理也很有用，在这些代理中，持续的反馈有助于随着时间的推移形成更好的响应，从而建立值得信赖的自主决策循环。

常见使用案例

- 红队特工与蓝队经纪人的比较
- 生成、评估和修改代码或计划的代理
- 质量保证、幻觉检测和风格强制执行

功能

- 支持使用不同的模型进行解耦生成和评估（例如，Claude 用于生成，Mistral 用于评估）
- 反馈是结构化的，用于提示修改后的产出
- 支持多次迭代或收敛阈值

结论

LLMs 提供了现代软件代理的认知核心，但是原始模型调用不足以实现有目的、强大和可控的智能。要从输出生成转向结构化推理和目标一致的行为， LLMs 必须嵌入到定义模型如何处理输入、管理上下文和协调操作的有意的工作流程模式中。

法学硕士工作流程引入了构建代理认知模块的基础：

- 提示链接将复杂的推理分解为模块化、可审计的步骤。
- 路由支持智能任务分类和定向委派。
- 并行化可加快吞吐量并促进多样化推理。
- 代理编排通过任务分解和基于角色的执行来构建多代理协作。
- Evaluator（反射精炼循环）可实现自我完善、质量控制和校准检查。

每个工作流程都代表一种可组合模式，可以根据代理的需求、任务的复杂性和用户的期望进行调整。这些工作流程并不相互排斥。它们是构建块，通常组合成支持动态推理、多代理协调和企业级可靠性的混合架构。

当你过渡到关于代理工作流程模式的下一章时，这些 LLM 工作流程将作为嵌入式结构重新出现在更大的系统中，支持目标授权、工具编排、决策循环和生命周期自主权。掌握这些法学硕士工作流程对于设计软件代理至关重要，这些代理不仅可以预测文本，还可以有目的地进行推理、调整和行动。

代理工作流程模式

Agentic 工作流程模式将模块化软件代理与结构化大型语言模型 (LLM) 工作流程集成在一起，从而实现自主推理和操作。虽然受到传统的无服务器和事件驱动架构的启发，但这些模式将核心逻辑从静态代码转移到增强的 LLM 代理，从而增强了适应性和上下文决策。这种演变将传统的云架构从确定性系统转变为能够进行动态解释和智能增强的架构，同时保持了可扩展性和响应能力的基本原则。

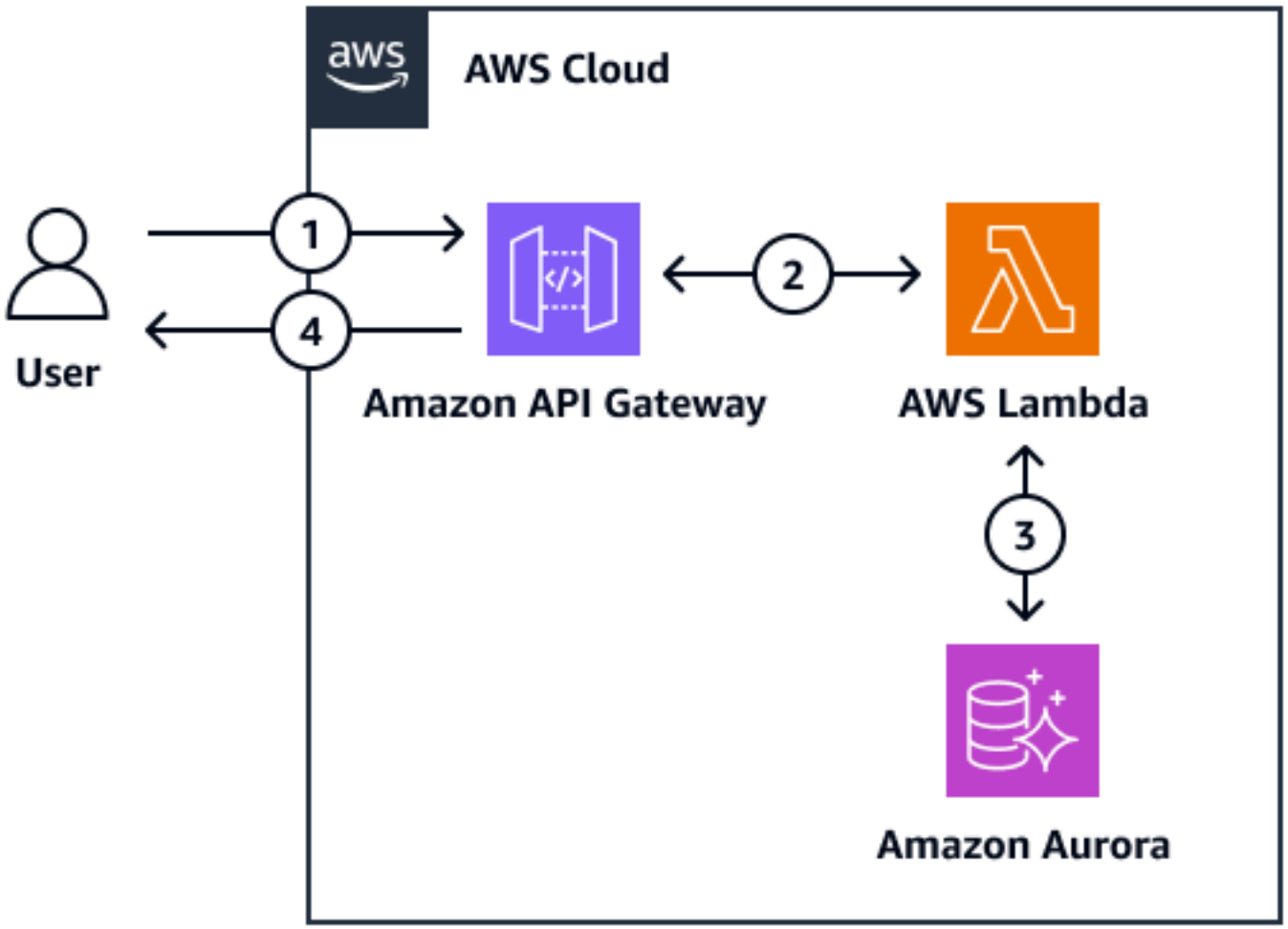
本节内容

从事件驱动到认知增强系统

现代云架构，尤其是那些基于无服务器和事件驱动原则的云架构，传统上依赖路由、扇出和丰富等模式来创建响应迅速、可扩展的系统。Agentic AI 系统建立在这些基础之上，同时围绕法学硕士增强推理和认知灵活性对其进行重新构建。这种方法允许更复杂的问题解决和自动化功能，有可能彻底改变云环境中处理复杂任务的方式。

事件驱动型架构

下图显示了一个典型的分布式系统：

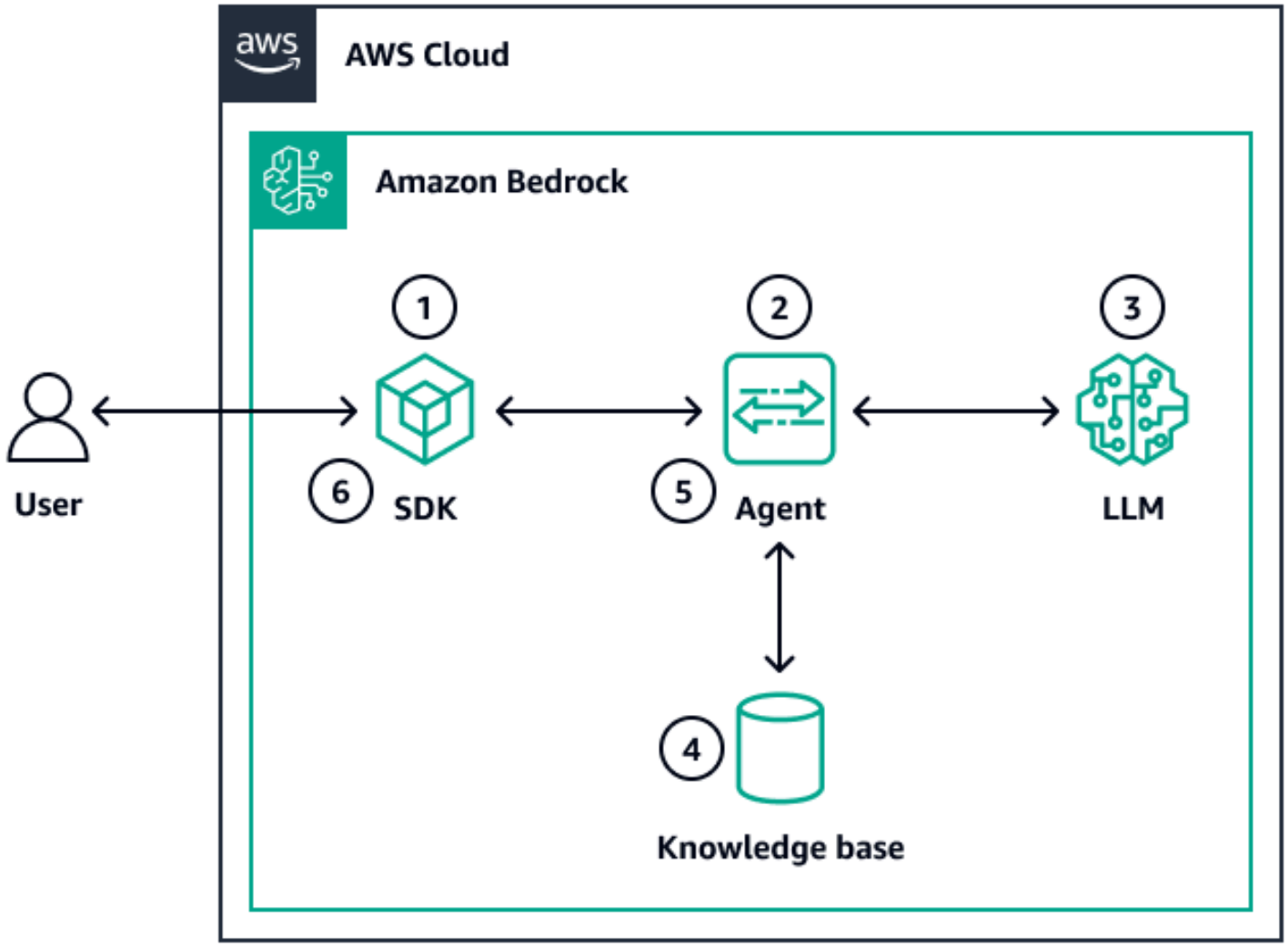


1. 用户向 Amazon API Gateway 提交请求。
2. Amazon API Gateway 会将请求路由到一个 AWS Lambda 函数。
3. AWS Lambda 通过查询 Amazon Aurora 数据库来进行数据扩充
4. Amazon API Gateway 会将充实的有效负载返回给调用方。

这种结构既可靠又可扩展，但它基本上是静态的。必须对业务规则和逻辑路径进行明确编码，适应不断变化的上下文或不完整信息是有限的。

认知增强工作流程

Agentic 架构为事件驱动的系统增加了认知增强功能。下图显示了代理等效物：



1. 用户通过 SDK 或 API 调用提交查询。
2. Amazon Bedrock 代理收到查询。
3. 代理通过调用 LLM 来解释查询
4. 代理通过搜索 Amazon Bedrock 知识库或其他外部数据源来进行语义扩充。
5. 法学硕士综合了上下文丰富、目标一致的回应。
6. 系统向用户返回合成响应。

在此流程中，法学硕士使用逻辑，了解意图，检索并组合相关上下文，然后决定如何最好地做出响应。这种模式反映了传统的扩充模式，即消息在进一步路由之前先用外部数据进行扩充。但是，在代理系统中，这种浓缩并不是一种静态的查找。相反，丰富是动态的、语义指导的，由目的驱动。

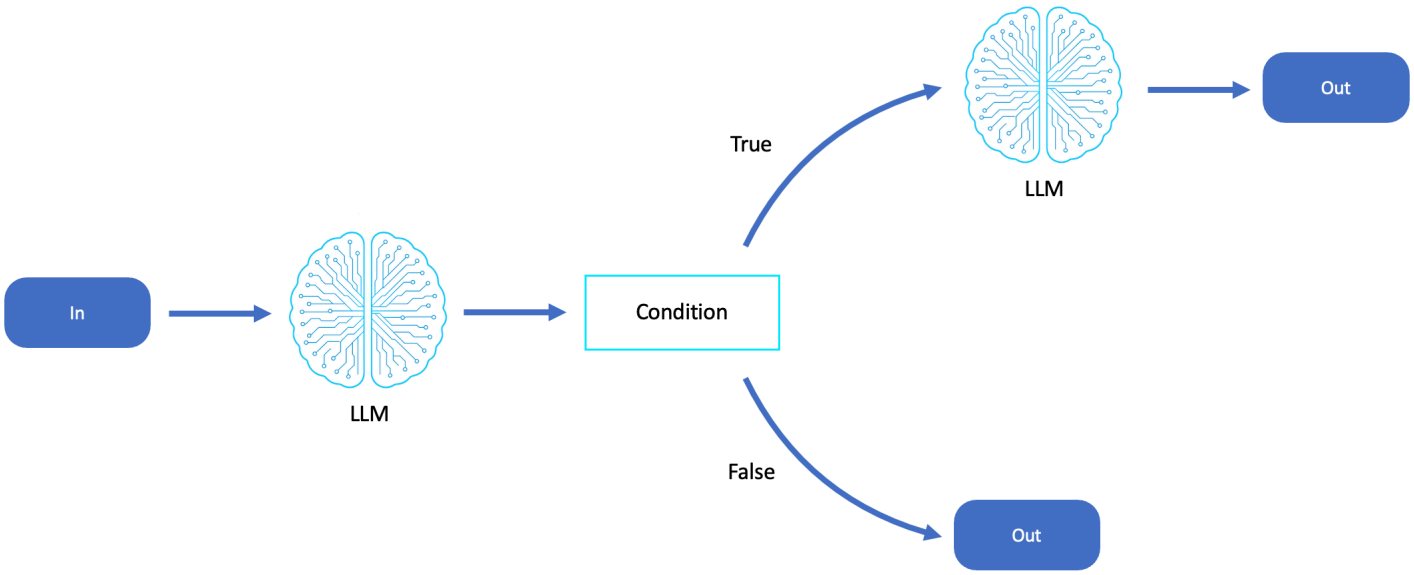
核心见解

每个 LLM 工作流程都可以映射到代理工作流程模式，该模式反映并演变了传统的事件驱动架构风格。代理工作流程的一个基本组成部分是能够使用数据、工具和内存来增强法学硕士的上下文。这会创建一个明智、自适应且符合用户意图的推理循环。传统系统通过查找数据来丰富消息，而代理系统使软件的行为与其说是脚本，不如说是智能协作者。

提示链接传奇模式

通过将 LLM 提示链重新构想为一个事件驱动的传奇，我们解锁了一种新的运营模式：工作流程变得分散、可恢复，并且可以跨自治代理进行语义协调。每个提示响应步骤都被重新定义为原子任务，作为事件发出，由专用代理使用，并富含上下文元数据。

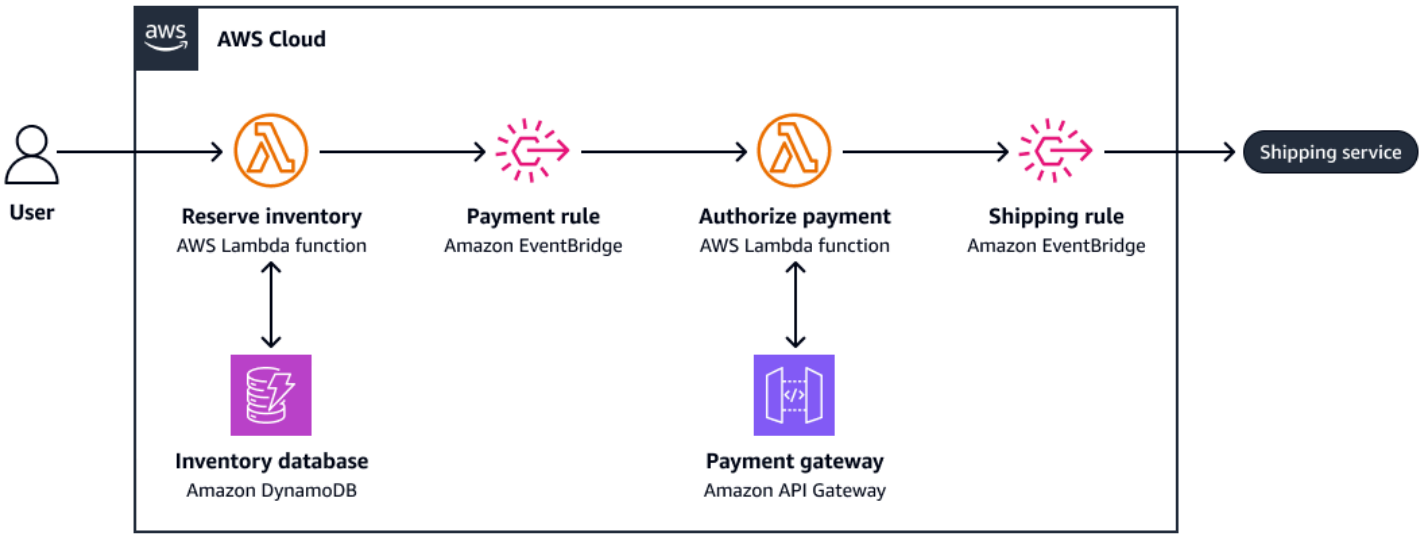
下图是 LLM 提示符链接的示例：



saga 编配

传奇编舞模式是一种在没有中央协调器的分布式系统中的实现方法。相反，每个服务或组件都会发布触发下一个工作流程操作的事件。这种模式广泛用于分布式系统中，用于管理跨多个服务的交易。在传奇中，该系统运行了一系列协调的本地交易。如果一个失败，系统会触发补偿操作以保持一致性。

下图是传奇编舞的示例：



1. 预留库存
2. 授权付款
3. 创建配送订单

如果步骤 3 失败，系统将调用补偿操作（例如，取消付款或发放库存）。

这种模式在事件驱动架构中特别有价值，在这种架构中，服务是松散耦合的，即使存在部分故障，也必须随着时间的推移一致地解析状态。

提示链接模式

提示链接在结构和目的上都类似于传奇模式。它执行一系列推理步骤，这些步骤按顺序构建，同时保留上下文并允许回滚和修订。

特工编舞

1. LLM 解释复杂的用户查询并生成假设
2. 法学硕士详细制定了解决任务的计划
3. LLM 执行子任务（例如，通过使用工具调用或检索知识）
4. 如果LLM认为结果不令人满意，则会完善输出或重新审视之前的步骤

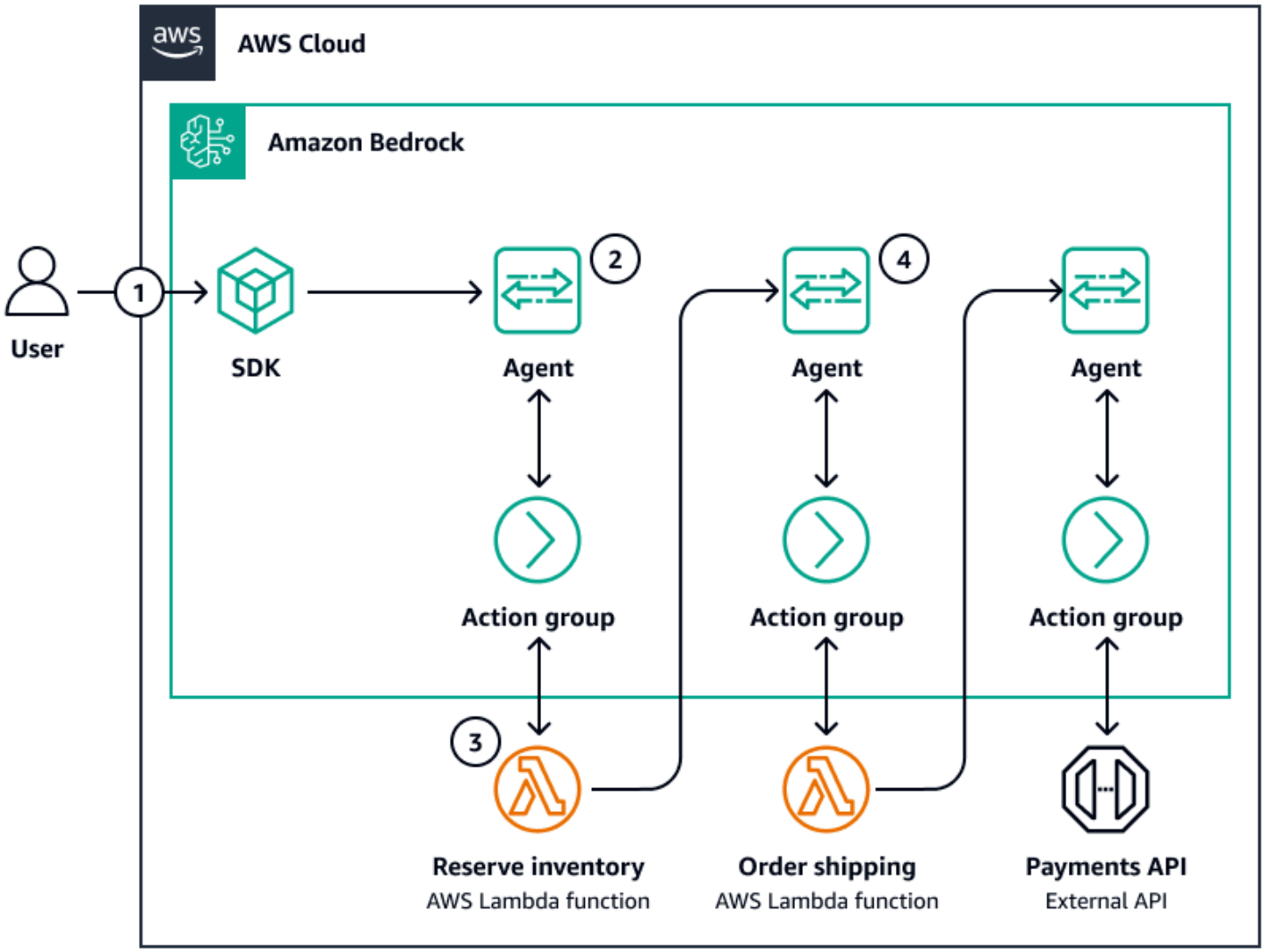
如果中间结果存在缺陷，则系统可以执行以下操作之一：

- 使用其他方法重试这些步骤

- 恢复到之前的提示并重新计划
- 使用赋值器循环（例如，来自赋值器-优化器模式）来检测和纠正故障

与传奇模式一样，提示链接允许部分进度和回滚机制。这是通过迭代优化和 LLM 指导的校正来实现的，而不是通过补偿数据库事务来实现的。

下图是特工编舞的示例：



1. 用户通过 SDK 提交查询。
2. Amazon Bedrock 代理通过以下方式精心策划推理：
   - 口译（法学硕士）
   - 规划（法学硕士）
   - 通过工具或知识库执行

- 响应构建

3. 如果工具出现故障或返回的数据不足，代理可以动态地重新计划或重新表述任务。

4. 内存（例如，短期向量存储）可以跨步骤保持其状态

外卖

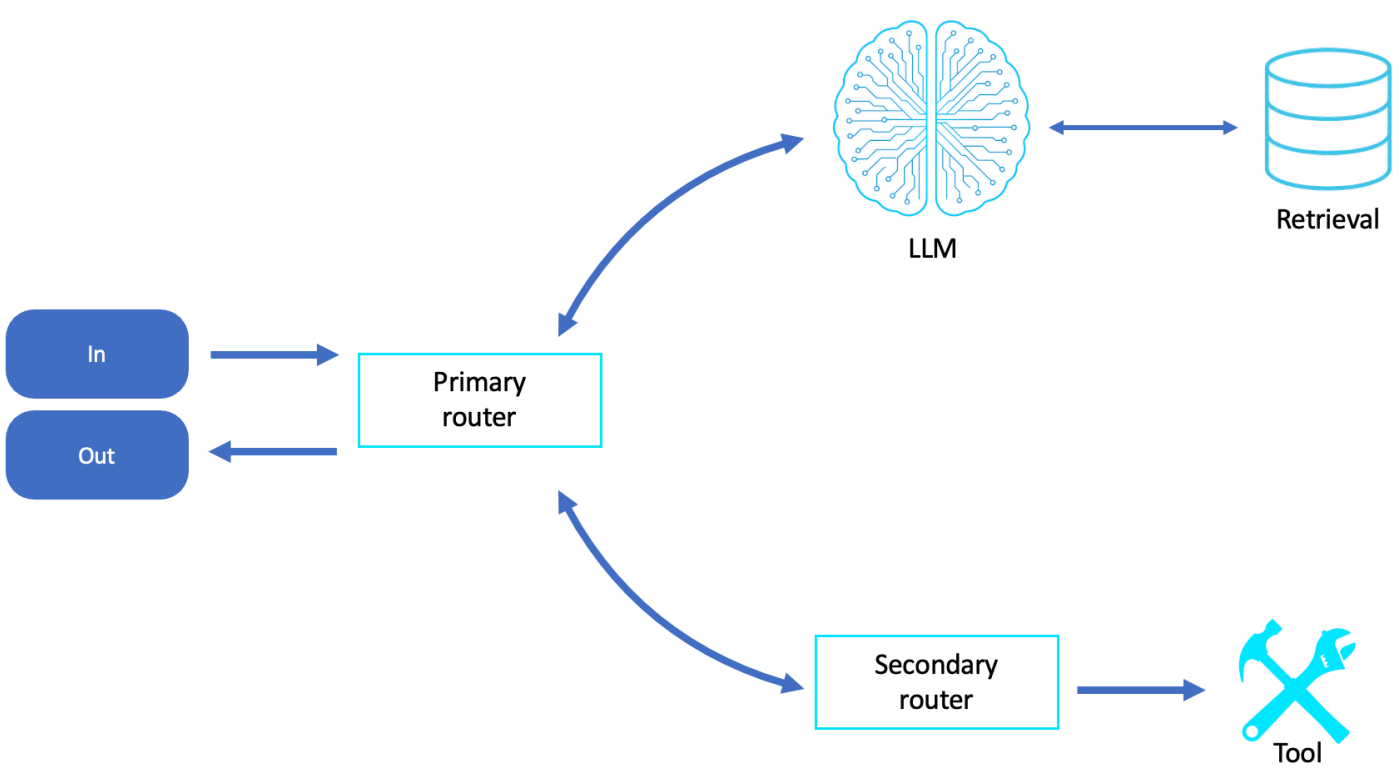
传奇模式使用补偿逻辑管理分布式服务调用，而提示链则通过反射式排序和自适应重新规划来管理推理任务。这两个系统都允许渐进式进展、分散的决策点和故障恢复，所有这些都是通过明智的推理而不是僵硬的回滚来实现的。

提示链接引入了交易推理，这在认知上等同于传奇。也就是说，作为更广泛的目标导向对话的一部分，每个 “想法” 都会被重新评估、修改或放弃。

路由动态调度模式

在现代代理系统中，任务范围从文档解析到自主软件生成，将请求动态路由到功能最强的大型语言模型 (LLM) 或代理的能力变得至关重要。静态路由逻辑通常嵌入在编排脚本或 API 层中，缺乏实时、多模型、多功能环境所需的适应性。为了解决这个问题，可以将 LLM 路由工作流转换为事件驱动的架构，该架构利用动态调度模式，将 LLM 调用转换为智能路由、上下文感知的事件。

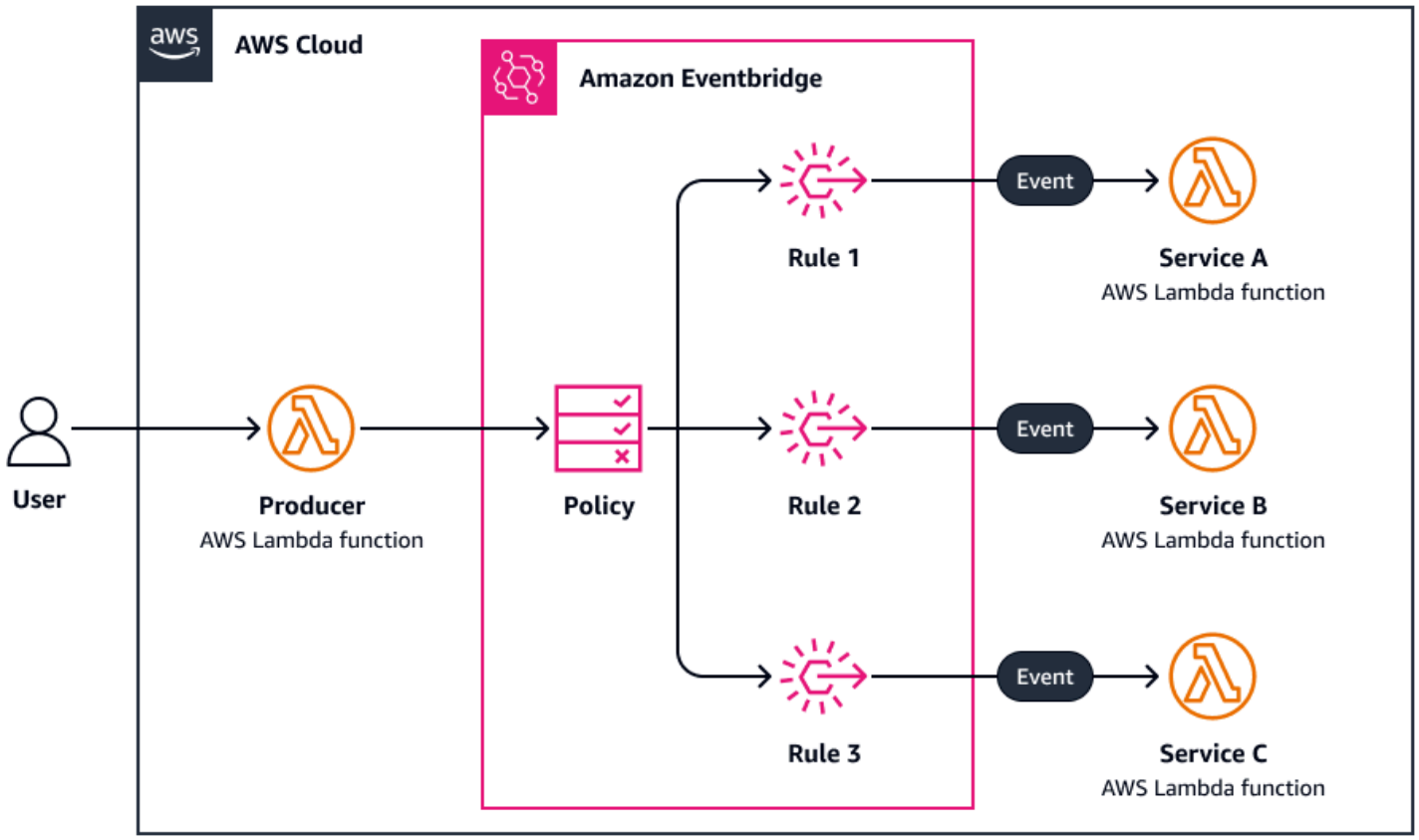
下图是 LLM 路由的示例：



动态调度

在传统的分布式系统中，动态调度模式根据传入的事件属性（例如事件类型、源和负载）在运行时选择和调用特定的服务。这通常使用 Amazon 实现 EventBridge，亚马逊可以评估传入的事件并将其路由到相应的目标（例如 AWS Step Functions， AWS Lambda 函数或亚马逊弹性容器服务任务）。

下图是动态调度的示例：



1. 应用程序会发出一个事件（例如，{"type": “orderCreated”，“优先级”：“高”}）。
2. Amazon EventBridge 会根据其路由规则评估事件。
3. 根据事件的属性，系统会动态调度到以下内容：
   - `HighPriorityOrderProcessor`（服务 A）
   - `StandardOrderProcessor`（服务 B）
   - `UpdateOrderProcessor`（服务 C）

此模式支持松散耦合、基于域的专业化和运行时可扩展性。这使系统能够智能地响应不断变化的需求和事件语义。

基于 LLM 的路由

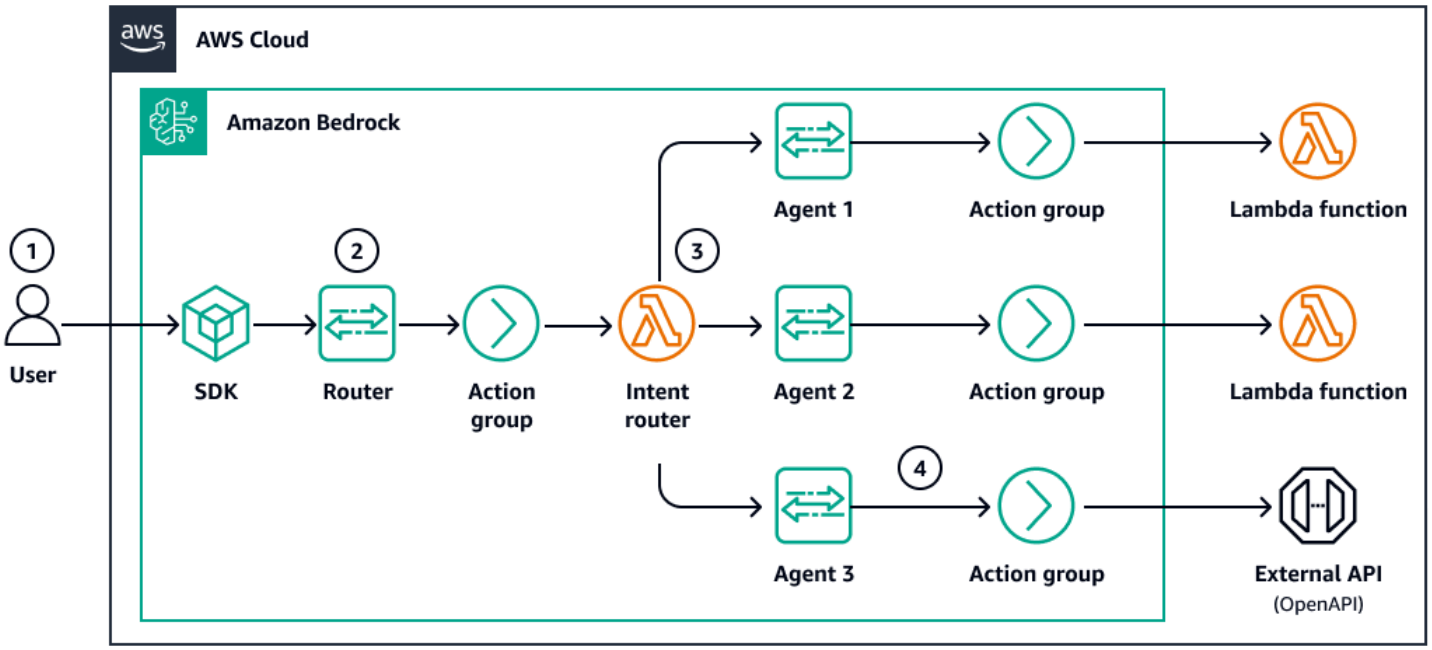
在代理系统中，路由还会执行动态任务委派，但是 LLM 不是亚马逊 EventBridge 规则或元数据筛选器，而是通过自然语言对用户的意图进行分类和解释。结果是一种灵活、语义化和自适应的调度形式。

代理路由器

这种架构支持基于意图的丰富调度，无需预定义架构或事件类型，非常适合非结构化输入和复杂查询。

1. 用户提交了请求“你能帮我查看合同条款吗？”
2. 法学硕士将其解释为一项法律文件任务。
3. 代理将任务路由到以下一项或多项：
   - 合同审查提示模板
   - 法律推理子代理
   - 文档解析工具

下图是代理路由器的示例：



1. 用户通过 SDK 提交自然语言请求。
2. Amazon Bedrock 代理使用 LLM 对任务进行分类（例如，法律、技术或日程安排）。
3. 代理通过操作组动态路由任务以调用所需的代理：
   - 特定域名代理
   - 专业工具链
   - 自定义提示符配置
4. 选定的处理者处理任务并返回定制的响应。

外卖

传统动态调度使用基于结构化事件属性的Amazon EventBridge 规则进行路由，而代理路由则根据含义和意图对任务进行语义分类和路由。 LLMs 这通过启用以下功能扩展了系统的灵活性：

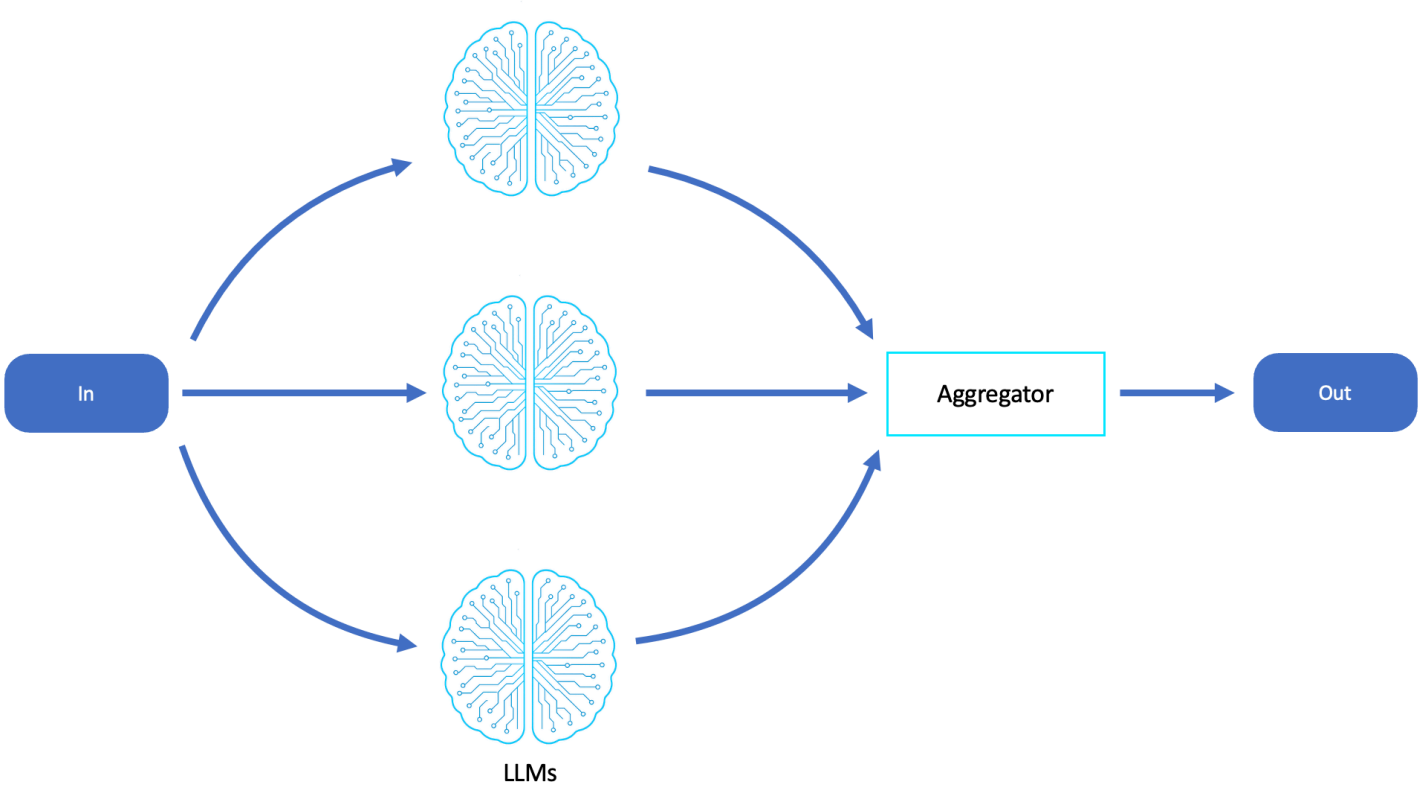
- 更广泛的输入理解
- 智能备用和工具选择
- 通过新的代理角色或提示样式实现自然的可扩展性

代理路由用动态认知调度取代了严格的规则，这允许系统随着语言而不是代码的变化而发展。

并行化和分散-聚集模式

许多高级推理和生成任务（例如总结大型文档、评估多个解决方案路径或比较不同的视角）都受益于提示的并行执行。当需要可扩展性、响应能力和容错能力时，传统的顺序工作流程是不够的。为了克服这个问题，可以使用事件驱动的分散聚集模式重新构想基于 LLM 的并行化，在这种模式下，任务被动态分散给自主代理，然后智能地合成结果。

下图是 LLM 并行化工作流程的示例：



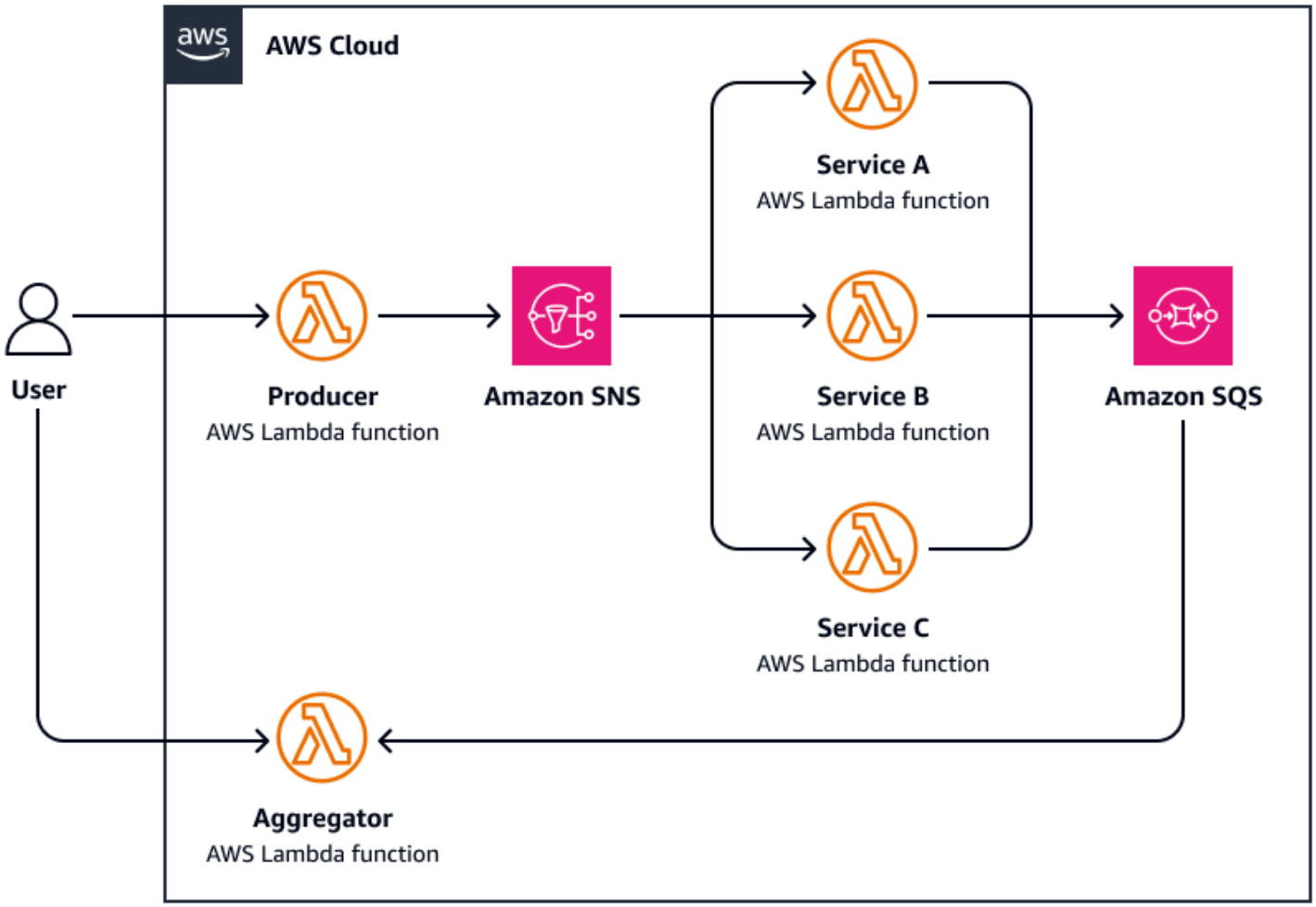
分散收集

在分布式系统中，分散聚集模式将任务并行发送到多个服务或处理单元，等待它们的响应，然后将结果聚合到合并的输出中。与扇出不同，scatter-gather 是协调的，因为它需要响应，并且通常会应用逻辑来合并、比较和选择结果。

并行化和分散收集的常见实现包括以下内容：

- AWS Step Functions 映射并行任务执行的状态
- AWS Lambda 使用并发性，协调来自多个调用函数的结果
- EventBridge 具有关联 IDs 和聚合工作流程的 Amazon
- 使用亚马逊简单存储服务 (Amazon S3)、亚马逊 DynamoDB 或队列管理扇出和收集结果的自定义控制器模式

下图是分散聚集的示例：



1. 用户向中央协调器功能发送请求，该功能通过向亚马逊简单通知服务 (Amazon SNS) Simple Notification Service 主题发布并行消息来分散任务。
2. 每条消息都包含任务元数据，并发送给专业工作人员 AWS Lambda。
3. 每个工作人员都 AWS Lambda 独立处理其分配的子任务（例如，查询外部 API、处理文档和分析数据）。
4. 结果会写入公共存储层，例如亚马逊简单队列服务 (Amazon SQS) Simple Queue Service。
5. 聚合器函数等待所有响应完成，然后执行以下操作：
   - 收集和汇总结果（例如，合并摘要、选择最佳匹配项）
   - 发送最终响应或触发下游工作流程

分散聚集模式的常见用例包括以下几种：

- 联合搜索
- 价格比较引擎
- 汇总数据分析
- 多模型推理

基于 LLM 的并行化（分散-聚集认知）

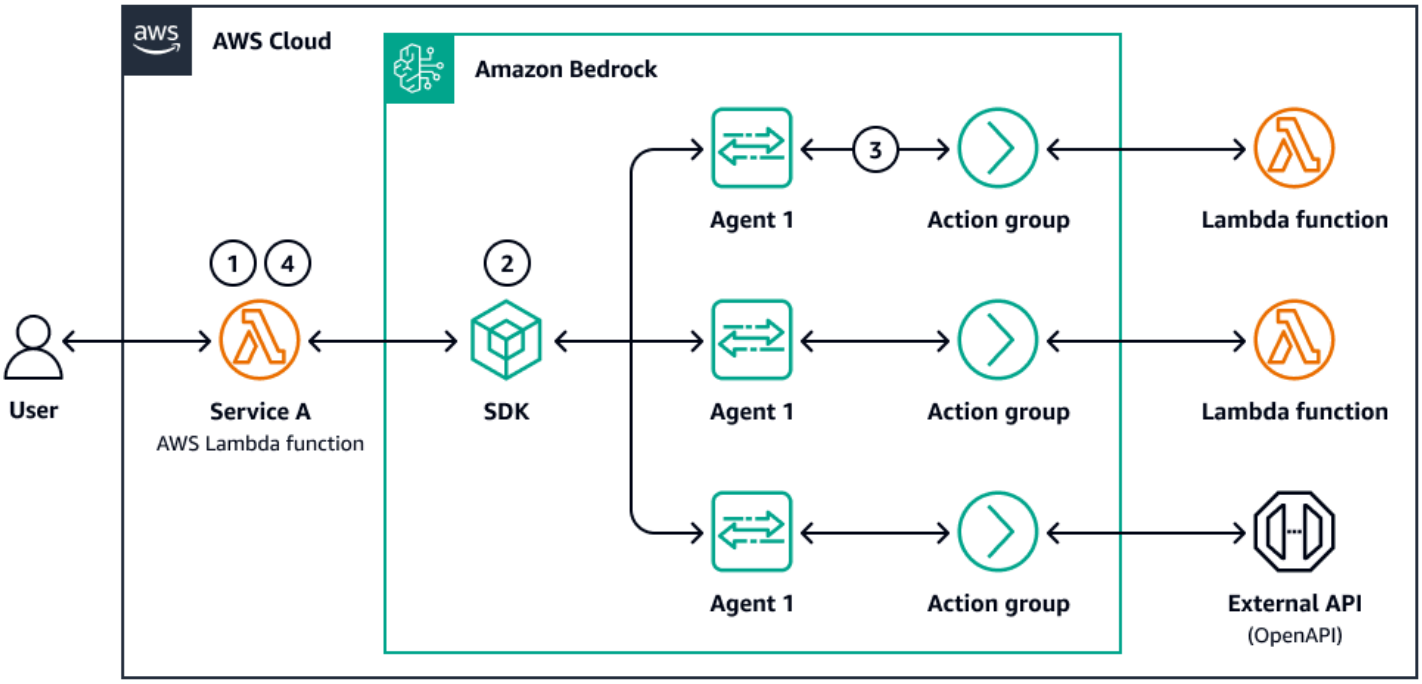
在代理系统中，并行化通过在多个 LLM 调用或代理之间分配子任务来密切反映分散聚集，每个调用或代理独立推理问题的一部分。返回的结果由聚合过程收集和合成，聚合过程通常是另一个 LLM 或控制器代理。

代理并行化

1. 代理提交了 “汇总这 10 份报告的见解” 请求。
2. 它将报告分散到 10 个并行的 LLM 摘要任务中。
3. 当它返回所有摘要时，代理会执行以下操作：
   - 将摘要汇总成统一的简报
   - 识别主题或矛盾
   - 将合成后的输出发送给用户

这种代理工作流程支持可扩展、模块化和自适应的并行推理。这非常适合需要高认知吞吐量的用例。

下图是代理并行化的示例：



1. 用户提交分段查询或文档集。
2. 控制器 AWS Lambda 或步进函数分配子任务。每个任务都会使用自己的提示调用 Amazon Bedrock LLM 调用或子代理。
3. 调用和子任务完成后，结果将被存储（例如，在 Amazon S3 或内存存储中），然后聚合步骤会合并、比较或筛选输出。
4. 系统将最终响应返回给用户或下游代理。

该系统具有分布式推理循环，具有可追溯性、容错能力以及可选的结果加权或选择逻辑。

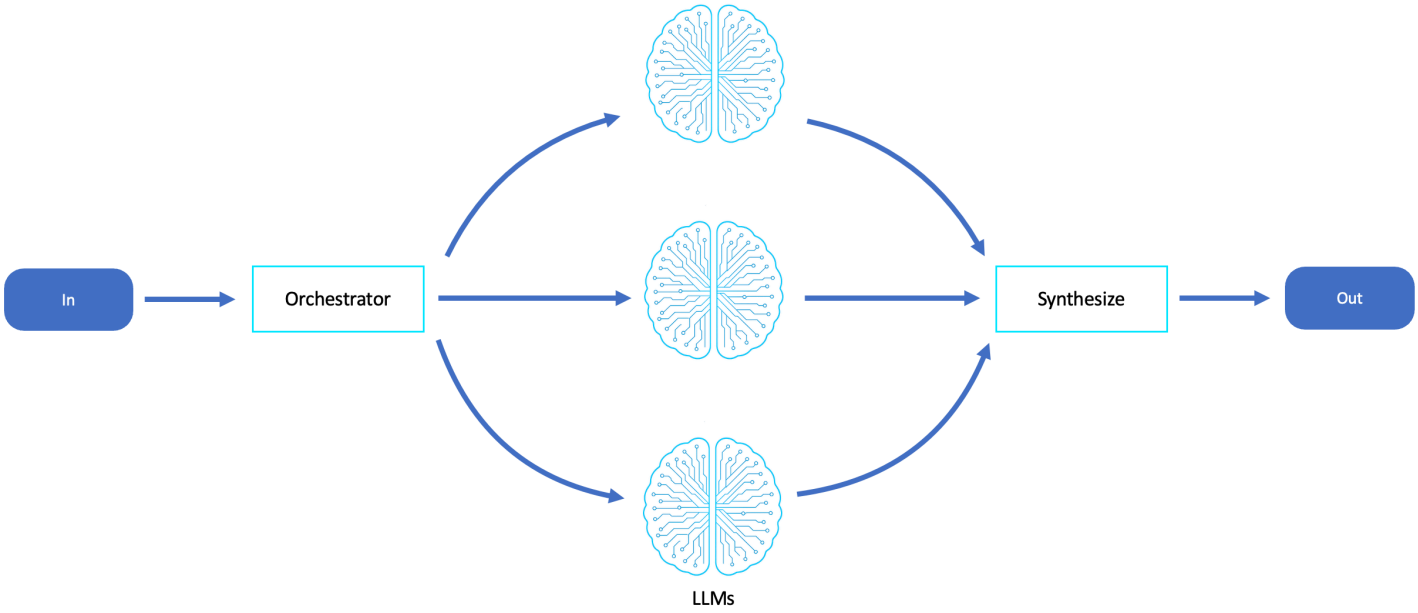
外卖

代理并行化使用分散聚集模式来分发 LLM 任务，从而实现并行处理和智能结果合成。

传奇编排模式

随着由驱动的工作流程 LLMs 变得越来越复杂，跨越提示链、数据处理步骤、工具调用和代理协作，对智能编排的需求变得至关重要。这些工作流程可以作为事件驱动的编排模式来实现，而不是依赖紧密耦合的脚本或静态的预先确定的执行流程，从而使基于 LLM 的系统能够动态地协调、监控和调整自主代理之间的多步骤任务。

下图是协调器的示例：



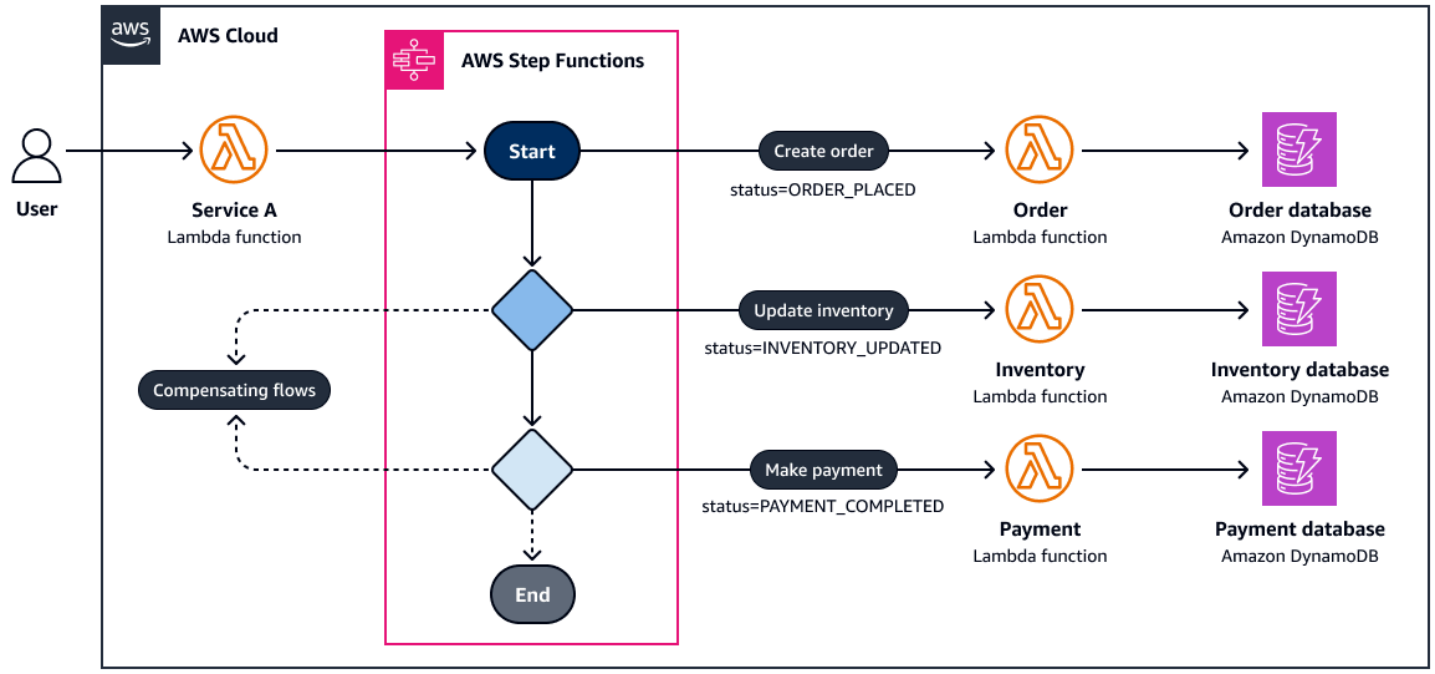
活动编排

在传统的分布式系统中，事件编排是指一种模式，在这种模式中，中央协调员通过明确指挥多个服务或任务的控制流来管理复杂的工作流程。与事件编排（每项服务独立做出反应）不同，编排提供了对整个流程的集中逻辑、可见性和控制。

这通常使用以下工具来实现：

- AWS Step Functions— 定义和执行有状态的工作流程
- AWS Lambda— 在精心策划的流程中执行离散的任务
- 亚马逊 SQS 或亚马逊 EventBridge — 触发异步步骤或响应

下图是传奇编排的示例：



AWS Step Functions 工作流管理客户订单流程：

1. 创建订单 (AWS Lambda)
2. 更新库存 (AWS Lambda)
3. 付款 (AWS Lambda)

协调器通过管理重试、并行分支、超时和失败来协调每个步骤。

基于角色的代理系统（协调器）

在代理系统中，协调器模式反映了事件编排，但将逻辑分配给多个推理代理，每个推理代理都有明确的角色或专长。中央协调器代理解释整体任务，将其分解为子任务，然后将这些任务委托给工作代理，每个代理都针对特定领域（例如研究、编码、摘要、审查）进行了优化。

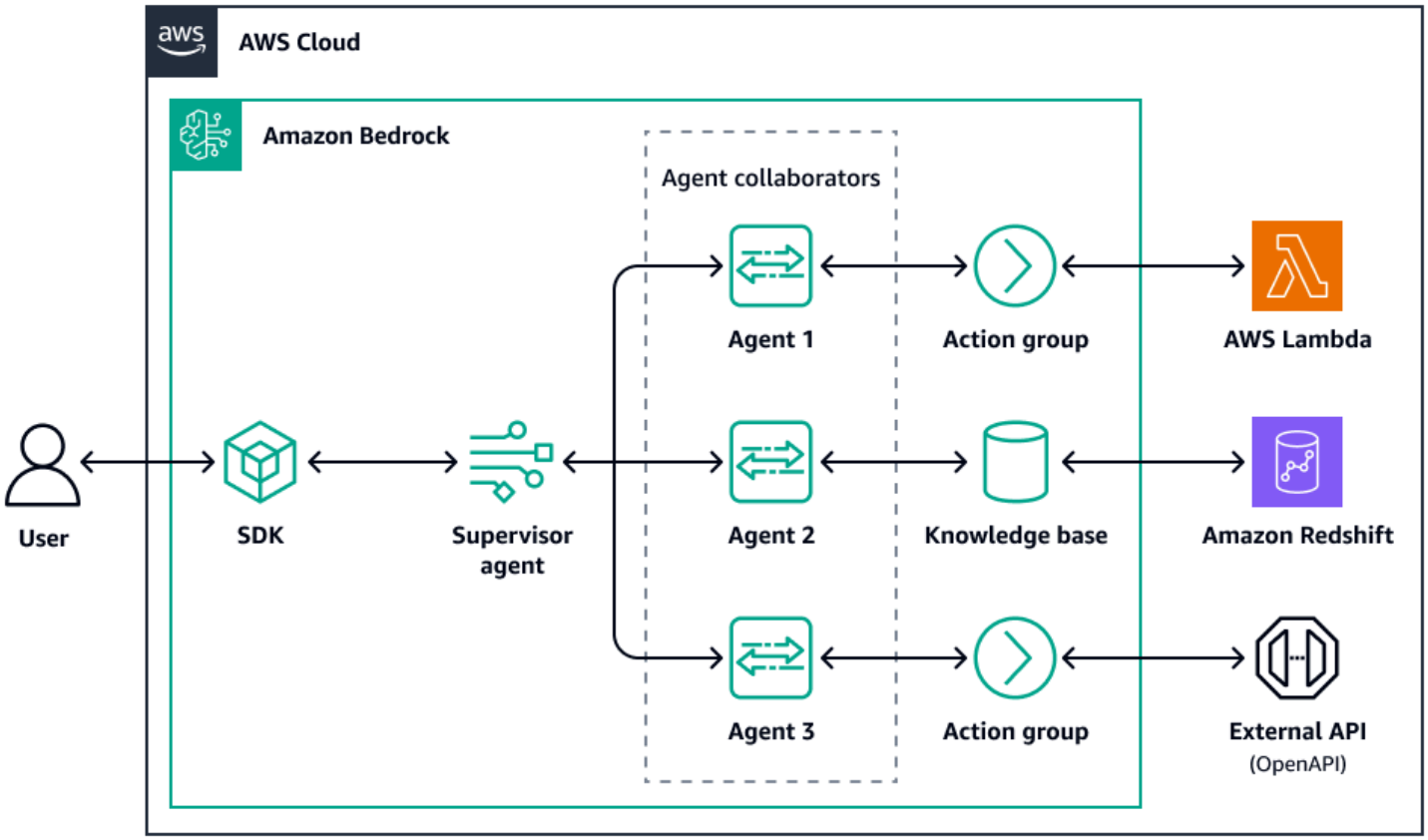
主管

1. 用户提交了 “创建项目简介并总结排名前五的竞争对手” 的查询。
2. 协调器代理执行以下操作：
   - 指派研究代理人寻找竞争对手的数据
   - 将原始发现结果发送给汇总代理
   - 将结果传递给简报撰稿人代理

- 为用户编译最终输出

每个代理独立运行，但协调器负责协调任务。这就像处理工作流程任务的 Lambda 函数。

下图是主管的示例：



1. 用户向 Amazon Bedrock 主管代理提交任务。
2. 主管代理将请求解析为每个代理协作者的子任务。
3. 每个子任务都分配给带有特定角色提示或工具链的协作者代理。
4. 工作人员代理通过操作组呼叫外部人员 APIs 或工具。
5. 每个工作代理都以结构化格式返回输出。
6. 当所有工作人员返回结果时，主管会评估、综合并返回最终响应。

这种结构允许在复杂的多步骤代理工作流程中实现模块化、适应性和内省。

外卖

事件编排使用集中控制（例如 AWS Step Functions）来指导服务执行，而基于角色的代理系统则使用由 LLM 提供支持的 Orchestrator 代理来推理目标，将子任务委托给工作代理并合成最终输出。

在这两种范式中，协调器都会执行以下操作：

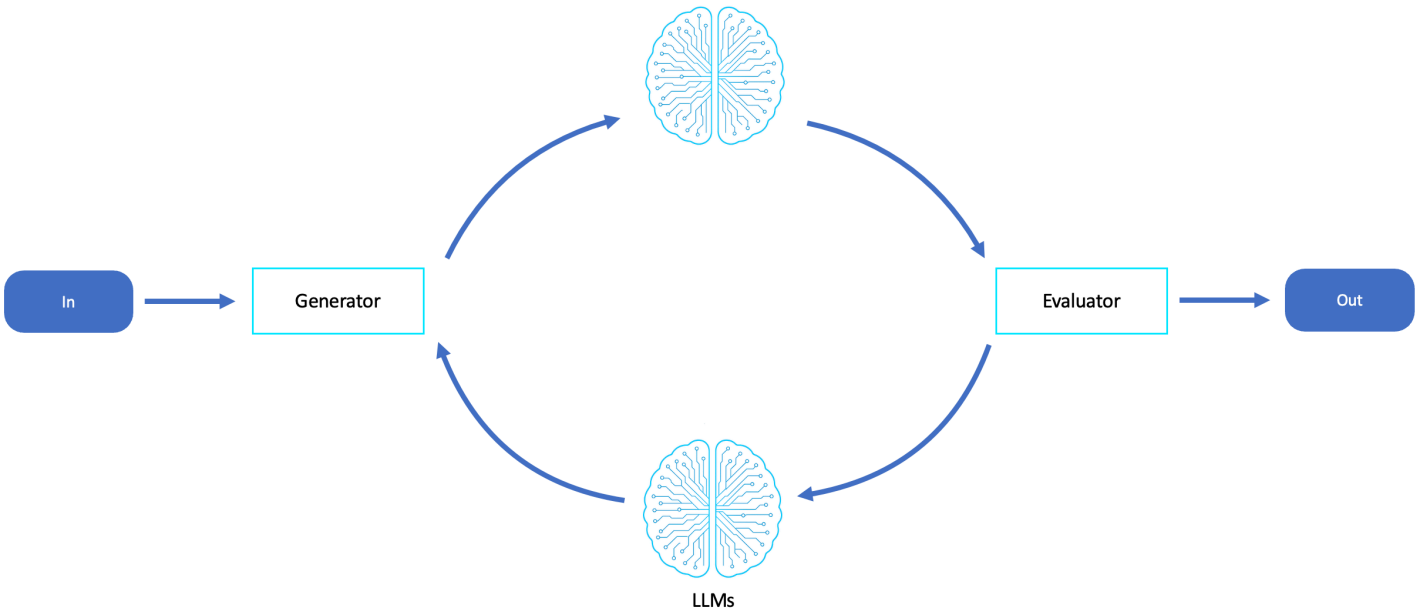
- 维护上下文和执行流程
- 处理分支、排序和错误处理
- 通过分布式组件生成统一的结果

但是，代理编排增加了推理、适应性和语义委托。这使得它非常适合开放式、模棱两可和不断演变的任务。

赋值器反射-优化循环模式

诸如代码生成、摘要或自主决策之类的任务可以从运行时反馈中受益匪浅，从而使系统能够通过观察和完善来发展。为了实现这一点，可以将反射-精炼周期作为事件驱动的反馈控制回路来实现，这种模式的灵感来自系统工程，适用于自主、智能的工作流程。

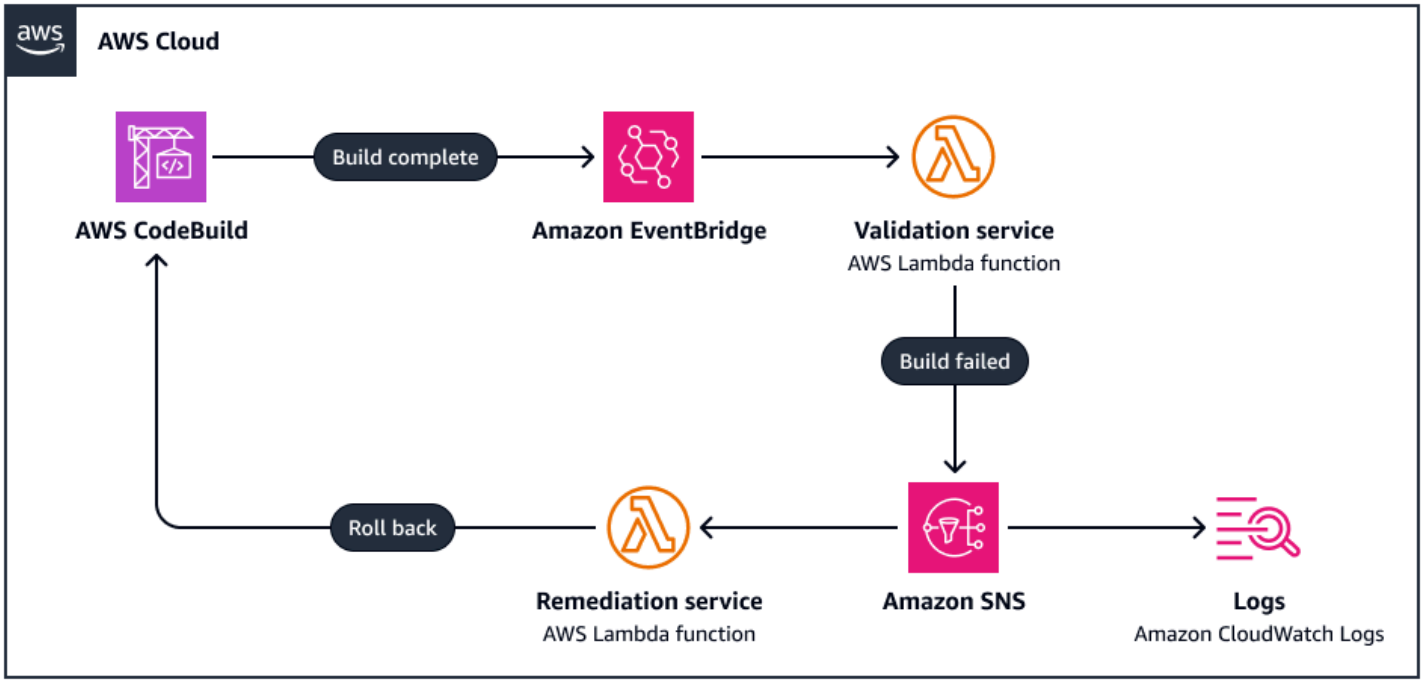
下图是评估器反射精简反馈回路的示例：



反馈控制回路

反馈控制回路是一种模式，它监视自己的输出和行为，根据定义的标准或所需状态对其进行评估，然后相应地调整其动作。该架构的灵感来自控制理论，是自动化、持续集成和持续交付 (CI/CD) 管道以及机器学习操作等领域的基础。

下图是反馈控制回路的示例：



1. 部署管道会发出 buildComplete 事件。
2. 该事件会触发自动测试或评估作业，以验证构建。
3. 如果验证失败（例如，由于测试失败、安全问题或违反策略），则系统：
   - 发出 BuildComplete 事件
   - 记录问题或发送通知
   - 触发补救或更正操作，例如回滚、修补或重试

循环一直持续到产生可接受的结果或升级，或者出现超时。此模式通常用于以下用途：

- Amazon 将事件路由到评估或补救任务的 EventBridge 规则
- AWS Step Functions 用于迭代重试逻辑和对评估结果进行分支
- 亚马逊简单通知服务 (Amazon SNS) Simple Notification Service 或 A CloudWatch mazon 警报，用于发送反馈触发器和警报

- AWS Lambda 职能或容器化工作人员采取纠正措施

反馈控制回路（评估器）

评估者工作流程是由 LLMs 我们的推理代理提供支持的认知反馈循环。该过程包括以下内容：

1. 生成器代理或 LLM 生成输出（例如，计划、答案或草稿）。
2. 评估人员使用批评提示或评估标准来审查结果。
3. 根据反馈，原始代理或新的优化器代理修改输出。

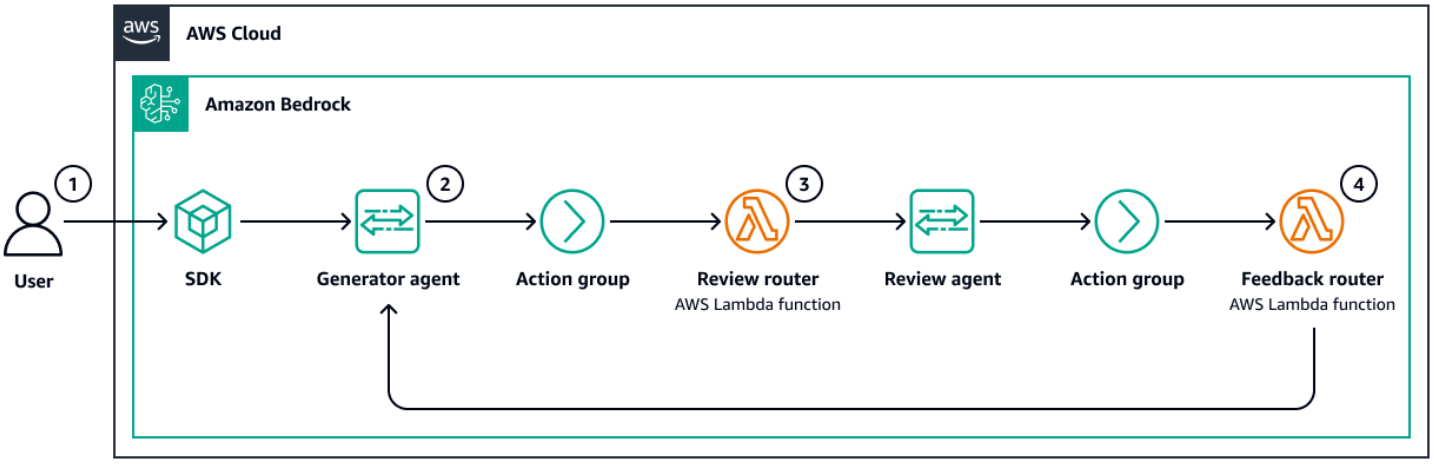
循环会重复，直到结果符合一组标准、获得批准或达到重试限制。

评估者

1. 用户要求代理撰写策略摘要。
2. 生成器代理起草它。
3. 评估人员检查覆盖范围、语气和法律正确性。
4. 如果响应不充分，则会对其进行完善并重新提交，直到反馈回路趋于一致。

这可以实现自我评估、迭代完善和自适应输出控制，所有这些都无需人工输入。

下图是反馈控制回路（赋值器）的示例：



1. 用户下达任务（例如，起草业务策略）。
2. Amazon Bedrock 代理使用 LLM 生成初始草稿。

3. 第二个代理（或后续提示）执行结构化评估（例如，“根据清晰度、完整性和语气对输出进行评分”）。
4. 如果评级低于阈值，则通过以下方式对响应进行修改：
   - 使用嵌入式评论重新调用生成器
   - 将反馈发送给专业的炼油厂代理商
   - 迭代直到得到可接受的响应

可选组件，例如 AWS Lambda 控制器或 AWS Step Functions 可以管理反馈阈值、重试和回退策略。

外卖

传统的反馈控制回路使用事件、指标和补救逻辑来验证和调整系统行为，而代理评估器循环则使用推理代理来动态评估、反映和修改输出。

在这两种范式中：

- 输出在生成后进行评估
- 根据反馈触发纠正或完善措施
- 系统不断适应目标质量或目标

代理版本将静态验证转换为语义反射，从而使自我完善的代理能够评估自己的有效性。

在上设计代理工作流程 AWS

本指南中的每种模式都可以使用构建 AWS 服务。Amazon Bedrock 代理提供编排、数据访问和互动渠道。

组件	AWS 服务	目的
法学硕士推理	Amazon Bedrock	代理逻辑、规划、工具使用
工具执行	AWS Lambda、亚马逊 ECS、亚马逊 SageMaker	为代理提供外部工具
内存和 RAG	亚马逊 Bedrock 知识库，亚马逊 S3， OpenSearch	持久记忆和语义记忆

编排	AWS Step Functions	多步骤任务和代理协调
事件路由	亚马逊 EventBridge、亚马逊 SQS	解耦代理间消息
用户界面	亚马逊 API Gateway、 AWS AppSync、SDK	应用程序或系统的入口点
监控	亚马逊 CloudWatch、 AWS X-Ray、 AWS Distro for OpenTelemetry	可观察性和代理内省

结论

代理工作流程模式是事件驱动架构的下一个演变阶段，其中业务逻辑不是静态定义的，而是通过使用大型语言模型 (LLM)（增强认知能力）来动态推理。通过将传统的云原生基元与法学硕士工作流程和代理设计模式相结合，组织可以构建自适应、智能和模块化的系统，这些系统可以有针对性地响应并从经验中学习。

在这些模式中，Amazon Bedrock 是通往代理认知的门户，它允许基于 LLM 的代理访问事件工作流程，与工具和内存进行交互，并提供结构化、可追溯和一致的结果。

在您设计和部署代理系统时，这些工作流程模式为构建自主、可组合的 AI 架构提供了蓝图。这些系统以无服务器最佳实践为基础，并通过智能基础模型进行了增强。

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	—	2025 年 7 月 14 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：

- 重构/重新架构：充分利用云原生功能来提高敏捷性、性能和可扩展性，以迁移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：将本地 Oracle 数据库迁移到 Amazon Aurora PostgreSQL 兼容版。
- 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中，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

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。

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？

人工智能操作 (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 AIOps AWS 迁移策略中使用的更多信息，请参阅操作集成指南。

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。

蓝/绿部署

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 AWS Well-Architected Guidance 中的 Implement break-glass procedures 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。

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，创建着陆区、定义 CCo E、建立运营模型）
- 迁移 - 迁移单个应用程序
- 重塑 - 优化产品和服务，在云中创新

Stephen Orban在 AWS Cloud 企业战略博客的博客文章《云优先之旅和采用阶段》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 管线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，以自定义您的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 Well-Architecte AWS d Framework 中安全支柱的一个组成部分。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。

defense-in-depth

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，一种 defense-in-depth方法可以结合多因素身份验证、网络分段和加密。

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 Well-Architected Fr ame AWS work 中的 “工作负载灾难恢复：云端 AWS 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。Eric Evans 在其著作领域驱动设计：软件核心复杂性应对之道（Boston: Addison-Wesley Professional, 2003）中介绍了这一概念。有关如何将领域驱动设计与 strangler fig 模式结合使用的信息，请参阅使用容器和 Amazon API Gateway 逐步将原有的 Microsoft ASP.NET（ASMX）Web 服务现代化。

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。大端序系统先存储最高有效字节。小端序系统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，请参阅使用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。此技术是上下文内学习的一种应用，其中模型可以从提示中嵌入的示例（样本）中学习。对于需要特定格式、推理或领域知识的任务，少样本提示可能非常有效。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，一直在广义和未标记数据的大量数据集上进行训练。 FMs 能够执行各种各样的一般任务，例如理解语言、生成文本和图像以及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。

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

帮助管理各组织单位的资源、策略和合规性的高级规则 (OUs)。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。

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。

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 AWS Well-Architected Framework 中的使用不可变基础设施进行部署最佳实践。

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 安全参考架构建议设置您的网络帐户，包括入站、出站和检查， VPCs 以保护您的应用程序与更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 Klaus Schwab 在 2016 年提出，指的是通过连接、实时数据、自动化、分析和 AI/ML 的进步来实现制造流程的现代化。

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) 数字化转型战略。

检查 VPC

在 AWS 多账户架构中，一种集中式 VPC，用于管理对 VPCs（相同或不同 AWS 区域）、互联网和本地网络之间的网络流量的检查。AWS 安全参考架构建议设置您的网络帐户，包括入站、出站和检查， VPCs 以保护您的应用程序与更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，请参阅使用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 LLMs。

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。

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 Well-Architect AWS ed 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）

一种基于发布/订阅模式的轻量级 machine-to-machine (M2M) 通信协议，适用于资源受限的物联网设备。

微服务

一种小型的独立服务，通过明确的定义进行通信 APIs ，通常由小型的独立团队拥有。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通过定义明确的接口进行通信。 APIs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

跨职能团队，通过自动化、敏捷的方法简化工作负载迁移。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、开发人员和冲刺 DevOps 领域的专业人员。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，Well-Architect AWS ed Framework 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 machine-to-machine（M2M）通信协议。OPC-UA 提供了一个包含数据加密、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 Framework 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 安全参考架构建议设置您的网络帐户，包括入站、出站和检查， VPCs 以保护您的应用程序与更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

一个容器，其中包含有关您希望 Amazon Route 53 如何响应针对一个或多个 VPCs域名及其子域名的 DNS 查询的信息。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，相互独立，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，用于集中控制组织中所有账户的权限 AWS Organizations。 SCPs 定义防护措施或限制管理员可以委托给用户或角色的操作。您可以使用 SCPs 允许列表或拒绝列表来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。

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 模型

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 逐步将原有的 Microsoft ASP.NET（ASMX）Web 服务现代化。

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

键值对，用作组织资源的元数据。 AWS 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。

中转网关

一个网络传输中心，可用于将您的网络 VPCs 和本地网络互连。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。有关更多信息，请参阅量化深度学习系统中的不确定性指南。

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

两者之间的连接 VPCs ，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。