上的客服人員 AI 架構、平台、通訊協定和工具 AWS

AWS 方案指引

AWS 方案指引: 上的客服人員 AI 架構、平台、通訊協定和工具 AWS

Table of Contents

上的客服人員 AI 架構、平台、通訊協定和工具 AWS

Aaron Sempf、Ansley Verzosa 和 Joshua Samuel，Amazon Web Services (AWS)

2026 年 1 月 (文件歷史記錄)

代理式 AI 是 AI、分散式系統和軟體工程交集的強大範例。這是一個智慧型系統類別，由使用 AI 模型並與工具和資源整合的自動、非同步軟體代理程式組成。客服人員代表使用者或系統，展現代理、可以感知內容、對目標的推理、做出決策，以及採取有目的的動作。這些代理程式通常在分散式環境中獨立運作，旨在追求具有內嵌智慧、記憶體和意圖的委派目標。

在 上 AWS，組織可以利用代理式 AI 來自動化複雜的工作流程、增強決策程序，以及建立更回應的系統。本指南提供有關建置有效代理式 AI 解決方案所需的關鍵元件的資訊：

- 架構會描述目前的代理式 AI 架構，包括檢閱其優點和使用案例。了解這些架構如何減少跨模式、通訊協定和工具的繁重工作。了解金鑰選擇條件，為您的需求選擇正確的架構。
- 平台提供代理式 AI 平台（受管代理程式、開放原始碼協同運作和混合）的概觀，以及選擇或設計的考量事項。
- 通訊協定探索客服人員互動所需的基本標準化通訊協定。Agent-to-agent程式通訊協定正在出現，例如開放原始碼模型內容通訊協定 (MCP) 和 Agent2Agent (A2A)，以及其他專屬實作。探索常見的通訊協定如何讓不同的通訊協定無縫互動。
- 工具提供有關通訊協定型工具（例如 MCP)、架構原生工具和中繼工具的資訊。組織可以建置工具組，與其工作流程中的關鍵系統整合，同時啟用最終使用者和伺服器型代理程式工作流程。

目標對象

本指南適用於尋求在現代雲端原生應用程式中利用 AI 驅動軟體代理程式的架構師、開發人員和技術領導者。

目標

本指南可協助您執行以下操作：

- 比較不同的客服人員 AI 架構，為您的使用案例選擇最適合的架構。
- 了解客服人員 AI 平台，這些平台提供將個別客服人員轉換為協調、適應性系統的功能。
- 了解開放式通訊協定在建置永續性代理式 AI 架構方面的優勢。

- 在建置代理程式系統時，建立適當的工具整合策略。

關於此內容系列

本指南是代理式 AI 相關系列的一部分 AWS。如需詳細資訊和檢視此系列中的其他指南，請參閱 AWS 規範性指導網站上的 客服人員 AI 。

架構

上的代理式 AI 基礎 AWS會檢查啟用自動目標導向行為的核心模式和工作流程。實作這些模式的核心是架構的選擇。框架是預先編寫程式碼的軟體基礎，可提供結構化環境和常見功能，用於建置和管理建置生產就緒的自動 AI 代理器所需的工具和協同運作功能。

有效的代理式 AI 架構提供數種基本功能，可將原始大型語言模型 (LLM) 互動轉換為協調的智慧型系統，能夠推理、協作和採取行動：

- 客服人員協調協調跨單一或多個客服人員的資訊流程和決策，無需人工介入即可實現複雜的目標。
- 工具整合可讓客服人員與外部系統、APIs和資料來源互動，將功能延伸到語言處理之外。如需詳細資訊，請參閱 Strands Agents 文件中的工具概觀。
- 記憶體管理提供持久性或工作階段型狀態，以維護互動之間的內容，對於長時間執行或適應性任務至關重要。更進階的架構整合了長期記憶體來儲存摘要和使用者偏好設定，從而實現個人化和上下文感知的客服人員體驗。如需詳細資訊，請參閱 LangChain 部落格上的如何考慮客服人員架構。
- 工作流程定義支援結構化模式，例如鏈、路由、平行化和反射迴圈，可實現複雜的自主推理。
- 部署和監控透過自動化系統的可觀測性，促進從開發到生產的轉換。如需詳細資訊，請參閱 Amazon Bedrock AgentCore 一般可用性公告。

這些功能會在架構環境中以不同的方法和階段實作，每個功能都為不同的自動代理程式使用案例和組織內容提供不同的優勢。

本節將介紹並比較建置代理式 AI 解決方案的領導架構，並著重於其優勢、限制和自動化操作的理想使用案例：

- Strands 代理程式
- LangChain 和 LangGraph
- CrewAI
- AutoGen
- ???
- 比較代理式 AI 架構

> **Note**
>
> 本節涵蓋專門支援 AI 代理程式的架構，且不涵蓋沒有 代理程式的前端界面或生成式 AI。

Strands Agents

Strands Agents 是一種開放原始碼 SDK，最初由 發行 AWS，如AWS 開放原始碼部落格中所述。Strands Agents 旨在使用模型優先的方法建置自主 AI 代理器。它提供靈活、可擴展的架構，旨在無縫搭配 使用， AWS 服務 同時保持開放與第三方元件整合。Strands Agents 非常適合建置完全自主的解決方案。

的主要功能 Strands Agents

Strands Agents 包含下列主要功能：

- 模型優先設計 – 以基礎模型是代理程式智慧的核心概念為基礎，實現複雜的自主推理。如需詳細資訊，請參閱 Strands Agents 文件中的客服人員迴圈。
- 多客服人員協作模式 – 內建協調模式，例如 Swarm、Graph 和工作流程模式，可跨分散式客服人員網路進行可擴展的協作和管理。如需詳細資訊，請參閱 Strands Agents 文件中的多重代理程式模式。
- MCP 整合 – 原生支援模型內容通訊協定 (MCP)，為 LLMs提供標準化內容佈建，以實現一致的自動操作。
- AWS 服務 整合 – 與 Amazon Bedrock AWS Lambda、 和其他 AWS 服務 無縫連線 AWS Step Functions，以實現全面的自主工作流程。如需詳細資訊，請參閱AWS 每週總和 (AWS 部落格 ）。
- 基礎模型選擇 – 支援 Amazon Bedrock 上的各種基礎模型，包括 Anthropic Claude、Amazon Nova (Premier、Pro、Lite 和 Micro) 等，以針對不同的自動推理功能進行最佳化。如需詳細資訊，請參閱 Strands Agents 文件中的 Amazon Bedrock。
- LLM API 整合 – 與不同 LLM 服務介面的彈性整合，包括 Amazon Bedrock、OpenAI 和其他用於生產部署的介面。如需詳細資訊，請參閱 Strands Agents 文件中的 Amazon Bedrock 基本用量。
- 多模態功能 – 支援多種模態，包括文字、語音和影像處理，以進行全面的自動代理程式互動。如需詳細資訊，請參閱 Strands Agents 文件中的 Amazon Bedrock 多模式支援。
- 工具生態系統 – 豐富的 AWS 服務 互動工具集，具有擴展自動化功能的自訂工具的可擴展性。如需詳細資訊，請參閱 Strands Agents 文件中的工具概觀。

使用時機 Strands Agents

Strands Agents 特別適合自動代理程式案例，包括：

- 以希望原生與 整合 AWS 服務 以進行自動化工作流程的 AWS 基礎設施為基礎的組織
- 需要生產自動化系統企業級安全性、可擴展性和合規功能的團隊
- 需要跨不同供應商靈活選擇模型的專案，以進行專門的自動化任務
- 需要與現有 AWS 工作流程和資源緊密整合以進行端對端自動化程序的使用案例

的實作方法 Strands Agents

Strands Agents 為業務利益相關者提供直接的實作方法，如其快速入門指南所述。架構可讓組織：

- 根據特定業務需求，在 Amazon Bedrock 上選取基礎模型，例如 Amazon Nova (Premier、Pro、Lite 或 Micro)。
- 定義連接到企業系統和資料來源的自訂工具。
- 處理多個模態，包括文字、影像和語音。
- 部署可以自動回應業務查詢並執行任務的代理程式。

這種實作方法可讓業務團隊快速開發和部署自動代理程式，而無需 AI 模型開發的深厚技術專業知識。

的實際範例 Strands Agents

AWS Transform for .NET 使用 Strands Agents 為其應用程式現代化功能提供支援，如 AWS Transform for .NET 中所述，這是第一個大規模現代化 .NET 應用程式的代理式 AI 服務 (AWS 部落格）。此生產服務採用多個專門的自主代理程式。代理程式會共同分析舊版 .NET 應用程式、規劃現代化策略，以及執行程式碼轉換至雲端原生架構，無需人工介入。 AWS Transform for .NET 示範 Strands Agents企業自動化系統的生產準備程度。

LangChain 和 LangGraph

LangChain 是代理式 AI 生態系統中最成熟的架構之一。 LangGraph擴展其功能以支援複雜且具狀態的代理程式工作流程，如 LangChain 部落格中所述。它們共同提供全方位解決方案，以建置複雜的自主 AI 代理器，並具有豐富的協同運作功能，用於獨立操作。

LangChain 和 的主要功能 LangGraph

LangChain 和 LangGraph包含下列主要功能：

- 元件生態系統 – 適用於各種自動代理程式功能的大量預先建置元件程式庫，可快速開發專用代理程式。如需詳細資訊，請參閱 LangChain 文件中的 Quickstart。
- 基礎模型選擇 – 支援 Amazon Bedrock 上的各種基礎模型，包括 Anthropic Claude、Amazon Nova 模型 (Premier、Pro、Lite 和 Micro)，以及其他具有不同推理功能的模型。如需詳細資訊，請參閱 LangChain 文件中的輸入和輸出。
- LLM API 整合 – 適用於多個大型語言模型 (LLM) 服務提供者的標準化界面，包括 Amazon BedrockOpenAI、 和其他可靈活部署的 。如需詳細資訊，請參閱 LangChain 文件中的 LLMs。
- 多模態處理 – 內建支援文字、影像和音訊處理，以啟用豐富的多模態自動代理程式互動。如需詳細資訊，請參閱 LangChain 文件中的多模態。
- 以圖形為基礎的工作流程 – LangGraph可將複雜的自主代理程式行為定義為狀態機器，支援複雜的決策邏輯。如需詳細資訊，請參閱LangGraph平台 GA 公告。
- 記憶體抽象化 – 短期和長期記憶體管理的多個選項，這對隨著時間維持內容的自動代理程式至關重要。如需詳細資訊，請參閱 LangChain 文件中的如何將記憶體新增至聊天機器人。
- 工具整合 – 跨各種服務和 APIs 的豐富工具整合生態系統，擴展自動代理程式功能。如需詳細資訊，請參閱 LangChain 文件中的工具。
- LangGraph 平台 – 生產環境的受管部署和監控解決方案，支援長時間執行的自動代理程式。如需詳細資訊，請參閱LangGraph平台 GA 公告。

何時使用 LangChain和 LangGraph

LangChain 和 LangGraph特別適合自動代理程式案例，包括：

- 複雜的多步驟推理工作流程，需要複雜的協同運作來進行自動決策
- 需要存取大型生態系統的專案，這些生態系統包含預先建置的元件和整合，以實現各種自動化功能
- 與想要建置自動化系統的現有 Python型機器學習 (ML) 基礎設施和專業知識的團隊
- 需要跨長時間執行的自動代理程式工作階段進行複雜狀態管理的使用案例

LangChain 和 的實作方法 LangGraph

LangChain 並為業務利益相關者LangGraph提供結構化實作方法，如 LangGraph 文件所述。此架構可讓組織：

- 定義代表業務流程的複雜工作流程圖表。
- 使用決策點和條件式邏輯建立多步驟推理模式。
- 整合多模式處理功能，以處理各種資料類型。
- 透過內建的檢閱和驗證機制實作品質控制。

這種以圖形為基礎的方法可讓業務團隊將複雜的決策程序建模為自動化工作流程。團隊清楚了解推理程序的每個步驟，以及稽核決策路徑的能力。

LangChain 和 的實際範例 LangGraph

Vodafone 已使用 LangChain (和 LangGraph) 實作自動代理程式，以增強其資料工程和操作工作流程，如LangChain企業案例研究中所述。他們建立了內部 AI 助理，可透過自然語言互動自動監控效能指標、從文件系統擷取資訊，以及提供可行的洞見。

Vodafone 實作使用LangChain模組化文件載入器、向量整合和對多個 LLMs(OpenAI、LLaMA#3 和 Gemini) 的支援，來快速建立原型並對這些管道進行基準測試。然後，它們透過部署模組化子代理程式LangGraph來建構多代理程式協同運作。這些代理程式會執行收集、處理、摘要和推理任務。 透過APIs將這些代理程式LangGraph整合到其雲端系統中。

CrewAI

CrewAI 是一種開放原始碼架構，專門專注於自動多代理程式協同運作，可在 GitHub 上取得。它提供結構化方法來建立專業自主代理程式團隊，以協作解決複雜的任務，而無需人工介入。 CrewAI強調以角色為基礎的協調和任務委派。

的主要功能 CrewAI

CrewAI 提供下列主要功能：

- 以角色為基礎的代理程式設計 – 自治代理程式是以特定角色、目標和背景案例定義，以實現專業專業知識。如需詳細資訊，請參閱 CrewAI 文件中的建立有效的代理程式。
- 任務委派 – 內建機制，可根據任務的功能自動將任務指派給適當的客服人員。如需詳細資訊，請參閱 CrewAI 文件中的任務。
- 客服人員協作 – 用於自動客服人員間通訊和知識分享的架構，無需人工調解。如需詳細資訊，請參閱 CrewAI 文件中的協同合作。
- 程序管理 – 用於循序和平行自主任務執行的結構化工作流程。如需詳細資訊，請參閱 CrewAI 文件中的處理程序。

- 基礎模型選擇 – 支援 Amazon Bedrock 上的各種基礎模型，包括 Anthropic Claude、Amazon Nova 模型 (Premier、Pro、Lite 和 Micro)，以及針對不同自主推理任務進行最佳化的其他模型。如需詳細資訊，請參閱 CrewAI 文件中的 LLMs。
- LLM API 整合 – 與多個 LLM 服務介面的彈性整合，包括 Amazon BedrockOpenAI、 和本機模型部署。如需詳細資訊，請參閱 CrewAI 文件中的提供者組態範例。
- 多模式支援 – 用於處理文字、影像和其他模態的新興功能，以實現全面的自動代理程式互動。如需詳細資訊，請參閱 CrewAI 文件中的使用多模式代理程式。

使用時機 CrewAI

CrewAI 特別適合自動代理程式案例，包括：

- 受益於專業、以角色為基礎的專業知識的複雜問題，可自動運作
- 需要多個自主代理程式之間明確協同合作的專案
- 以團隊為基礎的問題分解改善自動解決問題的使用案例
- 需要在不同自動代理程式角色之間明確分離疑慮的情況

的實作方法 CrewAI

CrewAI 為業務利益相關者提供以角色為基礎的 AI 代理程式團隊方法實作，如 CrewAI 文件入門中所述。此架構可讓組織：

- 定義具有特定角色、目標和專業知識領域的專業自主代理程式。
- 根據客服人員的專門功能，將任務指派給客服人員。
- 在任務之間建立明確的相依性，以建立結構化工作流程。
- 協調多個客服人員之間的協同合作，以解決複雜的問題。

這種以角色為基礎的方法反映了人類團隊結構，讓業務領導者能夠直覺地了解和實作。組織可以建立具有專業知識領域的自主團隊，以協同合作實現業務目標，類似於人類團隊的運作方式。不過，自動化團隊可以在無需人工介入的情況下持續運作。

的實際範例 CrewAI

AWS 已如CrewAI已發佈的案例研究所述，使用與 Amazon Bedrock 整合的CrewAI」實作自動多代理程式系統。 AWS 並CrewAI開發了安全、廠商中立的架構。CrewAI 開放原始碼「流程和資源」架構與 Amazon Bedrock 基礎模型、記憶體系統和合規防護機制無縫整合。

實作的關鍵元素包括：

- 藍圖和開放原始碼 – AWS 以及CrewAI發佈的參考設計，可將CrewAI代理程式映射至 Amazon Bedrock 模型和可觀測性工具。他們還發佈了範例系統，例如多代理程式 AWS 安全稽核人員、程式碼現代化流程和消費者包裝商品 (CPG) 後台自動化。
- 可觀測性堆疊整合 – 解決方案使用 Amazon CloudWatch、 AgentOps和 嵌入監控LangFuse，實現從概念驗證到生產的可追蹤性和偵錯。
- 已證明投資報酬率 (ROI) – 早期試行者展現了重大改進：大型程式碼現代化專案的執行速度快 70#%，CPG 後台流程的處理時間縮短約 90#%。

AutoGen

AutoGen 是 最初發行的開放原始碼架構Microsoft。 AutoGen 著重於啟用對話式和協作式自主 AI 代理器。它提供靈活的架構，用於建置多代理程式系統，強調代理程式之間用於複雜自主工作流程的非同步、事件驅動的互動。

的主要功能 AutoGen

AutoGen 提供下列主要功能：

- 對話式客服人員 – 以自動客服人員之間的自然語言對話為基礎，透過對話實現複雜的推理。如需詳細資訊，請參閱 AutoGen 文件中的多客服人員對話架構。
- 非同步架構 – 非封鎖自動代理程式互動的事件驅動設計，支援複雜的平行工作流程。如需詳細資訊，請參閱 AutoGen 文件中的解決非同步聊天序列中的多個任務。
- Human-in-the-loop – 支援人類視需要選擇性參與自主代理程式工作流程。如需詳細資訊，請參閱 AutoGen 文件中的允許客服人員的人工意見回饋。
- 程式碼產生和執行 – 適用於程式碼導向自動代理程式的特殊功能，可寫入和執行程式碼。如需詳細資訊，請參閱 AutoGen 文件中的程式碼執行。
- 可自訂的行為 – 針對各種使用案例的彈性自動代理程式組態和對話控制。如需詳細資訊，請參閱 AutoGen 文件中的 agentchat.conversable_agent。

- 基礎模型選擇 – 支援 Amazon Bedrock 上的各種基礎模型，包括 Anthropic Claude、Amazon Nova 模型 (Premier、Pro、Lite 和 Micro)，以及其他具有不同自動推理功能的模型。如需詳細資訊，請參閱 AutoGen 文件中的 LLM 組態。
- LLM API 整合 – 多個 LLM 服務介面的標準化組態，包括 Amazon BedrockOpenAI、 和 Azure OpenAI。如需詳細資訊，請參閱 AutoGen API 參考中的 oai.openai_utils。
- 多模態處理 – 支援文字和影像處理，以啟用豐富的多模態自動代理程式互動。如需詳細資訊，請參閱 AutoGen 文件中的在 中使用多模態模型：GPT-4VAutoGen。

使用時機 AutoGen

AutoGen 特別適合自動代理程式案例，包括：

- 需要自動代理程式之間自然對話流程的應用程式，以進行複雜的推理
- 同時需要完全自主操作和選用人工監督功能的專案
- 涉及自動程式碼產生、執行和偵錯的使用案例，無需人工介入
- 需要彈性、非同步自主代理程式通訊模式的案例

的實作方法 AutoGen

AutoGen 為業務利益相關者提供對話式實作方法，如 AutoGen 文件入門中所述。此架構可讓組織：

- 建立自主代理程式，透過自然語言對話進行通訊。
- 在多個客服人員之間實作非同步、事件驅動的互動。
- 必要時，結合全自主操作與選用的人工監督。
- 為透過對話協作的不同業務職能開發專門的客服人員。

這種對話式方法可讓自動化系統的推理透明化，並可供商業使用者存取。決策者可以觀察客服人員之間的對話，以了解如何得出結論，並在需要人工判斷時選擇性地參與對話。

的實際範例 AutoGen

Magentic-One 是一種開放原始碼、通則多代理程式系統，旨在自主解決各種環境的複雜多步驟任務，如 Microsoft AI Frontiers 部落格中所述。其核心是 Orchestrator 代理程式，它會分解高階目標，並使用結構化總帳追蹤進度。此代理程式會將子任務委派給專業代理程式（例如 WebSurfer、Coder、 和 ComputerTerminal)FileSurfer，並在必要時重新規劃以動態調整。

系統是以AutoGen架構為基礎，且與模型無關，預設為 GPT-4o。它可在 GAIA、 AssistantBench 和 等基準測試中實現最先進的效能WebArena，完全無需進行任務特定的調校。此外，它還透過 AutoGenBench建議支援模組化可擴展性和嚴格的評估。

LlamaIndex

LlamaIndex 是一種資料架構，專門用於將大型語言模型 (LLMs) 與外部資料來源連線，以啟用複雜的擷取增強生成 (RAG) 和代理式 AI 應用程式。此架構為代理程式系統、自訂協同運作模式和系統整合提供抽象和加速開發工作流程，以減少知識驅動型 AI 解決方案的time-to-production。

的主要功能 LlamaIndex

LlamaIndex 提供一組完整的功能，特別適合企業代理式 AI 應用程式：

- 以資料為中心的架構 – 從超過 100 種資料格式擷取、編製索引和擷取資訊的 ExcelPDFs、MicrosoftWord 文件、試算表等。框架會將企業資料轉換為可查詢的知識庫，這些知識庫已針對 AI 代理器進行最佳化。如需詳細資訊，請參閱 LlamaIndex 文件。
- 生產就緒部署 – 透過 LlamaIndex同時提供開放原始碼架構和受管服務LlamaCloud，提供企業級功能，包括安全控制、可擴展性、可觀測性整合和部署彈性。如需詳細資訊，請參閱LlamaIndex架構文件。
- 進階文件處理 – LlamaCloud提供文件剖析、擷取、索引和擷取功能，可處理複雜的配置、巢狀資料表、多模態內容，甚至是手寫筆記。這種複雜的剖析可讓客服人員有效地處理包含圖表、圖表和複雜格式的真實世界企業文件。如需詳細資訊，請參閱 LlamaCloud 文件。
- 工作流程協同運作 – LlamaAgents提供事件驅動的非同步優先協同運作引擎，用於建置多步驟代理系統。工作流程支援複雜的模式，包括迴圈、平行執行、條件式分支和有狀態恢復，使其非常適合複雜的客服人員互動。如需詳細資訊，請參閱LlamaIndex工作流程文件。
- 代理程式擷取功能 – 進階擷取模式，包括混合搜尋、語意搜尋和自動路由，可智慧地判斷每個查詢的最佳擷取策略。此架構支援跨多個知識庫的複合擷取，並透過重新排名來提高準確性。如需詳細資訊，請參閱 LlamaIndex RAG 文件。
- 可觀測性和評估 – 與各種可觀測性和評估工具LlamaIndex整合。此整合功能可協助您追蹤和偵錯應用程式、評估其效能，以及監控成本。如需詳細資訊，請參閱追蹤、偵錯和評估LlamaIndex文件。

使用時機 LlamaIndex

LlamaIndex 特別適合強調資料密集型工作流程和知識管理的客服人員 AI 案例：

- 文件密集型應用程式，需要客服人員處理、分析和擷取大量企業文件的洞見，例如合約、報告、手冊和法規文件
- 生產案例的快速原型設計，其中組織想要快速建置和部署以文件為中心的代理程式，而無需大量的基礎設施管理開銷
- 優先考慮擷取準確性和內容相關性的 RAG 架構，特別是在使用包含資料表、影像和結構化資料的複雜多模式文件時
- 多代理程式文件工作流程，需要專門的代理程式來處理文件的不同層面，例如剖析、分析、摘要和合規檢查

的實作方法 LlamaIndex

LlamaIndex 同時提供低階建置區塊和高階抽象，以適應不同的實作方法：

- 使用LlamaIndex高階 APIs，在幾行程式碼中快速開發功能性 RAG 應用程式。此方法可讓初次接觸代理式 AI 的業務團隊和開發人員LlamaIndex存取。
- 透過 LlamaHub 進行企業整合，適用於熱門的企業系統，包括 SharePoint、Amazon Simple Storage Service (Amazon S3)、資料庫和 APIs。此方法可無縫整合現有的資料基礎設施。
- 開放原始碼自我託管部署之間的彈性部署選項，可實現最大控制，或可降低營運開銷和企業功能的LlamaCloud受管服務。
- 應用程式可以從簡單的查詢引擎開始，並隨著需求的演進逐步新增代理程式功能、多重代理程式協同運作和複雜的工作流程。

的實際範例 LlamaIndex

此範例著重於一家專門提供航空導航和操作解決方案的航太公司的子公司。他們需要解決不斷增長的挑戰，這涉及試行不協調的 AI 聊天機器人試驗。這些試驗導致整個組織的重複工作、長開發週期、合規障礙和隔離實作。

他們開發了統一的代理程式架構，這是以LlamaIndex開放原始碼架構為基礎的可重複使用範本型解決方案，可讓代理程式建立更有效率。它們比較了數個競爭架構，包括鏈結導向和圖形型架構。最後，他們選擇了LlamaIndex三個關鍵優勢：其彈性設計、模組化元件和生產就緒的協同運作控制。

該平台將代理程式開發和部署時間從 512 縮短 87% 到 64 小時。透過讓團隊建置具有大約 50 行程式碼和 JSON 組態檔案的客服人員，即可實現此減少。團隊利用具有內建安全性、合規性和特殊權限系統存取權的統一架構。如需詳細資訊，請參閱LlamaIndex客戶案例研究。

比較代理式 AI 架構

選擇適用於自動代理程式開發的代理程式 AI 架構時，請考慮每個選項如何符合您的特定需求。不僅考慮其技術能力，還考慮其組織適用性，包括團隊專業知識、現有基礎設施和長期維護需求。許多組織可能受益於混合式方法，利用多個架構來實現自動化 AI 生態系統的不同元件。

下表比較各架構在關鍵技術維度之間的成熟度等級（最強、強、適當或弱）。對於每個架構，資料表也包含生產部署選項和學習曲線複雜性的相關資訊。

架構	AWS 整合	自動多重代理程式支援	自主工作流程複雜性	多模式功能	基礎模型選擇	LLM API 整合	生產部署	學習曲線
AutoGen	脆弱	強大	強大	適當	適當	強大	自行執行 (DIY)	陡峭
CrewAI	脆弱	強大	適當	脆弱	適當	適當	DIY	適中
LangChain / LangGraph	適當	強大	最強	最強	最強	最強	平台或 DIY	陡峭
LlamaIndex	適當	適當	強大	適當	強大	強大	平台或 DIY	適中
Strands Agents	最強	強大	最強	強大	強大	最強	DIY	適中

選擇代理式 AI 架構時的考量事項

開發自動代理程式時，請考慮下列關鍵因素：

- AWS 基礎設施整合 – 大量投資的組織 AWS 將受益於 Strands Agents 與 的原生整合 AWS 服務，以進行自動化工作流程。如需詳細資訊，請參閱AWS 每週總和 (AWS 部落格）。

- 基礎模型選擇 – 根據自動代理程式的推理需求，考慮哪個架構為您的偏好基礎模型（例如 Amazon Bedrock 或 Anthropic Claude 上的 Amazon Nova 模型）提供最佳支援。如需詳細資訊，請參閱 Anthropic 網站上的建置有效的代理程式。
- LLM API 整合 – 根據架構與您偏好的大型語言模型 (LLM) 服務介面（例如 Amazon Bedrock 或 OpenAI) 的整合來評估架構，以進行生產部署。如需詳細資訊，請參閱 文件中的 Strands Agents模型界面。
- 多模態需求 – 對於需要處理文字、影像和語音的自動代理程式，請考慮每個架構的多模態功能。如需詳細資訊，請參閱 LangChain 文件中的多模態。
- 自主工作流程複雜性 – 具有複雜狀態管理的更複雜自主工作流程可能偏好進階狀態機器功能。 的 LangGraph。
- 自主團隊協作 – 需要專業客服人員之間明確角色型自主協作的專案，可以從 的團隊導向架構中受益 CrewAI。
- 自主開發範例 – 偏好自動代理程式對話、非同步模式的團隊可能會偏好 的事件驅動型架構 AutoGen。
- 受管或程式碼型方法 – 希望以最少編碼獲得全受管體驗的組織應考慮 Amazon Bedrock 代理程式。需要更深入自訂的組織可能會偏好 Strands Agents或其他具有專門功能的架構，以更符合特定的自主代理程式需求。
- 自動系統的生產準備程度 – 考慮生產自動代理器的部署選項、監控功能和企業功能。

平台

代理式 AI 平台提供部署、擴展和管理生產級代理系統所需的基礎執行期、協同運作和整合層。架構定義客服人員的建置方式，而通訊協定則控管客服人員的通訊方式。平台提供這些代理程式大規模安全操作、協作和發展的環境。

代理程式平台將模型執行、內容管理、工具整合、可觀測性和控管功能結合到統一的環境中。這些平台可讓組織從實驗轉移到企業規模的部署。

在本節中：

為什麼平台很重要

對於尋求在生產環境中操作自動化系統的組織而言，代理式 AI 平台至關重要。它們提供下列功能：

- 提供用於託管、擴展和協調代理程式的執行時間協調。
- 跨多代理程式工作流程管理狀態、內容和記憶體。
- 提供符合企業標準的安全性、身分和控管控制。
- 透過標準 APIs 或通訊協定與工具生態系統和外部系統整合。
- 在客服人員互動和事件流程中啟用可觀測性和可稽核性。
- 支援跨模型互通性，允許客服人員在單一環境中使用多個基礎模型。

這些功能將個別代理程式轉換為協調的自適應系統，可在企業和法規界限內可靠地運作。

代理式 AI 平台的類型

代理式 AI 平台通常屬於下列一或多個類別：

- 受管代理程式 – 全受管平台提供內建的基礎設施、記憶體和協同運作功能。它們可減少營運開銷並加速生產時間。
- 開放原始碼協同運作 – 開放原始碼代理平台為偏好可自訂環境或內部部署的組織提供彈性和透明度。
- 混合企業 – 混合平台整合了受管和自我託管元件，將雲端受管服務的可擴展性與企業系統的控制相結合。

平台選擇考量事項

選取或設計代理式 AI 平台時，組織應考慮下列事項：

- 整合深度 – 評估平台與現有資料來源、工具和通訊協定的整合能力。
- 可擴展性 – 確保平台可以動態擴展以支援自主工作負載和多代理程式協同合作。
- 安全性與合規 – 根據組織和區域需求評估資料隱私權、加密和控管功能。
- 可擴展性 – 選擇具有模組化架構的平台，以允許隨時間新增新工具、模型或代理程式。
- 可觀測性 – 偏好為客服人員互動提供詳細遙測、可追蹤性和稽核日誌的平台。
- 成本效率 – 考慮無伺服器或以用量為基礎的模型，以最佳化可變工作負載的成本。

Amazon Bedrock 代理程式

Amazon Bedrock Agents 是一項全受管服務，可讓您在應用程式中建置和設定自動代理程式。它可以協調基礎模型、資料來源、軟體應用程式和使用者對話之間的互動。其建立代理程式的簡化方法不需要您佈建容量、管理基礎設施或撰寫自訂程式碼。

Amazon Bedrock 代理程式的主要功能

Amazon Bedrock 代理程式包含下列主要功能：

- 全受管服務 – 完整的基礎設施管理，無需佈建容量或管理基礎系統。如需詳細資訊，請參閱 Amazon Bedrock 文件中的使用 AI 代理器自動化應用程式中的任務。
- API 驅動的開發 – 透過簡單的 API 呼叫，透過指定模型、指示、工具和組態參數來定義和執行代理程式。如需詳細資訊，請參閱 Amazon Bedrock 文件中的手動建立和設定代理程式。
- 動作群組 – 透過使用 API 結構描述建立動作群組，定義您的代理程式可執行的特定動作。如需詳細資訊，請參閱《Amazon Bedrock 文件》中的使用動作群組來定義代理程式要執行的動作。

- 知識庫整合 – 無縫連線至 Amazon Bedrock 知識庫，以使用組織的資料增強客服人員回應。如需詳細資訊，請參閱 Amazon Bedrock 文件中的為具有知識庫的代理程式產生增強回應。
- 進階提示範本 – 透過提示範本自訂客服人員行為，以進行預先處理、協同運作、知識庫回應產生和後製處理。如需詳細資訊，請參閱《Amazon Bedrock 文件》中的使用 Amazon Bedrock 中的進階提示範本增強代理程式的準確性。
- 追蹤和可觀測性 – 使用內建的追蹤功能追蹤代理程式step-by-step推理程序。如需詳細資訊，請參閱 Amazon Bedrock 文件中的使用追蹤追蹤追蹤代理程式step-by-step推理程序。
- 版本控制和別名 – 建立多個版本的代理程式，並透過別名部署它們以進行受控推展。如需詳細資訊，請參閱 Amazon Bedrock 文件中的在您的應用程式中部署和使用 Amazon Bedrock 代理程式。

何時使用 Amazon Bedrock 代理程式

Amazon Bedrock Agents 特別適合自動代理程式案例，包括：

- 希望獲得完全受管體驗的組織，無需管理基礎設施即可建置和部署代理程式
- 需要透過組態而非程式碼快速開發和部署代理程式的專案
- 受益於與其他 Amazon Bedrock 功能緊密整合的使用案例，例如知識庫和護欄
- 沒有內部資源的團隊從頭開始建置代理程式，但需要生產就緒的自動化功能

Amazon Bedrock 代理程式的實作方法

Amazon Bedrock Agents 為業務利益相關者提供以組態為基礎的實作方法。服務可讓組織：

- 透過 AWS 管理主控台 或 API 呼叫定義客服人員，無需撰寫複雜的程式碼。
- 建立動作群組，指定代理程式可執行APIs 和操作。
- 連接知識庫，以提供特定網域的資訊給代理程式。
- 透過視覺化界面測試和反覆執行代理程式行為。

這種受管方法可讓業務團隊快速開發和部署自動代理程式，而無需 AI 模型開發或基礎設施管理的深厚技術專業知識。

Amazon Bedrock 代理程式的實際範例

本AWS 部落格文章所述的財務操作 (FinOps) 解決方案使用 Amazon Bedrock 多代理程式架構來建立 AI 驅動的雲端成本管理助理。經濟實惠的 Amazon Nova 基礎模型為中央 FinOps 主管代理程式將任

務委派給專業代理程式的解決方案提供支援。這些客服人員會使用 擷取和分析 AWS 支出資料， AWS Cost Explorer 並使用 產生節省成本的建議 AWS Trusted Advisor。

系統包括透過 Amazon Cognito 的安全使用者存取、託管在 上的前端 AWS Amplify，以及用於即時分析和預測 AWS Lambda 的動作群組。財務團隊可以詢問自然語言查詢，例如「2025 年 2 月我的成本是多少？」 系統會回應詳細的明細、最佳化建議和預測，全都使用 部署在可擴展的無伺服器架構內 AWS CloudFormation。

Amazon Bedrock AgentCore

Amazon Bedrock AgentCore 是一種代理程式平台，可使用任何架構、模型或通訊協定大規模安全地建置、部署和操作功能強大的代理程式。使用 AgentCore，您可以執行以下操作，無需任何基礎設施管理：

- 更快速地建置代理程式。
- 讓客服人員跨工具和資料採取動作。
- 使用低延遲和延長的執行時間安全地執行代理程式。
- 在生產環境中監控客服人員。

AgentCore 消除了建置專用代理程式基礎設施的無差別繁重工作，可讓您加速代理程式進入生產環境。其服務可以一起使用或獨立使用，並且與任何架構相容，包括 CrewAI、LlamaIndex、 LangGraph 和 Strands Agents。AgentCore 也相容於 Amazon Bedrock 內外提供的任何基礎模型，提供最大的彈性。

AgentCore 由數個關鍵服務組成：

- Amazon Bedrock AgentCore 執行期 – 提供安全、無伺服器、可擴展的環境來託管和執行您的代理程式，而無需管理部署和執行 AI 代理程式或工具所需的任何基礎設施。
- Amazon Bedrock AgentCore 記憶體 – 提供受管記憶體系統，透過保持即時和長期知識，讓客服人員保留互動的內容，以進行更個人化和一致的對話。
- Amazon Bedrock AgentCore Gateway – 簡化為客服人員建立、保護和尋找正確工具的程序。透過 AgentCore Gateway，開發人員可以將 APIs、Lambda 函數和現有服務轉換為與模型內容協定 (MCP) 相容的工具，並將其提供給客服人員。
- Amazon Bedrock AgentCore Identity – 提供安全、可擴展的代理程式身分和存取管理服務，可加速 AI 代理程式開發。使用 AgentCore Identity，您可以將唯一、可驗證的身分指派給客服人員，進而實現精細存取控制，並保護客服人員與企業系統的互動。

- Amazon Bedrock AgentCore 內建工具 – 可讓您使用內建工具來增強開發和測試工作流程。使用這些工具與您的應用程式有效互動，讓 AI 代理器能夠在沙盒環境中安全地撰寫和執行程式碼。使用瀏覽器工具讓 AI 代理器大規模與網站互動。
- Amazon Bedrock AgentCore 可觀測性 – 提供記錄和監控功能，讓您即時了解代理程式的效能和行為，以利偵錯和最佳化。

AgentCore 的主要功能

AgentCore 包含下列主要功能：

- 完全受管且可擴展 – AgentCore 是一種完全受管的服務，這表示 AWS 會處理基礎基礎設施和維護。它也是可擴展的，可讓您自訂和增強代理程式的功能。如需詳細資訊，請參閱 AgentCore 文件中的開始使用 AgentCore 執行期。 AgentCore
- 長期和短期記憶體 – 透過為客服人員配備記憶體系統，以從目前對話和長期知識中回收內容，提供更個人化的相關互動。如需詳細資訊，請參閱 AgentCore 文件中的開始使用 AgentCore 記憶體。 AgentCore
- 簡化工具開發和整合 – 讓您的代理程式能夠透過單一安全端點探索和使用工具。只需幾行程式碼，即可快速將現有的企業資源轉換為客服人員就緒工具，讓開發人員專注於建置獨特的功能。如需詳細資訊，請參閱 AgentCore 文件中的 AgentCore Gateway 入門。 AgentCore
- 安全且可擴展的基礎設施 – AgentCore 為部署和操作代理程式提供安全且可擴展的環境。它包含身分和存取管理、資料加密和網路安全的功能。如需詳細資訊，請參閱 AgentCore 文件中的開始使用 AgentCore Identity。 AgentCore
- 與各種工具整合 – 可讓您將代理程式與各種工具整合，包括程式碼解譯器，以及您可以使用 AgentCore 內建工具建置的瀏覽器工具。如需詳細資訊，請參閱 AgentCore 文件中的開始使用 AgentCore Code Interpreter 和開始使用 AgentCore 瀏覽器。 AgentCore
- 全方位的可觀測性和監控 – 使用全方位工具來追蹤、偵錯和監控其在生產環境中的效能，進而深入了解您的代理程式。視覺化代理程式的整個執行路徑，以稽核其推理並解決失敗。使用即時儀表板和標準化遙測資料來追蹤關鍵操作指標。如需詳細資訊，請參閱 AgentCore 文件中的將可觀測性新增至 Amazon Bedrock AgentCore 資源。

何時使用 AgentCore

AgentCore 特別適合自動代理程式案例，包括：

- 想要透過處理基礎設施、安全性、內建工具、可觀測性和擴展的全受管服務加速開發並降低營運開銷的組織

- 需要彈性的專案，搭配可共同運作或獨立運作的模組化服務，且與任何架構相容，例如 CrewAI或 LangGraph，以及來自任何來源的任何基礎模型
- 需要有狀態的對話客服人員的使用案例，這些客服人員需要維護內容並從過去的互動中學習，以提供個人化和相關的回應
- 透過與各種應用程式、資料來源和 APIs代理程式

AgentCore 的實作方法

AgentCore 專為希望將 AI 代理器從使用開放原始碼或自訂代理程式架構建置的概念驗證移至生產環境的組織而設計。透過 AgentCore，組織可以執行下列動作：

- 在無伺服器基礎設施上安全地部署代理程式，支援任何架構和模型，具有工作階段隔離和內建身分和存取管理，以實現end-to-end安全和合規。使用入門工具組，快速為主要代理程式架構建立 AgentCore 執行期代理程式。
- 透過整合持久性記憶體以保留內容，透過 AgentCore Gateway 簡化工具開發和整合，來增強代理程式。利用內建瀏覽器工具和程式碼解譯器進行進階工作流程。
- 使用採用 Amazon CloudWatch Application Insights 和 技術的可觀測性儀表板，追蹤生產環境中的 AI 代理器OpenTelemetry，追蹤 AgentCore 資源 （執行時間、記憶體、閘道和工具） 的關鍵指標。
- 使用任何代理程式架構和模型提供者，透過全受管、模組化服務、可組合區塊一起或獨立地加速部署和創新。這種靈活性有助於組織更快地從原型遷移到生產環境。

這種受管方法可讓組織快速安全地建置、部署和執行任何規模的企業級 AI 代理程式和多代理程式系統。

AgentCore 的實際範例

AWS 已觀察到拉丁美洲最大的銀行之一已使用 AI/ML 多年，以提供超個人化且安全的數位銀行體驗。銀行使用 AgentCore 來擴展代理式 AI 服務，為客戶提供直覺式互動、增強安全性和更高的自動化能力。根據 CTO，AgentCore 預期會支援其大規模履行客戶承諾的工作。AgentCore 為其開發人員提供工具和彈性來建置和管理代理程式，同時協助確保符合金融法規。

通訊協定

AI 代理器需要標準化的通訊協定，才能與其他代理程式和服務互動。實作代理程式架構的組織在互通性、廠商獨立性以及未來的投資方面面臨重大挑戰。

本節可協助您導覽agent-to-agent程式通訊協定環境，著重於將彈性和互通性最大化的開放標準。（如需agent-to-tool通訊協定的相關資訊，請參閱本指南稍後的工具整合策略。）

本節重點介紹模型內容通訊協定 (MCP)，這是 最初Anthropic在 2024 年開發的開放標準。今天，透過對通訊協定的開發和實作的貢獻， AWS 主動支援 MCP。 AWS 正在與包括 LangGraph、 CrewAI 和 等主要開放原始碼代理程式架構協作LlamaIndex，以在通訊協定上塑造代理程式間通訊的未來。如需詳細資訊，請參閱開啟客服人員互通性通訊協定第 1 部分：MCP 上的客服人員間通訊 (AWS 部落格）。

在本節中：

- 為什麼通訊協定選擇很重要
- Agent-to-agent通訊協定
- 選取代理程式通訊協定
- 代理程式通訊協定的實作策略
- MCP 入門
- ???

為什麼通訊協定選擇很重要

通訊協定選擇從根本上塑造了如何建置和發展 AI 代理器架構。透過選擇支援代理程式架構之間可攜性的通訊協定，您可以靈活地結合不同的代理程式系統和工作流程，以滿足特定需求。

開放式通訊協定可讓您跨多個架構整合客服人員。例如，使用 LangChain快速原型設計和實作生產系統與 Strands Agents，透過常見的通訊協定進行通訊，例如 MCP 或 Agent2Agent (A2A) 通訊協定。這種靈活性可降低對特定 AI 供應商的依賴性、簡化與現有系統的整合，並可讓您隨著時間增強代理程式功能。

精心設計的通訊協定也會建立一致的安全模式，以跨代理程式生態系統進行身分驗證和授權。最重要的是，通訊協定可攜性可保留您在新代理程式架構和功能出現時採用這些架構和功能的自由。選擇開放通訊協定可保護您的代理程式開發投資，同時保持與第三方系統的互通性。

開放通訊協定的優點

實作您自己的擴充功能或建置自訂代理程式系統時，開放式通訊協定會提供令人信服的優勢：

- 文件和透明度 – 通常提供全面的文件和透明實作
- 社群支援 – 存取更廣泛的開發人員社群以進行故障診斷和最佳實務
- 互通性保證 – 更佳保證您的延伸模組可跨不同的實作運作
- 未來相容性 – 降低中斷變更或棄用的風險
- 對開發的影響 – 有機會為通訊協定演變做出貢獻

Agent-to-agent通訊協定

下表提供代理程式通訊協定的概觀，可讓多個代理程式協同合作、委派任務和共用資訊。

通訊協定	適用於	考量
MCP 代理程式間通訊	尋求彈性客服人員協同合作模式的組織	• 所提議模型內容通訊協定 (MCP) 的延伸 AWS ，以其現有的agent-to-agent通訊基礎為基礎 • 啟用與 OAuth 型安全性的無縫代理程式協同合作
A2A 通訊協定	跨平台代理程式生態系統	• 後端為 Google • 較 MCP 採用更嚴格的新標準

在通訊協定選項之間決定

實作agent-to-agent通訊時，請將您的特定通訊需求與適當的通訊協定功能配對。不同的互動模式需要不同的通訊協定功能。下表概述常見的通訊模式，並針對每個案例建議最適合的通訊協定選擇。

模式	Description	理想的通訊協定選擇
簡單請求和回應	客服人員之間的一次性互動	具有無狀態流程的 MCP

狀態對話	持續對話與內容	具有工作階段管理的 MCP
多代理程式協同合作	多個客服人員之間的複雜互動	MCP 代理程式間或 AutoGen
以團隊為基礎的工作流程	具有定義角色的階層式客服人員團隊	MCP 代理程式間CrewAI、 或 AutoGen

除了通訊模式之外，數個技術和組織因素可能會影響您的通訊協定選擇。下表概述可協助您評估哪些通訊協定最符合您的特定實作需求的重要考量。

考量事項	Description	範例
安全模型	身分驗證和授權要求	MCP 中的 OAuth 2.0
部署環境	客服人員將在其中執行和通訊	分散式或單一機器
生態系統相容性	與現有代理程式架構整合	LangChain 或 Strands Agents
可擴展性需求	客服人員互動的預期成長	MCP 的串流功能

選取代理程式通訊協定

對於大多數建置生產代理程式系統的組織，模型內容通訊協定 (MCP) 為agent-to-agent之間的通訊提供了最完整且受支援的基礎。MCP 受益於來自 AWS 和開放原始碼社群的積極開發貢獻。

對於希望有效實作代理式 AI 的組織而言，選擇正確的代理程式通訊協定非常重要。考量事項因組織內容而異。

代理程式通訊協定選擇考量事項

選擇代理式 AI 系統的通訊協定時，組織應考慮下列最佳實務：

- 優先考慮開放標準 – 組織應採用 MCP 等開放通訊協定，以協助確保長期互通性、可擴展性，並降低廠商鎖定的風險。
- 平衡速度和靈活性 – 新創公司和早期採用者可以從支援良好的專屬通訊協定開始，以快速開發，但應該定義開放標準的遷移路徑，隨著系統成熟。
- 實作抽象層 – 企業應實作通訊協定抽象，以簡化遷移、啟用混合採用，以及符合未來需求的整合策略。

- 強調安全性和合規性 – 受監管產業的組織應選擇具有強大身分驗證、加密和稽核功能的通訊協定，以滿足控管和合規要求。
- 評估生態系統成熟度 – 所有組織都應評估每個通訊協定的運作狀態、採用和社群支援，以確保永續性並將技術負債降至最低。
- 參與標準開發 – Organizations 應參與標準機構或開放原始碼社群，以協助塑造通訊協定演變並影響最佳實務。
- 考慮資料主權 – 政府和受監管部門應確保通訊協定選擇符合跨部署區域的資料落地和主權要求。
- 利用受管服務 – 盡可能使用代理程式通訊協定的受管或無伺服器實作，以減少操作複雜性並加速部署。

代理程式通訊協定的實作策略

若要在整個組織中有效實作代理程式通訊協定，請考慮下列策略步驟：

1. 從標準一致性開始 – 盡可能採用已建立的開放式通訊協定。
2. 建立抽象層 – 在您的系統和特定通訊協定之間實作轉接器。
3. 致力於開放標準 – 參與通訊協定開發社群。
4. 監控通訊協定演變 – 隨時掌握新出現的標準和更新。
5. 定期測試互通性 – 驗證您的實作是否仍然相容。

MCP 入門

AWS 透過對通訊協定的開發和實作的貢獻，積極支援模型內容通訊協定 (MCP)。 AWS 正在與包括 LangGraph、 CrewAI和 在內的領導開放原始碼代理程式架構協作LlamaIndex，以在通訊協定上塑造代理程式間通訊的未來。

若要在代理程式架構中實作 MCP，請採取下列動作：

1. 探索 Strands Agents SDK 等架構中的 MCP 實作。
2. 檢閱模型內容通訊協定技術文件。
3. 閱讀客服人員互通性的開放式通訊協定第 1 部分：MCP 上的客服人員間通訊 (AWS 部落格），以了解客服人員互通性。
4. 加入 MCP 社群，以影響通訊協定的演變。

MCP 提供通訊層，可讓客服人員與外部資料和服務互動，也可以用來讓客服人員與其他客服人員互動。通訊協定的可串流 HTTP 傳輸實作為開發人員提供一組完整的互動模式，而不必重新打造輪子。這些模式支援無狀態請求/回應流程，以及具有持久 IDs的狀態工作階段管理。

透過採用 MCP 等開放式通訊協定，您可以將組織定位為建置代理程式系統，以隨著 AI 技術的演進而保持彈性、互通性和適應性。如需agent-to-tool通訊協定實作的相關資訊，請參閱本指南稍後的工具整合策略。

A2A 入門

Agent2Agent (A2A) 通訊協定可讓客服人員透過共用語意層進行分散式協同合作。A2A 不會透過中央協調器路由所有工作，而是允許客服人員彼此探索、公告其功能、交涉任務，以及使用輕量型 JSON 型通訊協定共用內容。每個客服人員都會發佈功能資訊清單。

下列範例顯示簡化的 A2A 功能資訊清單，公告代理程式支援的動作、所需的輸入和操作中繼資料，以啟用探索和任務交涉：

```
{
  "can": ["summarize.text", "extract.keywords"],
  "needs": ["document.input"],
  "meta": { "version": "1.0.3", "latencyMs": 120 }
}
```

此模型可啟用動態功能比對、中任務委派和跨組織協同合作。客服人員可以自行整理任務、形成臨時工作群組，並在新功能進入或退出系統時進行調整。

A2A 支援從簡單的無狀態請求到多步驟交涉工作階段的互動，包括：

- 用於低延遲協同合作的直接peer-to-peer傳訊
- 語意任務交涉，其中客服人員會選取最適合的對等
- 以能力為基礎的探索，實現緊急的人力分配
- 具狀態多步驟互動的工作階段錨定

透過採用開放的客服人員原生通訊協定，例如 A2A，組織可建立模組化、可互通且能夠跨邊界協作的 AI 系統。A2A 可確保客服人員生態系統保持彈性，並可隨著引進新的客服人員、團隊或外部系統而發展，而不需要剛性協同運作層或先前的耦合。

若要在代理程式架構中實作 A2A 通訊協定，請採取下列動作：

1. 檢閱 A2A 通訊協定規格 – 讀取 Agent2Agent (A2A) 通訊協定規格的最新版本，以了解功能資訊清單、交涉流程和客服人員交握的運作方式。
2. 探索 A2A-compatible執行時間 – 評估架構，例如 Strands Agents SDK 或支援 A2A-style功能資訊清單和peer-to-peer交涉的自訂執行時間層。
3. 為您的客服人員實作功能資訊清單 – 定義每個客服人員的 `can`、 `needs`和 `meta` 欄位，以啟用探索、配對和意圖層級協同合作。
4. 實驗 A2A 交涉模式 – 使用 request-offer-accept 迴圈、結構化功能查詢或 gossip 型探索，以了解客服人員應處理任務的人員原因。
5. 在混合基礎設施環境中測試 A2A – 結合 A2A 對等交涉與 AWS 透過 Amazon EventBridge 原生 的事件路由，以評估混合協調模式。
6. 加入 A2A 社群 – 與開放的工作群組互動，以掌握最新的延伸模組、安全建議和跨供應商互通性改進，並有助於通訊協定的開發。

工具

AI 代理器透過與外部工具、APIs和資料來源互動來提供價值，以執行有用的任務。正確的工具整合策略會直接影響代理程式的功能、安全狀態和長期彈性。

本節協助您導覽工具整合環境，專注於將自由和靈活性最大化的開放標準。本節重點介紹工具整合的模型內容通訊協定 (MCP)，並檢閱架構特定的工具和特殊的中繼工具，以增強代理程式工作流程。

在本節中：

工具類別

建置代理程式系統包含三個主要類別的工具。

通訊協定型工具

通訊協定型工具使用標準化通訊協定進行agent-to-tool的通訊：

- MCP 工具 – 使用本機和遠端執行選項，開啟跨架構運作的標準工具。
- OpenAI 函數呼叫 – 專屬於OpenAI模型的專屬工具。
- Anthropic 工具 – 函數OpenAI呼叫 Claude Anthropic 模型專屬工具的變化。

架構原生工具

架構原生工具直接建置在特定代理程式架構中：

- Strands Agents 工具 – 架構特有的輕量、quick-to-implement工具Strands Agents。
- LangChain 工具 – 與LangChain生態系統緊密整合的 Python型工具。
- LlamaIndex 工具 – 已針對 中的資料擷取和處理進行最佳化的工具LlamaIndex。

中繼工具

Meta-tools 可增強代理程式工作流程，無需直接採取外部動作：

- 工作流程工具 – 管理代理程式執行流程、分支邏輯和狀態管理。
- 客服人員圖形工具 – 在複雜的工作流程中協調多個客服人員。
- 記憶體工具 – 提供跨客服人員工作階段的持久性儲存和擷取資訊。
- 反射工具 – 可讓客服人員分析和改善自己的效能。

通訊協定型工具

考慮通訊協定型工具時，模型內容通訊協定 (MCP) 為工具整合提供了最全面且靈活的基礎。如AWS 開放原始碼部落格的客服人員互通性文章所述， AWS 採用 MCP 作為策略通訊協定，積極為其開發做出貢獻。

下表說明 MCP 工具部署的選項。

部署模型	Description	適用於	實作
以本機 stdio 為基礎的	工具在與代理程式相同的程序中執行	開發、測試和簡單工具	快速實作，無需網路額外負荷
本機伺服器傳送事件 (SSE) 型	工具會在本機執行，但透過 HTTP 進行通訊	具有分離疑慮的更複雜本機工具	更好的隔離，但仍低延遲
遠端 HTTP 可串流	在遠端伺服器上執行的工具	生產環境和共用工具	可擴展且集中管理

官方 MCP SDKs可用於建置 MCP 工具：

- Python SDK – 完整通訊協定支援的全面實作
- TypeScript SDK – 適用於 Web 應用程式的 JavaScript/TypeScript 實作
- Java SDK – 企業應用程式的 Java 實作

這些SDKs提供以您慣用語言建立 MCP 相容工具的建置區塊，以及一致的通訊協定規格實作。

此外， AWS 已在 Strands Agents SDK 中實作 MCP。Strands Agents 開發套件提供直接的方式來建立和使用與 MCP 相容的工具。Strands Agents GitHub 儲存庫提供完整的文件。對於更簡單的使用案例或在Strands Agents框架之外工作時，官方 MCP SDKs以多種語言直接實作通訊協定。

MCP 工具的安全功能

MCP 工具的安全功能包括下列項目：

- OAuth 2.0/2.1 身分驗證 – 產業標準身分驗證
- 許可範圍 – 工具的精細存取控制
- 工具功能探索 – 可用工具的動態探索
- 結構化錯誤處理 – 一致的錯誤模式

MCP 工具入門

若要實作 MCP 進行工具整合，請採取下列動作：

1. 探索適用於生產就緒 MCP 實作的 Strands Agents 開發套件。
2. 檢閱 MCP 技術文件以了解核心概念。
3. 使用此AWS 開放原始碼部落格文章中所述的實際範例。
4. 從簡單的本機工具開始，然後再繼續進行遠端工具。
5. 加入 MCP 社群，以影響通訊協定的演變。

探索 AgentCore Gateway

Amazon Bedrock AgentCore Gateway 為開發人員提供簡單且安全的方式來大規模建置、部署、探索和連線至 MCP 工具和其他目標端點。透過 AgentCore Gateway，開發人員可以將 APIs、 AWS Lambda 函數和現有服務轉換為與 MCP 相容的工具。然後，只要幾行程式碼，他們就可以透過 AgentCore Gateway 端點將這些工具提供給客服人員。AgentCore Gateway 支援 OpenAPI、 Smithy 和 Lambda 作為輸入類型，並且是唯一在全受管服務中提供完整輸入身分驗證和輸出身分驗證的解決方案。

架構原生工具

雖然模型內容通訊協定 (MCP) 提供最靈活的基礎，但架構原生工具可為特定使用案例提供優勢。

Strands Agents 開發套件提供以 Python為基礎的工具，其輕量型設計為特徵，只需最少的負荷即可進行簡單的操作。它們可以快速實作，並允許開發人員使用幾行程式碼來建立工具。此外，它們已緊密整合，可在Strands Agents架構內順暢運作。

下列範例示範如何使用 建立簡單的天氣工具Strands Agents。開發人員可以將Python函數快速轉換為客服人員可存取的工具，並將程式碼額外負荷降至最低，並自動從函數的 Docstring 產生適當的文件。

```
#Example of a simple Strands native tool

@tool

def weather(location: str) -> str:

"""Get the current weather for a location""" #

Implementation here

return f"The weather in {location} is sunny."
```

對於快速原型或簡單的使用案例，架構原生工具可以加速開發。不過，對於生產系統，MCP 工具提供比架構原生工具更好的互通性和未來彈性。

下表提供其他架構特定工具的概觀。

架構	工具類型	優點	考量
AutoGen	函數定義	強大的多代理程式支援	Microsoft 生態系統
LangChain	Python 類別	預先建置工具的大型生態系統	架構鎖定
LlamaIndex	Python 函式	針對資料操作進行最佳化	限制為 LlamaIndex

中繼工具

中繼工具不會直接與外部系統互動。反之，他們會實作代理程式模式來增強代理程式功能。本節討論工作流程、代理程式圖形和記憶體中繼工具。

工作流程中繼工具

工作流程中繼工具可管理代理程式執行的流程：

- 狀態管理 – 維護多個客服人員互動的內容
- 分支邏輯 – 啟用條件式執行路徑
- 重試機制 – 使用複雜的重試策略處理失敗

具有工作流程中繼工具的範例架構包括 LangGraph和 Strands Agents工作流程功能。

代理程式圖形中繼工具

代理程式圖形中繼工具會協調多個同時運作的代理程式：

- 任務委派 – 將子任務指派給專業客服人員
- 結果彙總 – 合併來自多個客服人員的輸出
- 衝突解決 – 解決客服人員之間的分歧

像 AutoGen和 之類的架構CrewAI專門處理客服人員圖形協調。

記憶體中繼工具

記憶體中繼工具提供持久性儲存和擷取：

- 對話歷史記錄 – 維護跨工作階段的內容
- 知識庫 – 存放和擷取特定網域的資訊
- 向量存放區 – 啟用語意搜尋功能

MCP 的資源系統提供標準化的方式來實作跨不同代理程式架構運作的記憶體中繼工具。

工具整合策略

您選擇的工具整合策略會直接影響代理程式可以完成的內容，以及系統可以輕鬆發展的程度。優先考慮開放式通訊協定，例如模型內容通訊協定 (MCP)，同時策略性地使用架構原生工具和中繼工具。如此一來，您就能建置工具生態系統，隨著 AI 技術發展而保持彈性和強大。

下列工具整合的策略方法可最大限度地提高靈活性，同時滿足組織的立即需求：

1. 採用 MCP 作為您的基礎 – MCP 提供標準化方法，將代理程式連接到具有強大安全功能的工具。從 MCP 開始，做為下列項目的主要工具通訊協定：
   - 將用於多個代理程式實作的策略工具。
   - 需要強大身分驗證和授權的安全敏感工具。
   - 在生產環境中需要遠端執行的工具。
2. 適當時使用架構原生工具 – 考慮架構原生工具：
   - 初始開發期間的快速原型設計。
   - 具有最低安全需求的簡單非關鍵工具。
   - 利用唯一功能的架構特定功能。
3. 實作複雜工作流程的中繼工具 – 新增中繼工具以增強您的代理程式架構：
   - 從基本工作流程模式開始簡單。
   - 隨著使用案例成熟，增加複雜性。
   - 標準化代理程式和中繼工具之間的界面。
4. 規劃進化 – 以未來靈活性為重心建置：
   - 獨立於實作的文件工具界面。
   - 在客服人員和工具之間建立抽象層。
   - 建立從專屬通訊協定到開放通訊協定的遷移路徑。

工具整合的安全最佳實務

工具整合會直接影響您的安全狀態。本節概述要為您的組織考慮的最佳實務。

身分驗證和授權

利用下列強大的存取控制：

- 使用 OAuth 2.0/2.1 – 實作遠端工具的產業標準身分驗證。
- 實作最低權限 – 僅授予工具所需的許可。
- 輪換登入資料 – 定期更新 API 金鑰和存取權杖。

資料保護

為了協助保護資料，請採用下列措施：

- 驗證輸入和輸出 – 為所有工具互動實作結構描述驗證。
- 加密敏感資料 – 對所有遠端工具通訊使用 TLS。
- 實作資料最小化 – 僅將必要資訊傳遞至工具。

監控和稽核

使用以下機制來維持可見性和控制：

- 記錄所有工具叫用 – 維護完整的稽核線索。
- 監控異常 – 偵測不尋常的工具使用模式。
- 實作速率限制 – 透過過多的工具呼叫防止濫用。

模型內容通訊協定 (MCP) 安全模型可全面解決這些問題。如需詳細資訊，請參閱 MCP 文件中的安全考量。

結論

代理式 AI 的前景持續快速發展，為組織提供強大的新方法來建置智慧、自動化系統。本指南探索了成功實作的三個基本元件：提供基礎的架構、提供環境的平台、啟用通訊的通訊協定，以及擴展功能的工具。

隨著架構的成熟，您可以預期互通性、模型內容通訊協定 (MCP) 等通訊協定的標準化，以及自動化代理程式更複雜的協同運作功能。目前使用這些架構建立專業知識的組織，將能夠妥善地建置越來越自主的智慧型代理程式，以提供顯著的商業價值。

平台提供代理系統運作所在的執行、控管和生命週期環境。它們處理諸如身分、安全界限、可觀測性、記憶體管理、工作階段基礎以及與工具和資料的安全互動等問題。在 AWS 環境中，受管代理程式執行時間和協同運作服務等平台可讓組織大規模部署、監控、發展和管理自動代理程式和代理程式系統。平台會橋接基礎架構與實際操作需求。

客服人員通訊協定的選擇代表策略決策，在即時開發需求與長期彈性和互通性之間取得平衡。透過優先考慮開放式通訊協定並建立適當的抽象層，組織可以建置代理程式系統，以適應不斷發展的技術，同時滿足目前的業務需求。

對於大多數組織而言，MCP 代表堅實的基礎，因為它具有開放的標準、不斷成長的生態系統、agent-to-agent通訊模式的支援，以及工具整合功能。 AWS 採用 MCP 和 Agent2Agent (A2A) 作為策略通訊協定，積極為其開發做出貢獻，並在 Strands Agents SDK 等服務之間實作它們。透過使用 MCP 或 A2A 搭配適當的架構原生工具和中繼工具，您可以建置代理程式系統，以提供立即值，同時保持適應未來的創新。

Resources

使用下列 AWS 和其他與自動代理程式開發相關的資源。

AWS 部落格

- Amazon Bedrock AgentCore 記憶體：建置內容感知代理程式
- 使用 Amazon Bedrock Agents 建置強大生成式 AI 應用程式的最佳實務 – 第 1 部分
- 使用 Amazon Bedrock Agents 建置強大生成式 AI 應用程式的最佳實務 – 第 2 部分
- 使用 LlamaIndex和 Amazon Bedrock 建置強大的 RAG 管道
- 使用 Amazon Bedrock AgentCore 可觀測性建置值得信賴的 AI 代理器
- 使用 Amazon Bedrock LlamaIndex和 RAGAS 評估 RAG 回應
- Amazon Bedrock AgentCore 程式碼解譯器簡介
- Amazon Bedrock AgentCore Gateway 簡介：轉換企業 AI 代理程式工具開發
- Amazon Bedrock AgentCore Identity：大規模保護代理式 AI
- 簡介 Strands Agents，開放原始碼 AI 代理程式 SDK
- 客服人員互通性的開放式通訊協定第 1 部分：MCP 上的客服人員間通訊
- 在 Amazon Bedrock AgentCore 執行期上安全地啟動和擴展您的代理程式和工具
- AWS Transform for .NET 是第一個大規模現代化 .NET 應用程式的代理式 AI 服務
- AWS 每週總和： Strands Agents

AWS 方案指引

- 在 上操作代理式 AI AWS
- 上的代理式 AI 基礎 AWS
- 上的客服人員 AI 模式和工作流程 AWS
- 在 上建置代理式 AI 的無伺服器架構 AWS
- 在 上建置代理式 AI 的多租戶架構 AWS
- 上的代理式 AI 安全性 AWS
- 在 上擷取增強產生選項和架構 AWS

AWS 資源

- Amazon Bedrock 文件
- Amazon Bedrock AgentCore 文件
- Amazon Bedrock AgentCore Starter Toolkit (GitHub 儲存庫)
- Amazon Nova 文件
- AWS MCP 伺服器 (GitHub 儲存庫)

其他資源

- AutoGen 文件 (Microsoft)
- 建置有效的代理程式 (Anthropic)
- CrewAI GitHub 儲存庫
- LangChain 文件
- LangGraph 平台
- LlamaIndex 文件
- 模型內容通訊協定文件
- Strands Agents 文件
- Strands Agents 工具概觀
- Strands Agents 快速入門指南

文件歷史紀錄

下表描述了本指南的重大變更。如果您想收到有關未來更新的通知，可以訂閱 RSS 摘要。

變更	描述	日期
新增章節	新增的平台區段	2026 年 1 月 16 日
初次出版	—	2025 年 7 月 14 日

AWS 規範性指引詞彙表

以下是 AWS Prescriptive Guidance 提供的策略、指南和模式中常用的術語。若要建議項目，請使用詞彙表末尾的提供意見回饋連結。

數字

7 R

將應用程式移至雲端的七種常見遷移策略。這些策略以 Gartner 在 2011 年確定的 5 R 為基礎，包括以下內容：

- 重構/重新架構 – 充分利用雲端原生功能來移動應用程式並修改其架構，以提高敏捷性、效能和可擴展性。這通常涉及移植作業系統和資料庫。範例：將您的現場部署 Oracle 資料庫 遷移至 Amazon Aurora PostgreSQL 相容版本。
- 平台轉換 (隨即重塑) – 將應用程式移至雲端，並引入一定程度的優化以利用雲端功能。範例：將內部部署 Oracle 資料庫 遷移至 中的 Amazon Relational Database Service (Amazon RDS) for Oracle AWS 雲端。
- 重新購買 (捨棄再購買) – 切換至不同的產品，通常從傳統授權移至 SaaS 模型。範例：將您的客戶關係管理 (CRM) 系統 遷移至 Salesforce.com。
- 主機轉換 (隨即轉移) – 將應用程式移至雲端，而不進行任何變更以利用雲端功能。範例：將您的現場部署 Oracle 資料庫 遷移至 中 EC2 執行個體上的 Oracle AWS 雲端。
- 重新放置 (虛擬機器監視器等級隨即轉移) – 將基礎設施移至雲端，無需購買新硬體、重寫應用程式或修改現有操作。您可以將伺服器從內部部署平台遷移到相同平台的雲端服務。範例：將 Microsoft Hyper-V應用程式 遷移至 AWS。
- 保留 (重新檢視) – 將應用程式保留在來源環境中。其中可能包括需要重要重構的應用程式，且您希望將該工作延遲到以後，以及您想要保留的舊版應用程式，因為沒有業務理由來進行遷移。
- 淘汰 – 解除委任或移除來源環境中不再需要的應用程式。

A

ABAC

請參閱屬性型存取控制。

抽象服務

請參閱 受管服務。

ACID

請參閱原子性、一致性、隔離性、耐久性。

主動-主動式遷移

一種資料庫遷移方法，其中來源和目標資料庫保持同步 (透過使用雙向複寫工具或雙重寫入操作)，且兩個資料庫都在遷移期間處理來自連接應用程式的交易。此方法支援小型、受控制批次的遷移，而不需要一次性切換。它更靈活，但比主動-被動遷移需要更多的工作。

主動-被動式遷移

一種資料庫遷移方法，其中來源和目標資料庫保持同步，但只有來源資料庫會在資料複寫至目標資料庫時處理來自連線應用程式的交易。目標資料庫在遷移期間不接受任何交易。

彙總函數

在一組資料列上運作的 SQL 函數，會計算群組的單一傳回值。彙總函數的範例包括 SUM和 MAX。

AI

請參閱人工智慧。

AIOps

請參閱人工智慧操作。

匿名化

永久刪除資料集中個人資訊的程序。匿名化有助於保護個人隱私權。匿名資料不再被視為個人資料。

反模式

經常用於經常性問題的解決方案，其中解決方案具有反生產力、無效或比替代解決方案更有效。

應用程式控制

一種安全方法，僅允許使用核准的應用程式，以協助保護系統免受惡意軟體攻擊。

應用程式組合

有關組織使用的每個應用程式的詳細資訊的集合，包括建置和維護應用程式的成本及其商業價值。此資訊是產品組合探索和分析程序的關鍵，有助於識別要遷移、現代化和優化的應用程式並排定其優先順序。

人工智慧 (AI)

電腦科學領域，致力於使用運算技術來執行通常與人類相關的認知功能，例如學習、解決問題和識別模式。如需詳細資訊，請參閱什麼是人工智慧？

人工智慧操作 (AIOps)

使用機器學習技術解決操作問題、減少操作事件和人工干預以及提高服務品質的程序。如需有關如何在 AWS 遷移策略中使用 AIOps 的詳細資訊，請參閱操作整合指南。

非對稱加密

一種加密演算法，它使用一對金鑰：一個用於加密的公有金鑰和一個用於解密的私有金鑰。您可以共用公有金鑰，因為它不用於解密，但對私有金鑰存取應受到高度限制。

原子性、一致性、隔離性、耐久性 (ACID)

一組軟體屬性，即使在出現錯誤、電源故障或其他問題的情況下，也能確保資料庫的資料有效性和操作可靠性。

屬性型存取控制 (ABAC)

根據使用者屬性 (例如部門、工作職責和團隊名稱) 建立精細許可的實務。如需詳細資訊，請參閱《 AWS Identity and Access Management (IAM) 文件》中的 ABAC for AWS。

授權資料來源

存放主要版本資料的位置，被視為最可靠的資訊來源。您可以將授權資料來源中的資料複製到其他位置，以處理或修改資料，例如匿名、修訂或假名化資料。

可用區域

中的不同位置 AWS 區域 ，可隔離其他可用區域中的故障，並提供相同區域中其他可用區域的低成本、低延遲網路連線能力。

AWS 雲端採用架構 (AWS CAF)

的指導方針和最佳實務架構 AWS ，可協助組織制定高效且有效的計劃，以成功地移至雲端。 AWS CAF 將指導方針組織到六個重點領域：業務、人員、治理、平台、安全和營運。業務、人員和控管層面著重於業務技能和程序；平台、安全和操作層面著重於技術技能和程序。例如，人員層面針對處理人力資源 (HR)、人員配備功能和人員管理的利害關係人。因此， AWS CAF 為人員開發、訓練和通訊提供指引，協助組織做好成功採用雲端的準備。如需詳細資訊，請參閱 AWS CAF 網站和 AWS CAF 白皮書。

AWS 工作負載資格架構 (AWS WQF)

評估資料庫遷移工作負載、建議遷移策略並提供工作預估值的工具。 AWS WQF 隨附於 AWS Schema Conversion Tool (AWS SCT)。它會分析資料庫結構描述和程式碼物件、應用程式程式碼、相依性和效能特性，並提供評估報告。

B

錯誤的機器人

旨在中斷或傷害個人或組織的機器人。

BCP

請參閱業務持續性規劃。

行為圖

資源行為的統一互動式檢視，以及一段時間後的互動。您可以將行為圖與 Amazon Detective 搭配使用來檢查失敗的登入嘗試、可疑的 API 呼叫和類似動作。如需詳細資訊，請參閱偵測文件中的行為圖中的資料。

大端序系統

首先儲存最高有效位元組的系統。另請參閱 Endianness。

二進制分類

預測二進制結果的過程 (兩個可能的類別之一)。例如，ML 模型可能需要預測諸如「此電子郵件是否是垃圾郵件？」等問題 或「產品是書還是汽車？」

Bloom 篩選條件

一種機率性、記憶體高效的資料結構，用於測試元素是否為集的成員。

藍/綠部署

一種部署策略，您可以在其中建立兩個不同但相同的環境。您可以在一個環境 （藍色） 中執行目前的應用程式版本，並在另一個環境 （綠色） 中執行新的應用程式版本。此策略可協助您快速復原，並將影響降至最低。

機器人

透過網際網路執行自動化任務並模擬人類活動或互動的軟體應用程式。有些機器人有用或有益，例如在網際網路上編製資訊索引的 Web 爬蟲程式。某些其他機器人稱為惡意機器人，旨在中斷或傷害個人或組織。

殭屍網路

受到惡意軟體感染且受單一方控制之機器人的網路，稱為機器人繼承器或機器人運算子。殭屍網路是擴展機器人及其影響的最佳已知機制。

分支

程式碼儲存庫包含的區域。儲存庫中建立的第一個分支是主要分支。您可以從現有分支建立新分支，然後在新分支中開發功能或修正錯誤。您建立用來建立功能的分支通常稱為功能分支。當準備好發佈功能時，可以將功能分支合併回主要分支。如需詳細資訊，請參閱關於分支 (GitHub 文件)。

碎片存取

在特殊情況下，以及透過核准的程序，讓使用者快速取得他們通常無權存取 AWS 帳戶 之 的存取權。如需詳細資訊，請參閱 Well-Architected 指南中的 AWS 實作打破玻璃程序指標。

棕地策略

環境中的現有基礎設施。對系統架構採用棕地策略時，可以根據目前系統和基礎設施的限制來設計架構。如果正在擴展現有基礎設施，則可能會混合棕地和綠地策略。

緩衝快取

儲存最常存取資料的記憶體區域。

業務能力

業務如何創造價值 (例如，銷售、客戶服務或營銷)。業務能力可驅動微服務架構和開發決策。如需詳細資訊，請參閱在 AWS上執行容器化微服務白皮書的圍繞業務能力進行組織部分。

業務連續性規劃 (BCP)

一種解決破壞性事件 (如大規模遷移) 對營運的潛在影響並使業務能夠快速恢復營運的計畫。

C

CAF

請參閱AWS 雲端採用架構。

Canary 部署

版本對最終使用者的緩慢和增量版本。當您有信心時，您可以部署新版本並完全取代目前的版本。

CCoE

請參閱 Cloud Center of Excellence。

CDC

請參閱變更資料擷取。

變更資料擷取 (CDC)

追蹤對資料來源 (例如資料庫表格) 的變更並記錄有關變更的中繼資料的程序。您可以將 CDC 用於各種用途，例如稽核或複寫目標系統中的變更以保持同步。

混沌工程

故意引入故障或破壞性事件，以測試系統的彈性。您可以使用 AWS Fault Injection Service (AWS FIS) 執行實驗，為您的 AWS 工作負載帶來壓力，並評估其回應。

CI/CD

請參閱持續整合和持續交付。

分類

有助於產生預測的分類程序。用於分類問題的 ML 模型可預測離散值。離散值永遠彼此不同。例如，模型可能需要評估影像中是否有汽車。

用戶端加密

在目標 AWS 服務 接收資料之前，在本機加密資料。

雲端卓越中心 (CCoE)

一個多學科團隊，可推動整個組織的雲端採用工作，包括開發雲端最佳實務、調動資源、制定遷移時間表以及領導組織進行大規模轉型。如需詳細資訊，請參閱 AWS 雲端 企業策略部落格上的 CCoE 文章。

雲端運算

通常用於遠端資料儲存和 IoT 裝置管理的雲端技術。雲端運算通常連接到邊緣運算技術。

雲端操作模型

在 IT 組織中，用於建置、成熟和最佳化一或多個雲端環境的操作模型。如需詳細資訊，請參閱建置您的雲端操作模型。

採用雲端階段

組織在遷移至 時通常會經歷的四個階段 AWS 雲端：

- 專案 – 執行一些與雲端相關的專案以進行概念驗證和學習用途
- 基礎 – 進行基礎投資以擴展雲端採用 (例如，建立登陸區域、定義 CCoE、建立營運模型)

- 遷移 – 遷移個別應用程式
- 重塑 – 優化產品和服務，並在雲端中創新

這些階段由 Stephen Orban 在部落格文章 The Journey Toward Cloud-First 和企業策略部落格上的採用階段中定義。 AWS 雲端 如需有關它們如何與 AWS 遷移策略相關的詳細資訊，請參閱遷移整備指南。

CMDB

請參閱組態管理資料庫。

程式碼儲存庫

透過版本控制程序來儲存及更新原始程式碼和其他資產 (例如文件、範例和指令碼) 的位置。常見的雲端儲存庫包括 GitHub或 Bitbucket Cloud。程式碼的每個版本都稱為分支。在微服務結構中，每個儲存庫都專用於單個功能。單一 CI/CD 管道可以使用多個儲存庫。

冷快取

一種緩衝快取，它是空的、未填充的，或者包含過時或不相關的資料。這會影響效能，因為資料庫執行個體必須從主記憶體或磁碟讀取，這比從緩衝快取讀取更慢。

冷資料

很少存取且通常是歷史資料的資料。查詢這類資料時，通常可接受慢查詢。將此資料移至效能較低且成本較低的儲存層或類別，可以降低成本。

電腦視覺 (CV)

AI 欄位???，使用機器學習從數位影像和影片等視覺化格式分析和擷取資訊。例如，Amazon SageMaker AI 提供 CV 的影像處理演算法。

組態偏離

對於工作負載，組態會從預期狀態變更。這可能會導致工作負載不合規，而且通常是漸進和無意的。

組態管理資料庫 (CMDB)

儲存和管理有關資料庫及其 IT 環境的資訊的儲存庫，同時包括硬體和軟體元件及其組態。您通常在遷移的產品組合探索和分析階段使用 CMDB 中的資料。

一致性套件

您可以組合的 AWS Config 規則和修補動作集合，以自訂您的合規和安全檢查。您可以使用 YAML 範本，將一致性套件部署為 AWS 帳戶 和 區域中或整個組織的單一實體。如需詳細資訊，請參閱 AWS Config 文件中的一致性套件。

持續整合和持續交付 (CI/CD)

自動化軟體發行程序的來源、建置、測試、暫存和生產階段的程序。CI/CD 通常被描述為管道。CI/CD 可協助您將程序自動化、提升生產力、改善程式碼品質以及加快交付速度。如需詳細資訊，請參閱持續交付的優點。CD 也可表示持續部署。如需詳細資訊，請參閱持續交付與持續部署。

CV

請參閱電腦視覺。

D

靜態資料

網路中靜止的資料，例如儲存中的資料。

資料分類

根據重要性和敏感性來識別和分類網路資料的程序。它是所有網路安全風險管理策略的關鍵組成部分，因為它可以協助您確定適當的資料保護和保留控制。資料分類是 AWS Well-Architected Framework 中安全支柱的元件。如需詳細資訊，請參閱資料分類。

資料偏離

生產資料與用於訓練 ML 模型的資料之間有意義的變化，或輸入資料隨時間有意義的變更。資料偏離可以降低 ML 模型預測的整體品質、準確性和公平性。

傳輸中的資料

在您的網路中主動移動的資料，例如在網路資源之間移動。

資料網格

架構架構，提供分散式、分散式資料擁有權與集中式管理。

資料最小化

僅收集和處理嚴格必要資料的原則。在 中實作資料最小化 AWS 雲端 可以降低隱私權風險、成本和分析碳足跡。

資料周邊

AWS 環境中的一組預防性防護機制，可協助確保只有信任的身分才能從預期的網路存取信任的資源。如需詳細資訊，請參閱在 上建置資料周邊 AWS。

資料預先處理

將原始資料轉換成 ML 模型可輕鬆剖析的格式。預處理資料可能意味著移除某些欄或列，並解決遺失、不一致或重複的值。

資料來源

在整個生命週期中追蹤資料的原始伺服器和歷史記錄的程序，例如資料的產生、傳輸和儲存方式。

資料主體

正在收集和處理資料的個人。

資料倉儲

支援商業智慧的資料管理系統，例如 分析。資料倉儲通常包含大量歷史資料，通常用於查詢和分析。

資料庫定義語言 (DDL)

用於建立或修改資料庫中資料表和物件之結構的陳述式或命令。

資料庫處理語言 (DML)

用於修改 (插入、更新和刪除) 資料庫中資訊的陳述式或命令。

DDL

請參閱資料庫定義語言。

深度整體

結合多個深度學習模型進行預測。可以使用深度整體來獲得更準確的預測或估計預測中的不確定性。

深度學習

一個機器學習子領域，它使用多層人工神經網路來識別感興趣的輸入資料與目標變數之間的對應關係。

深度防禦

這是一種資訊安全方法，其中一系列的安全機制和控制項會在整個電腦網路中精心分層，以保護網路和其中資料的機密性、完整性和可用性。當您在 上採用此策略時 AWS，您可以在 AWS Organizations 結構的不同層新增多個控制項，以協助保護資源。例如，defense-in-depth方法可能會結合多重要素驗證、網路分割和加密。

委派的管理員

在 中 AWS Organizations，相容的服務可以註冊 AWS 成員帳戶，以管理組織的帳戶和管理該服務的許可。此帳戶稱為該服務的委派管理員。如需詳細資訊和相容服務清單，請參閱 AWS Organizations 文件中的可搭配 AWS Organizations運作的服務。

deployment

在目標環境中提供應用程式、新功能或程式碼修正的程序。部署涉及在程式碼庫中實作變更，然後在應用程式環境中建置和執行該程式碼庫。

開發環境

請參閱 環境。

偵測性控制

一種安全控制，用於在事件發生後偵測、記錄和提醒。這些控制是第二道防線，提醒您注意繞過現有預防性控制的安全事件。如需詳細資訊，請參閱在 AWS上實作安全控制中的偵測性控制。

開發值串流映射 (DVSM)

一種程序，用於識別對軟體開發生命週期中的速度和品質造成負面影響的限制並排定優先順序。DVSM 擴展了原本專為精簡製造實務設計的價值串流映射程序。它著重於透過軟體開發程序建立和移動價值所需的步驟和團隊。

數位分身

真實世界系統的虛擬呈現，例如建築物、工廠、工業設備或生產線。數位分身支援預測性維護、遠端監控和生產最佳化。

維度資料表

在星星結構描述中，較小的資料表包含有關事實資料表中量化資料的資料屬性。維度資料表屬性通常是文字欄位或離散數字，其行為類似於文字。這些屬性通常用於查詢限制、篩選和結果集標記。

災難

防止工作負載或系統在其主要部署位置中實現其業務目標的事件。這些事件可能是自然災難、技術故障或人為動作的結果，例如意外設定錯誤或惡意軟體攻擊。

災難復原 (DR)

您用來將災難造成的停機時間和資料遺失降至最低的策略和程序。如需詳細資訊，請參閱 AWS Well-Architected Framework 中的 上工作負載災難復原 AWS：雲端中的復原。

DML

請參閱資料庫處理語言。

領域驅動的設計

一種開發複雜軟體系統的方法，它會將其元件與每個元件所服務的不斷發展的領域或核心業務目標相關聯。Eric Evans 在其著作 Domain-Driven Design: Tackling Complexity in the Heart of Software (Boston: Addison-Wesley Professional, 2003) 中介紹了這一概念。如需有關如何將領域驅動的設計與 strangler fig 模式搭配使用的資訊，請參閱使用容器和 Amazon API Gateway 逐步現代化舊版 Microsoft ASP.NET (ASMX) Web 服務。

DR

請參閱災難復原。

偏離偵測

追蹤與基準組態的偏差。例如，您可以使用 AWS CloudFormation 來偵測系統資源中的偏離，也可以使用 AWS Control Tower 來偵測登陸區域中可能影響控管要求合規性的變更。 https://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/using-cfn-stack-drift.html

DVSM

請參閱開發值串流映射。

E

EDA

請參閱探索性資料分析。

EDI

請參閱電子資料交換。

邊緣運算

提升 IoT 網路邊緣智慧型裝置運算能力的技術。與雲端運算相比，邊緣運算可以減少通訊延遲並改善回應時間。

電子資料交換 (EDI)

在組織之間自動交換商業文件。如需詳細資訊，請參閱什麼是電子資料交換。

加密

一種運算程序，可將人類可讀取的純文字資料轉換為加密文字。

加密金鑰

由加密演算法產生的隨機位元的加密字串。金鑰長度可能有所不同，每個金鑰的設計都是不可預測且唯一的。

端序

位元組在電腦記憶體中的儲存順序。大端序系統首先儲存最高有效位元組。小端序系統首先儲存最低有效位元組。

端點

請參閱 服務端點。

端點服務

您可以在虛擬私有雲端 (VPC) 中託管以與其他使用者共用的服務。您可以使用 建立端點服務，AWS PrivateLink 並將許可授予其他 AWS 帳戶 或 AWS Identity and Access Management (IAM) 委託人。這些帳戶或主體可以透過建立介面 VPC 端點私下連接至您的端點服務。如需詳細資訊，請參閱 Amazon Virtual Private Cloud (Amazon VPC) 文件中的建立端點服務。

企業資源規劃 (ERP)

一種系統，可自動化和管理企業的關鍵業務流程 （例如會計、MES 和專案管理）。

信封加密

使用另一個加密金鑰對某個加密金鑰進行加密的程序。如需詳細資訊，請參閱 AWS Key Management Service (AWS KMS) 文件中的信封加密。

環境

執行中應用程式的執行個體。以下是雲端運算中常見的環境類型：

- 開發環境 – 執行中應用程式的執行個體，只有負責維護應用程式的核心團隊才能使用。開發環境用來測試變更，然後再將開發環境提升到較高的環境。此類型的環境有時稱為測試環境。
- 較低的環境 – 應用程式的所有開發環境，例如用於初始建置和測試的開發環境。
- 生產環境 – 最終使用者可以存取的執行中應用程式的執行個體。在 CI/CD 管道中，生產環境是最後一個部署環境。
- 較高的環境 – 核心開發團隊以外的使用者可存取的所有環境。這可能包括生產環境、生產前環境以及用於使用者接受度測試的環境。

epic

在敏捷方法中，有助於組織工作並排定工作優先順序的功能類別。epic 提供要求和實作任務的高層級描述。例如， AWS CAF 安全概念包括身分和存取管理、偵測控制、基礎設施安全、資料保護和事件回應。如需有關 AWS 遷移策略中的 Epic 的詳細資訊，請參閱計畫實作指南。

ERP

請參閱企業資源規劃。

探索性資料分析 (EDA)

分析資料集以了解其主要特性的過程。您收集或彙總資料，然後執行初步調查以尋找模式、偵測異常並檢查假設。透過計算摘要統計並建立資料可視化來執行 EDA。

F

事實資料表

星狀結構描述中的中央資料表。它存放有關業務操作的量化資料。一般而言，事實資料表包含兩種類型的資料欄：包含度量的資料，以及包含維度資料表外部索引鍵的資料欄。

快速失敗

一種使用頻繁和增量測試來縮短開發生命週期的理念。這是敏捷方法的關鍵部分。

故障隔離界限

在 中 AWS 雲端，像是可用區域 AWS 區域、控制平面或資料平面等邊界會限制故障的影響，並有助於改善工作負載的彈性。如需詳細資訊，請參閱AWS 故障隔離界限。

功能分支

請參閱分支。

特徵

用來進行預測的輸入資料。例如，在製造環境中，特徵可能是定期從製造生產線擷取的影像。

功能重要性

特徵對於模型的預測有多重要。這通常表示為可以透過各種技術來計算的數值得分，例如 Shapley Additive Explanations (SHAP) 和積分梯度。如需詳細資訊，請參閱 的機器學習模型可解譯性 AWS。

特徵轉換

優化 ML 程序的資料，包括使用其他來源豐富資料、調整值、或從單一資料欄位擷取多組資訊。這可讓 ML 模型從資料中受益。例如，如果將「2021-05-27 00:15:37」日期劃分為「2021」、「五月」、「週四」和「15」，則可以協助學習演算法學習與不同資料元件相關聯的細微模式。

少量擷取提示

在要求 LLM 執行類似的任務之前，提供少量示範任務和所需輸出的範例。此技術是內容內學習的應用程式，其中模型會從內嵌在提示中的範例 (快照) 中學習。對於需要特定格式、推理或網域知識的任務，少量的提示可以有效。另請參閱零鏡頭提示。

FGAC

請參閱精細存取控制。

精細存取控制 (FGAC)

使用多個條件來允許或拒絕存取請求。

閃切遷移

一種資料庫遷移方法，透過變更資料擷取使用連續資料複寫，以盡可能在最短的時間內遷移資料，而不是使用分階段方法。目標是將停機時間降至最低。

FM

請參閱基礎模型。

基礎模型 (FM)

大型深度學習神經網路，已在廣義和未標記資料的大量資料集上進行訓練。FMs 能夠執行各種一般任務，例如了解語言、產生文字和影像，以及以自然語言交談。如需詳細資訊，請參閱什麼是基礎模型。

G

生成式 AI

已針對大量資料進行訓練的 AI 模型子集，可使用簡單的文字提示建立新的內容和成品，例如影像、影片、文字和音訊。如需詳細資訊，請參閱什麼是生成式 AI。

地理封鎖

請參閱地理限制。

地理限制 (地理封鎖)

Amazon CloudFront 中的選項，可防止特定國家/地區的使用者存取內容分發。您可以使用允許清單或封鎖清單來指定核准和禁止的國家/地區。如需詳細資訊，請參閱 CloudFront 文件中的限制內容的地理分佈。

Gitflow 工作流程

這是一種方法，其中較低和較高環境在原始碼儲存庫中使用不同分支。Gitflow 工作流程被視為舊版，而以幹線為基礎的工作流程是現代、偏好的方法。

黃金影像

系統或軟體的快照，做為部署該系統或軟體新執行個體的範本。例如，在製造中，黃金映像可用於在多個裝置上佈建軟體，並有助於提高裝置製造操作的速度、可擴展性和生產力。

綠地策略

新環境中缺乏現有基礎設施。對系統架構採用綠地策略時，可以選擇所有新技術，而不會限制與現有基礎設施的相容性，也稱為棕地。如果正在擴展現有基礎設施，則可能會混合棕地和綠地策略。

防護機制

有助於跨組織單位 (OU) 來管控資源、政策和合規的高層級規則。預防性防護機制會強制執行政策，以確保符合合規標準。透過使用服務控制政策和 IAM 許可界限來將其實作。偵測性防護機制可偵測政策違規和合規問題，並產生提醒以便修正。它們是透過使用 AWS Config AWS Security Hub CSPM、Amazon GuardDuty、、Amazon Inspector AWS Trusted Advisor和自訂 AWS Lambda 檢查來實作。

H

HA

請參閱高可用性。

異質資料庫遷移

將來源資料庫遷移至使用不同資料庫引擎的目標資料庫 (例如，Oracle 至 Amazon Aurora)。異質遷移通常是重新架構工作的一部分，而轉換結構描述可能是一項複雜任務。AWS 提供有助於結構描述轉換的 AWS SCT。

高可用性 (HA)

在遇到挑戰或災難時，工作負載能夠在不介入的情況下持續運作。HA 系統的設計目的是自動容錯移轉、持續提供高品質的效能，以及處理不同的負載和故障，並將效能影響降至最低。

歷史現代化

一種方法，用於現代化和升級操作技術 (OT) 系統，以更好地滿足製造業的需求。歷史資料是一種資料庫，用於從工廠中的各種來源收集和存放資料。

保留資料

從用於訓練機器學習模型的資料集中保留的部分歷史標記資料。您可以使用保留資料，透過比較模型預測與保留資料來評估模型效能。

異質資料庫遷移

將您的來源資料庫遷移至共用相同資料庫引擎的目標資料庫 (例如，Microsoft SQL Server 至 Amazon RDS for SQL Server)。同質遷移通常是主機轉換或平台轉換工作的一部分。您可以使用原生資料庫公用程式來遷移結構描述。

熱資料

經常存取的資料，例如即時資料或最近的轉譯資料。此資料通常需要高效能儲存層或類別，才能提供快速的查詢回應。

修補程序

緊急修正生產環境中的關鍵問題。由於其緊迫性，通常會在典型 DevOps 發行工作流程之外執行修補程式。

超級護理期間

在切換後，遷移團隊在雲端管理和監控遷移的應用程式以解決任何問題的時段。通常，此期間的長度為 1-4 天。在超級護理期間結束時，遷移團隊通常會將應用程式的責任轉移給雲端營運團隊。

I

IaC

請參閱基礎設施即程式碼。

身分型政策

連接至一或多個 IAM 主體的政策，可定義其在 AWS 雲端 環境中的許可。

閒置應用程式

90 天期間 CPU 和記憶體平均使用率在 5% 至 20% 之間的應用程式。在遷移專案中，通常會淘汰這些應用程式或將其保留在內部部署。

IIoT

請參閱工業物聯網。

不可變的基礎設施

為生產工作負載部署新基礎設施的模型，而不是更新、修補或修改現有的基礎設施。不可變基礎設施本質上比可變基礎設施更一致、可靠且可預測。如需詳細資訊，請參閱 AWS Well-Architected Framework 中的使用不可變基礎設施部署最佳實務。

傳入 (輸入) VPC

在 AWS 多帳戶架構中，接受、檢查和路由來自應用程式外部之網路連線的 VPC。AWS 安全參考架構建議您使用傳入、傳出和檢查 VPC 來設定網路帳戶，以保護應用程式與更廣泛的網際網路之間的雙向介面。

增量遷移

一種切換策略，您可以在其中將應用程式分成小部分遷移，而不是執行單一、完整的切換。例如，您最初可能只將一些微服務或使用者移至新系統。確認所有項目都正常運作之後，您可以逐步移動其他微服務或使用者，直到可以解除委任舊式系統。此策略可降低與大型遷移關聯的風險。

工業 4.0

2016 年 Klaus Schwab 推出的術語，透過連線能力、即時資料、自動化、分析和 AI/ML 的進展，指製造程序的現代化。

基礎設施

應用程式環境中包含的所有資源和資產。

基礎設施即程式碼 (IaC)

透過一組組態檔案來佈建和管理應用程式基礎設施的程序。IaC 旨在協助您集中管理基礎設施，標準化資源並快速擴展，以便新環境可重複、可靠且一致。

工業物聯網 (IIoT)

在製造業、能源、汽車、醫療保健、生命科學和農業等產業領域使用網際網路連線的感測器和裝置。如需詳細資訊，請參閱建立工業物聯網 (IIoT) 數位轉型策略。

檢查 VPC

在 AWS 多帳戶架構中，集中式 VPC 可管理 VPCs 之間 （在相同或不同的 中 AWS 區域)、網際網路和內部部署網路之間的網路流量檢查。AWS 安全參考架構建議您使用傳入、傳出和檢查 VPC 來設定網路帳戶，以保護應用程式與更廣泛的網際網路之間的雙向介面。

物聯網 (IoT)

具有內嵌式感測器或處理器的相連實體物體網路，其透過網際網路或本地通訊網路與其他裝置和系統進行通訊。如需詳細資訊，請參閱什麼是 IoT？

可解釋性

機器學習模型的一個特徵，描述了人類能夠理解模型的預測如何依賴於其輸入的程度。如需詳細資訊，請參閱 的機器學習模型可解譯性 AWS。

IoT

請參閱物聯網。

IT 資訊庫 (ITIL)

一組用於交付 IT 服務並使這些服務與業務需求保持一致的最佳實務。ITIL 為 ITSM 提供了基礎。

IT 服務管理 (ITSM)

與組織的設計、實作、管理和支援 IT 服務關聯的活動。如需有關將雲端操作與 ITSM 工具整合的資訊，請參閱操作整合指南。

ITIL

請參閱 IT 資訊庫。

ITSM

請參閱 IT 服務管理。

L

標籤型存取控制 (LBAC)

強制存取控制 (MAC) 的實作，其中使用者和資料本身都會獲得明確指派的安全標籤值。使用者安全標籤和資料安全標籤之間的交集會決定使用者可以看到哪些資料列和資料欄。

登陸區域

登陸區域是架構良好的多帳戶 AWS 環境，可擴展且安全。這是一個起點，您的組織可以從此起點快速啟動和部署工作負載與應用程式，並對其安全和基礎設施環境充滿信心。如需有關登陸區域的詳細資訊，請參閱設定安全且可擴展的多帳戶 AWS 環境。

大型語言模型 (LLM)

預先訓練大量資料的深度學習 AI 模型。LLM 可以執行多個任務，例如回答問題、摘要文件、將文字翻譯成其他語言，以及完成句子。如需詳細資訊，請參閱什麼是 LLMs。

大型遷移

遷移 300 部或更多伺服器。

LBAC

請參閱標籤型存取控制。

最低權限

授予執行任務所需之最低許可的安全最佳實務。如需詳細資訊，請參閱 IAM 文件中的套用最低權限許可。

隨即轉移

請參閱 7 個 R。

小端序系統

首先儲存最低有效位元組的系統。另請參閱 Endianness。

LLM

請參閱大型語言模型。

較低的環境

請參閱 環境。

M

機器學習 (ML)

一種使用演算法和技術進行模式識別和學習的人工智慧。機器學習會進行分析並從記錄的資料 (例如物聯網 (IoT) 資料) 中學習，以根據模式產生統計模型。如需詳細資訊，請參閱機器學習。

主要分支

請參閱分支。

惡意軟體

旨在危及電腦安全或隱私權的軟體。惡意軟體可能會中斷電腦系統、洩露敏感資訊，或取得未經授權的存取。惡意軟體的範例包括病毒、蠕蟲、勒索軟體、特洛伊木馬程式、間諜軟體和鍵盤記錄器。

受管服務

AWS 服務 會 AWS 操作基礎設施層、作業系統和平台，而您會存取端點來存放和擷取資料。Amazon Simple Storage Service (Amazon S3) 和 Amazon DynamoDB 是受管服務的範例。這些也稱為抽象服務。

製造執行系統 (MES)

一種軟體系統，用於追蹤、監控、記錄和控制生產程序，將原物料轉換為現場成品。

MAP

請參閱遷移加速計劃。

機制

建立工具、推動工具採用，然後檢查結果以進行調整的完整程序。機制是在操作時強化和改善自身的循環。如需詳細資訊，請參閱 AWS Well-Architected Framework 中的建置機制。

成員帳戶

除了屬於組織一部分的管理帳戶 AWS 帳戶 之外的所有 AWS Organizations。帳戶一次只能是一個組織的成員。

製造執行系統

請參閱製造執行系統。

訊息佇列遙測傳輸 (MQTT)

根據發佈/訂閱模式的輕量型machine-to-machine(M2M) 通訊協定，適用於資源受限的 IoT 裝置。

微服務

一種小型的獨立服務，它可透過定義明確的 API 進行通訊，通常由小型獨立團隊擁有。例如，保險系統可能包含對應至業務能力 (例如銷售或行銷) 或子領域 (例如購買、索賠或分析) 的微服務。微服務的優點包括靈活性、彈性擴展、輕鬆部署、可重複使用的程式碼和適應力。如需詳細資訊，請參閱使用無 AWS 伺服器服務整合微服務。

微服務架構

一種使用獨立元件來建置應用程式的方法，這些元件會以微服務形式執行每個應用程式程序。這些微服務會使用輕量型 API，透過明確定義的介面進行通訊。此架構中的每個微服務都可以進行更新、部署和擴展，以滿足應用程式特定功能的需求。如需詳細資訊，請參閱在 上實作微服務 AWS。

Migration Acceleration Program (MAP)

一種 AWS 計畫，提供諮詢支援、訓練和服務，協助組織建立強大的營運基礎，以移至雲端，並協助抵銷遷移的初始成本。MAP 包括用於有條不紊地執行舊式遷移的遷移方法以及一組用於自動化和加速常見遷移案例的工具。

大規模遷移

將大部分應用程式組合依波次移至雲端的程序，在每個波次中，都會以更快的速度移動更多應用程式。此階段使用從早期階段學到的最佳實務和經驗教訓來實作團隊、工具和流程的遷移工廠，以透過自動化和敏捷交付簡化工作負載的遷移。這是 AWS 遷移策略的第三階段。

遷移工廠

可透過自動化、敏捷的方法簡化工作負載遷移的跨職能團隊。遷移工廠團隊通常包括營運、業務分析師和擁有者、遷移工程師、開發人員以及從事 Sprint 工作的 DevOps 專業人員。20% 至 50% 之間的企業應用程式組合包含可透過工廠方法優化的重複模式。如需詳細資訊，請參閱此內容集中的遷移工廠的討論和雲端遷移工廠指南。

遷移中繼資料

有關完成遷移所需的應用程式和伺服器的資訊。每種遷移模式都需要一組不同的遷移中繼資料。遷移中繼資料的範例包括目標子網路、安全群組和 AWS 帳戶。

遷移模式

可重複的遷移任務，詳細描述遷移策略、遷移目的地以及所使用的遷移應用程式或服務。範例：使用 AWS Application Migration Service 重新託管遷移至 Amazon EC2。

遷移組合評定 (MPA)

線上工具，提供驗證商業案例以遷移至 的資訊 AWS 雲端。MPA 提供詳細的組合評定 (伺服器適當規模、定價、總體擁有成本比較、遷移成本分析) 以及遷移規劃 (應用程式資料分析和資料收集、應用程式分組、遷移優先順序，以及波次規劃)。MPA 工具 （需要登入） 可供所有 AWS 顧問和 APN 合作夥伴顧問免費使用。

遷移準備程度評定 (MRA)

使用 AWS CAF 取得組織雲端整備狀態的洞見、識別優缺點，以及建立行動計劃以消除已識別差距的程序。如需詳細資訊，請參閱遷移準備程度指南。MRA 是 AWS 遷移策略的第一階段。

遷移策略

用來將工作負載遷移至 的方法 AWS 雲端。如需詳細資訊，請參閱此詞彙表中的 7 個 Rs 項目，並請參閱調動您的組織以加速大規模遷移。

機器學習 (ML)

請參閱機器學習。

現代化

將過時的 (舊版或單一) 應用程式及其基礎架構轉換為雲端中靈活、富有彈性且高度可用的系統，以降低成本、提高效率並充分利用創新。如需詳細資訊，請參閱《》中的現代化應用程式的策略 AWS 雲端。

現代化準備程度評定

這項評估可協助判斷組織應用程式的現代化準備程度；識別優點、風險和相依性；並確定組織能夠在多大程度上支援這些應用程式的未來狀態。評定的結果就是目標架構的藍圖、詳細說明現代化程序的開發階段和里程碑的路線圖、以及解決已發現的差距之行動計畫。如需詳細資訊，請參閱《》中的評估應用程式的現代化準備 AWS 雲端程度。

單一應用程式 (單一)

透過緊密結合的程序作為單一服務執行的應用程式。單一應用程式有幾個缺點。如果一個應用程式功能遇到需求激增，則必須擴展整個架構。當程式碼庫增長時，新增或改進單一應用程式的功能也會變得更加複雜。若要解決這些問題，可以使用微服務架構。如需詳細資訊，請參閱將單一體系分解為微服務。

MPA

請參閱遷移產品組合評估。

MQTT

請參閱訊息佇列遙測傳輸。

多類別分類

一個有助於產生多類別預測的過程 (預測兩個以上的結果之一)。例如，機器學習模型可能會詢問「此產品是書籍、汽車還是電話？」 或者「這個客戶對哪種產品類別最感興趣？」

可變基礎設施

更新和修改生產工作負載現有基礎設施的模型。為了提高一致性、可靠性和可預測性， AWS Well-Architected Framework 建議使用不可變基礎設施做為最佳實務。

O

OAC

請參閱原始存取控制。

OAI

請參閱原始存取身分。

OCM

請參閱組織變更管理。

離線遷移

一種遷移方法，可在遷移過程中刪除來源工作負載。此方法涉及延長停機時間，通常用於小型非關鍵工作負載。

OI

請參閱 操作整合。

OLA

請參閱操作層級協議。

線上遷移

一種遷移方法，無需離線即可將來源工作負載複製到目標系統。連接至工作負載的應用程式可在遷移期間繼續運作。此方法涉及零至最短停機時間，通常用於關鍵的生產工作負載。

OPC-UA

請參閱開放程序通訊 - 統一架構。

開放程序通訊 - 統一架構 (OPC-UA)

用於工業自動化的machine-to-machine(M2M) 通訊協定。OPC-UA 提供資料加密、身分驗證和授權機制的互通性標準。

操作水準協議 (OLA)

一份協議，闡明 IT 職能群組承諾向彼此提供的內容，以支援服務水準協議 (SLA)。

操作整備審查 (ORR)

問題和相關最佳實務的檢查清單，可協助您了解、評估、預防或減少事件和可能失敗的範圍。如需詳細資訊，請參閱 AWS Well-Architected Framework 中的操作整備審查 (ORR)。

操作技術 (OT)

使用實體環境控制工業操作、設備和基礎設施的硬體和軟體系統。在製造中，整合 OT 和資訊技術 (IT) 系統是工業 4.0 轉型的關鍵重點。

操作整合 (OI)

在雲端中將操作現代化的程序，其中包括準備程度規劃、自動化和整合。如需詳細資訊，請參閱操作整合指南。

組織追蹤

由 建立的線索 AWS CloudTrail ，會記錄 AWS 帳戶 組織中所有 的所有事件 AWS Organizations。在屬於組織的每個 AWS 帳戶 中建立此追蹤，它會跟蹤每個帳戶中的活動。如需詳細資訊，請參閱 CloudTrail 文件中的建立組織追蹤。

組織變更管理 (OCM)

用於從人員、文化和領導力層面管理重大、顛覆性業務轉型的架構。OCM 透過加速變更採用、解決過渡問題，以及推動文化和組織變更，協助組織為新系統和策略做好準備，並轉移至新系統和策略。在 AWS 遷移策略中，此架構稱為人員加速，因為雲端採用專案所需的變更速度。如需詳細資訊，請參閱 OCM 指南。

原始存取控制 (OAC)

CloudFront 中的增強型選項，用於限制存取以保護 Amazon Simple Storage Service (Amazon S3) 內容。OAC 支援所有 S3 儲存貯體中的所有伺服器端加密 AWS KMS (SSE-KMS) AWS 區域，以及對 S3 儲存貯體的動態PUT和DELETE請求。

原始存取身分 (OAI)

CloudFront 中的一個選項，用於限制存取以保護 Amazon S3 內容。當您使用 OAI 時，CloudFront 會建立一個可供 Amazon S3 進行驗證的主體。經驗證的主體只能透過特定 CloudFront 分發來存取 S3 儲存貯體中的內容。另請參閱 OAC，它可提供更精細且增強的存取控制。

ORR

請參閱操作整備審核。

OT

請參閱操作技術。

傳出 (輸出) VPC

在 AWS 多帳戶架構中，處理從應用程式內啟動之網路連線的 VPC。AWS 安全參考架構建議您使用傳入、傳出和檢查 VPC 來設定網路帳戶，以保護應用程式與更廣泛的網際網路之間的雙向介面。

P

許可界限

附接至 IAM 主體的 IAM 管理政策，可設定使用者或角色擁有的最大許可。如需詳細資訊，請參閱 IAM 文件中的許可界限。

個人身分識別資訊 (PII)

當直接檢視或與其他相關資料配對時，可用來合理推斷個人身分的資訊。PII 的範例包括名稱、地址和聯絡資訊。

PII

請參閱個人身分識別資訊。

手冊

一組預先定義的步驟，可擷取與遷移關聯的工作，例如在雲端中提供核心操作功能。手冊可以採用指令碼、自動化執行手冊或操作現代化環境所需的程序或步驟摘要的形式。

PLC

請參閱可程式設計邏輯控制器。

PLM

請參閱產品生命週期管理。

政策

可定義許可的物件（請參閱身分型政策）、指定存取條件（請參閱資源型政策），或定義組織中所有帳戶的最大許可 AWS Organizations（請參閱服務控制政策）。

混合持久性

根據資料存取模式和其他需求，獨立選擇微服務的資料儲存技術。如果您的微服務具有相同的資料儲存技術，則其可能會遇到實作挑戰或效能不佳。如果微服務使用最適合其需求的資料儲存，則可以更輕鬆地實作並達到更好的效能和可擴展性。

組合評定

探索、分析應用程式組合並排定其優先順序以規劃遷移的程序。如需詳細資訊，請參閱評估遷移準備程度。

述詞

傳回 `true`或 的查詢條件`false`，通常位於 `WHERE`子句中。

述詞下推

一種資料庫查詢最佳化技術，可在傳輸前篩選查詢中的資料。這可減少必須從關聯式資料庫擷取和處理的資料量，並改善查詢效能。

預防性控制

旨在防止事件發生的安全控制。這些控制是第一道防線，可協助防止對網路的未經授權存取或不必要變更。如需詳細資訊，請參閱在 AWS上實作安全控制中的預防性控制。

委託人

中可執行動作和存取資源 AWS 的實體。此實體通常是 AWS 帳戶、IAM 角色或使用者的根使用者。如需詳細資訊，請參閱 IAM 文件中角色術語和概念中的主體。

依設計的隱私權

透過整個開發程序將隱私權納入考量的系統工程方法。

私有託管區域

一種容器，它包含有關您希望 Amazon Route 53 如何回應一個或多個 VPC 內的域及其子域之 DNS 查詢的資訊。如需詳細資訊，請參閱 Route 53 文件中的使用私有託管區域。

主動控制

旨在防止部署不合規資源的安全控制。這些控制項會在佈建資源之前對其進行掃描。如果資源不符合控制項，則不會佈建。如需詳細資訊，請參閱 AWS Control Tower 文件中的控制項參考指南，並參閱實作安全控制項中的主動控制項。 AWS

產品生命週期管理 (PLM)

管理產品整個生命週期的資料和程序，從設計、開發和啟動，到成長和成熟，再到拒絕和移除。

生產環境

請參閱 環境。

可程式設計邏輯控制器 (PLC)

在製造中，高度可靠、可調整的電腦，可監控機器並自動化製造程序。

提示鏈結

使用一個 LLM 提示的輸出做為下一個提示的輸入，以產生更好的回應。此技術用於將複雜任務分解為子任務，或反覆精簡或展開初步回應。它有助於提高模型回應的準確性和相關性，並允許更精細、個人化的結果。

擬匿名化

以預留位置值取代資料集中個人識別符的程序。假名化有助於保護個人隱私權。假名化資料仍被視為個人資料。

發佈/訂閱 (pub/sub)

一種模式，可啟用微服務之間的非同步通訊，以提高可擴展性和回應能力。例如，在微服務型 MES 中，微服務可以將事件訊息發佈到其他微服務可訂閱的頻道。系統可以新增新的微服務，而無需變更發佈服務。

Q

查詢計劃

一系列步驟，如指示，用於存取 SQL 關聯式資料庫系統中的資料。

查詢計劃迴歸

在資料庫服務優化工具選擇的計畫比對資料庫環境進行指定的變更之前的計畫不太理想時。這可能因為對統計資料、限制條件、環境設定、查詢參數繫結的變更以及資料庫引擎的更新所導致。

R

RACI 矩陣

請參閱負責、負責、諮詢、告知 (RACI)。

RAG

請參閱擷取增強產生。

勒索軟體

一種惡意軟體，旨在阻止對計算機系統或資料的存取，直到付款為止。

RASCI 矩陣

請參閱負責、負責、諮詢、告知 (RACI)。

RCAC

請參閱資料列和資料欄存取控制。

僅供讀取複本

用於唯讀用途的資料庫複本。您可以將查詢路由至僅供讀取複本以減少主資料庫的負載。

重新架構師

請參閱 7 Rs。

復原點目標 (RPO)

自上次資料復原點以來可接受的時間上限。這會決定最後一個復原點與服務中斷之間可接受的資料遺失。

復原時間目標 (RTO)

服務中斷與服務還原之間的可接受延遲上限。

重構

請參閱 7 Rs。

區域

地理區域中的 AWS 資源集合。每個 AWS 區域 都獨立於其他 ，以提供容錯能力、穩定性和彈性。如需詳細資訊，請參閱指定 AWS 區域 您的帳戶可以使用哪些。

迴歸

預測數值的 ML 技術。例如，為了解決「這房子會賣什麼價格？」的問題 ML 模型可以使用線性迴歸模型，根據已知的房屋事實 (例如，平方英尺) 來預測房屋的銷售價格。

重新託管

請參閱 7 Rs。

版本

在部署程序中，它是將變更提升至生產環境的動作。

重新定位

請參閱 7 個 R。

Replatform

請參閱 7 個 R。

回購

請參閱 7 Rs。

彈性

應用程式抵禦中斷或從中斷中復原的能力。在 中規劃彈性時，高可用性和災難復原是常見的考量 AWS 雲端。如需詳細資訊，請參閱AWS 雲端 彈性。

資源型政策

附接至資源的政策，例如 Amazon S3 儲存貯體、端點或加密金鑰。這種類型的政策會指定允許存取哪些主體、支援的動作以及必須滿足的任何其他條件。

負責者、當責者、事先諮詢者和事後告知者 (RACI) 矩陣

定義所有涉及遷移活動和雲端操作之各方的角色和責任的矩陣。矩陣名稱衍生自矩陣中定義的責任類型：負責人 (R)、責任 (A)、諮詢 (C) 和知情 (I)。支援 (S) 類型為選用。如果您包含支援，則矩陣稱為 RASCI 矩陣，如果您排除它，則稱為 RACI 矩陣。

回應性控制

一種安全控制，旨在驅動不良事件或偏離安全基準的補救措施。如需詳細資訊，請參閱在 AWS上實作安全控制中的回應性控制。

保留

請參閱 7 個 R。

淘汰

請參閱 7 Rs。

檢索增強生成 (RAG)

一種生成式 AI 技術，其中 LLM 會在產生回應之前參考訓練資料來源以外的授權資料來源。例如，RAG 模型可能會對組織的知識庫或自訂資料執行語意搜尋。如需詳細資訊，請參閱什麼是 RAG。

輪換

定期更新秘密的程序，讓攻擊者更難存取登入資料。

資料列和資料欄存取控制 (RCAC)

使用已定義存取規則的基本、彈性 SQL 表達式。RCAC 包含資料列許可和資料欄遮罩。

RPO

請參閱復原點目標。

RTO

請參閱復原時間目標。

執行手冊

執行特定任務所需的一組手動或自動程序。這些通常是為了簡化重複性操作或錯誤率較高的程序而建置。

S

SAML 2.0

許多身分提供者 (IdP) 使用的開放標準。此功能會啟用聯合單一登入 (SSO)，讓使用者可以登入 AWS 管理主控台 或呼叫 AWS API 操作，而不必為您組織中的每個人在 IAM 中建立使用者。如需有關以 SAML 2.0 為基礎的聯合詳細資訊，請參閱 IAM 文件中的關於以 SAML 2.0 為基礎的聯合。

SCADA

請參閱監督控制和資料擷取。

SCP

請參閱服務控制政策。

秘密

您以加密形式存放的 AWS Secrets Manager機密或限制資訊，例如密碼或使用者登入資料。它由秘密值及其中繼資料組成。秘密值可以是二進位、單一字串或多個字串。如需詳細資訊，請參閱 Secrets Manager 文件中的 Secrets Manager 秘密中的什麼內容？。

依設計的安全性

透過整個開發程序將安全性納入考量的系統工程方法。

安全控制

一種技術或管理防護機制，它可預防、偵測或降低威脅行為者利用安全漏洞的能力。安全控制有四種主要類型：預防性、偵測性、回應性和主動性。

安全強化

減少受攻擊面以使其更能抵抗攻擊的過程。這可能包括一些動作，例如移除不再需要的資源、實作授予最低權限的安全最佳實務、或停用組態檔案中不必要的功能。

安全資訊與事件管理 (SIEM) 系統

結合安全資訊管理 (SIM) 和安全事件管理 (SEM) 系統的工具與服務。SIEM 系統會收集、監控和分析來自伺服器、網路、裝置和其他來源的資料，以偵測威脅和安全漏洞，並產生提醒。

安全回應自動化

預先定義和程式設計的動作，旨在自動回應或修復安全事件。這些自動化可做為偵測或回應式安全控制，協助您實作 AWS 安全最佳實務。自動化回應動作的範例包括修改 VPC 安全群組、修補 Amazon EC2 執行個體或輪換登入資料。

伺服器端加密

由接收資料的 AWS 服務 在其目的地加密資料。

服務控制政策 (SCP)

為 AWS Organizations中的組織的所有帳戶提供集中控制許可的政策。SCP 會定義防護機制或設定管理員可委派給使用者或角色的動作限制。您可以使用 SCP 作為允許清單或拒絕清單，以指定允許或禁止哪些服務或動作。如需詳細資訊，請參閱 AWS Organizations 文件中的服務控制政策。

服務端點

的進入點 URL AWS 服務。您可以使用端點，透過程式設計方式連接至目標服務。如需詳細資訊，請參閱 AWS 一般參考 中的 AWS 服務 端點。

服務水準協議 (SLA)

一份協議，闡明 IT 團隊承諾向客戶提供的服務，例如服務正常執行時間和效能。

服務層級指標 (SLI)

服務效能方面的測量，例如其錯誤率、可用性或輸送量。

服務層級目標 (SLO)

代表服務運作狀態的目標指標，由服務層級指標測量。

共同責任模式

描述您與 共同 AWS 承擔雲端安全與合規責任的模型。 AWS 負責雲端的安全，而 負責雲端的安全。如需詳細資訊，請參閱共同責任模式。

SIEM

請參閱安全資訊和事件管理系統。

單一故障點 (SPOF)

應用程式的單一關鍵元件故障，可能會中斷系統。

SLA

請參閱服務層級協議。

SLI

請參閱服務層級指標。

SLO

請參閱服務層級目標。

先拆分後播種模型

擴展和加速現代化專案的模式。定義新功能和產品版本時，核心團隊會進行拆分以建立新的產品團隊。這有助於擴展組織的能力和服務，提高開發人員生產力，並支援快速創新。如需詳細資訊，請參閱 中的階段式應用程式現代化方法 AWS 雲端。

SPOF

請參閱單一故障點。

星狀結構描述

使用一個大型事實資料表來存放交易或測量資料的資料庫組織結構，並使用一或多個較小的維度資料表來存放資料屬性。此結構旨在用於資料倉儲或商業智慧用途。

Strangler Fig 模式

一種現代化單一系統的方法，它會逐步重寫和取代系統功能，直到舊式系統停止使用為止。此模式源自無花果藤，它長成一棵馴化樹並最終戰勝且取代了其宿主。該模式由 Martin Fowler 引入，作為重寫單一系統時管理風險的方式。如需有關如何套用此模式的範例，請參閱使用容器和 Amazon API Gateway 逐步現代化舊版 Microsoft ASP.NET (ASMX) Web 服務。

子網

您 VPC 中的 IP 地址範圍。子網必須位於單一可用區域。

監控控制和資料擷取 (SCADA)

在製造中，使用硬體和軟體來監控實體資產和生產操作的系統。

對稱加密

使用相同金鑰來加密及解密資料的加密演算法。

合成測試

以模擬使用者互動的方式測試系統，以偵測潛在問題或監控效能。您可以使用 Amazon CloudWatch Synthetics 來建立這些測試。

系統提示

一種向 LLM 提供內容、指示或指導方針以指示其行為的技術。系統提示有助於設定內容，並建立與使用者互動的規則。

T

標籤

做為中繼資料以組織 AWS 資源的鍵/值對。標籤可協助您管理、識別、組織、搜尋及篩選資源。如需詳細資訊，請參閱標記您的 AWS 資源。

目標變數

您嘗試在受監督的 ML 中預測的值。這也被稱為結果變數。例如，在製造設定中，目標變數可能是產品瑕疵。

任務清單

用於透過執行手冊追蹤進度的工具。任務清單包含執行手冊的概觀以及要完成的一般任務清單。對於每個一般任務，它包括所需的預估時間量、擁有者和進度。

測試環境

請參閱 環境。

訓練

為 ML 模型提供資料以供學習。訓練資料必須包含正確答案。學習演算法會在訓練資料中尋找將輸入資料屬性映射至目標的模式 (您想要預測的答案)。它會輸出擷取這些模式的 ML 模型。可以使用 ML 模型，來預測您不知道的目標新資料。

傳輸閘道

可以用於互連 VPC 和內部部署網路的網路傳輸中樞。如需詳細資訊，請參閱 AWS Transit Gateway 文件中的什麼是傳輸閘道。

主幹型工作流程

這是一種方法，開發人員可在功能分支中本地建置和測試功能，然後將這些變更合併到主要分支中。然後，主要分支會依序建置到開發環境、生產前環境和生產環境中。

受信任的存取權

將許可授予您指定的服務，以代表您在組織中 AWS Organizations 及其帳戶中執行任務。受信任的服務會在需要該角色時，在每個帳戶中建立服務連結角色，以便為您執行管理工作。如需詳細資訊，請參閱 文件中的 AWS Organizations 搭配使用 AWS Organizations 與其他 AWS 服務。

調校

變更訓練程序的各個層面，以提高 ML 模型的準確性。例如，可以透過產生標籤集、新增標籤、然後在不同的設定下多次重複這些步驟來訓練 ML 模型，以優化模型。

雙比薩團隊

兩個比薩就能吃飽的小型 DevOps 團隊。雙披薩團隊規模可確保軟體開發中的最佳協作。

U

不確定性

這是一個概念，指的是不精確、不完整或未知的資訊，其可能會破壞預測性 ML 模型的可靠性。有兩種類型的不確定性：認知不確定性是由有限的、不完整的資料引起的，而隨機不確定性是由資料中固有的噪聲和隨機性引起的。如需詳細資訊，請參閱量化深度學習系統的不確定性指南。

未區分的任務

也稱為繁重工作，這是建立和操作應用程式的必要工作，但不為最終使用者提供直接價值或提供競爭優勢。未區分任務的範例包括採購、維護和容量規劃。

較高的環境

請參閱 環境。

V

清空

一種資料庫維護操作，涉及增量更新後的清理工作，以回收儲存並提升效能。

版本控制

追蹤變更的程序和工具，例如儲存庫中原始程式碼的變更。

VPC 對等互連

兩個 VPC 之間的連線，可讓您使用私有 IP 地址路由流量。如需詳細資訊，請參閱 Amazon VPC 文件中的什麼是 VPC 對等互連。

漏洞

危害系統安全性的軟體或硬體瑕疵。

W

暖快取

包含經常存取的目前相關資料的緩衝快取。資料庫執行個體可以從緩衝快取讀取，這比從主記憶體或磁碟讀取更快。

暖資料

不常存取的資料。查詢這類資料時，通常可接受中等速度的查詢。

視窗函數

SQL 函數，對與目前記錄在某種程度上相關的資料列群組執行計算。視窗函數適用於處理任務，例如根據目前資料列的相對位置計算移動平均值或存取資料列的值。

工作負載

提供商業價值的資源和程式碼集合，例如面向客戶的應用程式或後端流程。

工作串流

遷移專案中負責一組特定任務的功能群組。每個工作串流都是獨立的，但支援專案中的其他工作串流。例如，組合工作串流負責排定應用程式、波次規劃和收集遷移中繼資料的優先順序。組合工作串流將這些資產交付至遷移工作串流，然後再遷移伺服器和應用程式。

WORM

請參閱寫入一次，讀取許多。

WQF

請參閱AWS 工作負載資格架構。

寫入一次，讀取許多 (WORM)

儲存模型，可一次性寫入資料，並防止刪除或修改資料。授權使用者可以視需要多次讀取資料，但無法變更資料。此資料儲存基礎設施被視為不可變。

Z

零時差入侵

利用零時差漏洞的攻擊，通常是惡意軟體。

零時差漏洞

生產系統中未緩解的缺陷或漏洞。威脅行為者可以使用這種類型的漏洞來攻擊系統。開發人員經常因為攻擊而意識到漏洞。

零鏡頭提示

提供 LLM 執行任務的指示，但沒有可協助引導任務的範例 (快照)。LLM 必須使用其預先訓練的知識來處理任務。零鏡頭提示的有效性取決於任務的複雜性和提示的品質。另請參閱少量擷取提示。

殭屍應用程式

CPU 和記憶體平均使用率低於 5% 的應用程式。在遷移專案中，通常會淘汰這些應用程式。

本文為英文版的機器翻譯版本，如內容有任何歧義或不一致之處，概以英文版為準。