aws

上的模型內容通訊協定策略 AWS

AWS 方案指引

AWS 方案指引: 上的模型內容通訊協定策略 AWS

Table of Contents

上的模型內容通訊協定策略 AWS

Amazon Web Services （貢獻者）

2026 年 3 月 (文件歷史記錄)

本指南可協助您在整個組織中開發和實作模型內容協定 (MCP) 策略，以支援您的代理式 AI 旅程。隨著客服人員和語言模型逐漸成為業務營運的核心，建立 MCP 策略對於成功的客服人員解決方案至關重要。

本指南探討建置 MCP 策略的三個基礎支柱：MCP 工具設計、MCP 伺服器託管和 MCP 控管。透過解決這些互連元件，組織可以建立可擴展、安全且有效的系統，以跨其 AI 實作管理模型內容。本指南為組織在 AI 旅程的任何階段提供可行的洞察和策略指導，從初始實驗到完整規模的生產部署。這有助於他們開發量身訂做的 MCP 解決方案，以符合其特定需求和目標。

這些最佳實務衍生自企業規模部署 MCP 的組織的實際實作、目前 MCP 規格標準的分析，以及生產環境中自訂大型語言模型 (LLM) 應用程式的經驗教訓。

AI 系統在各種使用案例中使用越來越複雜且強大的 LLMs。LLMs擅長了解自然語言、產生類似人類的回應，以及推斷複雜的資訊。不過，若要將 LLMs 從對話界面轉換為可自動完成複雜任務的系統，組織正在採用 代理式 AI 架構、可感知其環境的 AI 系統、目標原因、自主決策、跨多個步驟協調，以及採取動作來代表使用者實現目標。此代理方法可協助組織建置 AI 系統，透過自然語言了解使用者意圖、自動協調多個資料來源和工具，並以傳統請求回應模式無法達成的規模提供個人化體驗。為了讓這些客服人員更有能力，組織需要提供現有工具和資料的存取權，以充實客服人員的情境理解，並允許其代表使用者採取行動。

MCP 提供 AI 工具整合的標準化通訊協定，可讓客服人員與外部資源之間進行一致的通訊。雖然 MCP 本身定義了通訊標準，但有效地實作它需要仔細考慮架構模式、安全模型、操作實務和效能最佳化策略，以實現可擴展、安全和可維護的解決方案。

本指南合成從企業 MCP 部署中學到的經驗教訓，提供符合 AWS Well-Architected Framework 的可行建議。它涵蓋 MCP 工具設計、MCP 伺服器託管和 MCP 控管的策略，這對於建置您自己的 MCP 解決方案至關重要。本指南中的建議對應至 AWS Well-Architected Framework 的下列五大支柱：

- 安全性 – 字符隔離、縮小範圍憑證、單獨的讀取/寫入授權
- 卓越營運 – 工具選擇準確度指標、用於迴歸測試的黃金資料集
- 可靠性 – 每個使用者和每個工具速率限制、負載卸載

- 效能效率 – 工作流程範圍工具、工具篩選、語意搜尋以減少內容時段用量
- 成本最佳化 – 跨團隊可重複使用的 MCP 伺服器，透過工具篩選降低每個請求權杖的成本

目標對象

本指南適用於在其組織中實作代理式 AI 解決方案的架構師、開發人員和技術領導者。若要了解本指南中的概念，您應該了解 LLMs的運作方式，並具備 MCP、工具和提示工程的基礎知識。

目標

建置可立即生產的客服人員 AI 系統，意味著一起解決控管、最佳化和安全性問題，以支援組織的政策。以下說明本指南如何處理這些目標：

- 控管 – 如果沒有集中控管，您就無法回答有關 AI 工作負載的稽核問題，包括哪些客服人員存取了哪些資料、具有哪些許可，以及何時存取了哪些資料。您也無法強制執行版本控制。本指南的 MCP 託管策略區段說明使用者如何執行因為缺乏系統強制執行而具有已知漏洞的過期本機 MCP 伺服器。

  對於受管制的產業，控管至關重要。稽核人員想要從單一窗格查看所有客服人員的政策強制執行和工具用量追蹤。MCP 控管提供此功能。

  透過遵循本指南中的建議，您可以在對等審核的基準中將任務準確性提高 28-32%。如需詳細資訊，請參閱 MARCO：多客服人員即時聊天協調 (ACL Anthology 網站）。控管不僅與合規有關，還改善了您的代理 AI 系統效能。
- 最佳化 – 您的團隊可能會多次建置相同的整合。例如，當五個不同的團隊為其 AI 應用程式編寫自己的資料庫查詢指令碼來與其資料庫通訊時，這相當於開發成本的五倍，以及要維護的五組錯誤清單。MCP 可讓您建置一次，並在整個工程社群中共用。隨著您的代理程式計數增加，節省的複合數會增加。

  也有大多數團隊一開始不會注意到的每次請求成本問題。每個工具定義都會使用內容視窗字符。在 20 個工具中，您只會在描述上花費 5，000-10，000 個字符，以及使用者查詢。這會增加延遲和 LLM 推論成本，並在模型難以從可用工具清單中選擇正確的工具時降低準確性。

  與直接存取 APIs 的代理程式相比，使用結構化工具包裝函式的代理程式在資料庫任務上的準確度約為三倍（如需詳細資訊，請參閱 LLMs的中介軟體：工具是複雜環境中語言代理程式的檢測工具)。如何設計和向 AI 模型展示工具非常重要。本指南建議為工具提供清晰的結構描述、將它們限定為實際工作流程而非原始端點，以及限制內容視窗中的資訊。本指南的 MCP 工具設計策略章節深入探討這些層面。

- 安全與合規 – 想像一個代理式 AI 系統，該系統會美化清除步驟，並嘗試刪除生產資料庫。如果代理繼承了使用者的完整管理員登入資料，則刪除可能會經歷。透過只授予讀取和建立存取權的權杖隔離和縮小範圍憑證，它會安全地失敗。

  受管制的工作流程會進一步強化這一點。本指南提供範例（在處理病患資料之前需要 HIPAA 驗證和個人身分識別資訊匿名化的醫療保健管道）。在 MCP 工具中嵌入此類邏輯意味著每次合規都會發生確定性。

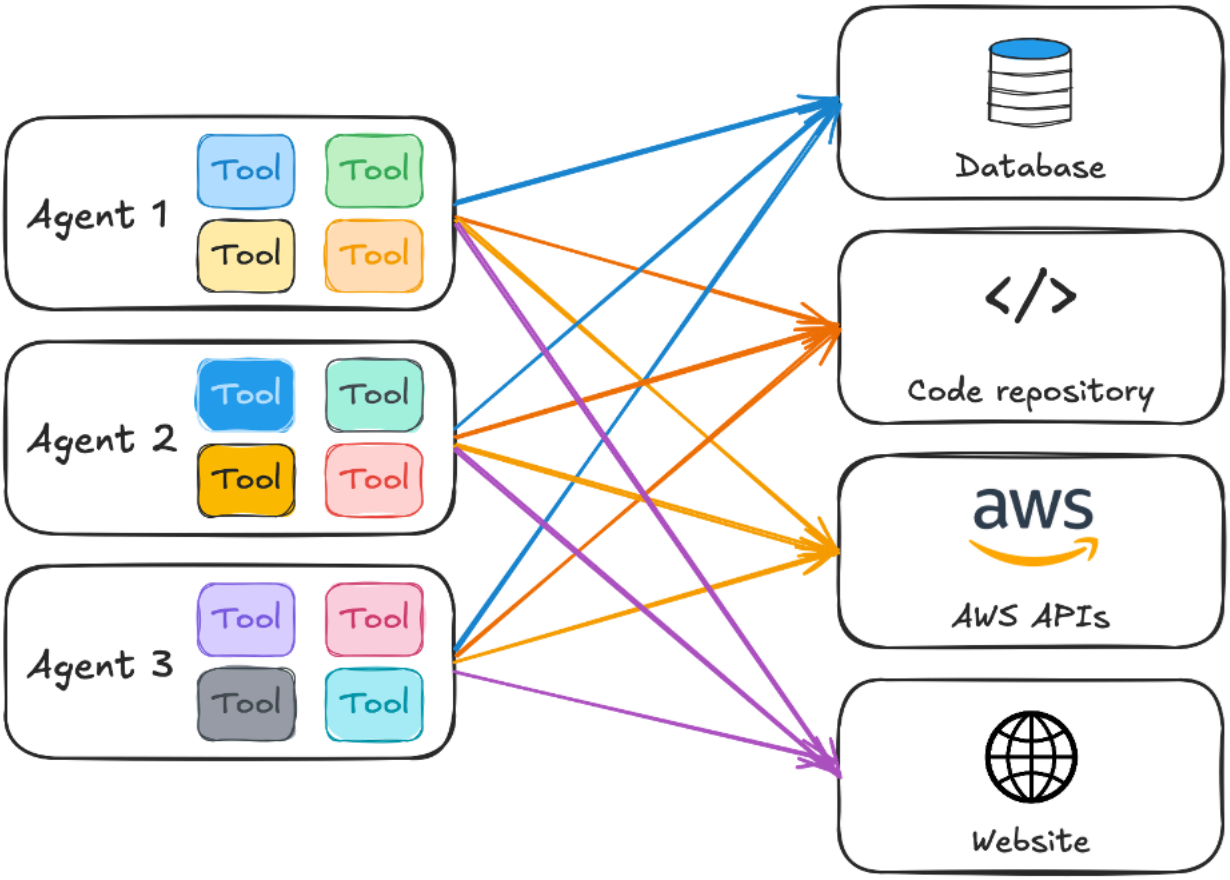
什麼是 MCP？

LLMs的運作方式是根據其訓練資料，預測提示的答案。這表示 LLM 只能提供已看到之資料和事件的答案。擷取增強生成 (RAG) 和知識庫等方法可讓您包含內容資料。不過，如果您要詢問 LLM 明天的天氣預測會是什麼，或資料庫中有多少客戶，則可能會幻覺或無法提供答案，因為這些情況不在 LLM 預先訓練的知識範圍內。為了能夠回答這類問題，客服人員需要存取 LLM 原生內容之外的外部功能、資料和 APIs。

了解工具

我們可以讓 LLM 透過 工具存取其他系統和內容。 工具是提供給 LLM 以達成明確目標的函數。工具可以呼叫 API、查詢資料庫、執行計算器操作、操作程式碼沙盒、執行 Web 搜尋，甚至叫用另一個 AI 系統或agent-as-a-tool。每個工具都應包含說明，告知 LLM 工具的功能、使用時機，以及接受的參數。這可讓 LLM 根據使用者的輸入，對要叫用的工具或工具組合做出細微的決策。LLM 會被告知代理程式可使用哪些工具，讓其產生回應，指示代理程式叫用該工具。例如，當您詢問 LLM 資料庫中有多少客戶時，LLM 會將回應傳回給代理程式，要求 使用特定輸入參數執行`query_database`工具。LLM 會決定要叫用的工具和工具呼叫的輸入。代理程式接著會執行 工具，將自然語言輸入轉換為語法上正確的函數呼叫，並執行查詢。代理程式會根據 LLM 的指示叫用工具，並將這些結果傳回 LLM。這利用 LLM 對文字型輸入進行推理，並為任務選取適當工具的能力。

下圖顯示每個代理程式如何針對每個目標管理自己的工具集。



擴展工具存取可能會為代理式 AI 解決方案帶來挑戰：

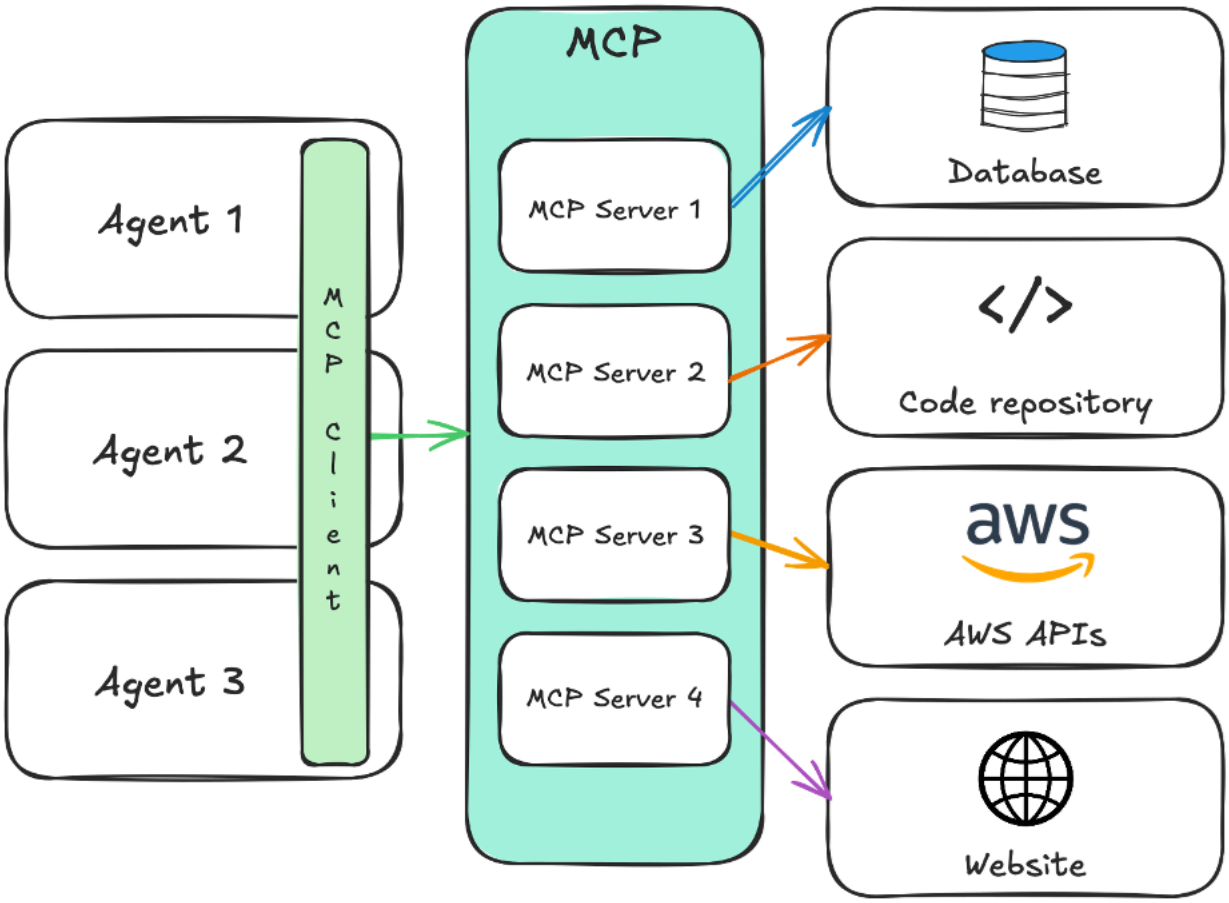
- 如果每個開發人員都為相同的外部功能建立自己的工具，則有許多重複的工作和與這些外部功能互動的非標準化方式。這會在您的客服人員之間產生不一致的實作。雖然您可以透過在程式庫中開發標準工具並分配它們來解決這個問題，但這缺乏集中式控管。這使得實施安全政策、追蹤工具用量、跨團隊管理版本控制，或確保符合組織標準變得困難。此外，當您直接將工具嵌入代理程式時，您必須在每次建立新工具或更新現有工具時重新部署代理程式。
- 提供工具給 LLM 會使用其內容視窗。內容視窗是模型可以隨時考慮的字符數量 (LLMs處理的文字單位，通常代表單字、單字部分或標點符號）。LLMs具有內容視窗限制。工具和其文件會使用該有限內容視窗，以及系統提示和使用者提示。當內容視窗填滿時，LLMs可能會因為多種因素而遇到效能降低：識別相關資訊困難、處理複雜性提高，以及推理容量降低。當工具定義、系統提示和對話歷史記錄爭奪有限內容時段空間時，挑戰會更為複雜，因為它們在每次 LLM 調用時都會提供。

因此，工具數量及其記錄方式會直接影響 LLM 的效能，例如回應時間和準確性。

MCP 建立將客服人員連線至外部功能的通用標準。它通常稱為「USB-C for AI 應用程式」。MCP 伺服器不是直接向代理程式註冊工具，而是做為透過 JSON-RPC 2.0 發現和調用之託管工具的媒

介。MCP 可讓您註冊 MCP 伺服器，以封裝代理程式可存取的工具，而不是將數十或數百種不同的工具新增至您的代理程式，並隨著時間進行維護。此方法會標準化工具的封裝、呈現和叫用方式。這有助於解決客服人員內工具使用量的擴展和管理挑戰。它還會將代理程式開發和操作與其用於外部功能的工具分離。

下圖顯示使用 MCP 存取外部資源的客服人員。



不過，MCP 標準無法解決所有擴展和控管挑戰。MCP 伺服器的實作必須與有效的工具設計、託管和企業控管策略結合。本指南提供每個策略的最佳實務，協助您建置和使用 MCP 做為代理式 AI 解決方案的一部分。

何時使用 MCP

MCP 提供策略基礎設施，以擴展您的代理式 AI 計畫。透過集中工具管理和控管，MCP 伺服器可降低跨多個代理程式建置和維護自訂整合的累積成本。這會隨著客服人員生態系統的擴展而增加回報。

在下列情況下，MCP 可能會成為您策略的一部分：

- 您需要集中控管客服人員如何存取企業系統和服務，例如資料庫、APIs、內部工具和第三方整合。

- 開發人員花太多時間撰寫跨實作不一致的自訂整合。
- 您有重複的工具，可提供常見的功能。
- 您想要透過標準化、受管的 MCP 介面，將您的專屬工具或資料提供給外部消費者或第三方代理系統，以解鎖新的收入串流，同時維護安全性和控制能力。

在您決定 MCP 伺服器將成為策略的一部分後，請評估現有的開放原始碼 MCP 伺服器實作是否符合您的需求、是否需要增強，或是否需要建置自訂伺服器。許多預先建置的 MCP 伺服器實作可在公有儲存庫中使用，涵蓋常見的功能，例如檔案系統存取、Web 瀏覽、程式碼沙盒、資料庫存取和 API 整合。

在許多情況下，預先存在的 MCP 伺服器已足夠。例如， AWS 提供 AWS MCP 伺服器受管遠端 MCP 伺服器，它為 AI 助理和客服人員提供 AWS 服務 透過自然語言互動的安全、已驗證存取。您可以使用 AWS MCP 伺服器 來執行複雜的多步驟 AWS 任務，方法為結合對 AWS 文件的即時存取、語法上更正 API 呼叫，以及遵循 AWS 最佳實務的預先建置工作流程，稱為客服人員 SOPs。 AWS 會持續測試 AWS MCP 伺服器 ，以確保客戶客服人員可以成功使用它們。

您應該使用代理程式測試這些現有的 MCP 伺服器，以判斷它們是否符合您的使用案例。如果客服人員無法完成工作流程、產生不正確或次佳的回應、無法導覽複雜的多步驟程序，或錯過重要的領域特定最佳實務或安全性考量事項，您應該考慮在幾個方面進行增強。

當現有的 MCP 伺服器無法完全滿足您的需求，且難以正確使用現有工具或產生準確的回應時，請在建置自訂伺服器之前考慮這些增強方法：

- 豐富的代理程式內容 – 如果您的代理程式難以正確或有效地使用現有 MCP 伺服器中的工具，請考慮使用其他文件或範例來補充這些工具定義。這有助於為 LLM 提供額外的內容。
- 新增補充工具 – 使用工具擴展現有的 MCP 伺服器，以存取客服人員成功完成工作流程所需的其他組織資料或內容。
- 改善基礎 APIs – 簡化您的服務 APIs，藉由降低參數複雜性、提供更清楚的錯誤訊息，以及提供客服人員可以使用的合理預設值，讓它們更易於 LLM。

雖然使用現有的 MCP 伺服器實作可加速常見功能的開發，但當您的使用案例需要特殊功能時，建置自訂 MCP 伺服器是必要的。自訂 MCP 伺服器可協助您封裝網域專業知識、強制執行組織標準、改善複雜工作流程的代理程式可靠性，以及支援安全要求的合規性。考慮在下列情況下建置自訂 MCP 伺服器：

- 特定網域的工作流程 – 當 API 文件中未擷取必要的知識時，需要網域專業知識的多步驟工作流程應該封裝在自訂 MCP 工具中。 例如，不要讓客服人員協調複雜的醫療保健資料管道，而這些管道必須驗證健康保險流通與責任法案 (HIPAA) 合規、匿名化 PII，以及轉換為 HL7 FHIR 格式，而是提供

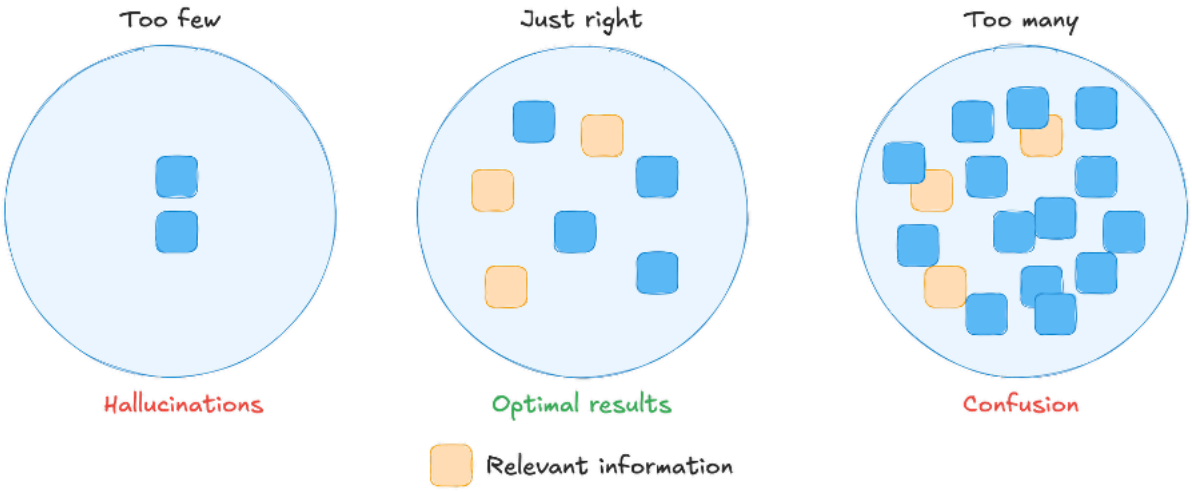
直接嵌入網域專業知識process_patient_data的工具。這會移除 LLM 的相依性，以正確協調和執行工作流程步驟，從而提高一致性和合規性。

- 黃金路徑抽象 – 客服人員可能難以實作最佳方法，因為他們缺乏組織內容，預設為基本模式，而不是組織最佳實務。在這些案例中，您可以在自訂 MCP 工具中封裝這些黃金路徑，以強制執行成本、效能或安全性的規範性標準。例如，提供直接嵌入組織標準的工具， deploy_secure_infrastructure而不是讓客服人員使用可能不安全或效率不佳的預設設定來部署基礎設施。
- 複雜的多服務協同運作 – 您可以確定在 MCP 工具內建置該邏輯，而不是透過嘗試推斷每個步驟中使用的正確序列和一組服務，讓代理程式協同運作複雜的工作流程。您可能也想要提供有關客服人員可能不知道的最佳服務整合模式的專業知識。 這也可以改善代理程式的準確性和效率。
- 服務特定的最佳實務 – 這適用於以安全為重心的工具，這些工具可協助客服人員實作加密政策、存取控制，以及透過客服人員工具存取服務特有的合規模式。此外，如果有不明顯的服務特定操作最佳實務，使用 MCP 伺服器可協助您確保它們已實作，且不會讓客服人員進行說明。

MCP 工具設計策略

MCP 用戶端和伺服器的主要任務是探索工具並將其呈現給 LLM，以便使用它們來改善其回應。這使得 MCP 工具設計成為建置有效 MCP 解決方案最重要的策略之一。從模型的角度來看，工具是他們可以根據需要叫用以提供更準確和完整回應的函數。函數界面會抽象工具的基礎實作，範圍可以從圍繞單一 API 呼叫的包裝函式到複雜的工作流程邏輯。

不過，您必須達到與提供給 LLM 之工具數量的平衡。如果工具太少，LLM 可能無法收集正確的內容和資訊，因此會採用模型中可用資訊的最佳猜測。如果工具太多，LLM 可能會混淆正確的工具選擇和序列，導致幻覺。您的目標是取得恰到好處的工具數量。下圖顯示工具太少和太多的挑戰。



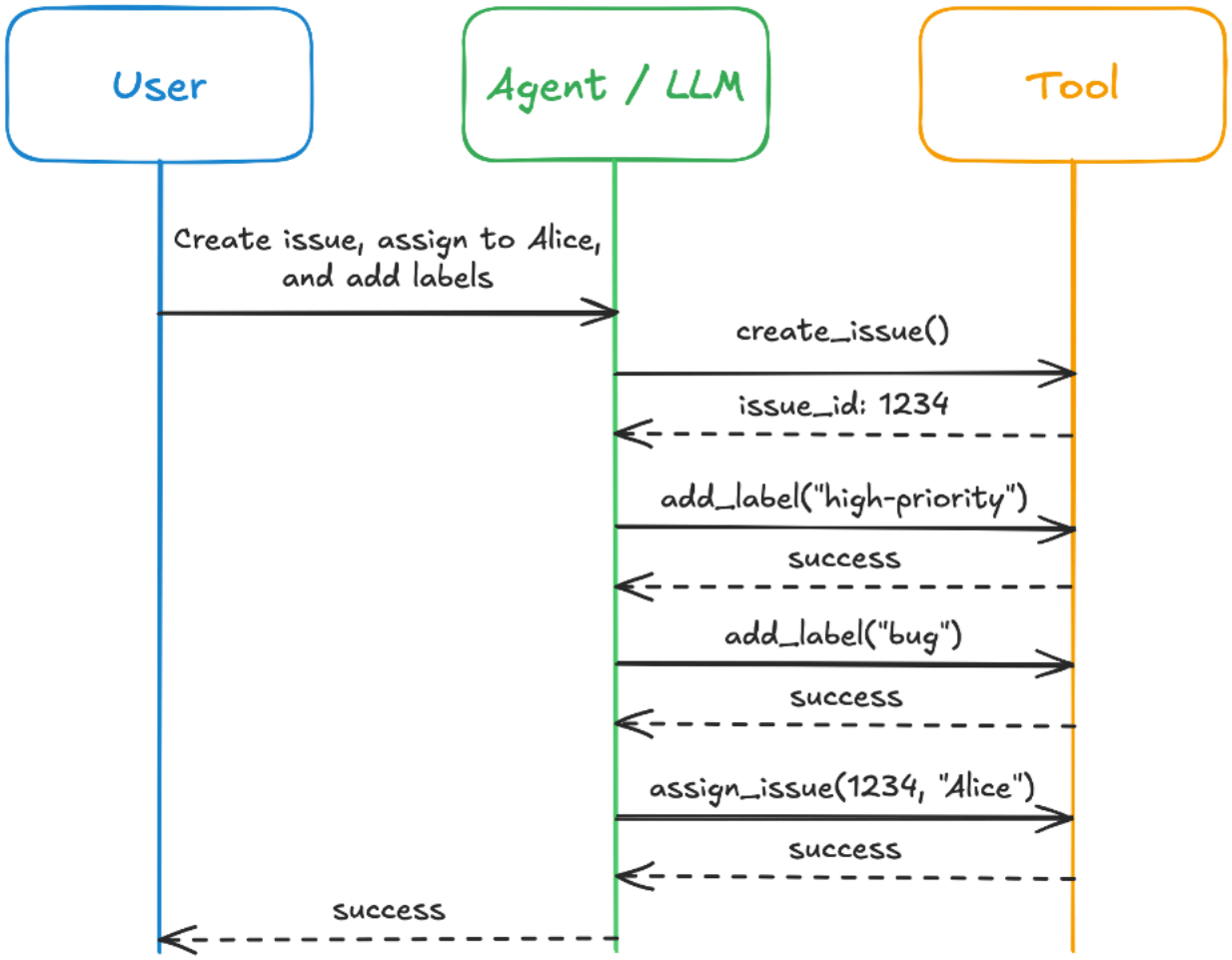
解決方案需要了解要提供多少 工具， 以及如何限制每個工具的範圍。無論工具對應至個別 API 呼叫或完成工作流程，工具的精細程度都會直接影響客服人員所需的工具總數，以及其使用方式。本節提供界定 MCP 工具範圍、建立工具定義、探索工具和組織工具的最佳實務。

工具範圍

有兩種開發工具的方法：精細和粗粒。

精細

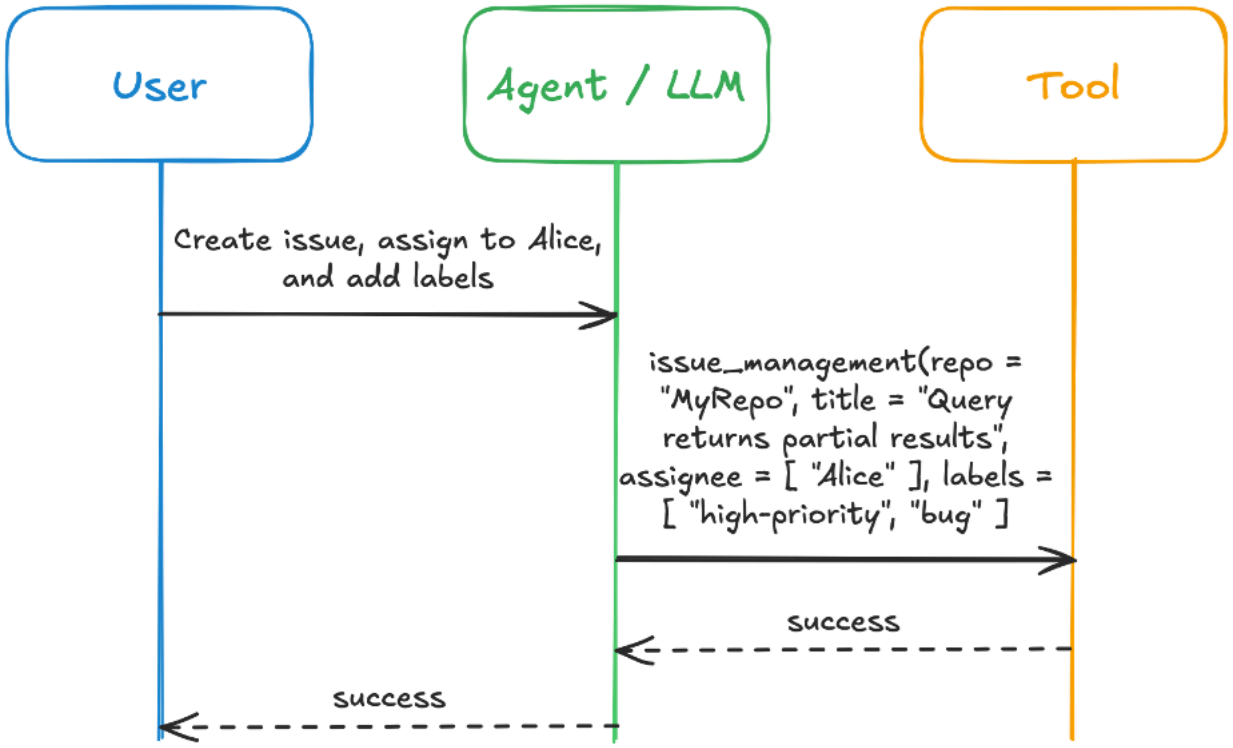
在精細的方法中，您會為每個 API、動作或查詢建立工具。例如，您可以為 Git `get_issue`儲存庫建立 `create_issue`、`assign_issue`、、 `add_label`和 `close_issue`工具。這可讓 LLM 對每個 API 進行精細呼叫，並視需要協調每個 API。考慮以下提示：「為名為「僅查詢傳回部分結果」的產品服務建立問題，將其標記為錯誤和高優先順序，並將其指派給 Alice。」 下圖顯示每個 tool-per-API方法如何回應此提示。



在此方法中，系統提示和每個已註冊的工具定義會在每次呼叫時提供給 LLM。這會消耗額外的內容並產生延遲懲罰，因為每個工具呼叫代表對 LLM 的個別呼叫。它也會增加在工作流程中處理錯誤的複雜性。

粗粒

粗略粒度或工作流程驅動的方法將是工作流程導向的工具。此工具著重於end-to-end使用者對 API 結構的意圖。您有一個工具可確定地呼叫許多 APIs，而不是tool-per-API 的工具。使用先前的 Git 儲存庫範例，您可以建立由代理程式呼叫一次`create_and_setup_issue`的工具。工具實作會根據提供給工具的參數，建立問題、新增標籤，並將其指派給使用者。下圖顯示粗粒方法如何處理相同的提示。



此方法顯示 LLM 層如何隱藏所有複雜性。當協同運作邏輯內嵌在工具實作中時，所有循序步驟、記錄、重試邏輯、斷路器和速率限制都會在工具中決定性地執行。工作流程驅動型方法可讓 LLM 更輕鬆地使用正確的參數叫用正確的工具。請務必注意，某些 APIs可能已經提供工作流程意圖，例如 Amazon EC2 `RunInstances` API。在這些情況下，tool-per-API可能會提供您想要的工作流程導向設計。

不過，工具也可能變得太粗粒。如果您的單一工作流程工具嘗試執行太多動作，並且有許多可能的參數，LLM 可能會難以判斷如何正確使用工具。它也可以透過參數選擇和錯誤處理來產生挑戰。因此，工具開發必須達到與使用者意圖一致的平衡，並避免單一工具中的功能太少或太多。我們建議您根據完整的使用者工作流程設計工具，綁定通常一起發生的操作（例如三個或更多的 API 呼叫）。我們也建議您分解超過八個或更多參數的工具，或處理多個不同的使用者意圖。使用實際提示 進行測試，以確認客服人員是否可以正確使用每個工具。

如果您有複雜且動態的工作流程，無法輕鬆地封裝為確定性工具，您可以考慮使用agent-as-tool模式。專門的代理程式可以充當工具，而不是您的主要代理程式嘗試在工作流程中協調複雜的任務。這些類型的工具可以實作進階決策和分支，而且可以處理無法在確定性程式碼中輕鬆管理的錯誤和重試邏輯。這類似於但不同於 Agent2Agent (A2A) 通訊協定。A2A 通訊協定是互補的，可在任何代理程式架構中的代理程式之間提供互通性和協同合作。

我們建議您從工作流程分析開始，映射最常見的使用者工作流程，以識別每個客服人員所需的核心功能。這會建立您的最低可行工具集。根據我們大規模開發 MCP 伺服器的經驗，我們建議下列實務。當這些實務發生衝突時，請優先考慮使用者意圖和工作流程。

MCP 工具範圍的最佳實務

- 思考使用者案例並綁定常見操作 – 工具應直接映射以完成使用者互動，而不是需要協調多個操作。如果工作流程通常需要三個以上的個別呼叫，請將它們合併為單一工具。這可減少 LLM 的認知負載、將工具呼叫次數降至最低、減少完成任務所需的內容耗用和延遲，並改善準確性和延遲。
- 將參數限制為八個或更少 – 如果工具超過八個參數，請將它分解為多個工具。隨著複雜性的增加，LLMs難以選擇參數。

> **Note**
>
> 如果綁定操作需要八個以上的參數，則優先於參數計數的綁定，因為簡化工作流程比嚴格的參數限制更有價值。

- 個別的讀取和寫入操作 – 提供不同的工具來讀取資料並進行修改。當客服人員執行潛在破壞性操作、啟用不同的授權政策，並降低在資訊收集期間意外修改的風險時，這種區隔會明確顯示。
- 提供合理的預設值 – 設計工具，讓 LLM 只需要指定個別請求特有的參數。預設值可降低 LLM 必須考量的資訊，藉此降低參數複雜性並改善工具選擇的準確性。
- 偏好確定性執行 – 盡可能讓工具執行和輸出確定性。確定性工具更可靠且更容易測試。對於需要智慧型協同運作、分支邏輯或進階錯誤處理的複雜工作流程，如果無法在確定性程式碼中輕鬆管理，請考慮使用專用代理程式做為工具。不過，請選擇性地使用此模式，因為它會增加複雜性。

工具定義

當 LLM 收到無法直接處理的請求時，將會檢閱可用的工具，以協助完成請求。LLM 會根據其對所提供工具的名稱和描述的語意理解，以及提示中提供的任何指示來選取工具。然後，它會根據定義的輸入結構描述建立輸入，並根據輸出結構描述預期輸出。因此，建立描述性工具定義和經過驗證的輸入和輸出結構描述，對於協助 LLM 有效選取工具至關重要。建立此文件通常有兩種方法：工具規格方法和文件方法。

工具規格方法

建議的方法是在定義工具時直接遵循 MCP 工具規格。以下範例使用 Strands Agent 工具裝飾工具顯示：

```
@tool(
  name = "search_website",
  description = "This tool searches the provided website for semantic matches to the
 query provided",
  inputSchema = {
    "json": {
      "type": "object",
      "properties": {
        "url": {
          "type": "string",
          "description": "The url of the website to load and search."
        },
        "query": {
          "type": "string",
          "description": "The content you want to try and match in the website."
      }
    },
    "required": ["url", "query"]
  },
  outputSchema = {
    "json": {
      "type": "object",
      "properties": {
      "results": {
          "type": "array",
          "items": {
            "type": "string"
          }
        }
      }
    }
  }
)
def search_website:
  …
```

使用標準欄位，例如 name、inputSchema、 description和 ，outputSchema確保每個工具都有 LLM 和人類都可以理解的一致文件。每個工具都應至少定義這些欄位，並選擇性地提供標題和註釋，這些是工具行為的選用提示。盡可能使用參數值的列舉，讓 LLM 可以輕鬆選取正確的選項。列舉最適合無限集，例如狀態或優先順序值，但不適用於自由格式文字、動態值、任意數字或資源識別符。在這些情況下，請改為提供明確的描述和範例。同時盡可能包含預設值，以便 LLM 不必猜測正確的選項。

請記住，工具定義會在每次調用時包含在 LLM 提示中，並耗用內容時段空間以及系統指示和對話歷史記錄。

Docstring 方法

如果您以 Python 撰寫工具，另一種方法是使用 docstring 來提供工具的描述、用量和輸出。以下是此方法的範例：

```
def search_website(url: str, query: str) -> list:

    """
    This tool loads the specified website and then attempts to find content that
 matches the provided query through semantic search. It provides back a list of strings
 that are the sentences that match the query.
    Args:
      url: the website url to load
      query: the content you want to semantically match in the website
    """
```

文件字串不會強制執行結構描述或標準化格式。根據工具開發人員選擇記錄每個工具的方式，使用此方法可能會產生不一致的結果。如果您遵循此方法，則定義和強制執行整個組織的標準至關重要。

MCP 工具定義的最佳實務

- 遵循 MCP 工具規格 – 為每個工具提供 `name`、`inputSchema`、 `description`和 `outputSchema` 欄位。對於 Python 實作，請使用 Pydantic 模型透過欄位描述提供內嵌文件、自動類型驗證，以及透過列舉提供限制值。這可讓結構描述自我記錄，並改善 LLM 對有效參數選項的了解。
- 將描述撰寫為提示 – 工具描述是指導 LLM 決策的說明。包含工具用途（工具的功能）、使用時機（使用者意圖模式或案例）、輸出內容（輸出的用途）、參數和錯誤條件的基本元件。
- 提供具體範例 – 包含工作流程範例與實際值是引導 LLMs正確工具用量的最有效方法。
- 明確記錄相依性 – 包含先決條件、編號序列、狀態變更和後續動作。

工具探索

有三種方法可讓您透過 MCP 伺服器在代理程式中探索和註冊工具：靜態定義、動態探索和搜尋函數。

靜態定義

首先，您可以直接在客服人員程式碼中靜態定義可用的工具。在此方法中，您會為 MCP 用戶端存取的 MCP 伺服器所提供的每個工具定義遠端工具 (Strands Agent SDK 等架構中的用戶端參考物件）。下列範例使用可串流的 HTTP 傳輸：

```
from mcp.client.streamable_http import streamablehttp_client
from strands import Agent
from strands.tools.mcp import MCPClient

streamable_http_mcp_client = MCPClient(
    lambda: streamablehttp_client("https://mcp1:8000/mcp")
)

reverse_text = RemoteTool(
    name="reverseText",
    client=streamable_http_mcp_client
)

agent = Agent(tools=[reverse_text])
```

個別工具註冊可協助您非常選擇您提供給 LLM 的工具，這可將使用的內容時段量降至最低。權衡是，它需要知道可用工具的名稱，如果 MCP 伺服器中的可用工具變更，它可能會很脆弱。

動態探索

下一個方法是使用動態探索並向代理程式註冊所有可用的工具。隨著將更多工具新增至 MCP 伺服器，此方法會線性使用內容。以下是此方法的範例：

```
from mcp.client.streamable_http import streamablehttp_client
from strands import Agent
from strands.tools.mcp import MCPClient

streamable_http_mcp_client = MCPClient(
    lambda: streamablehttp_client("https://mcp1:8000/mcp")
)

with streamable_http_mcp_client:
    tools = streamable_http_mcp_client.list_tools_sync()
    agent = Agent(tools=tools)
```

假設典型的工具定義大約使用 250–500 個 字符 （ 包括名稱、描述和結構描述 ）。註冊 20 個 工具將消耗內容時段的 5，000–10，000 個 字符。當您有少量的 MCP 伺服器，而且您可以控制工具的數量時，此選項是最簡單的實作方式。不過，如果預期工具清單會成長，它可能會在您的客服人員中建立無提示的內容管理問題。這種方法的替代變化是呼叫 時使用工具篩選參數`list_tools`，例如 Strands Agents SDK 提供的參數，以減少向代理程式註冊的工具數量。

搜尋函數

第三個選項是使用搜尋函數在執行時間尋找相關工具。您可以從 MCP 伺服器列出所有可用的工具，然後根據使用者提示對這些工具執行語意搜尋。然後，產生的工具會向 代理程式註冊。Amazon Bedrock AgentCore Gateway 提供原生語意搜尋功能，可讓這類解決方案更容易實作。

MCP 工具探索的最佳實務

- 內容時段保留 – 選擇盡可能節省內容時段的工具探索和註冊方法。
- 使用工具篩選或語意搜尋功能 – 動態提供 LLM 一組縮小範圍的工具以供選擇，從而提高選擇正確工具的準確性和有效性。工具篩選可以在工具名稱 （ 完全相符或模式 ）、工具描述 （ 語意相符 ） 或網域或類別標籤上操作。語意搜尋對於比對使用者意圖與工具描述特別有效。這兩種方法都會減少內容時段的使用。

工具組織

探索正確的工具並確保 LLM 可以有效地使用這些工具，是有效工具開發的最關鍵部分之一。當您開始開發 MCP 伺服器時，您需要決定下列項目的策略：

- 有多少工具進入一個 MCP 伺服器
- 哪些工具不應放入相同的 MCP 伺服器
- 如何命名工具以使其可搜尋並防止名稱衝突 （ 具有相同名稱的不同工具 ）
- 如何記錄工具和 MCP 伺服器，以方便 LLM 使用

命名空間組織是一種設計模式，可防止工具名稱衝突、群組相關功能，並促進 LLMs 的有效工具識別。模式會建立結構式分類，類似於組織式儲存系統，而不是非結構式累積。我們建議使用 domain-noun-verb 模式來命名工具。例如，`github_issue_create`、`github_issue_list`、`github_issue_update`、`github_pullreques`
`github_pullrequest_create`或 `github_pullrequest_merge`。檢查字母排序行為時，此模式的優勢顯而易見。依字母順序列出工具時，所有與問題相關的操作叢集 (`create`、`list`、`update`)，

後面接著提取請求操作 (`create`、`list`、`merge`)。名詞 （資源類型） 做為組織界限。此結構有助於 LLM 工具掃描和人工文件導覽，因為相關功能自然分組在一起。

MCP 伺服器應該在網域層級繫結，但可以根據其所提供功能的責任分離進行細分。例如，您可能有個別的 MCP 伺服器，用於資料庫的寫入操作和讀取操作。若要強制執行此區隔，建議您在代理程式層級實作護欄，以根據使用者意圖和許可限制可存取的 MCP 伺服器。這可以透過以下組合來實現：

- 條件式伺服器載入 – 只有在代理程式在使用者輸入中偵測到讀取操作時，才載入唯讀 MCP 伺服器。
- 許可型篩選 – 使用使用者授權僅授予適當 MCP 伺服器的存取權。

最後，您會想要在 MCP 伺服器提供的工具數量上建立上限。請勿假設客服人員將如何使用您的 MCP 伺服器。他們可能會從中列出所有可用的工具，並將其全部提供給 LLM。如果您在單一伺服器中有 50 個以上的工具，您應該考慮將其分割成多個伺服器。

MPC 工具組織的最佳實務

- 使用工具的 domain-noun-verb 命名標準 – 實作策略，以防止 MCP 伺服器和代理程式中的名稱衝突。
- 設定上限 – 限制單一 MCP 伺服器中的工具數量。
- 分割 MCP 伺服器 – 使用職責分離將 MCP 伺服器分割為邏輯群組。

MCP 託管策略

將可用工具抽象化為 MCP 伺服器會將您的代理程式開發與可用工具分離。這將介紹託管 MCP 伺服器的位置，以及在這些伺服器內如何組織工具的挑戰。

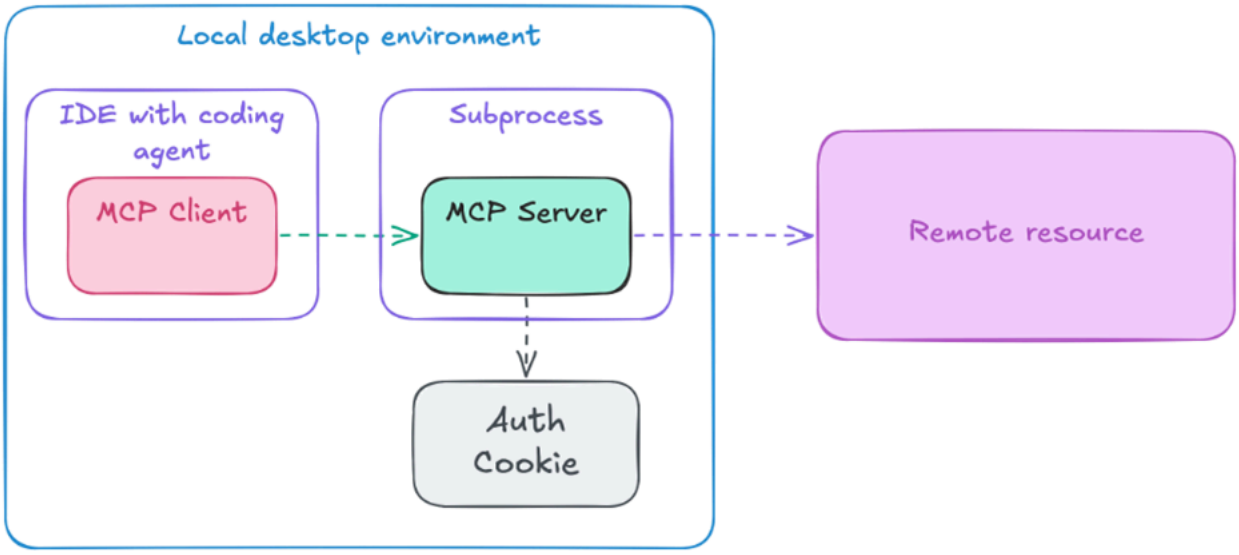
託管方法

託管 MCP 伺服器有三個選項：在終端使用者機器本機上執行它們、遠端託管它們，或透過 MCP 閘道託管它們。每個選項都有優點和權衡。

本機託管

本機託管會在本機電腦上執行 MCP 伺服器作為子程序，以及透過標準輸入和輸出串流使用 JSON-RPC 與伺服器通訊的代理程式。此方法不需要用戶端和伺服器之間的身分驗證。工具可以與本機應用程式和檔案互動、使用本機儲存的登入資料，以及繼承使用者本機電腦的網路存取權。這是最簡單的託管模式，有幾個優點。

許多客戶使用本機伺服器開始使用 MCP。它們可讓工程師快速迭代和解決其本機環境中的各種問題。請考慮連線至工程師編碼助理我們使用的 Git 儲存庫的 MCP 伺服器。將 MCP 伺服器保留在本機很有意義，因為它可以使用工程師的唯一登入資料來存取儲存庫，而且不會將額外的網路呼叫新增至遠端 MCP 伺服器。下圖顯示與 IDE 中的編碼代理程式搭配使用的本機託管 MCP 伺服器。



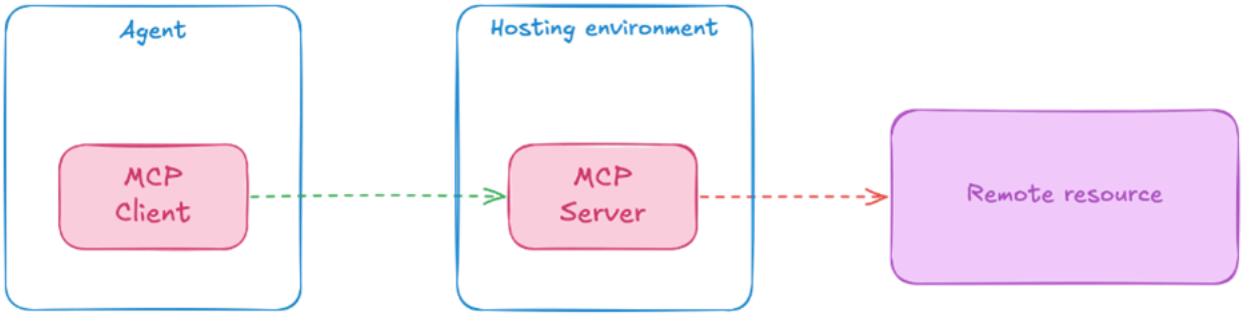
對於這些類型的部署，您必須考慮如何開發和分發 MCP 伺服器。大多數客戶都會開發 MCP 登錄檔，供最終使用者註冊和下載伺服器。這與容器登錄非常類似，使用者可以搜尋特定功能，並找到符合其需求的 MCP 伺服器。

有公有 MCP 登錄檔，例如官方 MCP 登錄檔，並且有私有託管登錄檔。組織通常會將其 MCP 登錄檔策略與開放原始碼軟體分發、容器登錄檔和內部套件管理相關的現有政策保持一致。您應該考慮安全掃描、核准工作流程和合規要求等因素。

不過，本機託管引入了組織應考慮的操作挑戰。首先，最終使用者必須獨立探索、下載和設定 MCP 伺服器。這可能會增加複雜性，以開始使用其在本機使用的每個個別 MCP 伺服器。其次，您無法控制 MCP 伺服器生命週期，這表示使用者可能會繼續在本機執行具有安全漏洞或缺少功能的過時版本。這可能會使符合合規要求變得複雜。有些 IDEs 和 CLI 工具，例如 Kiro，可讓組織管理和控制可用的 MCP 工具，確保團隊之間的一致性和安全性。

遠端託管

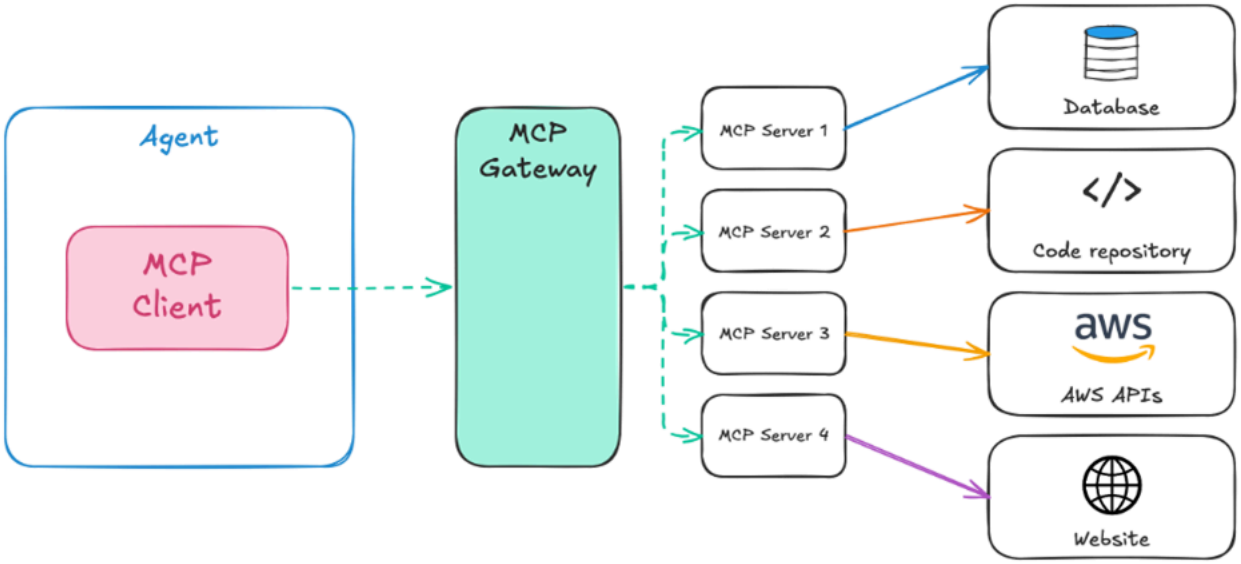
第二個選項是託管透過 HTTP 或 HTTPS 存取的遠端 MCP 伺服器。這可讓您存取任何網路連線的用戶端。使用遠端託管可讓您集中控制對 MCP 資源和功能的存取、實作身分驗證和授權，以及控制 MCP 伺服器邏輯的版本控制和更新。遠端託管仍然需要使用 MCP 登錄檔，以便最終使用者可以探索他們想要與代理程式搭配使用的 MCP 伺服器。下圖顯示遠端託管方法。



從客服人員開發的角度來看，無論 MCP 伺服器是本機或遠端，體驗都很類似。最重要的變更是實作身分驗證和授權，包括代理程式對 MCP 伺服器的存取，以及伺服器對外部資源的存取。必須仔細規劃遠端 MCP 伺服器實作，以考慮多租戶存取和權限管理。MCP 控管策略章節包含有關身分驗證和授權考量的詳細資訊。

MCP 閘道

最後一個選項是使用 MCP 閘道。MCP 閘道可做為 MCP 用戶端和伺服器之間的集中式代理，並協調對已註冊 MCP 伺服器的存取。如果沒有閘道，每個代理程式都需要註冊其可能想要使用的每個遠端 MCP 伺服器。閘道可讓代理程式連線至管理身分驗證、授權、路由和通訊協定轉譯的單一端點。新的 MCP 伺服器和工具可以動態新增，並立即提供給代理程式。下圖顯示 MCP 閘道方法。



有些閘道解決方案，例如 Docker MCP Gateway，也會管理 MCP 伺服器的生命週期，並視需要啟動伺服器。MCP 閘道，例如 Amazon Bedrock AgentCore Gateway，也可以提供原生語意搜尋功能，協助管理工具探索。這為客服人員提供單一端點來與 MCP 用戶端連線，並協助最佳化其內容時段用量。結果是簡單的代理程式，可以有效地選擇和使用 MCP 工具。不過，它具有與遠端 MCP 伺服器方法類似的身分相關挑戰。

託管 MCP 伺服器的最佳實務

- 託管選項的頻譜不是適合所有人的大小。目前大部分的 MCP 伺服器使用都是本機的。
- 當您開始使用遠端 MCP 伺服器時，您的主要考量是對 MCP 伺服器的一致身分驗證和授權，以及 MCP 伺服器如何對下游資源執行身分驗證和授權。
- MCP 閘道可簡化託管多個遠端 MCP 伺服器的連線、身分驗證和授權。他們也提供功能，透過搜尋適用的工具來改善內容時段管理。

MCP 控管策略

MCP 為組織提供的其他重要功能支援集中式控管。您的 MCP 控管策略應針對 MCP 伺服器及其存取的資源進行身分驗證和授權。它還應解決速率限制，以保護下游資源、監控工具用量和效能的操作指標，以及管理 MCP 伺服器的部署和分佈。

身分驗證和授權

身分驗證和授權策略最重要的部分之一，就是從 MCP 伺服器管理下游資源存取。當使用者呼叫代理程式時，會執行身分驗證和授權，以確保使用者具有呼叫代理程式的許可。然後，代理程式會協調呼叫 MCP 伺服器中的特定工具。您需要決定如何根據每個工具授權存取。

其中一個選項是machine-to-machine授權，其中不需要使用者同意或互動。例如，以時間為基礎的代理程式調用使用 MCP 伺服器從應用程式收集日誌並對其進行分析。在此案例中，代理程式會預先授權存取指定的資料。第二個選項是使用者委派的存取，其中使用者同意存取使用者特定的資料和資源。

下表顯示身分驗證和授權模式。

因素	使用者委派的存取	Machine-to-machine
資料擁有權	資料的使用者特定授權	系統或整個組織的資料
使用者互動	使用者存在且可以同意	沒有使用者互動
操作時間	互動式或即時	背景、排程或批次
許可範圍	許可因使用者而異	客服人員層級的一致許可

使用者委派存取需要謹慎實作，並且應該與您的安全團隊一起開發。客服人員必須能夠評估 LLM 已選取的工具，以及是否需要額外的授權。MCP 工具必須包含說明，以指出其身分驗證和授權需求，以及擷取存取權杖的位置。用戶端應用程式必須支援中繼身分驗證請求，且 MCP 用戶端必須針對每個工具呼叫，將擷取的登入資料傳回給客服人員。

您應該確保 MCP 工具一律擁有自己的權杖來存取外部功能，以及記錄和稽核存取權。使用者登入資料不應透過您的代理系統傳播。例如，您的 MCP 伺服器不應使用相同的字符來存取用來叫用代理程式的資料。下游呼叫應使用明確範圍、用途產生的權杖。這有助於提供額外的護欄，以防止代表動作進行意外的資料存取。它也可以協助防止幻覺產生意外的結果。假設具有完整管理員許可的使用者要求代

理程式複製生產資料庫，以便在生產前使用。若要這樣做，使用者只需要 READ和 CREATE許可。假設 LLM 會美化並認為需要清除舊資料庫，作為此請求的一部分。如果重複使用使用者的登入資料，可能會成功，因為使用者的原始登入資料具有DELETE許可。反之，如果 MCP 伺服器只針對具有 READ和 CREATE許可的請求使用刻意縮小範圍的權杖，則刪除生產資料庫的嘗試將會失敗。

您可以使用 Amazon Bedrock AgentCore Identity 來協助實作這些模式。請務必刻意選擇列出和叫用 MCP 伺服器託管工具的許可是否意味著 MCP 伺服器公開的外部功能的許可。此身分從 MCP 伺服器流向資源，再流回使用者，取決於使用的身分驗證和授權服務類型。您必須決定 MCP 伺服器的大規模處理方式。

設計身分驗證和授權模式時，請實作權杖隔離機制，針對每個要存取的工具擷取不同的存取權杖。請勿在工具和伺服器之間重複使用權杖。AgentCore Identity 提供此字符隔離功能。它會自動管理工作負載字符（用於machine-to-machine身分驗證）和使用者字符（用於使用者委派存取），以確保適當的分離並防止許可升級。這在整合遠端 MCP 伺服器或 MCP 閘道時特別重要。

MCP 身分驗證和授權的最佳實務

- 權杖分離 – 請勿將承載權杖從發起人傳遞至下游服務。驗證 aud (audience) 欄位是否符合接收字符的伺服器。對象宣告指定權杖要用於哪個服務，防止在不同 MCP 伺服器上未經授權的權杖重複使用。
- 選取存取方法 – 針對 MCP 伺服器提供的每個工具，選擇machine-to-machine和使用者委派存取。考慮在使用相同身分驗證模式的相同 MCP 伺服器中將工具分組在一起。

控制負載

如同任何分散式系統，您必須考慮如何控制 MCP 伺服器機群中的負載。首先，您會考慮是否要在 MCP 伺服器中實作速率限制，以及在何處實作限制。如果您選擇不實作速率限制，則會傳遞下游資源執行的任何速率限制。許多系統會根據請求屬性選擇評分限制，例如使用者或帳戶 ID。驗證傳送至下游服務的請求是否具有這些相同的屬性，以便多個使用者不受另一個使用者驅動的負載影響。

如果您選擇實作速率限制，建議的方法是在 MCP 伺服器層級實作主要速率限制，使用後端服務提供次要保護，且客服人員會根據速率限制回饋調整其行為。考慮速率限制是每個 MCP 伺服器還是每個工具。每個 MCP 伺服器速率限制有助於保護多租戶環境中的 MCP 伺服器機群和服務。不過，這可能非常粗粒。每個工具速率限制旨在防止可能無法自行充分速率限制的大量下游資源。如果工具呼叫多個 APIs，您應該設定速率限制，以符合這些 APIs 允許的最低速率。

在 HTTP 標頭中傳遞速率限制資訊也可能是使用者和自動化系統的實用指標，以協助管理自己的請求速率和重試策略。例如，您可以從 MCP 伺服器將這些標頭傳回給代理程式，如下列範例所示：

```
X-RateLimit-Limit: 100
X-RateLimit-Remaining: 45
X-RateLimit-Reset: 1640995200
```

此外，當沒有單一客戶超過速率限制，但負載影響系統效能時，請考慮卸載以保護整體服務。

控制負載的最佳實務

- 選擇速率限制方法 – 計劃根據個別使用者使用下游資源或透過其使用 MCP 伺服器和工具來對其進行速率限制。
- 考慮卸載 – 保護您的 MCP 伺服器機群免受非由單一或少數客戶驅動的一般過載影響。

操作指標

要擷取 MCP 實作的關鍵指標應專注於他們提供的客戶體驗。這些指標通常包括字符用量、工具選擇準確性、向代理程式註冊的工具數量，以及工具延遲。例如，監控每個工具傳回的輸出字符，可讓您在工具超過內容時段用量的閾值時設定警示。當工具超過該閾值時，您可能想要檢閱工具的行為。這也與 MCP 工具設計策略相關。工具選擇準確性指標指出客服人員為指定任務選擇適當工具的程度，而執行速度和成功率則強調效能瓶頸和可靠性問題。

例如，為了評估工具選擇和工具使用準確性指標， AWS 團隊建立了黃金資料集進行迴歸測試。在使用者查詢時，使用歷史 API 調用日誌LLMs，以合成方式產生資料集。使用預先定義的工具選擇和工具使用指標（例如工具選擇準確度、工具參數準確度和多迴轉函數呼叫準確度）， AWS 團隊可以客觀地評估 AI 代理器在對話迴轉期間正確識別適當工具、填入其參數，以及維持一致工具調用序列的能力。

測量向 代理程式註冊的工具數量指標，可協助您識別潛在的內容時段管理挑戰，以及 MCP 伺服器所提供可用工具的變更。您應該定期檢閱指出 MCP 伺服器和工具使用者體驗的操作指標。

貢獻者

編寫

- Alex Torres，資深解決方案架構師 AWS
- Saikat Gomes，資深客戶解決方案經理 AWS
- Mike Haken，資深首席解決方案架構師， AWS
- Sreeja Das，首席工程師 AWS

檢閱

- Ted Swinyar，解決方案架構師經理 AWS
- Raju Patil，資深資料科學家， AWS

技術寫入

- 資深技術作者，Lilly AbouHarb AWS

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
初次出版	—	2026 年 3 月 16 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：將您的現場部署 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

A2A Agent-to-Agent)

支援任務委派和狀態轉移的agent-to-agent協同合作的狀態通訊協定。

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。

客服人員

一種 AI 系統，可使用工具自動推理、規劃和採取行動來實現目標。

客服人員操作

在生產環境中大規模建置、測試、部署和執行 AI 代理器的操作實務。

彙總函數

在一組資料列上操作並計算群組單一傳回值的 SQL 函數。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。

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)

一種工具，可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。 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，並透過核准的程序，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 實作碎片程序指標。

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。

公民開發人員

在沒有專業技術技能的情況下，使用無程式碼/低程式碼平台建立 AI 應用程式的商業使用者。

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

部落格文章中的 Stephen Orban 定義了這些階段： AWS 雲端 企業策略部落格上的邁向雲端優先之旅和採用階段。如需有關它們如何與 AWS 遷移策略關聯的資訊，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)

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的 AI 欄位。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護欄，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

正在收集和處理其資料的個人。

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 成員帳戶來管理組織的帳戶，並管理該服務的許可。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 延伸了原本專為精簡製造實務設計的價值串流映射程序。它著重於在軟體開發過程中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 中的在 雲端中復原工作負載的災難 AWS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)

組織之間商業文件的自動交換。如需詳細資訊，請參閱什麼是電子資料交換。

加密

將人類可讀取的純文字資料轉換為加密文字的運算程序。

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，少量的提示非常有效。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，以及自然語言的交談。如需詳細資訊，請參閱什麼是基礎模型。

FM 閘道

集中式中介，可控制和標準化對基礎模型的存取。也稱為 LLM 閘道。

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。

護欄 (AI)

可篩選、驗證和限制代理程式輸入和輸出的安全機制，以協助確保負責任且安全的 AI 行為。

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)

工作負載在遇到挑戰或災難時持續運作的能力，無需介入。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，並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。

human-in-the-loop (HitL)

一種工作流程模式，其中代理程式執行會在關鍵決策點暫停進行人工審核和核准。

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 Rs。

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、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。

MCP

請參閱模型內容通訊協定。

模型內容通訊協定 (MCP)

用於代理程式對工具通訊的無狀態通訊協定。

MCP 伺服器

透過模型內容通訊協定公開一或多個工具的服務。

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

屬於組織一部分的管理帳戶 AWS 帳戶 以外的所有 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)

此 AWS 計畫提供諮詢支援、訓練和服務，以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，請參閱本詞彙表中的 7 個 Rs 項目，並請參閱動員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 中的操作準備審查 (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)

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。

設計隱私權

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

請參閱擷取增強生成。

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 個 R。

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 個 R。

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 個 R。

版本

在部署程序中，它是將變更提升至生產環境的動作。

重新放置

請參閱 7 個 R。

Replatform

請參閱 7 個 R。

回購

請參閱 7 個 R。

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

矩陣，定義所有參與遷移活動和雲端操作之各方的角色和責任。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 個 R。

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。

斯卡達

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。它包含秘密值及其中繼資料。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 秘密中的內容？ Secrets Manager 文件中的 。

設計安全性

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。

陰影 AI

在組織內受管管道之外建置或使用的未授權 AI 應用程式。

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。

tool

代理程式可以叫用以在外部系統中執行操作的函數或 API。

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。

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，通常可接受中等緩慢的查詢。

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。