AWS 使用生成式 AI 加速 上的軟體開發生命週期

AWS 方案指引

AWS 方案指引: AWS 使用生成式 AI 加速 上的軟體開發生命週期

Table of Contents

AWS 使用生成式 AI 加速 上的軟體開發生命週期

Chetan Makvana，Amazon Web Services

2025 年 4 月 (文件歷史記錄)

高品質軟體的需求不斷增長，促使組織不斷尋求加速其軟體開發生命週期 (SDLC) 的方法。隨著組織努力保持競爭，請務必減少上市時間，同時維持或改善產品品質。為了因應這些挑戰，軟體開發體驗必須演進和使用尖端技術、方法和實務，以簡化程序並使軟體開發團隊更具生產力和創造力。新一代開發體驗的出現，意味著軟體的構想、建置、測試和部署方式發生了重大變化。它整合了各種功能，包括雲端原生開發、AI 驅動的自動化、進階專案管理、協作工具和 DevSecOps，共同增強 SDLC 的效率和有效性。

此轉型的前沿是軟體工程中生成式 AI 的崛起。根據「Gartner」，到 2027 年，40% 的平台工程團隊將使用 AI 來擴增 SDLC 的每個階段，而 2023 年只有 5%。此報告還說明軟體工程領導者現在必須準備在更廣泛的領域採用生成式 AI，這些領域對開發程序至關重要。在另一份報告中，McKinsey 研究顯示，開發人員速度索引較高的公司將營收成長 4 到 5 倍、獲得 60% 的股票報酬率，以及 55% 的創新能力。透過只採用程式碼產生的生成式 AI，組織可以在軟體開發工作流程中釋放新的效率、生產力和創新水準。這可以減少手動工作量、表面洞察，並增強人類專業知識。

目標

此策略文件概述架構、基礎功能、使用案例、最佳實務和成功指標，可協助您使用生成式 AI 加速 SDLC。它說明如何在所有開發階段有效整合生成式 AI，以改善產品品質和效率。

此策略文件可協助您和組織達成下列目標：

- 實作架構、基礎功能、使用案例、最佳實務和成功指標，以透過生成式 AI 加速 SDLC。
- 在所有開發階段有效整合生成式 AI，以改善產品品質、發行速度和開發效率。
- 透過整合尖端 AI 技術、方法和實務來簡化程序並強化開發團隊，以適應新一代軟體開發。

目標對象

此策略文件適用於希望透過將生成式 AI 套用至其開發實務來加速軟體開發生命週期的 IT 領導者、工程經理、技術長和軟體開發團隊。

了解軟體開發體驗

軟體開發體驗包含開發團隊在整個軟體開發生命週期 (SDLC) 中使用的環境、工具和程序。它包含整合式開發環境 (IDE)、協作平台、測試架構、知識管理系統、部署管道等。

精心設計的開發體驗可簡化工作流程、減少手動工作，並讓您的團隊能夠專注於高價值的任務，最終加速 SDLC。例如，與需要手動交接和內容切換的分段工具鏈相比，透過無縫整合 IDE、版本控制系統和部署工具，可讓開發人員以更高的速度和效率撰寫、測試和部署程式碼。同樣地，整合強大的知識管理架構可協助團隊輕鬆存取和共用機構知識、最佳實務和文件。這可增強其整體生產力和問題解決能力。

軟體開發體驗會直接影響軟體開發團隊的整體效能和成功。次佳的體驗可能會導致下列情況：

- 生產力降低 – 工具效率低下、工作流程複雜，以及缺乏自動化阻礙了團隊生產力，這使得功能和更新的交付速度變慢。
- 技術負債增加 – 整合不佳的工具和臨機操作程序可能會導致技術負債，這使得隨著時間的推移維護和擴展軟體系統更具挑戰性。
- 減少創新 – 當受到手動重複性任務的影響 時，您的團隊探索新技術和推動創新的能力會受到限制。
- 品質受損 – 分段測試和部署程序會增加軟體瑕疵和漏洞的風險。這可能會對交付軟體的整體品質產生負面影響。

透過投資設計良好的軟體開發體驗，您可以釋放顯著的優勢，例如更快的上市時間、更高的軟體品質、增強的軟體開發團隊滿意度，以及更高的業務敏捷性。

使用生成式 AI 為軟體開發體驗提供支援

將生成式 AI 整合到軟體開發生命週期 (SDLC) 代表整個軟體開發團隊如何構想、設計、實作和維護軟體解決方案的範式轉移。生成式 AI 有可能徹底改變 SDLC 的每個階段，包括專案管理、需求收集、設計、編碼、測試、部署和維護。

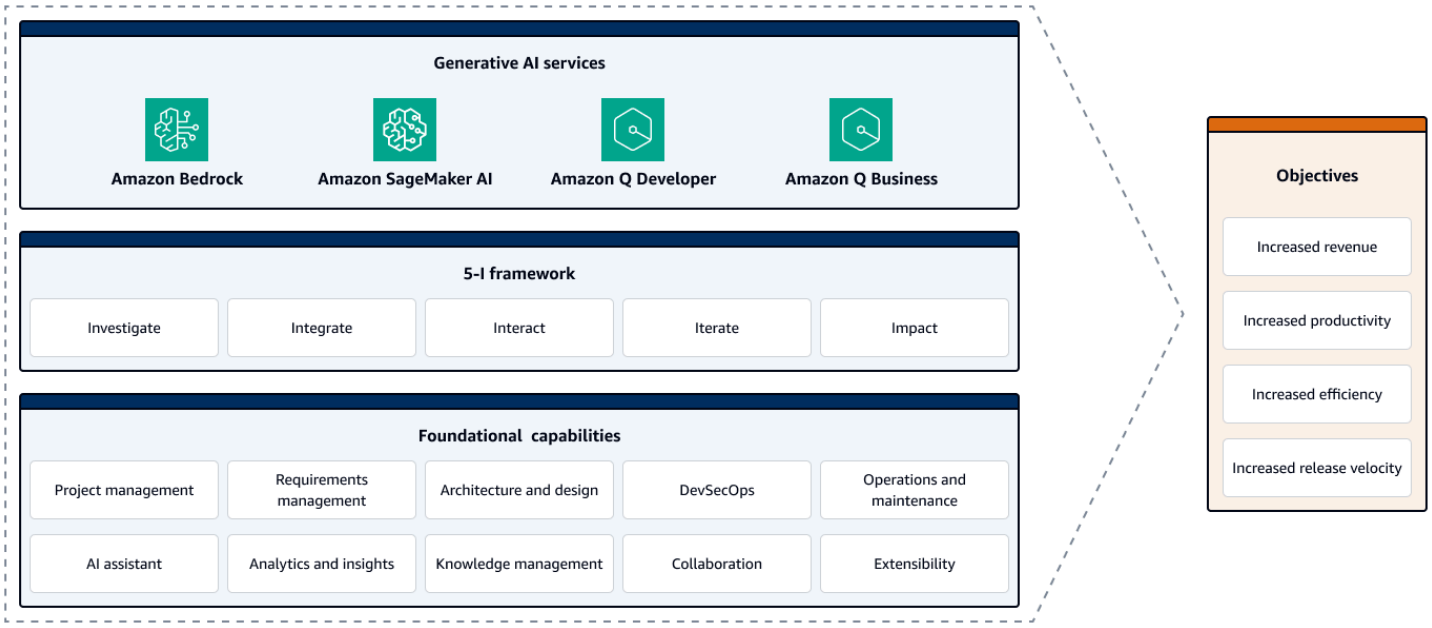
產生式 AI 支援開發體驗的核心是做為整個軟體開發團隊的智慧協作者，包括產品經理、設計師、解決方案架構師、開發人員、測試人員和營運人員。它提供內容感知協助、產生成品（例如使用者案例、設計模型、程式碼片段和測試案例）、提供近乎即時的建議，甚至在潛在問題出現之前預測問題。這種 AI 擴增方法可大幅降低團隊成員的認知負載。這可以讓他們專注於高階策略決策和複雜的問題解決，而生成式 AI 可以處理更單調、重複性更高的任務。

生成式 AI 也可做為知識擴大器。它可協助團隊成員從大量的資料儲存庫快速存取相關資訊、最佳實務和模式。這可以有效地讓整個組織的專業知識普及化。透過在整個開發工具鏈中無縫整合生成式 AI 功能，您可以為整個軟體開發團隊建立更直覺、更有效率且更具生產力的環境。此增強型開發體驗可加速 SDLC 並改善整體品質。它也可以減少錯誤並促進創新，因為團隊成員可以更快地探索新想法和方法。

若要在您的組織中採用生成式 AI 驅動的開發體驗，請考慮下列關鍵元素：

- 5-I 架構 – 5-I 架構由五個維度組成， 提供全方位方法來導覽現代軟體開發的程序。它提供結構化方法，可協助您在 SDLC 的所有階段有系統地套用生成式 AI。
- 基礎功能 – 若要充分利用現代軟體開發維度的生成式 AI 功能，您需要建立一組強大的基礎功能。這些功能構成 AI 驅動開發體驗的骨幹。這些功能可協助您在整個 SDLC 中整合和使用生成式 AI。

5-I 架構和基礎功能共同形成了重新構想軟體開發體驗的策略。五個維度提供套用生成式 AI 的策略架構，而基礎功能可讓您的組織準備好支援這種 AI 驅動型方法。 AWS 服務例如 Amazon Bedrock、Amazon SageMaker AI、Amazon Q Developer 和 Amazon Q Business，可提供生成式 AI 功能和功能，您可以將這些功能整合到您的軟體開發體驗中。



AI 驅動的軟體開發體驗的 5-I 架構

5-I 架構為軟體開發團隊提供結構化方法，以有效地將生成式 AI 整合到其開發實務中。它可協助您為在整個 SDLC 中使用生成式 AI 建立堅實的基礎。它還可協助您設定正確的開發實務、工作流程和思維，以充分利用生成式 AI 的潛力。

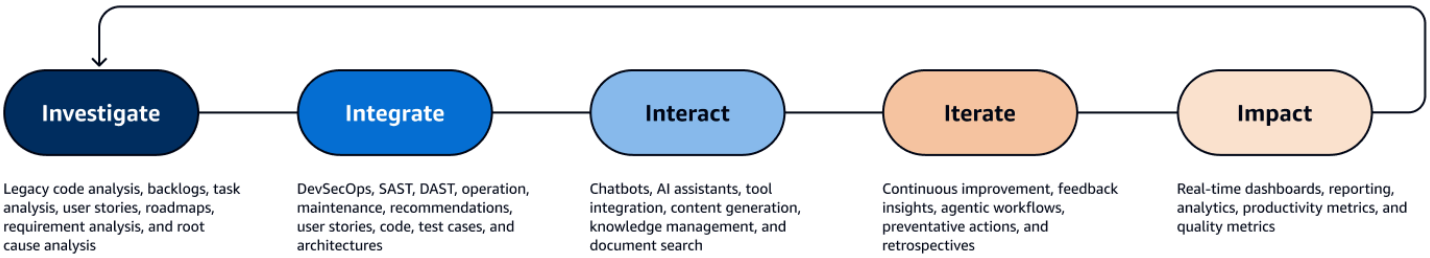
本節包含下列主題：

- 架構概觀
- 與軟體開發生命週期整合

架構概觀

5-I 架構是以五個關鍵維度建置：調查、整合、互動、反覆運算和影響。每個維度代表生成式 AI 大幅增強軟體開發程序的關鍵區域。透過策略性地整合這些維度的生成式 AI，架構可解決現代軟體開發不斷演進的需求。它可以減少認知負載並擴增創造潛力。它認識到理想的開發體驗不僅與工具有關，還與建立 AI 在每個階段無縫增強人類功能的環境有關。

下圖顯示採用 AI 的軟體開發的五個維度。對於每個維度，它會顯示您可以整合生成式 AI 的位置，以提高效率和創新。



以下是架構中的五個維度：

- 調查 – 使用生成式 AI 增強軟體開發過程中的每個分析任務。使用生成式 AI 來了解需求、處理大量資料、辨識模式，以及產生可能超過人類容量或產生更長時間的洞見。這些洞見可協助您做出更明智的決策、快速識別改善機會，以及更有效率地交付高品質的軟體。生成式 AI 可以是整個 SDLC 分析程序的智慧型合作夥伴。透過利用生成式 AI，您將深入分析套用至關鍵領域，例如需求收集、舊版程式碼庫檢查和產品待處理項目最佳化。例如，產品擁有者可以在建立使用者案例之前，使用生成式 AI 來分析使用者旅程或需求。開發團隊可以發現效率低下，並識別現有程式碼庫中的最佳化機會。DevOps 工程師可以套用根本原因分析，以快速診斷效能問題或安全漏洞，進而改善可靠性。
- 整合 – 整合生成式 AI，以在整個 SDLC 中自動化各種任務和程序。這包括自動產生程式碼片段、測試案例、架構設計、使用者案例和部署管道。透過自動化這些通常為手動的任務，團隊可以專注於更具策略和創新性的工作，從而加快上市時間和高品質的應用程式。整合維度代表軟體開發中的範例轉移，其中 AI 成為開發過程中不可或缺的一部分。它與您的軟體開發團隊合作，以提高生產力、提高品質並推動創新。這會導致更快的上市時間。它挑戰您的軟體開發團隊定期評估其程序和工作流程，方法是在每個步驟中詢問：「可以自動化嗎？」
- 互動 – 使用生成式 AI 輔助，為您的團隊提供跨各種任務和查詢的即時情境式支援。這些智慧型助理充當知識淵博的協作者，從大量的資訊儲存庫中擷取。他們可以回答編碼問題、提供設計建議、解釋標準操作程序，以及協助疑難排解複雜的問題。將這些 AI 助理整合到開發工作流程中可提高生產力，並促進更協作、解決問題的環境。
- 迭代 – 使用生成式 AI 在整個 SDLC 中啟用快速、資料驅動的調整。您可以持續分析來自客戶意見回饋、使用模式、市場趨勢和團隊績效指標等來源的資料，以便快速做出明智的決策。這種適應性會將您的軟體開發從靜態、預先定義的程序精簡為流暢、回應式的方法。它以各種方式顯示資訊清單，包括對待處理項目的動態優先順序、靈活的資源配置、適應性測試策略、不斷發展的文件和回應式部署程序。例如，產品管理員可以使用 AI 產生的洞見來重新排序其待處理項目，以近乎即時的方式整合新的客戶需求和市場趨勢。DevOps 工程師可以根據效能分析調整部署計劃和基礎設施組態，確保應用程式保持彈性和最佳化。開發團隊可以將衝刺回顧的意見回饋轉換為可行的改進，以供下一次反覆運算使用，進而推動持續程序增強的文化。
- 影響 – 套用生成式 AI 來評估軟體開發程序的有效性和效能。透過使用 AI 支援的分析和指標，您可以深入了解開發效率、程式碼品質、使用者參與度和整體應用程式效能。此資料驅動型方法可協助您

做出明智的決策、最佳化開發工作流程，並持續改善應用程式的品質和使用者體驗。評估軟體團隊生產力時，生成式 AI 會分析各種資料點，例如程式碼遞交頻率、問題解決時間、發行速度、功能交付率等。它也可以評估程式碼檢閱的品質、協作工具的有效性，以及不同開發實務對整體團隊輸出的影響。透過將這些指標與專案結果建立關聯，AI 會識別人類分析師可能遺漏的模式和趨勢，並且可以提供可採取行動的洞見來提高團隊生產力。此外，生成式 AI 可協助您根據產業標準或歷史資料對團隊效能進行基準測試，提供個人化的改善建議。它也可以預測開發過程中的潛在瓶頸或風險，以便您可以採取主動措施。

與軟體開發生命週期整合

SDLC 由多個階段組成，可能因組織而異。這些階段通常包括下列項目：需求和規劃、設計和架構、實作、測試、部署，以及操作和維護。

下表將 5-I 架構的維度映射至 SDLC 階段，並提供每個維度的整合層級。

架構維度	需求和規劃	設計和架構	實作	測試	部署	操作和維護
調查	高	低	低	低	低	中
整合	中	中	高	中	高	高
互動	高	高	高	中	中	高
反覆運算	中	低	低	低	低	中
影響	高	中	高	低	高	高

整合層級從高到低不等。映射會顯示每個維度的關鍵重點區域。例如，調查會在需求和規劃階段顯示高強度。Integrate 會在實作、部署、操作和維護階段展現高強度。

透過使用此映射，您可以有效地排定工作的優先順序。我們建議您專注於高、中、低。請務必採用平衡且具影響力的方法，透過生成式 AI 增強軟體開發體驗。

AI 驅動軟體開發體驗的基礎功能

若要成功實作生成式 AI 驅動的軟體開發體驗，您需要在組織中建立一組跨越多個角色的基礎功能。這些功能代表您能夠有效地部署資源、實作程序，並在採用 AI 的軟體開發環境中實現所需的成果。透過培養這些功能，您可以建立強大的基礎，協助您在 SDLC 的所有階段無縫整合生成式 AI。

AWS 提供關鍵服務，協助您實作這些功能。例如，Amazon Q Developer 透過擔任採用 AI 技術的助理，協助加速軟體開發。Amazon Q Business 可協助您取得快速且相關的問題加急解答、解決問題和產生內容。它也可以整合與軟體開發相關的工具，代表您採取行動。Amazon Bedrock 可讓您存取基礎模型和廣泛的功能，以自訂特定的開發工作流程和需求。

透過 培養這些功能 AWS 服務，您可以建立強大的基礎，協助您在 SDLC 的所有階段無縫整合生成式 AI。

以下是您應該關注的基本功能：

- 專案管理
- 需求管理
- 架構和設計
- 協作
- DevSecOps
- 操作和維護
- AI 助理
- 分析與洞察
- 知識管理
- 可擴展性

每個基礎功能都與架構維度和 SDLC 的不同階段整合。此整合可協助您在整個軟體開發過程中有效使用 AI 功能。可提升每個步驟的效率、品質和創新能力。這些基礎功能、架構和 SDLC 階段之間的協同合作，為 AI 驅動的軟體開發建立全面的生態系統。這可協助您充分利用生成式 AI 的潛力、推動持續改進、加速開發週期，以及提供高品質的軟體產品。

下表顯示基礎功能和子功能如何對應至架構維度和 SDLC 階段。

功能：子功能	調查	整合	互動	反覆運算	影響
專案管理：問題管理	需求和規劃	無	無	無	無
專案管理：Sprint 和任務管理	需求和規劃	需求和規劃	無	無	無

功能：子功能	調查	整合	互動	反覆運算	影響
專案管理：產品待處理項目管理	需求和規劃	無	無	需求和規劃	無
專案管理：使用者案例映射	需求和規劃	無	無	無	無
專案管理：報告和分析	需求和規劃	無	無	無	需求和規劃
專案管理：產品藍圖管理	需求和規劃	無	需求和規劃	無	無
專案管理：回饋迴圈	無	無	無	需求和規劃	無
專案管理：回溯性	無	無	無	需求和規劃	無
需求管理	需求和規劃	需求和規劃	無	無	無
架構和設計：解決方案設計	設計和架構	設計和架構	無	無	無
協同合作：文件管理	所有 SDLC 階段	無	所有 SDLC 階段	無	無
協同合作：知識分享	所有 SDLC 階段	無	所有 SDLC 階段	無	無
協同合作：專案資產管理	無	所有 SDLC 階段	所有 SDLC 階段	無	無
DevSecOps：CI/CD	測試、部署	實作、測試、部署	部署	無	無

功能：子功能	調查	整合	互動	反覆運算	影響
DevSecOps：DevOps 安全性	實作	實作、測試、操作和維護	無	實作、測試、操作和維護	無
DevSecOps：應用程式效能監控	無	操作和維護	無	無	無
DevSecOps：日誌彙總和分析	操作和維護	操作和維護	無	無	無
DevSecOps：AIOps	操作和維護	無	無	操作和維護	無
DevSecOps：持續改善	無	無	無	操作和維護	無
DevSecOps：儀表板監控	無	操作和維護	無	無	無
DevSecOps：效能洞察	操作和維護	無	無	操作和維護	無
操作和維護：事件管理	無	無	無	操作和維護	無
操作和維護：程式碼升級	無	操作和維護	無	無	無
操作和維護：程式碼最佳化	操作和維護	操作和維護	無	無	無

功能：子功能	調查	整合	互動	反覆運算	影響
操作和維護：技術債務管理	無	操作和維護	操作和維護	無	無
操作和維護：變更管理	無	實作、部署	無	無	無
操作和維護：反向工程	操作和維護	無	無	無	無
操作和維護：程式碼現代化	無	實作	無	無	無
操作和維護：效能最佳化	無	操作和維護	無	操作和維護	無
分析和洞見	無	需求和規劃	無	無	所有 SDLC 階段
AI 助理	無	無	所有 SDLC 階段	無	無
知識管理	無	無	所有 SDLC 階段	無	無
可擴展性	無	部署	無	無	無

用於專案管理的生成式 AI 使用案例

有效的專案管理是軟體開發成功的核心。在生成式 AI 的背景下，專案管理會採用新的維度。它可以變得更具預測性、適應性和資料驅動性。AI 支援的專案管理工具會分析歷史專案資料，以產生更準確的時間和資源預估。他們可以根據業務目標和團隊容量自動排定任務的優先順序，甚至可以在任務發生之前預測潛在的障礙。例如，專案經理可能會根據專案的需求和類似專案的歷史資料，使用生成式 AI 來建立初步專案計劃。然後，AI 可以建議考慮技能、工作負載和專案需求的最佳團隊組成。在整個專

案中，AI 驅動的儀表板會透過自動產生報告並反白顯示需要注意的區域，提供近乎即時的專案狀態洞見。

這種 AI 擴增的專案管理方法可以提高效率。它有助於專案經理專注於策略決策和團隊領導，而不是陷入例行管理任務中。

下表顯示專案管理使用案例，您可以使用生成式 AI 和負責這些使用案例的角色來增強這些案例。

子功能：使用案例	角色
問題管理：建立和指派問題	專案經理
問題管理：在測試期間偵測問題並將其記錄	測試工程師
問題管理：根據嚴重性排定問題的優先順序，並將其指派給開發人員	專案經理
問題管理：識別和合併重複的問題	專案經理
問題管理：追蹤和產生有關專案的關鍵問題、指標和整體運作狀態的報告	專案經理
衝刺和任務管理：估計任務的工作量，並根據團隊容量指派故事點	Scrum Master
衝刺和任務管理：在團隊成員之間分配任務，以在整個衝刺中實現平均工作負載	Scrum Master
衝刺和任務管理：促進衝刺規劃工作階段，使團隊的工作與衝刺目標保持一致	Scrum Master
產品積壓管理：根據商業價值、緊迫性和使用者意見回饋重新排序積壓項目	產品擁有者
產品待辦項目管理：將新的客戶意見回饋和市場洞察整合至產品待辦項目中，以進行近乎即時的優先順序	產品擁有者
產品積壓管理：識別和管理積壓項目之間的相依性，以簡化開發	產品管理員

子功能：使用案例	角色
使用者案例映射：建立使用者旅程的映射，以識別所有必要的功能及其對應的使用者案例	產品擁有者
使用者案例映射：識別使用者流程中的差距或缺少的步驟	業務分析師
使用者案例映射：根據使用者案例對商業價值的影響來排定其優先順序	產品管理員
報告和分析：產生近乎即時的儀表板，以視覺化關鍵專案指標，例如衝刺速度和問題解決率	專案經理
報告和分析：分析歷史資料並預測未來的專案結果，例如潛在的延遲或瓶頸	專案經理
報告和分析：建立針對不同利益相關者量身打造的自訂報告，例如團隊績效或專案狀態報告	專案經理
產品藍圖管理：建立和維護概述主要里程碑和發行日期的產品藍圖	專案經理
產品藍圖管理：根據專案優先順序或時間表的變更來更新藍圖	產品管理員
產品藍圖管理：與利益相關者共用藍圖，以提供產品方向的可見性	產品管理員
意見回饋迴圈：在每次衝刺後從團隊收集意見回饋，並識別需要改進的領域	Scrum Master
回顧性：將意見回饋轉換為下一個衝刺的可行項目，推動持續改進	Scrum Master
追溯性：追蹤從先前追溯性實作的變更的影響，以衡量其有效性	Scrum Master

用於需求管理的生成式 AI 使用案例

需求管理是與專案管理密切相關的重要程序。假設產品擁有者使用 AI 工具來分析客戶意見回饋、市場趨勢和利益相關者輸入。AI 工具可以產生一組完整的使用者案例和要求、自動分類它們、偵測潛在的衝突或差距，甚至根據業務價值和實作複雜性建議優先順序。隨著專案的進展和需求發展，AI 可以持續更新和精簡需求，以確保它們與不斷變化的業務需求和技術限制保持一致。這種動態、AI 驅動的需求管理方法有助於確保開發工作在整個專案生命週期中與使用者需求和業務目標保持緊密一致。

下表顯示您可以使用生成式 AI 和負責這些使用案例的角色來增強的需求管理使用案例。

使用案例	角色
建立業務需求	業務分析師
從 功能建立 epics	產品擁有者
透過監控其相關聯使用者案例的完成情況，追蹤史詩級的進度	產品管理員
建立使用者案例	產品擁有者
估計每個使用案例所需的工作量，並指派案例點	Scrum Master
定義每個使用者案例的接受條件	產品擁有者

適用於架構和設計的生成式 AI 使用案例

憑藉專案管理的堅實基礎和明確定義的需求，下一個關鍵功能是架構和設計。在這裡，生成式 AI 為建立強大、可擴展且高效率的軟體架構提供了新的可能性。採用 AI 的設計工具可以分析需求和限制條件，以建議最佳的架構模式和設計方法。它們會產生多個設計替代方案，每個都針對不同的優先順序進行最佳化，例如效能、可擴展性或可維護性。例如，解決方案架構師可能會使用 AI 助理，根據專案需求快速產生數個高階架構設計。這種 AI 擴增方法可加速設計程序，並協助架構師做出更明智的決策。這會導致更強大且面向未來的軟體設計。

下表顯示您可以使用生成式 AI 和負責這些使用案例的角色來增強的架構和設計使用案例。

使用案例	角色
建立架構文件	解決方案架構師
建立詳細的設計文件	技術主管
了解現有的架構和設計標準	解決方案架構師
開發使用者界面的詳細模擬和原型	UX/UI 設計師

用於協同合作的生成式 AI 使用案例

軟體開發本質上是協作工作。您可以使用生成式 AI 來增強軟體開發團隊的協作。採用 AI 技術的協作工具不僅止於簡單的訊息和檔案共用。他們透過摘要冗長的討論主題、強調關鍵決策，甚至根據團隊成員的排程和生產力模式建議會議的最佳時間，促進更有效的溝通。AI 可以透過自動識別潛在問題、建議改進，甚至向檢閱者解釋複雜的變更，來協助程式碼檢閱。在腦力激盪工作階段期間，AI 可以充當主持人、產生想法、協助組織想法，甚至主持討論，以確保聽到所有聲音。對於分散式團隊，AI 可協助彌補文化和語言障礙。它可以在聊天和視訊通話中提供近乎即時的語言翻譯，並提供文化內容以協助防止誤解。透過增強人類與 AI 的協作，此功能可協助團隊更有效率且有效地工作，進而促進創新並改善整體專案成果。

下表顯示如何使用生成式 AI 來增強協作使用案例。

子功能：使用案例	角色
文件管理：建立和維護集中式文件儲存庫	技術寫入器
文件管理：允許多個團隊成員即時協作處理文件	開發團隊
知識分享：使用討論論壇做為開發人員以協作方式提出問題、分享知識和疑難排解問題的平台	開發團隊
知識分享：使用討論論壇記錄和追蹤專案討論期間所做的決策，確保擷取並存取關鍵決策背後的理由以供日後參考	產品管理員
專案資產管理：協助輕鬆共用專案相關資源	開發團隊

子功能：使用案例	角色
專案資產管理：針對共用內容實作版本控制，讓團隊成員可以追蹤變更、還原至先前的版本，以及協同處理內容更新	開發團隊

DevSecOps 的生成式 AI 使用案例

採用 AI 技術的 DevSecOps 工具可自動化軟體交付管道的許多層面。例如，當開發人員編寫程式碼時，他們可以執行智慧程式碼檢閱、偵測潛在錯誤、偵測安全漏洞，以及近乎即時地識別效能問題。AI 會產生並執行全面的測試套件，並隨著程式碼庫的演進自動更新這些套件。這種 AI 擴增的 DevSecOps 方法可加速交付管道，並大幅增強交付軟體的安全性和可靠性。

下表顯示您可以使用生成式 AI 和負責這些使用案例的角色來增強的 DevSecOps 使用案例。

子功能：使用案例	角色
DevOps 和持續交付：自動化整個部署管道	DevOps 工程師
DevOps 和持續交付：接收有關程式碼品質和潛在問題的近乎即時意見回饋	軟體開發人員
DevOps 和持續交付：接收近乎即時的安全問題和修補建議	軟體開發人員
DevOps 和持續交付：接收近乎即時的程式碼和最佳實務建議	軟體開發人員
DevOps 和持續交付：自動化重複性任務，並將命令整合到指令碼中	DevOps 工程師
DevOps 和持續交付：在每個程式碼遞交後自動建置程式碼並產生成品	軟體開發人員
DevOps 和持續交付：根據組織的標準和架構建置程式碼	軟體開發人員

子功能：使用案例	角色
DevOps 和持續交付：自動對每個遞交執行單元測試，以在開發程序的早期發現錯誤	軟體開發人員
DevOps 和持續交付：分析單元測試的涵蓋範圍，以確保所有關鍵程式碼路徑都經過測試	軟體開發人員
DevOps 和持續交付：管理分支和合併變更	軟體開發人員
DevOps 和持續交付：管理程式碼和成品版本控制	軟體開發人員
DevOps 和持續交付：存放和管理建置成品和相依性	DevOps 工程師
DevOps 和持續交付：在建置過程中解決和擷取相依性	軟體開發人員
DevOps 和持續交付：產生並執行整合測試，以確保元件可如預期一起運作	測試工程師
DevOps 和持續交付：在整合測試期間使用模擬服務來模擬與外部系統的互動	測試工程師
DevOps 和持續交付：在不同負載下的基準應用程式效能	效能工程師
DevOps 和持續交付：模擬高流量案例，以測試應用程式的可擴展性和回應時間	效能工程師
DevOps 和持續交付：測試系統從故障中復原的能力，例如伺服器當機或網路中斷	站點可靠性工程師
DevOps 和持續交付：執行混沌工程	站點可靠性工程師
DevOps 和持續交付：執行測試以確認應用程式符合業務需求	QA 工程師
DevOps 和持續交付：執行使用者接受度測試	產品擁有者

子功能：使用案例	角色
DevOps 和持續交付：掃描相依性是否有漏洞和授權合規問題	安全工程師
DevOps 和持續交付：監控和管理開放原始碼相依性，以確保它們是最新且安全的	安全工程師
DevOps 和持續交付：產生和維護軟體物料清單 (SBOM)，以追蹤所有元件和相依性	安全工程師
DevOps 和持續交付：使用 SBOM 進行合規稽核	合規主管
DevOps 和持續交付：建立版本備註	版本管理員
DevOps 和持續交付：規劃和協調版本	版本管理員
DevOps 和持續交付：實作復原和發行管理的標準操作程序	版本管理員
DevOps 和持續交付：使用功能旗標在生產環境中啟用或停用功能，而無需部署新程式碼	產品管理員
DevOps 和持續交付：使用功能旗標執行 A/B 測試，以測量不同功能對使用者行為的影響	產品管理員
DevOps 和持續交付：分析和監控管道故障	DevOps 工程師
DevOps 和持續交付：建立和管理基礎設施資源	DevOps 工程師
DevOps 和安全性：掃描程式碼儲存庫以取得硬式編碼的秘密	DevOps 工程師
DevOps 和安全性：實作近乎即時的偵測，在秘密遞交至儲存庫時立即提醒開發人員	DevOps 工程師
DevOps 和安全性：強制執行持續的程式碼品質監控	軟體開發人員

子功能：使用案例	角色
DevOps 和安全性：偵測和標記程式碼中潛在安全漏洞的指標	軟體開發人員
DevOps 和安全性：實作 Open Worldwide Application Security Project (OWASP) 前 10 個安全風險的自動化測試，以確保應用程式符合業界標準的安全實務	安全工程師
DevOps 和安全性：透過將檢查整合到開發程序中，定期更新和教育開發人員 OWASP 風險	安全工程師
DevOps 和安全性：掃描第三方程式庫和相依性是否有已知的安全漏洞	DevOps 工程師
DevOps 和安全性：掃描應用程式程式碼和基礎設施以偵測漏洞	DevOps 工程師
DevOps 和安全性：在部署之前分析漏洞的程式碼	安全工程師
DevOps 和安全性：透過防止具有關鍵漏洞的程式碼合併來強制執行安全政策	安全工程師
DevOps 和安全性：實作角色型存取控制 (RBAC)，以限制對敏感系統和資料的存取，並確保只有獲得授權的人員才能存取關鍵資源	安全工程師
DevOps 和安全性：根據角色和責任調整存取控制，方法是適應團隊結構中的變更	DevOps 工程師
DevOps 和安全性：透過模擬生產環境的攻擊，近乎即時地測試執行中的應用程式是否有安全漏洞	安全工程師
DevOps 和安全性：持續監控部署的應用程式是否有安全漏洞	DevOps 工程師

子功能：使用案例	角色
DevOps 和安全性：排定所有環境的定期漏洞掃描，以識別和解決安全性漏洞	安全工程師
DevOps 和安全性：根據漏洞掃描結果套用修補程式和更新，以協助維護安全系統	DevOps 工程師
應用程式效能監控：近乎即時地持續監控應用程式效能，以在效能問題影響使用者之前對其進行偵測和診斷	站點可靠性工程師
應用程式效能監控：偵測效能異常，例如回應時間突然遽增或錯誤率提高，並啟動警示	DevOps 工程師
應用程式效能監控：追蹤透過分散式系統傳播的請求，以識別效能瓶頸和延遲問題	DevOps 工程師
應用程式效能監控：使用分散式追蹤來精確找出負責失敗或效能降低的確切服務或元件	DevOps 工程師
日誌彙總和分析：將多個來源的日誌彙總到集中式系統，以便於搜尋和分析，以識別趨勢和問題	站點可靠性工程師
日誌彙總和分析：實作自動日誌剖析，以擷取相關資訊並偵測可能指出問題的模式或異常	DevOps 工程師
日誌彙總和分析：收集和視覺化關鍵效能指標	站點可靠性工程師
日誌彙總和分析：根據預先定義的服務層級協議 (SLAs) 監控指標	產品管理員
AI 操作：偵測事件、分析根本原因，以及啟動修正動作，無需人工介入	DevOps 工程師
AI 操作：預測未來資源需求並最佳化容量規劃，以避免中斷	站點可靠性工程師
持續改進：監控與應用程式的真實使用者互動，以收集有關效能的洞見並識別需要改進的領域	UX 設計工具

子功能：使用案例	角色
持續改進：追蹤不同地理區域的應用程式效能，以確保全球一致的使用者體驗	產品管理員
儀表板監控：建立可自訂的儀表板，以近乎即時的方式視覺化關鍵指標、日誌和追蹤，以提供系統運作狀態的完整檢視	站點可靠性工程師
儀表板監控：為不同的團隊（例如開發、營運和產品團隊）建立儀表板，以根據其重點領域提供相關洞見	DevOps 工程師
效能洞察：對應用程式效能進行詳細分析，以識別效率低下並最佳化程式碼或基礎設施	軟體開發人員
效能洞見：使用效能洞見反覆改善應用程式效能，並隨著時間最佳化使用者體驗	產品管理員

用於操作和維護的生成式 AI 使用案例

在部署軟體之後，焦點會轉移到操作和維護。生成式 AI 可以透過提供更主動、更有效率的系統管理來增強傳統方法。AI 驅動的操作工具會持續監控系統效能，並在影響使用者之前預測潛在問題。當問題發生時，它們會執行自動化根本原因分析，大幅縮短解決的平均時間。AI 也會近乎即時地最佳化系統效能。它會根據不斷變化的負載模式和使用者行為自動調整組態。例如，營運團隊可能會使用 AI 助理來產生預測性維護排程、自動識別可能失敗的元件，並建議先佔動作。AI 也可以透過分析用量趨勢並以高準確度預測未來資源需求，協助規劃容量。

下表顯示您可以使用生成式 AI 和負責這些使用案例的角色來增強的操作和維護使用案例。

子功能：使用案例	角色
事件管理：透過將監控工具與聊天平台整合，讓團隊可以直接在聊天環境中偵測、討論和解決問題，以近乎即時的方式管理事件	站點可靠性工程師
事件管理：允許團隊直接從聊天界面啟動部署、執行指令碼和執行命令，以簡化操作	DevOps 工程師

子功能：使用案例	角色
程式碼升級：升級程式碼相依性和程式庫，以減少手動工作，並確保程式碼庫與最新版本保持最新狀態	軟體開發人員
程式碼最佳化：檢閱程式碼以取得最佳化機會	軟體開發人員
程式碼最佳化：識別程式碼中的瓶頸，並重構或最佳化程式碼以增強效能	軟體開發人員
技術負債管理：將技術負債記錄為開發程序的一部分	產品管理員
技術債務管理：根據影響、風險和成本來優先考慮和解決技術債務，並將其整合到定期衝刺規劃程序中	軟體開發人員
技術負債管理：減少現有應用程式程式碼中的技術負債	軟體開發人員
變更管理：實作變更核准程序，確保所有程式碼變更在部署之前都經過必要的利益相關者的檢閱、測試和核准	變更管理員
變更管理：對提議的變更執行影響分析	DevOps 工程師
反向工程：分析和了解舊版程式碼的結構和行為	解決方案架構師
反向工程：說明現有程式碼並產生文件	軟體開發人員
程式碼現代化：將程式碼從一種程式設計語言轉換為另一種程式設計語言	軟體開發人員
程式碼現代化：將舊版程式碼現代化為最新的程式設計語言	軟體開發人員
效能最佳化：透過最佳化資源配置、負載平衡和重新設定應用程式，持續監控和調整系統效能	站點可靠性工程師

子功能：使用案例	角色
效能最佳化：識別和重構導致效能降低的程式碼，以提高速度和系統回應能力	軟體開發人員

軟體開發中生成式 AI 助理的使用案例

AI 助理功能是生成式 AI 驅動開發體驗的核心。這個智慧型、內容感知系統可做為整個 SDLC 所有團隊成員的虛擬協作者。試想一名開發人員正在處理複雜的程式碼。他們可以簡單地向 AI 助理尋求協助，並且可以提供相關的程式碼片段、解釋複雜的演算法，甚至根據目前的內容和最佳實務建議最佳化。AI 助理可協助 ITOps 管理員根據內部文件了解標準操作程序。透過提供即時的情境式支援，AI 助理可大幅降低團隊成員的認知負載。這有助於他們專注於更高階的問題解決和創意任務。此功能可做為力乘數，增強軟體開發所有階段的生產力和品質。

下表顯示您可以使用 AI 助理和受益角色增強的使用案例。

使用案例	角色
透過回答有關需求、架構和標準操作程序等問題，為開發團隊提供即時協助	軟體開發團隊
從廣泛的文件中搜尋或擷取摘錄，或使用自然語言查詢產生摘要	軟體開發團隊
摘要長的技術文件，例如需求文件、架構設計文件和內部程序	軟體開發團隊
維護提示程式庫，供團隊用於常見任務	軟體開發團隊
將生成式 AI 無縫整合到現有的工具和系統中	軟體開發團隊
自動化跨各種平台、工具和內部系統的任務	軟體開發團隊
建立集中式的知識儲存庫，包括最佳實務、專案特定資訊和團隊知識，可供所有團隊成員存取	軟體開發團隊
根據任務的內容，從儲存庫擷取相關知識	軟體開發團隊

使用案例	角色
執行自動化程式碼檢閱、根本原因分析、建議改進、偵測潛在錯誤，以及執行故障診斷	軟體開發人員、DevOps 工程師和網站可靠性工程師
分析效能資料以識別趨勢和模式，這些趨勢和模式可為效能最佳化的決策提供資訊	站點可靠性工程師
提供改善效率、降低複雜性和增強安全性的建議	軟體開發人員
建議最佳化雲端資源用量，例如擴展建議或節省成本的策略	軟體開發人員、DevOps 工程師、網站可靠性工程師和解決方案架構師
產生新內容，例如以程式碼、使用者指南或產品功能版本為基礎的文件	軟體開發團隊

用於分析和洞察的生成式 AI 使用案例

分析和洞見功能有助於將大量資料轉換為可採取行動的洞見，以推動決策和持續改進。透過使用生成式 AI，此功能可處理各種來源的資料，包括程式碼儲存庫、專案管理工具和團隊協作平台，以提供開發程序和團隊生產力的整體檢視。生成式 AI 超越傳統指標，以提供預測和方案分析。它可以預測潛在問題，並建議有針對性的改善。例如，它可以分析程式碼遞交、錯誤解決率和特徵交付速度中的模式，以識別高效能團隊、找出瓶頸，並建議程序最佳化。此外，它可以提供團隊動力和個人績效的洞見。這些洞察有助於領導者對工作負載分佈、訓練需求和團隊組成做出資料驅動型決策。透過互動式儀表板呈現這些洞見，此功能可讓所有層級的利益相關者做出明智的決策、最佳化程序，並持續提高團隊生產力，進而加快交付高品質軟體的速度。

下表顯示您可以使用生成式 AI 和負責這些使用案例的角色來增強的分析使用案例。

使用案例	角色
監控個人和團隊的生產力	開發管理員
分析生產力趨勢以偵測潛在的倦怠，讓您可以採取主動措施來維護團隊的良好狀態和生產力	開發管理員
追蹤程式碼變更部署到生產環境的頻率，以衡量開發程序的速度和敏捷性	產品管理員

使用案例	角色
分析部署頻率資料，以識別低部署活動的期間，這可能表示程序效率低下或資源限制	產品管理員
測量程式碼遞交至部署之間的時間，以識別簡化開發和部署程序的機會	開發管理員
追蹤導致需要立即修復之失敗的部署百分比，以評估發行程序的可靠性	站點可靠性工程師
使用變更失敗率指標來識別經常導致問題的程式碼區域，以引導有針對性的重構和測試工作	軟體開發人員
監控中斷或事件發生後還原服務所需的時間，以便減少停機時間並改善整體系統彈性	站點可靠性工程師
分析還原時間的趨勢，以增強事件回應程序，並加快從系統故障中復原的速度	DevOps 工程師
建立自訂儀表板來彙總關鍵指標，例如部署頻率、前置時間和變更失敗率，以提供開發和營運運作狀態的完整檢視	產品管理員
建立專為不同團隊需求量身打造的儀表板，以針對其特定責任領域提供重點洞察，例如開發、營運或業務	產品管理員
追蹤業務關鍵績效指標 KPIs)，例如營收影響、客戶滿意度和市場佔有率，使開發工作與更廣泛的業務目標保持一致	產品管理員
分析新功能對業務 KPIs的影響，以評估其成功並引導未來的產品開發	業務分析師
監控程式碼品質指標，例如程式碼複雜性、測試涵蓋範圍和錯誤密度，以確保程式碼庫保持可維護且安全	軟體開發人員

使用案例	角色
識別需要重構的程式碼庫區域，以推動長期永續性並減少技術債務	解決方案架構師

用於知識管理的生成式 AI 使用案例

在任何軟體開發組織中，知識都是關鍵資產。知識管理功能採用生成式 AI，可增強此資產的擷取、組織和使用方式。傳統的知識管理系統通常包含太多資訊、包含過時的內容，或難以搜尋以快速尋找相關資訊。

生成式 AI 可解決這些挑戰。它會根據程式碼變更、對話和專案成品自動產生和更新文件。這可確保知識庫保持最新狀態，而不需要團隊成員的手動工作。更重要的是，AI 可讓您以直覺的方式存取這些知識。團隊成員可以自然語言提出問題，AI 可以提供相關的答案。AI 可以利用各種來源，例如官方文件、程式碼評論、討論執行緒，甚至是外部資源。例如，嘗試了解特定元件的新團隊成員可能會詢問 AI：「身分驗證模組如何運作？」 然後，AI 會提供相關程式碼區段、架構圖和最近變更的簡潔說明和連結。它甚至可以根據團隊成員的角色和專業水準來量身打造此資訊。

此功能可加速加入、減少重複性問題，並提升整個組織的知識分享。它有助於保留機構知識，讓團隊隨著時間的推移更輕鬆地維護和發展複雜的系統。

下表顯示知識管理使用案例，您可以使用生成式 AI 和負責這些使用案例的角色來增強這些案例。

使用案例	角色
建立統一的平台，讓您輕鬆存取所有專案相關的知識	軟體開發團隊
從各種開發活動擷取知識	軟體開發團隊
提供進階搜尋功能，以在儲存庫中快速尋找相關知識	軟體開發團隊
為團隊個人化學習模組和途徑	軟體開發團隊

擴充性的生成式 AI 使用案例

可擴展性可無縫整合現有工具和工作流程，同時讓組織根據其特定需求量身打造 AI 系統。此功能提供強大的 APIs、SDKs和可自訂的界面，有助於將 AI 功能整合到熱門的開發和專案管理工具中。例如，組織可以使用 AI 支援的功能來增強 Jira，以實現自動化票證優先順序、工作量估算和衝刺規劃。您可以使用 AI 增強 Jenkins 管道，以進行智慧型建置最佳化和預測測試選擇。

此外，可擴展性允許與整合開發環境 (IDEs)、版本控制系統和程式碼檢閱平台進行深度整合。AI 可協助程式碼、自動化程式碼檢閱，以及產生情境文件。

此功能也支援針對組織特定資料訓練和微調 AI 模型。這有助於 AI 了解公司特定的編碼模式、架構偏好設定和領域知識。結果在所有整合工具中具有更相關和內容感知的協助。透過提供這種程度的靈活性和整合，可擴展性可確保 AI 驅動的開發體驗隨著組織的發展而演進。它可以適應不斷變化的技術和業務需求，同時無縫地增強現有的工具鏈和工作流程。

下表顯示可擴展性使用案例，您可以使用生成式 AI 和負責這些使用案例的角色來增強這些使用案例。

使用案例	角色
將第三方工具整合到開發環境	DevOps 工程師
建立自訂自動化工作流程，根據團隊的獨特開發程序量身打造	DevOps 工程師
連線至各種 APIs和服務	DevOps 工程師
建立跨平台工具的連接器	DevOps 工程師

在軟體開發中使用生成式 AI 的最佳實務

本節說明將生成式 AI 整合到軟體開發生命週期 (SDLC) 的最佳實務。從實作無縫工具鏈和 DevSecOps 管道，到促進協作和自動化重複性任務，這些指導方針可協助您利用 AI 的力量來增強開發程序和體驗。透過遵循這些最佳實務，軟體開發團隊可以在工作中釋放新的效率、創新和品質水準。

本節討論下列最佳實務：

- 實作無縫的end-to-end整合工具鏈
- 實作適用於 DevSecOps end-to-end CI/CD 管道
- 採用協作工具和實務
- 自動化重複性任務
- 定期檢閱和反覆查看開發體驗
- 採用有效的專案管理實務
- 實作知識管理
- 提供可擴展性和自訂
- 最佳化 操作
- 使用資料驅動的洞察
- 採用以平台為基礎的方法

實作無縫的end-to-end整合工具鏈

實作無縫的end-to-end整合工具鏈，是建立生成式 AI 驅動開發體驗的基礎最佳實務。核心想法是建立工具和平台的凝聚生態系統，您的軟體團隊可以在整個 SDLC 中使用。團隊可以使用工具鏈來規劃、構想、程式碼、建置、測試、部署和管理持續操作。透過將生成式 AI 功能整合到此工具鏈中，您可以確保每個階段都提供 AI 協助。此整合可減少或消除手動交接、減少內容切換，並協助資料和成品在不同開發階段之間順暢流動。例如，來自整合開發環境 (IDE) 的 AI 產生的程式碼片段可以順暢地流入您的版本控制系統，而來自部署平台的 AI 驅動分析可以通知您的專案管理工具。這會建立持續的意見回饋迴圈，以改善您的開發程序。

實作適用於 DevSecOps end-to-end CI/CD 管道

若要建置此整合工具鏈，請實作適用於 DevSecOps end-to-end持續整合和持續部署 (CI/CD) 管道。這個採用 AI 技術的管道是簡化軟體交付程序的關鍵元件。它可協助您更快速且可靠地發佈新的應用程

式和更新。透過在整個 SDLC 中嵌入安全實務，您可以更早識別和解決漏洞，從而降低整體成本和風險。管道應在每個階段納入 AI，從持續整合和測試到安全檢查和部署。例如，您可以使用 AI 近乎即時地分析程式碼遞交，以便在可能發生整合問題之前預測問題。在 CI/CD 管道中，您也可以使用生成式 AI 根據最新的威脅情報自動更新安全政策。

採用協作工具和實務

當您強化開發基礎設施時，請不要忘記人工元素。軟體開發本質上是協作工作。它涉及由開發人員、設計師、產品經理、Scrum Masters、商業分析師和其他利益相關者組成的跨職能團隊。這些人共同努力，將想法帶入成果。透過使用現代協作工具並培養開放溝通和知識分享的文化，您可以大幅提升軟體開發團隊的生產力和效率。在採用 AI 技術的軟體開發體驗中，這些工具會採用新的維度。您可以將 AI 整合到協作平台中，以促進團隊成員之間更有效的溝通和知識分享。AI 助理可以回答常見問題、總結討論，甚至調解衝突。生成式 AI 可以透過自動建議改進或識別潛在問題來增強程式碼檢閱程序。此外，您可以使用 AI 建立動態的內容感知文件，隨著專案的演進而近乎即時更新，讓所有團隊成員都能存取最新且相關的資訊。

自動化重複性任務

透過使用生成式 AI 來處理例行、耗時的活動，您可以讓軟體團隊專注於推動創新和提供業務影響的高價值、創造性工作。重複性任務的範例包括產生樣板程式碼、建立測試資料、撰寫文件，甚至草擬初始專案計劃。透過將這些任務卸載至 AI，團隊成員可以專注於更具創造力和更具策略的工作。例如，AI 支援的程式碼完成工具可以根據內容和編碼模式建議相關程式碼片段，藉此大幅加速編碼程序。同樣地，生成式 AI 可以在程式碼變更時自動建立和更新技術文件。這可讓文件保持最新狀態，並減少此任務通常需要的手動工作量。在測試中，AI 可以根據需求和程式碼分析產生全面的測試案例，從而改善測試涵蓋範圍並降低忽略邊緣案例的可能性。透過智慧地自動化這些重複性任務，生成式 AI 可加速開發時間表、提高一致性並減少人為錯誤。結果是更高品質的軟體輸出。

定期檢閱和反覆查看開發體驗

您的軟體開發體驗本身應視為需要持續改進的產品。這包括建立系統性程序，以定期檢閱和反覆查看開發生命週期、工具和實務的所有層面。對整個工具鏈、工作流程和程序執行定期評估。收集所有團隊成員跨各種角色的意見回饋，包括產品經理、設計人員、架構師、開發人員、測試人員和營運人員。要求他們找出困擾點、瓶頸和增強的機會。例如，團隊可能會每季審查其 CI/CD 管道效能，並分析建置時間、部署頻率和錯誤率等指標，以識別要最佳化的領域。由於生成式 AI 功能持續快速發展，因此請務必持續評估新的 AI 工具和功能，以進一步簡化 SDLC 中所有角色的工作流程或增強功能。

採用有效的專案管理實務

若要有效地協調複雜的軟體開發工作，請採用 AI 增強型專案管理實務。在這種情況下，有效的專案管理超越了傳統方法。它採用 AI 擴增方法，可增強整個 SDLC 的規劃、執行和監控。敏捷架構可提升彈性、協同合作和快速迭代，而且您可以使用生成式 AI 來最佳化這些程序。例如，生成式 AI 可以分析歷史專案資料，以取得更準確的預估值、根據業務目標和客戶意見回饋自動產生使用者案例並排定其優先順序，以及提供團隊績效的智慧洞見。採用 AI 技術的專案管理工具可以預測潛在的障礙，並根據團隊成員的技能和工作負載建議最佳的任務指派。透過將 AI 驅動的功能整合到專案管理實務中，您可以實現更高的可見性、更快地制定資料驅動型決策，並確保團隊成員符合並有效率地朝共同目標邁進。

實作知識管理

隨著 AI 驅動的軟體開發體驗成熟，請實作強大的知識管理系統。強大的知識管理系統可協助您擷取、組織和授予對寶貴洞見、最佳實務和解決方案的存取權。跨 SDLC 的所有團隊成員都應能夠輕鬆存取系統。使用生成式 AI 建立動態、智慧的知識庫，隨著您的組織而演進。例如，AI 可以根據程式碼變更、對話和專案成品自動產生和更新文件，讓資訊保持最新狀態，無需手動介入。生成式 AI 也可以支援智慧型搜尋功能，並使用自然語言查詢協助團隊成員快速找到相關資訊，即使他們不知道確切的術語。此外，生成式 AI 可以根據團隊成員目前的任務或挑戰，主動向他們展示相關資訊。它充當虛擬指導者，可增強所有角色的決策和問題解決能力。透過實作 AI 驅動的知識管理系統，您可以打破孤島、加速加入、減少多餘的工作，並在整個軟體開發團隊中培養持續學習和創新的文化。

提供可擴展性和自訂

為了最大限度地提高軟體開發中生成式 AI 的優勢，請確保您的 AI 工具和平台具有可擴展性和可自訂性。這可協助您根據您的特定需求、工作流程和技術堆疊量身打造 AI 功能。例如，您可以在自己的程式碼庫和文件上微調 AI 模型、為特定任務建立自訂 AI 支援的工具，或將 AI 功能整合到現有的工具和程序中。此可擴展性可協助您發展 AI 驅動的開發體驗，以滿足組織不斷變化的需求。它還可協助您針對特定網域或專案類型最佳化體驗。

最佳化 操作

生成式 AI 在最佳化軟體操作和維護方面扮演重要角色。將 AI 功能整合到您的操作工具和程序中，以最佳化 操作。例如，使用生成式 AI 近乎即時地分析日誌資料、預測潛在的系統故障，以及自動化例行維護任務。生成式 AI 也可以透過將事件關聯到複雜的分散式系統，協助進行根本原因分析。這可改善系統可靠性、減少停機時間，並讓您的營運團隊專注於更具策略性的計畫。

使用資料驅動的洞察

在 AI 驅動的開發旅程中使用資料驅動的洞見。實作系統以收集、分析和處理 SDLC 所有階段的資料。這包括程式碼指標、測試結果、部署資料、使用者意見回饋和操作效能。使用生成式 AI 來發現人類觀察者可能不明顯的模式和洞見。然後，將這些洞見傳回您的開發程序，以通知從架構決策到特徵優先順序的所有資訊。

採用以平台為基礎的方法

若要完全實現軟體開發中生成式 AI 的優勢，請採用以平台為基礎的方法。建立全方位的整合平台，將 AI 功能整合到 SDLC 的各個層面。平台應提供一致的使用者體驗、集中式管理和資料，以及不同工具和程序之間的無縫整合。這可讓 AI 優勢在整個組織中統一可用，減少管理多個不同 AI 工具的額外負荷，並為 AI 功能的持續改善和擴展奠定基礎。

在軟體開發中衡量生成式 AI 的成功

若要有效衡量實作生成式 AI 驅動軟體開發體驗的效果，您需要建立一組全面的指標，這些指標橫跨軟體開發生命週期 (SDLC) 的各個維度。這些指標應擷取效率和生產力的立即改善，並反映軟體品質、團隊滿意度和商業價值的長期收益。

請執行下列動作，以有效地使用本節中建議的指標：

1. 建立基準 – 在深入實作採用 AI 技術的開發體驗之前，請花時間收集有關這些指標目前效能的完整資料。這提供了明確的起點，並可協助您稍後進行有意義的比較。
2. 設定逼真的目標 – 準備好基準後，為每個指標設定可實現的改善目標。要有遠大但逼真的。請記住，永續進度通常是增量的。
3. 實作持續監控 – 使用自動化工具，在您的環境中持續收集和分析這些指標的資料。近乎即時的監控可協助您監控進度，並快速識別任何問題或機會。
4. 執行定期審查 – 排定每季或每兩年的審查工作階段，您和您的團隊會徹底評估您的目標進度。使用這些工作階段來識別需要進一步改進的領域，並慶祝您的成功。
5. 反覆運算和調整 – 根據您所取得的洞見，持續精簡生成式 AI 實作，並視需要調整目標。

本節說明下列指標類別：

- 部署速度
- 程式碼品質
- 運營效率
- 團隊生產力和滿意度
- 業務影響

部署速度

請考慮測量下列部署速度指標。

指標	Description
上市時間	測量從構想到生產部署的時間減少

指標	Description
衝刺速度	追蹤您的團隊每次衝刺完成的故事點增加
程式碼遞交頻率	監控程式碼遞交的增加，這表示加速開發週期
提取請求解決時間	評估檢閱和合併儲存庫中程式碼變更所花費的時間減少
發行速度	測量每季或每年發行次數的增加

程式碼品質

請考慮測量下列程式碼品質指標。

指標	Description
瑕疵密度	測量軟體錯誤的減少
程式碼涵蓋範圍	追蹤整個程式碼庫的測試涵蓋範圍百分比增加
技術負債	監控已識別技術負債隨著時間的減少
靜態程式碼分析分數	根據您的自動化分析工具評估程式碼品質的改善

運營效率

請考慮測量下列操作效率指標。

指標	Description
部署頻率	測量成功部署的數量增加
復原的平均時間 (MTTR)	追蹤從系統故障中復原所需的時間減少
變更失敗率	監控導致部署失敗的變更百分比減少

團隊生產力和滿意度

請考慮測量下列團隊生產力和滿意度指標。

指標	Description
提高生產力	監控每個任務的生產力百分比增加
滿意度分數	定期進行調查，以衡量團隊士氣和工作滿意度的改善
知識分享效率	測量您的團隊花費在搜尋資訊或詢問重複性問題的時間減少
加入時間	追蹤新團隊成員提高生產力所需的時間減少

業務影響

請考慮測量下列業務影響指標。

指標	Description
功能採用率	使用您發行的新功能來衡量使用者參與度的增加
客戶滿意度分數	追蹤使用者意見回饋和評分的改善
收入影響 （直接和間接）	評估營收增加歸因於發行速度提高或生產力提高

結論

此策略文件提供生成式 AI 驅動軟體開發體驗的概觀。它會探索 5-I 架構中的五個維度：調查、整合、互動、反覆運算和影響。這些維度提供在整個軟體開發生命週期 (SDLC) 中整合生成式 AI 的策略藍圖。它還描述了成功實作此架構所需的基本功能。這些功能橫跨專案管理、DevSecOps、AI 助理、知識管理等領域。它提供整合生成式 AI 時要考慮的最佳實務，並可協助您使用指標來衡量生成式 AI 對軟體開發體驗的影響。

將生成式 AI 與軟體開發程序整合，代表了可能加速創新、改善品質並提高生產力的範式轉移。不過，請務必確認這不是一次性實作。這是一個持續的演變，需要持續的努力和持續的改進。

當您開始此旅程時，我們建議您從對組織目前功能和整備度的全面評估開始。AWS 評估工具是一種 AI 驅動的軟體開發評估工具，可協助您識別優先順序領域並建立量身打造的實作藍圖。

資源

識別關鍵優先順序領域後，下列資源可協助您實作藍圖：

AWS 文件

- 使用 Amazon Bedrock 自動化 AWS 基礎設施操作 (AWS 方案指引)
- 使用 Amazon Q Developer 產生內嵌和助理程式碼的最佳實務 (AWS 方案指引)
- 使用 Amazon Bedrock 代理程式和知識庫開發全自動化聊天式助理 (AWS 方案指引)
- AWS 使用生成式 AI 在 上轉換應用程式開發和維護操作模型 (AWS 方案指引)
- 使用 Amazon Q Developer 作為編碼助理，以提高您的生產力 (AWS 規範性指導)

AWS 部落格文章和教學課程

- Amazon Q 部落格文章
- 使用 Amazon Q 加速您的軟體開發生命週期 (AWS 部落格文章)
- 建置解決方案 AWS 架構師 AI 代理程式：利用 Amazon Bedrock 進行自動化架構和部署 (AWS 影片)
- 生成式 AI 技術操作 (AWS 部落格文章)
- 使用 Amazon Q Developer 現代化 Java 應用程式 (AWS 部落格文章)
- 使用 Amazon Bedrock 在您的軟體開發管道中產生、評估和了解程式碼 (AWS 部落格文章)

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	不適用	2025 年 4 月 18 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

在整個資料生命週期中追蹤資料的來源和歷史記錄的程序，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 擴展了最初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 上工作負載的災難復原 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界限會限制故障的影響，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。少量的提示對於需要特定格式、推理或網域知識的任務來說非常有效。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，已針對廣義和未標記資料的大量資料集進行訓練。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。

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，並有助於改善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)

安全機制可篩選、驗證和限制代理程式輸入和輸出，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，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。

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

將基礎設施視為程式碼。

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。

MCP

請參閱模型內容通訊協定。

模型內容通訊協定 (MCP)

適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)

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 建議使用不可變的基礎設施作為最佳實務。

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)

問題及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 中的操作準備度審查 (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，OT 和資訊技術 (IT) 系統的整合是工業 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、使用 AWS KMS (SSE-KMS) 的伺服器端加密 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。

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

將資料集中的個人識別符取代為預留位置值的程序。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 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 Rs。

Replatform

請參閱 7 Rs。

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

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。矩陣名稱衍生自矩陣中定義的責任類型：負責人 (R)、責任 (A)、已諮詢 (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 秘密中的內容？ 在 Secrets Manager 文件中。

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)

服務效能層面的測量，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，多次讀取。

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。