AWS Well-Architected Framework

卓越營運支柱

卓越營運支柱: AWS Well-Architected Framework

Table of Contents

卓越營運支柱 - AWS Well-Architected Framework

出版日期： 2023 年 10 月 3 日 (文件修訂)

本白皮書的重點是 AWS Well-Architected Framework 的卓越營運支柱。當中提供了相關指引，可協助您在設計、交付和維護 AWS 工作負載時套用最佳實務。

簡介

此 AWS Well-Architected Framework 可協助您了解在 AWS 上建置工作負載時所做決策的效益和風險。透過使用該架構，您將了解關於在雲端設計和操作可靠、安全、有效率、經濟實惠且永續的工作負載的營運和架構最佳實務。藉助該架構，即可根據最佳實務一致地量測營運和架構，並找出需要改進的方面。我們相信，擁有重視營運的 Well-Architected 工作負載可大幅提高企業成功的可能性。

此架構以六大支柱為基礎：

- 卓越營運
- 安全性
- 可靠性
- 效能達成效率
- 成本最佳化
- 永續性

本白皮書重點介紹了卓越營運支柱，以及如何將其套用為您架構良好的解決方案的基礎。營運被認為是與其支援的業務線和開發團隊截然不同的獨立職能，因此要在此環境中實現卓越營運極具挑戰性。透過採用本白皮書中的實務，您可以建置能夠洞見營運狀態的架構、實現有效的營運和事件回應，並且可以繼續改善和支援您的業務目標。

本白皮書適用於擔任技術職務的人員，例如技術長 (CTO)、架構師、開發人員和營運團隊成員。閱讀本白皮書之後，您將了解在設計卓越營運雲端架構時要使用的 AWS 最佳實務和策略。本白皮書並未提供實作的詳細資訊或架構模式。不過，其確實包含有關此資訊的適當資源的參考資料。

卓越營運

在 Amazon，我們將卓越營運視為一項承諾，致力妥善設計軟體，並提供絕佳的客戶體驗。其中包括組織團隊、設計工作負載、大規模運作工作負載，以及促使其隨時間進化的最佳實務。卓越營運有助於讓您的團隊專心設計對客戶有利的新功能，避免將時間耗費在維護與緊急應變上。為了正確建置，我們依循相關最佳實務，為您與團隊造就妥善運作的系統、平衡的工作負載，且最重要的是，為客戶創造絕佳體驗。

卓越營運的總目標是要快速且可靠地為客戶提供新功能和錯誤修正。持續投入卓越營運的組織不僅可滿足其客戶，同時也能打造新功能、勇於改變，並且積極面對失敗。在此過程中，卓越營運可協助開發人員持續達成高品質的成果，進而實現持續整合與持續交付 (CI/CD)。

設計原則

下列設計原則有助於實現雲端中的卓越營運：

- 以程式碼執行營運： 在雲端，您可以在整個環境中套用與您應用程式程式碼所用相同的工程原則。您可將整個工作負載 (應用程式、基礎設施等) 定義為程式碼，並以程式碼加以更新。您可以使用指令碼處理營運程序，並透過啟動這些指令碼來自動化其程序，進而回應事件。透過以程式碼執行營運，您可限制人為錯誤並建立對事件的一致回應。
- 進行頻繁、細微和可逆的變更： 設計可擴展且鬆散耦合的工作負載以允許定期更新元件。自動化部署技術加上較細微的增量變更可縮減影響範圍，並在發生故障時更快地反轉情況。這可讓您更有信心您能為工作負載帶來有益的變更，同時維持品質並迅速適應市場情況的變化。
- 經常改進營運程序： 隨著您工作負載的演進，您的營運也應該適當演進。在使用營運程序時，尋找機會予以改善。保持定期檢閱，並驗證所有程序是否有效以及團隊是否熟悉這些程序。如果發現漏洞，請相應地更新程序。向所有利害關係人和團隊傳達程序更新消息。將營運遊戲化以分享最佳實務並教導團隊。
- 預料失敗： 執行「事前剖析」演練，以識別潛在的失敗來源，進而排除或減少這些來源。測試您的失敗情境，並驗證您理解失敗情境會造成的影響。測試您的回應程序，以確保它們確實有效且團隊熟悉其流程。設定定期演練日，以測試工作負載和團隊對模擬事件的回應。
- 從所有營運失敗中學習經驗： 從所有營運事件和失敗中學習經驗，進而不斷推動改善。跨團隊及在整個組織中分享獲得的經驗。
- 使用受管服務： 盡可能地使用 AWS 受管服務以降低營運負擔。圍繞與這些服務的互動建置營運程序。

- 實作可觀測性以獲得可採取行動的見解： 全面了解工作負載的行為、效能、可靠性、成本和運作狀態。建立關鍵績效指標 (KPI) 並利用可觀測性遙測，以做出明智決策並在業務成果面臨風險時立即採取行動。根據可採取行動的可觀測性資料，主動改善效能、可靠性和成本。

定義

雲端有四種最佳實務領域可實現卓越營運：

- 組織
- 準備
- 營運
- 演進

組織的領導階層定義業務目標。貴組織必須了解要求和優先順序，並運用這些資訊規劃和進行用以幫助達成業務成果的工作。您的工作負載必須提供支援工作負載所需的資訊。透過自動化重複程序的方式實作服務以整合、部署及交付工作負載，將使得更多有益的變更發揮作用。

工作負載的操作本質上就可能存在著風險。您必須了解這些風險，並做出明智的決策才能進入生產階段。您的團隊必須能夠支援您的工作負載。從所需業務成果衍生的業務和營運指標，將協助您了解工作負載的運作狀態、營運活動，並回應事故。您的優先事項會隨著業務需求和業務環境的變化而改變。運用這些方面做為回饋迴圈，以持續推動貴組織的改善和工作負載的操作。

組織

您需要了解組織的優先事項、組織結構，以及組織如何支援您的團隊成員，進而助您實現業務成果。

若要實現卓越營運，您必須了解以下事項：

主題

組織優先事項

您的團隊需要對您的整個工作負載，以及團隊成員在其中的作用達成共識，並且擁有共同的業務目標，以便設定能助力業務成功的優先事項。明確定義的優先事項將實現工作的最大收益。定期審查您的優先事項，以便在組織需求變更時將其更新。

最佳實務

OPS01-BP01 評估外部客戶需求

讓關鍵利害關係人 (包括業務、開發和營運團隊) 參與進來，以確定將工作重點放在哪些外部客戶需求上。這將確保您對實現想要的業務成果所需的營運支援有透徹的了解。

常用的反模式：

- 您已決定不在核心上班時間以外的時間提供客戶支援，但尚未檢閱歷史支援請求資料。您不知道這是否會影響您的客戶。

- 您正在開發新功能，但尚未與客戶互動，以了解是否需要該功能，若需要又應以何種形式提供，而且未進行試驗以驗證交付的需求和方法。

建立此最佳實務的優勢： 需求獲得滿足的客戶更有可能持續回購。評估和了解外部客戶的需求，將讓您了解如何安排工作的優先順序來實現商業價值。

若未建立此最佳實務，暴露的風險等級為： 高

實作指引

- 了解業務需求：只有業務、開發及營運團隊等利害關係人擁有共同的目標並達成共識，方能讓業務取得成功。
  - 審查外部客戶的業務目標、需求和優先事項：與關鍵利害關係人 (包括業務、開發和營運團隊) 進行互動，以討論外部客戶的目標、需求和優先事項。這將確保您對實現業務和客戶成果所需的營運支援有透徹的了解。
  - 建立共識：在以下方面建立共識：工作負載的業務功能、每個團隊在工作負載營運過程中的角色，以及這些因素如何支援內外部客戶的共同業務目標。

資源

相關文件：

- AWS Well-Architected Framework 概念 – 反饋迴圈

OPS01-BP02 評估內部客戶需求

讓關鍵利害關係人 (包括業務、開發和營運團隊) 參與進來，以確定將工作重點放在哪些內部客戶需求上。這將確保您對實現業務成果所需的營運支援有透徹的了解。

利用您制定的優先事項，聚焦於改善作業，因為它們能發揮最大的影響力 (例如，發展團隊技能、改善工作負載效能、降低成本、自動化執行手冊或提升監控力)。根據需求變更更新您的優先順序。

常用的反模式：

- 您已決定在不向產品團隊諮詢的情況下，變更他們的 IP 地址配置，以便更輕鬆地管理網路。您不知道這會對您的產品團隊造成什麼影響。
- 您正在實作新的開發工具，但尚未與內部客戶互動，以了解是否需要該工具或其是否與現有的實務相容。

- 您正在實作新的監控系統，但尚未聯絡內部客戶，以了解他們是否有應該考慮的監控或報告需求。

建立此最佳實務的優勢： 評估和了解內部客戶的需求，將讓您了解如何安排工作的優先順序來實現商業價值。

若未建立此最佳實務，暴露的風險等級： 高

實作指引

- 了解業務需求：只有業務、開發及營運團隊等利害關係人擁有共同的目標並達成共識，方能讓業務取得成功。
  - 審查內部客戶的業務目標、需求和優先事項：與關鍵利害關係人 (包括業務、開發和營運團隊) 進行互動，以討論內部客戶的目標、需求和優先事項。這將確保您對實現業務和客戶成果所需的營運支援有透徹的了解。
  - 建立共識：在以下方面建立共識：工作負載的業務功能、每個團隊在工作負載營運過程中的角色，以及這些因素如何支援內外部客戶的共同業務目標。

資源

相關文件：

- AWS Well-Architected Framework 概念 – 反饋迴圈

OPS01-BP03 評估管控要求

管控是政策、規則或架構的集合，供公司用來達成其商業目標。管控要求產生自您的組織內部。這些要求可能會影響到您所選擇的技術類型，或是您操作工作負載的方式。將組織管控要求納入您的工作負載中。合規是指展現您已實作管控要求的能力。

預期成果：

- 管控要求會併入工作負載的架構設計和操作中。
- 您可以提供您已遵循管控要求的證明。
- 定期審查並更新管控要求。

常見的反模式：

- 您的組織規定根帳戶需進行多重要素驗證。您未能實行此要求，根帳戶遭到損害。
- 在設計工作負載期間，您選擇了未經 IT 部門核准的執行個體類型。您無法啟動工作負載，而必須執行重新設計。
- 您必須有災難復原計劃。您未建立該計劃，且工作負載遭逢長時間的中斷。
- 您的團隊想要使用新的執行個體，但您的管理要求尚未更新予以允許。

建立此最佳實務的優勢：

- 遵循管控要求，可讓您的工作負載符合較大組織的政策。
- 管控要求會反映組織的產業標準和最佳實務。

未建立此最佳實務時的風險暴露等級：高

實作指引

與利害關係人和管控組織共同識別管控要求。將管控要求納入您的工作負載中。能夠證明您已遵循管控要求。

客戶範例

在 AnyCompany Retail，雲端營運團隊與組織內的利害關係人共同制定管控要求。例如，他們禁止對 Amazon EC2 執行個體進行 SSH 存取。如果團隊需進行系統存取，他們必須使用 AWS Systems Manager Session Manager。雲端營運團隊會在新服務推出時定期更新管控要求。

實作步驟

1. 識別工作負載的利害關係人，包括任何集中團隊。
2. 與利害關係人共同識別管控要求。
3. 產生清單後，請排定改善項目的優先順序，並開始在您的工作負載中加以實作。
    a. 使用 AWS Config 之類的服務建立「管控即程式碼」，並驗證確實遵循了管控要求。
    b. 如果您使用 AWS Organizations，您可以利用服務控制政策來實作管控要求。
4. 提供驗證實作情形的文件。

實作計劃的工作量：中。實作遺漏的管控要求可能會導致工作負載重新作業。

資源

相關的最佳實務：

- OPS01-BP04 評估合規要求 - 合規與管控類似，但來自組織外部。

相關文件：

- AWS 管理與管控雲端環境指南
- 多帳戶環境中的 AWS Organizations 服務控制政策的最佳實務
- AWS 雲端 中的管控：敏捷和安全之間的正確平衡
- 什麼是管控、風險和合規 (GRC)？

相關影片：

- AWS 管理與管控：組態、合規和稽核 - AWS 線上技術會談
- AWS re:Inforce 2019：雲端時代的管控 (DEM12-R1)
- AWS re:Invent 2020：使用 AWS Config 實現合規即程式碼
- AWS re:Invent 2020：AWS GovCloud (US) 上的敏捷管控

相關範例：

- AWS Config 合規套件範例

相關服務：

- AWS Config
- AWS Organizations - 服務控制政策

OPS01-BP04 評估合規要求

法規、產業和內部合規要求是定義組織優先順序的重要因子。您的合規架構可能會禁止使用特定技術或地理位置。若未識別出外部合規架構，請運用盡職調查。產生驗證合規性的稽核或報告。

如果聲明您的產品符合特定的合規標準，您必須有內部程序來確保持續的合規性。合規標準的例子包括 PCI DSS、FedRAMP 和 HIPAA。適用的合規標準取決於各種因素，例如解決方案存放或傳輸的資料類型，以及解決方案支援的地理區域。

預期成果：

- 將法規、產業和內部合規要求併入架構選擇中。
- 您可以驗證合規性並產生稽核報告。

常見的反模式：

- 您的工作負載有部分屬於支付卡產業資料安全標準 (PCI-DSS) 架構下，但您的工作負載儲存信用卡資料時並未予以加密。
- 您的軟體開發人員和架構師不知道您的組織必須遵循的合規架構。
- 年度 Systems and Organizations Control (SOC2) Type II 稽核即將到來，但您無法驗證已設置控制。

建立此最佳實務的優勢：

- 評估和了解套用到工作負載的合規要求，可讓您了解如何安排工作的優先順序來實現商業價值。
- 您可以選擇與合規架構相符的適當位置和技術。
- 針對可稽核性設計工作負載，可讓您證明您確實遵循合規架構。

未建立此最佳實務時的風險暴露等級：高

實作指引

若實作此最佳實務，即表示您會在架構設計程序中併入合規要求。您的團隊成員將得知必要的合規架構。您會驗證合規性符合架構。

客戶範例

AnyCompany Retail 儲存客戶的信用卡資訊。卡片儲存團隊的開發人員了解他們必須遵從 PCI-DSS 架構。他們執行了相關步驟，驗證信用卡資訊以安全方式儲存和存取，並遵從 PCI-DSS 架構。他們每年都會與安全團隊共同驗證合規性。

實作步驟

1. 與安全和管控團隊合作，確認您的工作負載必須遵循哪些產業、法規或內部合規架構。在您的工作負載中併入合規架構。
   a. 使用 AWS Compute Optimizer 和 AWS Security Hub 之類的服務驗證 AWS 資源的持續合規性。
2. 讓團隊成員了解合規要求，使其能據以操作及設計工作負載。合規要求應包含在架構和技術選擇中。
3. 根據合規架構，您可能必須產生稽核或合規報告。請與組織合作，盡可能將此程序自動化。
   a. 使用 AWS Audit Manager 之類的服務驗證合規性並產生稽核報告。
   b. 您可以透過 AWS Artifact 下載 AWS 安全與合規文件。

實作計劃的工作量：中。實作合規架構可能並不容易。產生稽核報告或合規文件，會增添額外的複雜性。

資源

相關的最佳實務：

- SEC01-BP03 識別和驗證控制目標 - 安全控制目標是整體合規性的重要環節。
- SEC01-BP06 將管道中安全控制的測試和驗證自動化 - 在您的管道中驗證安全控制。您也可以產生新變更的合規文件。
- SEC07-BP02 定義資料保護控制 - 許多合規架構都以資料處理和儲存政策為基礎。
- SEC10-BP03 準備鑑識功能 - 鑑識功能有時可用來稽核合規性。

相關文件：

- AWS 合規中心
- AWS 合規資源
- AWS 風險與合規白皮書
- AWS 共同責任模式
- 範圍內的 AWS 服務 (依合規計劃)

相關影片：

- AWS re:Invent 2020：使用 AWS Compute Optimizer 實現合規即程式碼
- AWS re:Invent 2021 - 雲端合規、保證和稽核

- AWS Summit ATL 2022 - 在 AWS 上實作合規、保證和稽核 (COP202)

相關範例：

- AWS 上的 PCI DSS 和 AWS 基礎安全最佳實務

相關服務：

- AWS Artifact
- AWS Audit Manager
- AWS Compute Optimizer
- AWS Security Hub

OPS01-BP05 評估威脅態勢

評估對業務的威脅 (例如，競爭、業務風險和負債、營運風險和資訊安全威脅)，並將最新的資訊保存在風險登記表內。決定工作重點的領域時，加入風險影響。

AWS Well-Architected 架構 強調 學習、衡量和改善。它為您提供可評估架構並實作將隨時間擴展之設計的一致方法。AWS 提供 AWS Well-Architected Tool ，以協助您在部署前檢閱方法、在生產前檢閱工作負載狀態，以及檢閱生產中的工作負載狀態。您可以將它們與最新的 AWS 架構最佳實務做比較、監控工作負載的整體狀態，以及深入了解潛在風險。

AWS 客戶還有資格獲得對其關鍵任務工作負載的指導式 Well-Architected 審查， 進而依循 AWS 最佳實務 衡量其架構。企業支援客戶有資格進行 營運審查，該審查旨在助其識別在雲端營運的方法的差距。

這些審查的跨團隊參與有助於建立對您的工作負載以及團隊角色可如何助力成功的共識。透過審查識別的需求可以助您確定優先順序。

AWS Trusted Advisor 是一款可存取核心檢查集的工具，這些檢查提出了優化建議，可能有助您確定優先事項。 商業和企業支援客戶 可存取針對安全性、可靠性、效能和成本優化的其他檢查，從而進一步協助確定他們的優先事項。

常用的反模式：

- 您在產品中使用舊版的軟體程式庫。您不知道，程式庫的安全性更新是否存在可能對工作負載產生意外影響的問題。

- 您的競爭對手剛發佈的產品版本，可解決客戶對您產品的許多抱怨。您尚未排定處理這些已知問題之事項的優先順序。
- 監管機構一直在追尋像您這樣不符合法律法規合規要求的公司。您尚未排定處理任何未解決合規要求之事項的優先順序。

建立此最佳實務的優勢： 識別和了解組織和工作負載所面臨的威脅，讓您可以判斷要解決哪些威脅、它們的優先順序，以及執行此作業所需的資源。

若未建立此最佳實務，暴露的風險等級為： 中

實作指引

- 評估威脅態勢：評估對業務的威脅 (例如，競爭、業務風險和負債、營運風險和資訊安全威脅)，以便您在決定工作重點時考量其影響。
  - AWS 最新安全公告
  - AWS Trusted Advisor
  - 維護威脅模型：建立和維護用於識別潛在威脅、已規劃和就地緩解措施及其優先順序的威脅模型。審查顯示為事件的威脅的機率、從這些事件中復原的成本、導致的預期傷害，以及防止這些事件的成本。當威脅模型的內容變更時，修改優先順序。

資源

相關文件：

- AWS 雲端 合規
- AWS 最新安全公告
- AWS Trusted Advisor

OPS01-BP06 評估權衡

評估在相互衝突的利益或替代方法之間做出權衡的影響，以幫助您在確定工作重點或選擇行動方案時做出明智的決定。例如，新功能加速上市可能是成本優化所強調的重點，或您可為非關聯式資料選擇關聯式資料庫，以便更輕鬆地遷移系統，而非遷移至針對您的資料類型優化的資料庫並更新您的應用程式。

AWS 可以協助您教育您的團隊有關 AWS 及其服務的知識，從而增進他們對自己的選擇會如何影響工作負載的了解。您應使用 AWS Support (AWS 知識中心， AWS 開發論壇和 AWS Support中心) 和

AWS 文件 資源來教育您的團隊。透過 AWS Support中心聯絡 AWS Support，以獲取 AWS 相關問題的幫助。

AWS 也分享了我們透過 在 Amazon Builders' Library 中營運 AWS 所學到的最佳實務和模式。您可透過 AWS 部落格 和 官方 AWS 播客。

常用的反模式：

- 您使用關聯式資料庫來管理時間序列和非關聯式資料。有針對支援您使用的資料類型進行最佳化的資料庫選項，但您並不了解其優點，因為您尚未評估解決方案之間的權衡。
- 您的投資者要求您證明支付卡產業資料安全標準 (PCI DSS) 的合規性。您沒有考量滿足要求和繼續您目前開發工作之間的權衡取捨。相反地，您繼續開發工作，而不證明合規性。由於對平台安全性及其投資的擔憂，您的投資者會停止對公司的支援。

建立此最佳實務的優勢： 了解您所選擇的影響和後果，讓您可以排定選項的優先順序。

若未建立此最佳實務，暴露的風險等級： 中

實作指引

- 評估權衡：評估在相互衝突的利益之間做出權衡的影響，以幫助您在確定工作重點時做出明智的決定。例如，相比成本優化更強調新功能加速上市。
- AWS 可以協助您教育您的團隊有關 AWS 及其服務的知識，從而增進他們對自己的選擇會如何影響工作負載的了解。您應使用 AWS Support (AWS 知識中心、AWS 論壇和 AWS Support 中心) 和 AWS 文件中的資源來教育您的團隊。透過 AWS Support中心聯絡 AWS Support，以獲取 AWS 相關問題的幫助。
- AWS 也分享了我們透過在 Amazon Builders' Library 中營運 AWS 所學到的最佳實務和模式。您可透過 AWS 部落格和官方 AWS 播客獲得其他各種實用資訊。

資源

相關文件：

- AWS 部落格
- AWS 雲端 合規
- AWS 開發論壇
- AWS 文件

- AWS 知識中心
- AWS Support
- AWS Support中心
- 在 Amazon Builders' Library 中
- 官方 AWS 播客

OPS01-BP07 管理收益和風險

管理收益和風險，以便在確定工作重點時做出明智的決定。例如，部署具有未解決問題的工作負載可能有益，以便可以為客戶提供重要的新功能。相關風險可能得以減輕，也可能出現無法接受風險存在的事實，在此情況下，您將需要採取動作來解決風險。

您可能會發現，您在某個時間點會想要強調一小部分的優先事項。長期利用平衡的方法，以確保開發所需的功能和管理風險。根據需求變更更新您的優先順序

常用的反模式：

- 您已決定包含一個程式庫，該程式庫會執行其中一個開發人員在網際網路上找到的您所需的任何項目。您尚未評估從未知來源採用此程式庫的風險，並且不知道它是否包含弱點或惡意程式碼。
- 您已決定開發和部署新功能，而不是修正現有問題。在部署功能之前，您一直未評估將問題置之不理的風險，而且不知道會對客戶造成什麼影響。
- 由於合規團隊的不明疑慮，您決定不部署客戶經常請求的功能。

建立此最佳實務的優勢： 識別您選擇的可用優勢，並了解組織面臨的風險，讓您可以做出明智的決策。

若未建立此最佳實務，暴露的風險等級： 低

實作指引

- 管理收益和風險：在決策的收益與所涉及的風險之間取得平衡。
  - 確定收益：根據業務目標、需求和優先事項確定收益。範例包括上市時間、安全性、可靠性、效能和成本。
  - 確定風險：根據業務目標、需求和優先事項確定風險。範例包括上市時間、安全性、可靠性、效能和成本。
  - 評估風險與收益並做出明智決定：根據關鍵利害關係人 (包括業務、開發和營運團隊) 的目標、需求和優先事項，確定收益和風險的影響。評價收益的價值時要考慮發生風險的可能性及其代價。例

如，強調上市速度優先於可靠性，可能提供競爭優勢。不過，如果發生可靠性問題，則可能會縮短正常執行時間。

操作模式

您的團隊必須了解其在達成業務成果中所扮演的角色。團隊需要了解自己在促成其他團隊成功的過程中所扮演的角色、其他團隊在促進其成功的過程中所扮演的角色，以及擁有共同目標。了解責任、擁有權、決策方式，以及誰有權制定決策，將有助於找到工作重點，並充分發揮團隊的優勢。

團隊的需求將由其產業、組織、團隊組成，以及工作負載的特性形塑而成。合理來說，無法要求單一操作模式支援所有團隊及其工作負載。

存在於組織內的操作模式數量可能會隨著開發團隊數量增長。您可能需要使用一組操作模式。

採用標準和消耗服務可幫助簡化操作，並限制操作模式的支援重擔。根據共同標準進行的開發工作所得到的效益，會隨著已採用該標準的團隊數目，以及誰將採用新功能而放大。

機制存在的目的應為請求標準新增、變更及例外狀況，以支援團隊的活動。如果沒有此選項，標準就會限制創新。評估效益和風險後，若可行則應核准請求，並判斷請求是否合適。

明確定義的一組責任將可減少工作發生衝突或重複的頻率。當業務、開發和營運團隊之間合作無間、關係密切時，則更容易達成業務成果。

操作模式 2 x 2 表示法

這些操作模式 2 x 2 表示法屬於圖示，可協助您了解您環境中團隊間的關係。這些圖表著重於相應人員負責的相應工作和團隊間的關係，但我們也將討論在這些範例中的管控和決策制定。

視他們支援的工作負載而定，我們的團隊可能在數種模式中擔負數個部份的責任。您可能希望打破更專業的學科領域，而非描述的高階領域。隨著您劃分或彙整活動，或重疊團隊並提供更具體的詳細資訊，這些模式可能會有無限的變化。

您可能會發現跨團隊間會有能夠提供其他優勢或發揮效率的重疊或尚未辨識的功能。您也可識別貴組織內尚未滿足，但自己打算解決的需求。

評估組織變更時，請檢視模式間的權衡、您的個別團隊位於模式中的哪個階段 (目前和變更後)、您團隊的關係和責任將如何變更，以及效益是否能夠影響貴組織。

您可以使用以下四種操作模式獲得成功。部分模式更適用於特定使用案例，或部署中的特定時間點。在您的環境中使用時，部分模式可能比其他模式更具優勢。

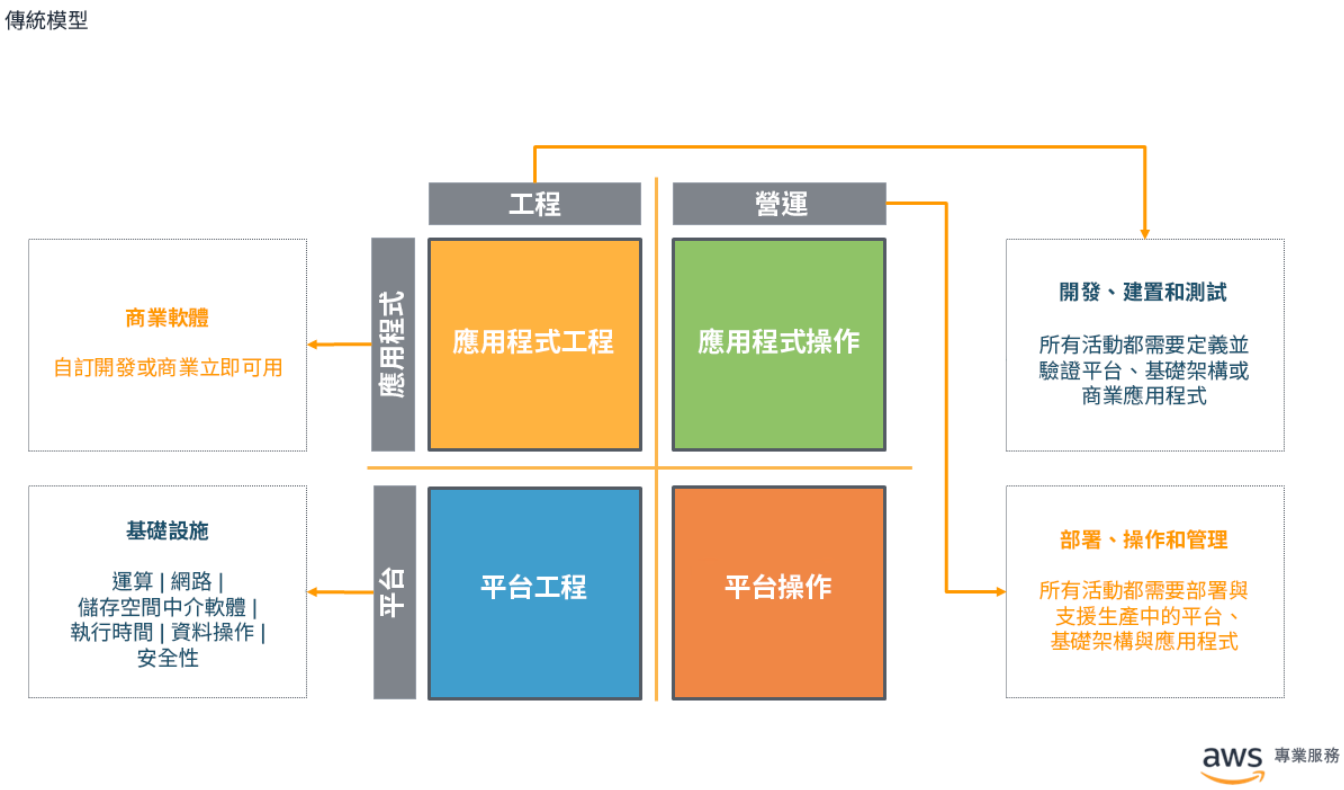
主題

- 完全獨立的操作模式
- 採用集中管控的獨立應用程式工程和操作 (AEO) 和基礎設施工程和操作 (IEO)
- 採用集中管控和服務供應商的獨立 AEO 和 IEO
- 採用集中管控和內部服務供應商諮詢合作夥伴的獨立 AEO 和 IEO
- 採用分散管控的獨立 AEO 和 IEO

完全獨立的操作模式

應用程式和基礎設施位於下圖的垂直軸上。應用程式是指幫助實現業務成果的工作負載，可以是自訂開發或採購的軟體。基礎設施是指實體和虛擬的基礎設施，以及支援該工作負載的其他軟體。

工程和操作位於水平軸上。工程是指開發、建置和測試應用程式和基礎設施。操作是指部署、更新及持續支援應用程式和基礎設施。



此「完全獨立」的模式存在於許多組織中。各象限中的活動由個別的團隊執行。 工作透過工作請求、工作佇列、票證或使用 IT 服務管理 (ITSM) 系統等機制，在團隊之間轉交。

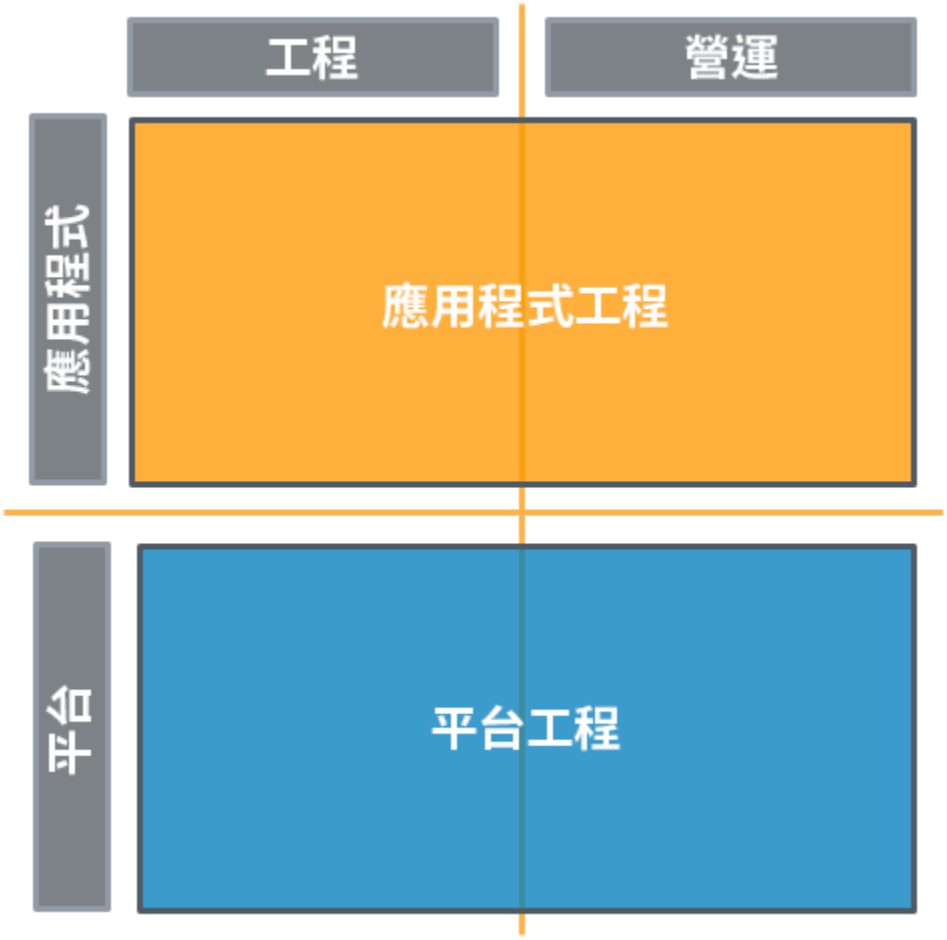
任務轉給團隊或團隊間的任務轉交會讓事情變得更複雜，並形成瓶頸與延誤。除非是要優先處理的請求，否則請求可能延誤處理。如果晚發現缺陷，可能需要大量重新作業，也可能需要再次交由相同的團隊和職務處理。如果發生需由工程團隊採取行動的事件，其回應會因遞交活動而受到延誤。

當業務、開發和營運團隊根據正在執行的活動或職務組織時，目標不一致的風險會升高。如此會導致讓團隊專注於其特定責任，而非專注於達成業務成果。團隊可能專精於狹隘的領域，或在實際或邏輯空間上遭到分離，因而阻礙溝通與合作。

採用集中管控的獨立應用程式工程和操作 (AEO) 和基礎設施工程和操作 (IEO)

此「獨立 AEO 和 IEO」模式依循「誰開發誰執行」方法。

您的應用程式工程師和開發人員會執行其工作負載的工程和操作。相同地，您的基礎設施工程師會執行其用於支援應用程式團隊的平台工程和操作。



在此範例中，我們將進行集中管控。標準會分發、提供給應用程式團隊，或與之共用。

您應該使用能集中管控跨帳戶環境的工具或服務，例如 AWS Organizations。諸如 AWS Control Tower 等服務會擴大此管理功能，讓您能定義帳戶設定的藍圖 (支援您的操作模式)、使用 AWS Organizations 套用持續管控，以及自動化新帳戶的佈建作業。

「誰開發誰執行」並不表示應用程式團隊負責完整的堆疊、工具鏈和平台。

平台工程團隊為應用程式團隊提供一組標準化服務 (例如，開發工具、監控工具、備份和復原工具及網路)。平台團隊也可將核准之雲端供應商服務、相同的特定組態或兩者的存取權提供給應用程式團隊。

提供部署核准之服務和組態的自助服務功能的機制，例如 Service Catalog，有助於限縮在強制執行管控時履行請求的相關延誤。

平台團隊可提供完整的堆疊能見度，以便應用程式團隊可以區分其應用程式元件和服務，以及其應用程式消耗之基礎設施元件所發生的問題。平台團隊也可提供設定這些服務的協助，以及如何改善應用程式團隊營運的指導。

如之前所述，機制存在的目的應為請求標準新增、變更及例外狀況，以支援團隊的活動及其應用程式的創新。

獨立 AEO IEO 模式提供健全的意見回饋迴圈給應用程式團隊。工作負載的日常操作會透過直接互動，或間接透過支援和功能請求，而增加與客戶的接觸。如此提高的能見度有助於應用程式團隊更迅速處理問題。透過更深入參與和更密切的關係能深入了解客戶需求，並更迅速地實現創新。

上述原則對於支援應用程式團隊的平台團隊也同樣適用。

採用的標準可能事先經過核准可以使用，進而減少進入生產所需的審查工作量。使用平台團隊提供的支援和經過測試的標準，可能會減少這些服務發生問題的頻率。應用程式團隊透過採用標準可以專注於差異化其工作負載。

採用集中管控和服務供應商的獨立 AEO 和 IEO

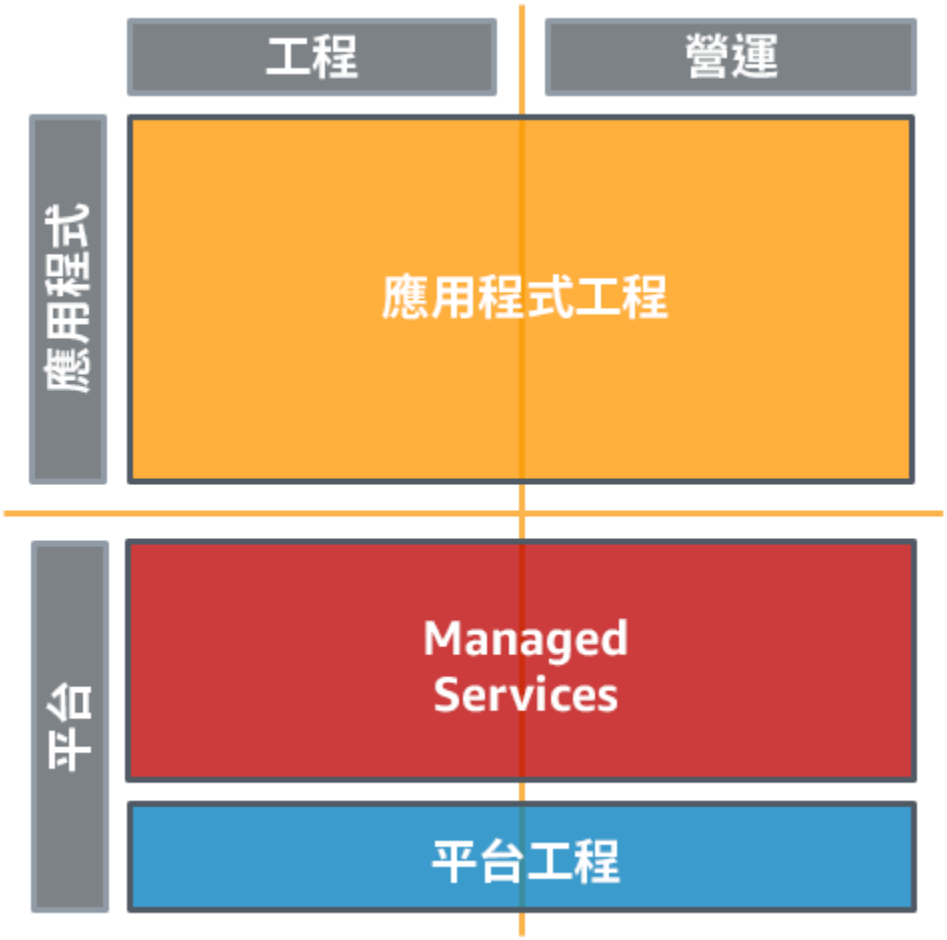
此「獨立 AEO 和 IEO」模式依循「誰開發誰執行」方法。

您的應用程式工程師和開發人員會執行其工作負載的工程和操作。

貴組織現有的技能或團隊成員可能無法支援專門的平台工程和營運團隊，或您可能不想要投入時間和人力物力來進行此工作。

或者，您可能想要永遠專注於建立讓您的企業脫穎而出之功能的平台團隊，但卻想要擺脫一成不變的日常作業，交給外包商執行。

諸如 AWS Managed Services、AWS Managed Services 合作夥伴等受管服務供應商，或 AWS 合作夥伴網路受管服務供應商，都會提供實作雲端環境的專業知識，並支援您的安全和合規要求及業務目標。



就此變化而言，我們將透過平台團隊進行集中管控和管理，使用 AWS Organizations 和 AWS Control Tower 建立帳戶並管理政策。

此模式的確需要您修改機制，以與您的服務供應商的機制合作。它不會處理因團隊間 (包括您的服務供應商) 轉交任務而導致的瓶頸和延誤，或因晚發現缺陷而導致的潛在重新作業。

您可以獲得供應商標準、最佳實務、流程和專業知識的優勢。您也可以從其服務產品持續開發中獲得效益。

將受管服務加入操作模式後，便可節省時間和資源，讓您的內部團隊精簡並專注於將使您的企業脫穎而出的策略性成果，而非開發新技能和功能。

採用集中管控和內部服務供應商諮詢合作夥伴的獨立 AEO 和 IEO

此「獨立 AEO 和 IEO」模式企圖建立「誰開發誰執行」方法。

您想要應用程式團隊為其工作負載執行工程和操作活動，並採用更類似於 DevOps 的文化。

您的應用程式團隊可能正在進行遷移、採用雲端或將工作負載現代化，但目前沒有技能足以支援雲端和雲端操作。在應用程式團隊能力或熟悉度方面的缺乏可能會成為您工作的障礙。

為了解決這個問題，您應建立一個雲端啟用中心團隊 (CCoE)，讓其提供一個論壇以便提出問題、討論需求，和識別解決方案。根據您組織的需求，CCoE 可以是一個專屬的專家團隊，也可以是一個虛擬團隊，其中參與者選自您的整個組織。CCoE 可讓團隊進行雲端轉型、建立集中式雲端管控，以及定義帳戶和組織管理標準。他們也會識別成功的參考架構和模式，供企業使用。

我們將 CCoE 稱為雲端啟用中心，而不是更常見的雲端卓越中心，以強調讓支援團隊能夠成功並實現商務成果。

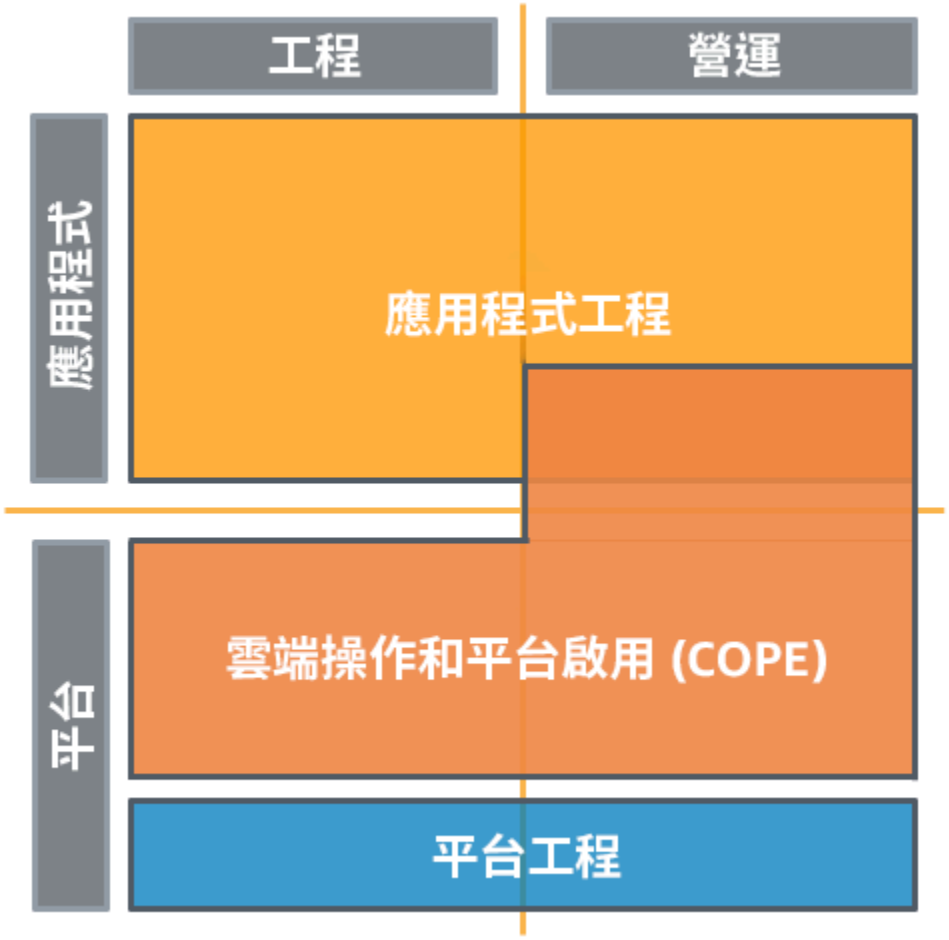
您的平台工程團隊會根據這些標準建置核心共用平台功能，供應用程式團隊採用。他們編纂了透過自助服務機制提供給應用程式團隊的企業參考架構和模式。使用 AWS Service Catalog 這類服務，應用程式團隊可以部署已核准的參考架構、模式、服務和組態，並預設符合集中管控和安全標準。

平台工程團隊也為應用程式團隊提供一組標準化服務 (例如，開發工具、監控工具、備份和復原工具及網路)。

您的組織擁有一個「內部 MSP 和諮詢合作夥伴」，用來管理和支援標準化服務，並協助應用程式團隊根據參考架構和模式建立其雲端服務。這個「雲端操作和平台啟用 (COPE)」團隊會與應用程式團隊合作，以協助他們建立基準操作，但隨著時間的推移，應用程式團隊逐漸對其系統和資源承擔更多責任。COPE 團隊與 CCoE 和平台工程團隊一起推動持續改進，並充當應用程式團隊的支持者。

應用程式團隊獲得協助設定環境、CICD 管道、變更管理、可觀察性和監控，以及視需要與 COPE 團隊一起建立事故和事件管理程序，而這些程序會與企業的類似程序整合。COPE 團隊與應用程式團隊一起參與這些操作活動的執行，一旦應用程式團隊獲得擁有權，就會取消 COPE 團隊的參與。

應用程式團隊受益於 COPE 團隊的技能和組織學到的經驗教訓。他們受到透過集中管控建立的防護機制保護。應用程式團隊建立在公認的成功之上，並在持續發展他們所採用的組織標準中獲益。他們透過建立可觀察性和監控的過程更深入地洞悉其工作負載的操作，並且能夠更好地了解他們對其工作負載所做變更的影響。



COPE 團隊保留支援操作活動、提供跨應用程式團隊的企業營運檢視，以及提供關鍵事故管理支援所需的存取權。COPE 團隊對視為無差別繁重工作的活動保留責任，其會透過可大規模支援的標準解決方案來滿足這些活動。他們也會繼續針對應用程式團隊管理熟知的程式設計和自動化操作活動，以便他們專注於區分其應用程式。

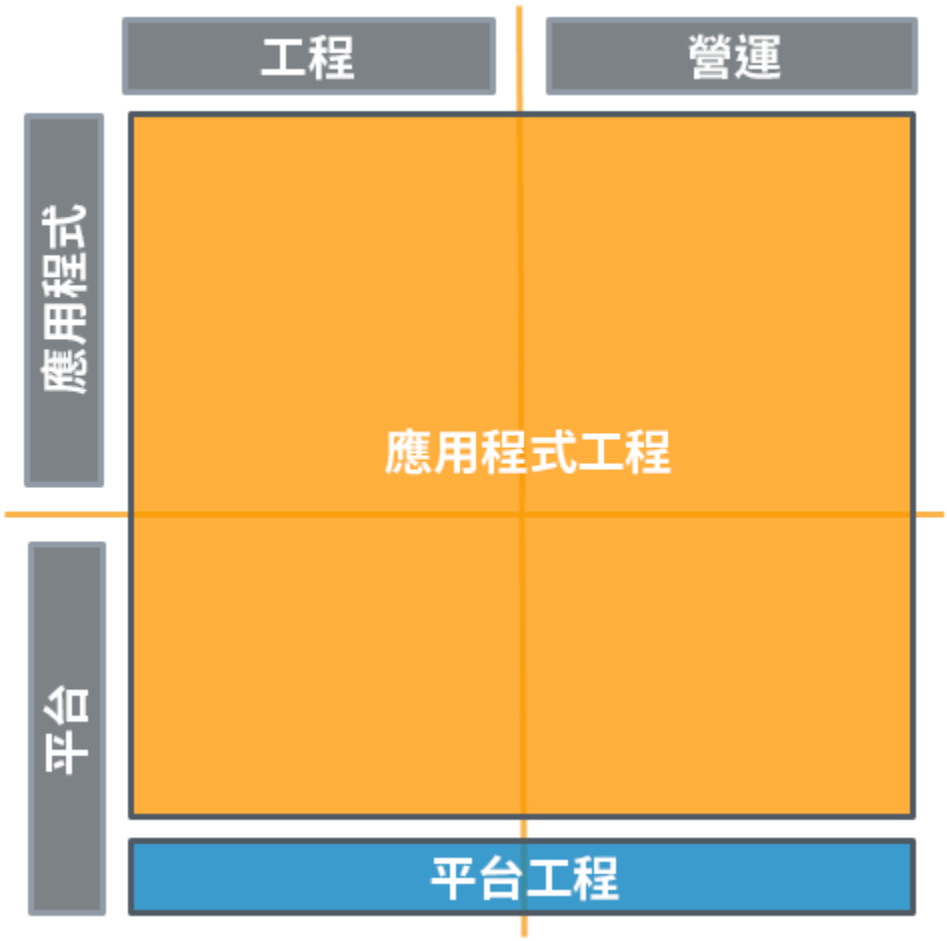
您可從組織標準、最佳實務、程序，以及衍生自團隊成功的專業知識獲得優勢。您可建立一種機制來複製這些成功模式，讓新團隊在雲端上採用或進行現代化。此模型強調 COPE 團隊能夠協助建立應用程式團隊，並轉移知識和成品。它會減輕應用程式團隊的操作負擔，但存在應用程式團隊無法獨立自主的風險。它會在 CCoE、COPE 和應用程式團隊之間建立關係，從而建立反饋迴圈以支援進一步的演進和創新。

建立您的 CCoE 和 COPE 團隊，同時定義全組織標準，可以促進雲端採用並支援現代化工作。藉由額外支援 COPE 團隊作為應用程式團隊的顧問和合作夥伴，您可以移除減緩應用程式團隊採用有益雲端功能的障礙。

採用分散管控的獨立 AEO 和 IEO

此「獨立 AEO 和 IEO」模式依循「誰開發誰執行」方法。

您的應用程式工程師和開發人員會執行其工作負載的工程和操作。相同地，您的基礎設施工程師會執行其用於支援應用程式團隊的平台工程和操作。



在此範例中，我們將進行分散管控。

標準仍會由平台團隊分發、提供給應用程式團隊，或與之共用，但應用程式團隊可自由設計和操作新的平台功能，以支援其工作負載。

在此模式中，應用程式團隊受到的限制較少，但伴隨而來的卻是責任大幅增加。必須擁有其他技能和潛在的團隊成員，以支援其他平台功能。如果技能組合不足，且無法及早辨識缺陷，則大量重新作業的風險會提高。

您應強制執行並非專門委派給應用程式團隊的政策。使用能集中管控跨帳戶環境的工具或服務，例如 AWS Organizations。諸如 AWS Control Tower 等服務會擴大此管理功能，讓您能定義帳戶設定的藍圖 (支援您的操作模式)、使用 AWS Organizations 套用持續管控，以及自動化新帳戶的佈建作業。

讓應用程式團隊設有請求標準新增和變更的機制，將會帶來好處。他們能夠貢獻可以使其他應用程式團隊受益的新標準。平台團隊可能會認為，為這些其他功能直接提供支援，能有效幫助實現業務成果。

此模式透過重要技能和團隊成員要求，減少對於創新的限制。它解決了團隊間轉交任務所導致的許多瓶頸和延誤，同時仍促進團隊與客戶之間建立有效的關係。

關係和擁有權

您的操作模式定義團隊之間的關係，並支援可識別的擁有權和責任。

最佳實務

- OPS02-BP01 已為資源識別擁有者
- OPS02-BP02 已為流程和程序識別擁有者
- OPS02-BP03 已為營運活動識別負責其效能的擁有者
- OPS02-BP04 團隊成員知道他們負責的項目
- OPS02-BP05 存在機制用來識別責任和擁有權
- OPS02-BP06 存在用於要求新增、變更和例外狀況的機制
- OPS02-BP07 團隊之間的責任是預先定義或經過協商的

OPS02-BP01 已為資源識別擁有者

工作負載的資源必須已識別變更控制、疑難排解和其他功能的擁有者。系統會為工作負載、帳戶、基礎設施、平台和應用程式指派擁有者。擁有權會使用集中註冊或連接至資源的中繼資料等工具來記錄。元件的商業價值會透露其適用的流程和程序。

預期成果：

- 資源已使用中繼資料或集中註冊識別擁有者。
- 團隊成員可識別誰擁有資源。
- 帳戶會盡可能擁有單一擁有者。

常見的反模式：

- AWS 帳戶 的替代聯絡人未填入。
- 資源缺少用來識別哪些團隊是其擁有者的標籤。
- 您有不具備電子郵件對應的 ITSM 佇列。
- 兩個團隊對於基礎設施的關鍵部件有重疊的擁有權。

建立此最佳實務的優勢：

- 有了指派的擁有權，資源的變更控制將是簡單明瞭的。
- 對問題進行疑難排解時，您將可接洽正確的擁有者。

未建立此最佳實務時的風險暴露等級：高

實作指引

定義擁有權對環境中的資源使用案例的意義。擁有權可表示誰負責監督資源的變更、誰支援疑難排解期間的資源，或財務責任由誰承擔。指定並記錄資源的擁有者，包括名稱、聯絡資訊、組織和團隊。

客戶範例

AnyCompany Retail 將擁有權定義為擁有資源變更和支援的團隊或個人。他們使用 AWS Organizations 來管理其 AWS 帳戶。替代帳戶聯絡人使用群組收件匣進行設定。每個 ITSM 佇列分別對應至一個電子郵件別名。標籤會指出誰擁有 AWS 資源。對於其他平台和基礎設施，他們有 Wiki 頁面會指出擁有權和聯絡資訊。

實作步驟

1. 首先為您的組織定義擁有權。擁有權可暗示資源的風險由誰承擔、誰擁有資源的變更，或誰支援疑難排解期間的資源。擁有權也可暗示資源的財務或管理擁有權。
2. 使用 AWS Organizations 管理帳戶。您可以集中管理帳戶的替代聯絡人。
   a. 只要使用公司擁有的電子郵件地址和電話號碼作為聯絡資訊，即使聯絡資訊所屬的個人已離職，您仍可存取這些資訊。例如，為帳單、營運和安全建立各別的電子郵件分發清單，在每個作用中 AWS 帳戶 中將這些設定為帳戶、安全和營運聯絡人。即使某人休假、職務變動或離職，仍有多人會收到 AWS 通知並且能有所回應。
   b. 如果帳戶未由 AWS Organizations 管理，替代帳戶聯絡人可協助 AWS 在必要時聯繫到適當人員。設定帳戶的替代聯絡人，將其指向群組而非個人。
3. 使用標籤來識別 AWS 資源的擁有者。您可以用個別的標籤指定擁有者及其聯絡資訊。
   a. 您可以使用 AWS Config 規則強制資源要有必要的擁有權標籤。

b. 如需如何為組織建置標記策略的深入指引，請參閱 AWS 標記最佳實務白皮書。

4. 對於其他資源、平台和基礎設施，請建立識別擁有權的文件。此文件應開放給所有團隊成員存取。

實作計劃的工作量：低。利用帳戶聯絡資訊和標籤指派 AWS 資源的擁有權。對於其他資源，您可以使用 Wiki 表格這類簡單的工具來記錄擁有權與聯絡資訊，或使用 ITSM 工具來對應擁有權。

資源

相關的最佳實務：

- OPS02-BP02 已為流程和程序識別擁有者 - 支援資源的流程和程序取決於資源擁有權。
- OPS02-BP04 團隊成員知道他們負責的項目 - 團隊成員應了解他們是哪些資源的擁有者。
- OPS02-BP05 存在機制用來識別責任和擁有權 - 擁有權必須可使用標籤或帳戶聯絡人等機制來探索。

相關文件：

- AWS 帳戶管理 - 更新聯絡資訊
- AWS Config 規則 - Required-Tags
- AWS Organizations - 更新組織中的替代聯絡人
- AWS 標記安全最佳實務白皮書

相關範例：

- AWS Config 規則 - Amazon EC2 使用必要標籤與有效值

相關服務：

- AWS Config
- AWS Organizations

OPS02-BP02 已為流程和程序識別擁有者

了解誰具有個別流程和程序的擁有權、為何使用特定流程和程序，以及為何該擁有權存在。了解使用特定流程和程序的原因，能夠幫助發現改進機會。

建立此最佳實務的優勢： 透過了解擁有權，可識別誰可以核准改進項目和/或實作這些改進項目。

若未建立此最佳實務，暴露的風險等級： 高

實作指引

- 已為流程和程序識別負責其定義的擁有者：擷取環境中使用的流程和程序，以及負責其定義的個人或團隊。
  - 識別流程和程序：識別為支援工作負載所執行的營運活動。將這些活動記錄在可探索的位置中。
  - 定義擁有流程或程序定義的人員：唯一識別負責活動規格的個人或團隊。他們負責確保具備適當技能的團隊成員能夠成功執行該活動，且該團隊成員具備正確許可、存取權和工具。如果執行該活動時發生問題，執行該活動的團隊成員需負責提供改善活動所需的詳細回饋。
  - 擷取活動成品中繼資料中的擁有權：在 AWS Systems Manager 等服務中，透過文件和做為函數的 AWS Lambda 自動化的程序，支援以標籤形式擷取中繼資料資訊。使用標籤或資源群組擷取資源擁有權，並指定擁有權和聯絡資訊。使用 AWS Organizations 建立標記政策，並確保擷取擁有權和聯絡資訊。

OPS02-BP03 已為營運活動識別負責其效能的擁有者

了解誰負責在已定義的工作負載上執行特定活動，以及為什麼該責任存在。透過了解誰負責執行活動，可得知誰將會進行活動、驗證結果，以及提供回饋給活動擁有者。

建立此最佳實務的優勢： 透過了解誰負責執行活動，可得知在需要採取動作時通知誰，以及誰將會執行動作、驗證結果，以及提供回饋給活動擁有者。

若未建立此最佳實務，暴露的風險等級： 高

實作指引

- 已為營運活動識別負責其效能的擁有者：擷取執行環境中所使用之流程和程序的責任
  - 識別流程和程序：識別為支援工作負載所執行的營運活動。將這些活動記錄在可探索的位置中。
  - 定義負責執行每個活動的人員：識別負責活動的團隊。確保他們擁有活動的詳細資訊，以及執行活動所需的技能和正確的許可、存取權和工具。他們必須了解活動執行條件 (例如，事件或排程)。讓此資訊可供探索，如此組織的成員便能夠識別他們針對特定需求需要聯絡的人員 (團隊或個人)。

OPS02-BP04 團隊成員知道他們負責的項目

透過了解您角色的責任以及您為業務成果做出貢獻的方式，可得知任務的優先順序以及您的角色為何很重要。如此可讓團隊成員辨識需求並適當地回應。

建立此最佳實務的優勢：透過了解您的責任，可得知您所做的決定、您採取的動作，以及如何將活動交給其適當的擁有者。

未建立此最佳實務時的風險暴露等級：高

實作指引

- 確保團隊成員了解其角色和責任：識別團隊成員的角色和責任，並確保他們了解其角色的期望。讓此資訊可供探索，如此組織的成員便能夠識別他們針對特定需求需要聯絡的人員 (團隊或個人)。

OPS02-BP05 存在機制用來識別責任和擁有權

如果沒有識別個人或團隊，就會有定義的向某人向上呈報的路徑，該人員有權指派擁有權或為需解決的需求進行規劃。

建立此最佳實務的優勢： 透過了解有責任或擁有權的人員，讓您可以聯絡適當的團隊或團隊成員，以提出請求或轉換任務。擁有有權指派責任或擁有權或為解決需求進行規劃的已識別人員，可降低無作為和需求得不到解決的風險。

若未建立此最佳實務，暴露的風險等級： 高

實作指引

- 存在機制用來識別責任和擁有權：為您的組織成員提供可存取的機制，以探索和識別擁有權和責任。這些機制讓他們可以針對特定需求識別要聯絡的人員 (團隊或個人)。

OPS02-BP06 存在用於要求新增、變更和例外狀況的機制

您可以向流程、程序和資源的擁有者提出要求。要求包含新增、變更和例外狀況。這些要求會經歷變更管理程序。評估收益和風險後，若可行並經判斷是合適的行為，則應制定明智的決策以核准要求。

預期成果：

- 您可以根據指派的擁有權提出變更流程、程序和資源的要求。

- 權衡利益與風險，審慎進行變更。

常見的反模式：

- 您必須更新您部署應用程式的方式，但無法透過營運團隊要求變更部署程序。
- 災難復原計劃必須更新，但沒有已識別的擁有者可接受變更的要求。

建立此最佳實務的優勢：

- 流程、程序和資源可能隨著要求的變更而演變。
- 擁有者可做出關於何時應變更的明智決策。
- 審慎進行變更。

未建立此最佳實務時的風險暴露等級：中

實作指引

若要實作此最佳實務，您必須能夠要求變更流程、程序和資源。變更管理程序可以精簡。記錄變更管理程序。

客戶範例

AnyCompany Retail 使用責任指派 (RACI) 矩陣來識別誰擁有流程、程序和資源的變更。他們記錄了精簡且容易遵循的變更管理程序。使用 RACI 矩陣和程序，任何人都能提交變更要求。

實作步驟

1. 識別您工作負載的流程、程序和資源，及其各自的擁有者。在您的知識管理系統中加以記錄。
   a. 如果您尚未實作 OPS02-BP01 已為資源識別擁有者、OPS02-BP02 已為流程和程序識別擁有者或 OPS02-BP03 已為營運活動識別負責其效能的擁有者，請先予以實作。
2. 與組織中的利害關係人合作制定變更管理程序。此程序應涵蓋資源、流程和程序的新增、變更與例外狀況。
   a. 您可以使用 AWS Systems Manager Change Manager 作為工作負載資源的變更管理平台。
3. 在您的知識管理系統中記錄變更管理程序。

實作計劃的工作量：中。制定變更管理程序時，必須在整個組織的多個利害關係人之間取得共識。

資源

相關的最佳實務：

- OPS02-BP01 已為資源識別擁有者 - 在您建置變更管理程序之前，資源必須要有已識別的擁有者。
- OPS02-BP02 已為流程和程序識別擁有者 - 在您建置變更管理程序之前，程序必須要有已識別的擁有者。
- OPS02-BP03 已為營運活動識別負責其效能的擁有者 - 在您建置變更管理程序之前，營運活動必須要有已識別的擁有者。

相關文件：

- AWS 方案指引 - AWS 大型遷移的基礎程序手冊：建立 RACI 矩陣
- 雲端中的變更管理白皮書

相關服務：

- AWS Systems Manager Change Manager

OPS02-BP07 團隊之間的責任是預先定義或經過協商的

團隊間已定義或協商說明如何相互配合及支援的協議 (例如，回應時間、服務水準目標或服務水準協議)。團隊間的溝通管道記錄於文件中。透過了解團隊工作對於業務成果和其他團隊及組織成果的影響，可得知其任務的優先順序，並協助他們適當地回應。

如果責任和擁有權未定義或未知，則您會面臨風險，不僅無法及時處理必要的活動，在解決這些需求時還會出現冗餘和可能相互衝突的工作。

預期成果：

- 團隊間的工作或支援協議經過議定並記錄於文件中。
- 相互支援或合作的團隊定義了溝通管道和回應預期。

常見的反模式：

- 生產過程發生問題，兩個不同的團隊各自起始了疑難排解。其各自為政的工作使中斷更為嚴重。
- 營運團隊需要開發團隊的協助，但雙方並未就回應時間達成協議。要求卡在積存中。

建立此最佳實務的優勢：

- 團隊知道如何彼此互動與支援。
- 眾人對回應能力有相同的預期。
- 溝通管道明確定義。

未建立此最佳實務時的風險暴露等級：低

實作指引

實作此最佳實務意味著，團隊間對於彼此的合作方式不會有歧義。正式協議明訂了團隊相互合作或支援的方式。團隊間的溝通管道記錄於文件中。

客戶範例

AnyCompany Retail 的 SRE 團隊與其開發團隊間有一份服務水準協議。無論開發團隊是否是在其票證系統提出要求的，應該都能在十五分鐘內獲得回應。如果發生站點中斷，SRE 團隊將主導調查，並由開發團隊提供支援。

實作步驟

1. 與組織中的利害關係人合作，根據流程和程序制定團隊之間的協議。
   a. 如果兩個團隊之間共用流程或程序，請制定關於團隊應如何共事的執行手冊。
   b. 如果團隊之間相互依賴，請協議要求的回應 SLA。
2. 在您的知識管理系統中記錄責任。

實作計劃的工作量：中。如果團隊之間目前沒有任何協議，與組織中的利害關係人達成協議可能會頗費周章。

資源

相關的最佳實務：

- OPS02-BP02 已為流程和程序識別擁有者 - 必須在設定團隊之間的協議之前識別程序擁有權。
- OPS02-BP03 已為營運活動識別負責其效能的擁有者 - 必須在設定團隊之間的協議之前識別營運活動擁有權。

相關文件：

- AWS Executive Insights - 透過雙披薩團隊增添創新動能
- DevOps on AWS 簡介 - 雙披薩團隊

組織文化

為您的團隊成員提供支援，讓他們能夠更有效地採取動作以及支援業務成果。

最佳實務

- OPS03-BP01 高層的支持
- OPS03-BP02 授權團隊成員在成果有風險時採取動作
- OPS03-BP03 鼓勵向上呈報
- OPS03-BP04 溝通需及時、清楚且可行
- OPS03-BP05 鼓勵進行試驗
- OPS03-BP06 團隊成員得以並受到鼓勵來維持和發展自己的技能集
- OPS03-BP07 適當地為團隊提供資源
- OPS03-BP08 鼓勵並尋求來自團隊內部和跨團隊的多樣化建議

OPS03-BP01 高層的支持

資深領導階層清楚地設定對組織的期望並評估成功情況。資深領導階層是採用最佳實務和組織演進的發起者、倡導者和推動者

建立此最佳實務的優勢： 積極參與的領導階層、清楚傳達的期望和共同目標，能夠確保團隊成員知道對他們的期望。評估成功情況可識別成功的障礙，因此可藉由發起者、倡導者及其代表的介入來解決這些障礙。

若未建立此最佳實務，暴露的風險等級： 高

實作指引

- 高層的支持：資深領導階層清楚設定對組織的期望並評估成功情況。資深領導階層是採用最佳實務和組織演進的發起者、倡導者和推動者
  - 設定期望：為您的組織定義和發佈目標，包括衡量目標的方式。
  - 追蹤目標的達成情況：定期衡量目標逐步達成的情況，並分享結果，以便在成果有風險時採取適當的動作。

- 提供實現目標所需的資源：根據新資訊、目標的變更、責任或您的業務環境等，定期檢閱資源是否仍然適當，或者是否需要其他資源。
- 倡導您的團隊：保持與團隊的合作，讓您了解團隊的情況，以及是否有外部因素正影響著他們。當您的團隊受到外部因素影響時，請重新評估目標並適當地調整目標。找出阻礙您團隊進度的障礙。代表您的團隊來協助解決障礙並消除不必要的負擔。
- 成為採用最佳實務的推動者：確認提供量化效益的最佳實務，並認可建立者和採用者。鼓勵進一步採用，以擴大已達成的效益。
- 成為團隊演變的推動者：建立持續改進的文化。鼓勵人員和組織的成長和發展。提供需要隨時間逐步達成的長期奮鬥目標。調整這個願景，以在需求、業務目標以及業務環境變化時，配合您的需求、業務目標和業務環境。

OPS03-BP02 授權團隊成員在成果有風險時採取動作

工作負載擁有者已定義指引和範圍，授權團隊成員在成果有風險時做出回應。當事件超出定義的範圍時，採取向上呈報機制來取得方向。

建立此最佳實務的優勢： 透過及早測試和驗證變更，您能以最低的成本來解決問題，並限制對客戶的影響。在部署前進行測試，可將引入的錯誤數量降到最低。

若未建立此最佳實務，暴露的風險等級為： 高

實作指引

- 授權團隊成員在成果有風險時採取動作：為您的團隊成員提供許可、工具和機會，以練習有效回應所需的技能。
  - 讓您的團隊成員有機會練習回應所需的技能：提供替代的安全環境，在其中可安全地測試和培訓流程及程序。執行演練日，讓團隊成員可以在模擬的安全環境中獲得回應實際事件的體驗。
  - 定義並認可團隊成員採取動作的權限：透過指派許可和對其支援的工作負載和元件的存取權，明確地定義團隊成員採取動作的權限。認可他們有權在成果有風險時採取動作。

OPS03-BP03 鼓勵向上呈報

如果團隊成員認為成果有風險，則其機制可協助將疑慮向上呈報至決策制定者和利害關係人，而且我們鼓勵這麼做。應該儘早且經常向上呈報，以便識別風險，並防止風險引發事件。

若未建立此最佳實務，暴露的風險等級： 高

實作指引

- 鼓勵儘早且經常向上呈報：組織認可儘早且經常呈報是最佳實務。組織認可並接受，向上呈報可能經證明是毫無根據的，然而有機會防止事件的發生，則好過於不向上呈報而錯過該機會。
  - 制定向上呈報的機制：制定定義進行向上呈報的時機與方式的記錄程序。記錄擁有越來越多採取動作或核准動作的權限的人員及其聯絡資訊。向上呈報應持續進行，直到團隊成員確信已將風險交給能夠解決問題的人員，或已聯絡承擔工作負載營運風險和責任的人員。該人員最終負責與工作負載相關的所有決策。向上呈報的內容應包括風險的本質、工作負載的關鍵性、受影響者、影響為何，以及緊急性 (也就是預期影響的時間為何)。
  - 保護向上呈報的員工：如果團隊成員圍繞無回應決策制定者或利害關係人向上呈報，則制定保護團隊成員免受報復的政策。制定機制以識別是否發生此情況，並適當地做出回應。

OPS03-BP04 溝通需及時、清楚且可行

存在的機制可用來及時通知團隊成員已知的風險和計劃的事件。提供必要的內容、詳細資訊和時間 (如果可能) 來支援判斷是否需要採取動作、需要什麼動作，並及時採取動作。例如，提供軟體漏洞的通知，以便加快修補的速度，或提供計劃的銷售促銷活動的通知，如此就能實作變更凍結，避免服務中斷的風險。計劃的事件可以記錄在變更行事曆或維護排程中，讓團隊成員可以確定哪些活動待處理。

預期成果：

- 溝通提供了情境、詳細資料和時間的預期。
- 團隊成員明確了解採取行動回應溝通的時機和方式。
- 利用變更行事曆提供預期變更的通知。

常見的反模式：

- 某個誤報的提醒一週發生了數次。每次通知出現時，您都將其靜音。
- 系統要求您對安全群組進行變更，但未提供其執行時機的相關預期。
- 您在系統擴充規模時持續在聊天中收到通知，但無須執行任何動作。您避開聊天管道，因而錯過了重要通知。
- 在未通知營運團隊的情況下，就做了生產方面的變更。該變更觸發了提醒，並啟用了值班團隊。

建立此最佳實務的優勢：

- 組織可避免出現警示疲勞。

- 團隊成員可依據必要的情境和預期採取行動。
- 變更可在變更時段內進行，因而降低風險。

未建立此最佳實務時的風險暴露等級：高

實作指引

若要實作此最佳實務，您必須與組織中的利害關係人共同達成溝通標準的協議。在組織內將這些標準公告週知。識別誤判或持續開啟的提醒，並加以移除。使用變更行事曆，讓團隊成員得知何時應採取行動，以及哪些活動擱置中。確認溝通可提供必要的情境，進而形成明確的行動。

客戶範例

AnyCompany Retail 以聊天作為其主要的溝通媒介。特定管道會填入提醒和其他資訊。人們必須展開行動時，將可明確參考預期成果，且在許多情況下都會有執行手冊或程序手冊可使用。他們可使用變更行事曆來排程生產系統的重大變更。

實作步驟

1. 分析您的提醒是否為誤判或是不斷觸發的提醒。加以移除或變更，使其僅在需要人為介入時觸發。如果觸發了提醒，請提供執行手冊或程序手冊。
   a. 您可以使用 AWS Systems Manager Documents 來建置提醒的程序手冊和執行手冊。
2. 已設立機制，以清楚且可行的方式提供風險或計劃事件的通知，並提供足夠的通知，以便適當的回應。使用電子郵件清單或聊天管道，在計劃性事件發之前傳送通知。
   a. AWS Chatbot 可在您的組織傳訊平台內用來傳送提醒及回應事件。
3. 提供可存取的資訊來源，您可以在其中發現計劃的事件。提供來自相同系統之計劃事件的通知。
   a. AWS Systems Manager 變更行事曆可用來建立可進行變更的變更時段。這可為團隊成員提供有關於何時可安全進行變更的通知。
4. 監控漏洞通知和修補程式資訊，了解外部漏洞以及與工作負載元件相關的潛在風險。提供通知給團隊成員，讓他們可以採取動作。
   a. 您可以訂閱 AWS 安全公告，以接收 AWS 相關漏洞的通知。

資源

相關的最佳實務：

- OPS07-BP03 使用執行手冊執行程序 - 在成果確知時提供執行手冊，使溝通化為實際行動。

- OPS07-BP04 使用程序手冊來調查問題 - 在成果不明的情況下，程序手冊可讓溝通化為實際行動。

相關文件：

- AWS 安全公告
- Open CVE

相關範例：

- Well-Architected 實驗室：清查和修補程式管理 (Level 100)

相關服務：

- AWS Chatbot
- AWS Systems Manager 變更行事曆
- AWS Systems Manager Documents

OPS03-BP05 鼓勵進行試驗

試驗是將新構想轉化為產品和功能的觸媒。試驗可加速學習，讓團隊成員保持興趣和參與度。我們鼓勵團隊成員經常進行試驗以推動創新。即便結果不如預期仍有其價值，至少我們了解到什麼是不該做的。團隊成員不會因取得不理想結果的成功試驗而受懲罰。

預期成果：

- 您的組織鼓勵試驗以促進創新。
- 試驗被視為一種學習機會。

常見的反模式：

- 您想要執行 A/B 測試，但沒有相關機制可執行試驗。您在沒有測試能力的情況下部署了 UI 變更。其結果導致了負面客戶體驗。
- 您的公司只有模擬和生產環境。沒有沙盒環境可用來試驗新功能或產品，因此您必須在生產環境內試驗。

建立此最佳實務的優勢：

- 試驗可帶動創新。
- 透過試驗，您可以更快回應使用者的意見反映。
- 組織可培養學習文化。

未建立此最佳實務時的風險暴露等級：中

實作指引

試驗應以安全的方式執行。利用多種環境進行試驗，而不會損害生產資源。使用 A/B 測試和功能旗標來測試試驗。為團隊成員提供在沙盒環境中執行試驗的能力。

客戶範例

AnyCompany Retail 鼓勵試驗。團隊成員可將其 20% 的工時投入於試驗或學習新技術。他們有沙盒環境可供創新之用。他們可對新功能進行 A/B 測試，用實際使用者的意見反映加以驗證。

實作步驟

1. 與組織中的領導階層共同推行試驗風氣。應鼓勵團隊成員以安全的方式執行試驗。
2. 為團隊成員提供可安全進行試驗的環境。他們必須能夠存取類似生產環境的環境。
   a. 您可以使用個別的 AWS 帳戶 建立沙盒環境，以供試驗之用。AWS Control Tower 可用來佈建這些帳戶。
3. 使用功能旗標和 A/B 測試安全地進行試驗，並收集使用者的意見反映。
   a. AWS AppConfig Feature Flags 提供建立功能旗標的能力。
   b. Amazon CloudWatch Evidently 可用來對受限部署執行 A/B 測試。
   c. 您可以使用 AWS Lambda 版本部署用於 Beta 測試的新版功能。

實作計劃的工作量：高。為團隊成員提供可安全執行試驗的環境，可能需要可觀的投資。為了使用功能旗標或支援 A/B 測試，您也可能需要修改應用程式程式碼。

資源

相關的最佳實務：

- OPS11-BP02 執行事故後分析 - 從事件中學習與試驗同樣為推動創新的重要因子。
- OPS11-BP03 實作回饋迴圈 - 回饋迴圈是試驗的重要環節。

相關文件：

- 深入了解 Amazon 文化：試驗、失敗、客戶至上
- 在 AWS 中建立和管理沙盒帳戶的最佳實務
- 樹立雲端造就的試驗文化
- 在 SulAmérica Seguros 透過雲端實行試驗和創新
- 試驗愈多次，就愈可能成功
- 使用多個帳戶整理您的 AWS 環境 - 沙盒 OU
- 使用 AWS AppConfig Feature Flags

相關影片：

- AWS On Air ft.Amazon CloudWatch Evidently | AWS Events
- AWS On Air San Fran Summit 2022 ft.AWS AppConfig Feature Flags 與 Jira 整合
- AWS re:Invent 2022 - 部署並非發行：使用功能旗標控制您的推出的項目 (BOA305-R)
- 透過 AWS Control Tower 以程式設計方式建立 AWS 帳戶
- 設定會使用 AWS Organizations 最佳實務的多帳戶 AWS 環境

相關範例：

- AWS 創新沙盒
- End-to-end Personalization 101 for E-Commerce

相關服務：

- Amazon CloudWatch Evidently
- AWS AppConfig
- AWS Control Tower

OPS03-BP06 團隊成員得以並受到鼓勵來維持和發展自己的技能集

團隊必須發展自己的技能集，以採用新技術，並支援需求和責任的變更，以支援您的工作負載。新技術的技能成長通常是團隊成員滿意度的來源，並可支援創新。支援團隊成員追求和維持產業認證，以驗證

和認可他們不斷成長的技能。交叉培訓以促進知識轉移，並在失去熟練的、經驗豐富且具備機構知識的成員時，降低重大影響的風險。提供學習專用的結構化時間。

AWS 提供了許多資源，包括 AWS 入門資源中心， AWS 部落格， AWS 線上技術會談， AWS 活動和網路研討會以及 AWS Well-Architected 實驗室，而這些資源均提供了可教育您的團隊的說明、範例和演練。

AWS 也分享了我們透過 在 Amazon Builders' Library 中 操作 AWS 所學到的最佳實務和模式，以及透過 不同途徑獲得的各種教材，例如 AWS 部落格 和 官方 AWS 播客。

您應該利用 AWS 提供的教育資源，例如 Well-Architected 實驗室、 AWS Support (AWS 知識中心， AWS 開發論壇和 AWS Support中心) 和 AWS 文件 資源來教育您的團隊。透過 AWS Support中心聯絡 AWS Support，以獲取 AWS 相關問題的幫助。

AWS 培訓 和認證 透過 AWS 基礎原理自主進度數位課程提供一些免費培訓。您還可以報名參加講師指導下的培訓，以進一步協助開發團隊的 AWS 技能。

若未建立此最佳實務，暴露的風險等級為： 中

實作指引

- 團隊成員得以並受到鼓勵來維持和發展自己的技能集：若要採用新技術、支援創新、需求和責任的變更以支援工作負載，必需進行持續的教育。
  - 為教育提供資源：提供專門的結構化時間、培訓教材和實驗室資源存取權，並支援參與會議和專業組織，這些會議和組織可為教育工作者和同儕提供學習的機會。為資淺團隊成員提供接近資深團隊成員的機會，讓資深團隊成員成為導師，或允許資深團隊成員伴隨資淺團隊成員工作，並展示他們的方法和技能。鼓勵學習與工作不直接相關的內容，以便取得更廣泛的視野。
  - 團隊教育和跨團隊參與：針對團隊成員持續的教育需求進行規劃。為團隊成員提供機會 (暫時或永久地) 加入其他團隊，以分享讓整個組織受益的技能和最佳實務
  - 支援產業認證的追求和維持：支援團隊成員取得與維持可驗證所學知識並認可其成就的產業認證。

資源

相關文件：

- AWS 入門資源中心
- AWS 部落格
- AWS 雲端 合規
- AWS 開發論壇

- AWS 文件
- AWS 線上技術會談
- AWS 活動和網路研討會
- AWS 知識中心
- AWS Support
- AWS 培訓 和認證
- AWS Well-Architected 實驗室，
- 在 Amazon Builders' Library 中
- 官方 AWS 播客。

OPS03-BP07 適當地為團隊提供資源

維持團隊成員能力，並提供工具和資源，以支援您的工作負載需求。為團隊成員指派過多的任務會增加因人為錯誤所造成的事件風險。對工具和資源的投資 (例如，為經常執行的活動提供自動化) 可以提高團隊的有效性，讓他們能夠支援其他的活動。

若未建立此最佳實務，暴露的風險等級： 中

實作指引

- 適當地為團隊提供資源：確保您了解團隊的成功，以及促成他們成功或不成功的因素。以適當的資源來支援團隊。
  - 了解團隊效能：衡量團隊營運成果的達成情況與資產的開發。追蹤一段時間內輸出和錯誤率的變更。與團隊合作，了解影響他們的工作相關挑戰 (例如，責任增加、技術變更、人員損失或客戶支援增加)。
  - 了解對團隊效能的影響：保持與團隊的合作，讓您了解團隊的情況，以及是否有外部因素正影響著他們。當您的團隊受到外部因素影響時，請重新評估目標並適當地調整目標。找出阻礙您團隊進度的障礙。代表您的團隊來協助解決障礙並消除不必要的負擔。
  - 提供團隊取得成功所需的資源：定期檢閱資源是否仍然適當、或是否需要額外資源，並做出適當的調整以支援團隊。

OPS03-BP08 鼓勵並尋求來自團隊內部和跨團隊的多樣化建議

利用跨組織的多樣性，尋求多種獨特的觀點。使用此觀點來增加創新、挑戰假設，並降低確認偏差的風險。在團隊中增加包容性、多樣性和可及性，以獲得有益的觀點。

組織文化對團隊成員工作滿意度和留任率有直接影響。讓團隊成員參與其中並習得能力，以便讓業務得以成功。

若未建立此最佳實務，暴露的風險等級： 低

實作指引

- 尋求多樣化的意見和觀點：鼓勵每個人做出貢獻。為代表性不足的群體發聲。在會議中輪換角色和責任。
  - 擴展角色和責任：為團隊成員提供機會，以擔任他們可能不會擔任的角色。他們會透過角色，以及與他們可能不會與之互動的新團隊成員互動，而獲得經驗和觀點。他們會將自己的經驗和觀點帶到新的角色，並帶給和他們互動的團隊成員。隨著觀點增加，可能會出現額外的商機，或者可能識別出新的改進機會。讓團隊內的成員輪流處理其他人通常執行的常見任務，以了解執行這些任務的需求和影響。
  - 提供安全且友善的環境：制定政策與控制措施，保護組織內團隊成員在精神和身體上的安全。團隊成員應該能夠在不擔心報復行為的情況下進行互動。當團隊成員感到安全且受歡迎時，他們才更有可能參與進來並具備生產力。您的組織越多樣化，您就越能了解所支援的人員，包括您的客戶。當您的團隊成員感到安心、可以自在的暢所欲言，而且有信心他們的聲音不會被淹沒，他們才更有可能分享寶貴的洞見 (例如，行銷機會、可及性的需求、尚未有服務的市場區段、環境中未確認的風險)。
  - 讓團隊成員能夠充分參與：提供員工充分參與所有與工作相關的活動所需的資源。面對日常挑戰的團隊成員已發展出解決挑戰的技能。這些以獨特方式發展的技能可為組織提供顯著的效益。為團隊成員提供必要住宿支援，將可提高從他們的貢獻中所獲得的效益。

準備

要為卓越營運做好準備，您必須了解您的工作負載及其預期行為。然後，您就能將其設計出來，以了解它們的狀態並建置可提供支援的程序。

要為卓越營運做好準備，您需要考慮以下事項：

主題

- 實作可觀測性
- 營運設計
- 緩解部署風險
- 營運準備度和變更管理

實作可觀測性

在工作負載中實作可觀測性，以便了解其狀態，並根據業務需求做出資料驅動的決策。

可觀測性不僅是單純的監控，還可根據系統的外部輸出全面了解系統的內部運作狀況。以指標、日誌和追蹤為根基，可觀測性提供了系統行換動態的洞見。有效的可觀測性能夠讓團隊辨別模式、異常情況和趨勢，以便主動解決潛在問題並維持最佳的系統運作狀態。

確定關鍵績效指標 (KPI) 至關重要，可確保監控活動與業務目標保持一致。這種一致性可確保團隊使用真正重要的指標來做出資料驅動的決策，進而最佳化系統效能和業務成果。

此外，可觀測性使得企業能夠化被動為主動。團隊能夠了解系統內的因果關係，預測並預防問題，而不只是被動回應問題。隨著工作負載的演進，務必重新檢視並改進可觀測性策略，以保持相關性和有效性。

最佳實務

- OPS04-BP01 識別關鍵績效指標
- OPS04-BP02 實作應用程式遙測
- OPS04-BP03 實作使用者體驗遙測
- OPS04-BP04 實作相依性遙測
- OPS04-BP05 實作分散式追蹤

OPS04-BP01 識別關鍵績效指標

想在工作負載中實作可觀測性，要先了解工作負載狀態，並根據業務需求做出資料驅動的決策。確保監控活動與業務目標保持一致的最有效方式之一，就是定義和監控關鍵績效指標 (KPI)。

預期成果： 有效率的可觀測性實作會與業務目標密切保持一致，確保監控工作始終能夠帶來實際的業務成果。

常見的反模式：

- 未定義 KPI：在沒有明確 KPI 的情況下工作，可能會導致監控過度或不足，而錯過重要訊號。
- 靜態 KPI：未隨著工作負載或業務目標發展而重新檢視或改進 KPI。
- 未能保持一致：專注於與業務成果沒有直接關係的技術指標，或難與實際問題相關聯的技術指標。

建立此最佳實務的優勢：

- 容易識別問題：業務 KPI 通常比技術指標更能清楚呈現問題所在。比起從眾多技術指標中苦苦尋找，業務 KPI 下降的現象，更能有效地指出問題所在。
- 業務一致性：確保監控活動可直接支援業務目標。
- 效率：優先監控資源並關注重要指標。
- 主動積極：找出並解決問題，不讓問題擴大影響業務。

未建立此最佳實務時的曝險等級： 高

實作指引

若要有效地定義工作負載 KPI：

1. 從業務成果開始著手： 在深入研究指標之前，請先了解所需的業務成果。想要增加銷售量、提高使用者參與度，還是加快回應時間？
2. 讓技術指標與業務目標相互關聯： 並非所有技術指標都會直接影響業務成果。找出有直接影響的技術指標，不過，通常更直接的方式是使用業務 KPI 找出問題。
3. 使用 Amazon CloudWatch： 採用 CloudWatch 定義和監控代表您的 KPI 的指標。
4. 定期檢閱和更新 KPI： 隨著工作負載和業務發展，保持 KPI 的相關性。
5. 讓利害關係人參與： 讓技術和業務團隊一起參與定義和檢閱 KPI 的過程。

實作計劃的工作量： 中

資源

相關的最佳實務：

- the section called “OPS04-BP02 實作應用程式遙測”
- the section called “OPS04-BP03 實作使用者體驗遙測”
- the section called “OPS04-BP04 實作相依性遙測”
- the section called “OPS04-BP05 實作分散式追蹤”

相關文件：

- AWS 可觀測性最佳實務
- CloudWatch 使用者指南
- AWS 可觀測性 Skill Builder 課程

相關影片：

- 研擬可觀測性策略

相關範例：

- One Observability 研討會

OPS04-BP02 實作應用程式遙測

應用程式遙測是工作負載可觀測性的基礎。提供遙測相當重要，因為能讓您獲得可付諸行動的洞見，深入了解應用程式的狀態以及實現的技術與業務成果。從疑難排解到衡量新功能的影響，或確保與業務關鍵績效指標 (KPI) 保持一致，應用程式遙測都能為您指出建置、操作和發展工作負載的方式。

指標，日誌和追蹤是構成可觀測性的三大要素。這些要素可做為診斷工具來描述應用程式的狀態。經過一段時間後，這些要素可協助建立基準和識別異常狀況。然而，為了確保監控活動與業務目標保持一致，就必須定義並監控 KPI。與單獨的技術指標相比，業務 KPI 通常更容易找出問題所在。

其他遙測類型 (例如實際使用者監控 (RUM) 和綜合交易) 可與這些主要資料來源相輔相成。RUM 提供即時使用者互動的洞見，而綜合交易則模擬可能的使用者行為，有助於在實際使用者遇到瓶頸之前便偵測到瓶頸。

預期成果： 獲得工作負載效能且可付諸行動的洞見。這些洞見可讓您做出有關效能最佳化的主動決策、提高工作負載穩定性、使 CI/CD 程序更順暢，並且有效利用資源。

常見的反模式：

- 不完整的可觀測性：忽略在工作負載的每一層納入可觀測性，導致出現可能遮蔽重要系統效能和行為洞見的盲點。
- 分散的資料檢視：當資料分散在多個工具和系統中時，便難以提供涵蓋工作負載運作狀況和效能的全面概覽。
- 使用者回報的問題：這種現象表示未能透過遙測和業務 KPI 監視進行主動問題偵測。

建立此最佳實務的優勢：

- 明智的決策：透過遙測和業務 KPI 獲得洞見，就能做出資料驅動的決策。
- 改善運作效率：資料驅動的資源利用率可帶來成本效益。
- 提高工作負載穩定性：更快偵測並解決問題，進而改善正常運作。
- 更順暢的 CI/CD 程序：從遙測資料獲得的洞見，有助於改進程序並交付可靠的程式碼。

未建立此最佳實務時的曝險等級： 高

實作指引

若要為您的工作負載實作應用程式遙測，請使用類似以下的 AWS 服務： Amazon CloudWatch 和 AWS X-Ray。Amazon CloudWatch 提供了一套全方位的監控工具，可讓您在 AWS 和內部部署環境中觀察資源和應用程式，還會收集、追蹤和分析指標、合併和監控日誌資料，並且回應資源的變更，以增進您對工作負載運作方式的了解。同時，AWS X-Ray 可讓您追蹤、分析和偵錯應用程式，藉此深入了解工作負載的行為。透過像是服務圖、延遲分佈情形和追蹤時間軸等功能，X-Ray 提供了洞見，讓您深入了解工作負載的效能及影響它的瓶頸。

實作步驟

1. 確定要收集的資料： 確定可提供工作負載運作狀況、效能和行為實質洞見的重要指標、日誌和追蹤。
2. 部署 CloudWatch 代理程式： CloudWatch 代理程式的作用在於，方便您從工作負載及其基礎設施中取得系統和應用程式指標和日誌。CloudWatch 代理程式也可用來收集 OpenTelemetry 或 X-Ray 追蹤，並傳送至 X-Ray。
3. 定義和監控業務 KPI： 建立 自訂指標 並與您的 業務成果保持一致。

4. 使用 AWS X-Ray 檢測您的應用程式： 除了部署 CloudWatch 代理程式之外，務必也要 檢測您的應用程式 以產生追蹤資料。此程序可提供工作負載行為和效能的進一步洞見。
5. 將整個應用程式的資料收集標準化： 將整個應用程式的資料收集實務標準化。採取一致的方式有助於找出資料關聯並進行分析，進而提供應用程式行為的全面概覽。
6. 分析資料並採取行動： 一旦有了資料收集和標準化的方式，就可使用 Amazon CloudWatch 進行指標和日誌分析，以及使用 AWS X-Ray 進行追蹤分析。這類分析可產生有關工作負載運作狀況、效能和行為的洞見，進而引導您進行決策。

實作計劃的工作量： 高

資源

相關的最佳實務：

- OPS04-BP01 識別關鍵績效指標
- OPS04-BP03 實作使用者體驗遙測
- OPS04-BP04 實作相依性遙測
- OPS04-BP05 實作分散式追蹤

相關文件：

- AWS 可觀測性最佳實務
- CloudWatch 使用者指南
- AWS X-Ray 開發人員指南
- 檢測分散式系統，以了解運作狀態
- AWS 可觀測性 Skill Builder 課程
- Amazon CloudWatch 最新消息
- AWS X-Ray 最新消息

相關影片：

- AWS re:Invent 2022 - Amazon 的可觀測性最佳實務
- AWS re:Invent 2022 - 研擬可觀測性策略

相關範例：

- One Observability 研討會
- AWS 解決方案程式庫：使用 Amazon CloudWatch 進行應用程式監控

OPS04-BP03 實作使用者體驗遙測

深入了解客戶體驗以及與應用程式的互動情形非常重要。實際使用者監控 (RUM) 和綜合交易正是合適的強大工具。從 RUM 提供的實際使用者互動相關資料，能夠獲悉真實的使用者滿意度，而綜合交易則會模擬使用者互動，有助於偵測潛在問題，提早防範問題影響實際使用者。

預期成果： 提供使用者體驗、主動偵測問題及最佳化使用者互動的整體概觀，從而獲得順暢的數位體驗。

常見的反模式：

- 沒有實際使用者監控 (RUM) 的應用程式：
  - 延遲偵測到問題：如果沒有 RUM，您可能直到收到使用者投訴，才察覺到效能瓶頸或問題。這種被動回應的方式可能導致客戶不滿意。
  - 缺乏使用者體驗洞見：未使用 RUM 代表您無法獲得使用者與應用程式實際互動情形的重要資料，因此也限制了您最佳化使用者體驗的能力。
- 沒有綜合交易的應用程式：
  - 缺少邊緣案例：綜合交易可協助您測試一般使用者可能不常使用，但對於某些業務功能來說相當關鍵的路徑和功能。缺少的話，這些路徑可能無法正常運作並遭到忽視。
  - 在應用程式未使用的情況下檢查問題：定期綜合測試可模擬實際使用者未積極與您的應用程式互動的情況，進而確保系統隨時正常運作。

建立此最佳實務的優勢：

- 主動偵測問題：找出並解決潛在問題，避免進一步影響實際使用者。
- 最佳化使用者體驗：RUM 提供持續的意見回饋，有助於改進並強化整體使用者體驗。
- 裝置和瀏覽器效能的相關洞見：了解您的應用程式在不同裝置和瀏覽器上的效能表現，以便進一步最佳化。
- 經驗證的業務工作流程：定期綜合交易可確保核心功能和重要路徑維持正常且高效率的運作。
- 增強應用程式效能：利用收集自實際使用者資料的洞見來改善應用程式回應能力和可靠性。

未建立此最佳實務時的曝險等級： 高

實作指引

為了利用 RUM 和綜合交易進行使用者活動遙測，AWS 提供了類似以下的服務： Amazon CloudWatch RUM 和 Amazon CloudWatch Synthetics。指標、日誌和追蹤搭配使用者活動資料，可提供深入應用程式運作狀態和使用者體驗的全方位檢視。

實作步驟

1. 部署 Amazon CloudWatch RUM： 將您的應用程式與 CloudWatch RUM 整合，以收集、分析和呈現實際使用者資料。
   a. 使用 CloudWatch RUM JavaScript 程式庫 將 RUM 與您的應用程式整合。
   b. 設定儀表板以視覺化和監控實際使用者資料。
2. 設定 CloudWatch Synthetics： 建立 Canary 或指令碼編寫的常式，以模擬使用者與應用程式的互動。
   a. 定義關鍵應用程式工作流程和路徑。
   b. 使用 CloudWatch Synthetics 指令碼 設計 Canary 以模擬這些路徑的使用者互動。
   c. 排定依指定間隔執行 Canary 並進行監控，確保一致的效能檢查。
3. 分析資料並採取行動： 利用來自 RUM 和綜合交易的資料獲得洞見，並於偵測到異常時採取修正措施。使用 CloudWatch 儀表板和警報隨時掌握情況。

實作計劃的工作量： 中

資源

相關的最佳實務：

- OPS04-BP01 識別關鍵績效指標
- OPS04-BP02 實作應用程式遙測
- OPS04-BP04 實作相依性遙測
- OPS04-BP05 實作分散式追蹤

相關文件：

- Amazon CloudWatch RUM 指南
- Amazon CloudWatch Synthetics 指南

相關影片：

- 透過最終使用者洞察與 Amazon CloudWatch RUM 最佳化應用程式
- AWS on Air ft.Amazon CloudWatch 的實際使用者監控

相關範例：

- One Observability 研討會
- Amazon CloudWatch RUM Web 用戶端的 Git 儲存庫
- 使用 Amazon CloudWatch Synthetics 測量頁面載入時間

OPS04-BP04 實作相依性遙測

對於監控工作負載所依賴的外部服務和元件運作狀況與效能，相依性遙測至關重要，可提供連線能力、逾時，以及像是 DNS、資料庫或第三方 API 等其他與相依性相關重要事件的寶貴洞見。藉由檢測應用程式，產生有關這些相依性的指標、日誌和追蹤，便可更清楚了解可能影響工作負載的潛在瓶頸、效能問題或故障。

預期成果： 工作負載所依賴的相依性如預期般正常運作，讓您能夠主動解決問題並確保最佳的工作負載效能。

常見的反模式：

- 忽略外部相依性：僅關注內部應用程式指標，而忽略與外部相依性相關的指標。
- 缺乏主動監控：等待問題出現，而非持續監控相依性的運作狀況與效能。
- 單獨運作的監控：使用多種分散的監控工具，如此可能導致僅片段掌握相依性運作狀況且獲得的資訊不一致。

建立此最佳實務的優勢：

- 改善工作負載可靠性：確保外部相依性穩定運作並保持最佳效能。
- 更快偵測並解決問題：主動找出並解決相依性相關問題，不讓問題影響工作負載。
- 全方位視角：獲得全方位視角，有效掌握影響工作負載運作狀況的內部和外部元件。
- 增強工作負載可擴展性：了解外部相依性的可擴展性限制與效能特性。

未建立此最佳實務時的曝險等級： 高

實作指引

從識別您的工作負載所依賴的服務、基礎設施和程序開始，實作相依性遙測。將相依性正常運作時的良好條件量化，然後判斷衡量時所需的資料。有了這些資訊，您就可以打造儀表板並設定警示，以便為營運團隊提供這些相依性狀態的洞見。相依性無法按需求運作時，使用 AWS 工具探索並量化其影響。不斷重新檢視您的策略，以考量優先順序、目標和所獲得洞見的變化。

實作步驟

若要有效實作相依性遙測：

1. 識別外部相依性： 與利害關係人協作，共同找出工作負載所依賴的外部相依性。外部相依性可能包含各種服務，像是外部資料庫、第三方 API、前往其他環境的網路連線能力路由，以及 DNS 服務。實現有效相依性遙測的第一步，就是徹底了解這些相依性。
2. 擬訂監控策略： 清楚了解外部相依性之後，就可以為其量身打造監控策略。這包括了解每一項相依性的重要性、預期行為，以及任何相關的服務層級協議或目標 (SLA 或 SLT)。設定主動警示，以便在發生狀態變更或效能偏差時通知您。
3. 利用 Amazon CloudWatch 網路監視器： 提供了深入全球網際網路的洞見，有助於了解可能影響外部相依性的中斷或干擾情況。
4. 透過 AWS Health Dashboard 隨時掌握資訊： 會在 AWS 發生可能影響服務的事件時，發出警示並提供修補指引。
5. 使用 AWS X-Ray 檢測您的應用程式： AWS X-Ray 提供了深入了解應用程式及其基礎相依性運作效能的洞見。透過從頭到尾追蹤請求，您就可以找出應用程式所依賴的外部服務或元件的瓶頸或故障。
6. 使用 Amazon DevOps Guru： 這項機器學習驅動的服務可識別操作問題，預測重大問題可能在什麼時候發生，並且建議可採取的特定行動。對於獲得相依性洞見並確定它們不是造成操作問題的根源來說，這項服務非常寶貴。
7. 定期監控： 持續監控與外部相依性相關的指標和日誌。針對非預期的行為或效能降低的情況設定警示。
8. 變更後驗證： 每當有任何外部相依性更新或變更，便驗證其效能並檢查是否符合您的應用程式需求。

實作計劃的工作量： 中

資源

相關的最佳實務：

- OPS04-BP01 識別關鍵績效指標
- OPS04-BP02 實作應用程式遙測
- OPS04-BP03 實作使用者體驗遙測
- OPS04-BP05 實作分散式追蹤

相關文件：

- 什麼是 AWS Health？
- 使用 Amazon CloudWatch 網路監視器
- AWS X-Ray 開發人員指南
- Amazon DevOps Guru 使用者指南

相關影片：

- 深入了解影響應用程式效能的網際網路問題
- Amazon DevOps Guru 簡介

相關範例：

- 使用 Amazon DevOps Guru 獲得 AIOps 的運作洞見
- AWS Health Aware

OPS04-BP05 實作分散式追蹤

分散式追蹤可讓您監控和以視覺化的方式了解，在分散式系統中各種來回移動元件的請求。透過從多個來源擷取追蹤資料並在統一的檢視中進行分析，團隊就能更了解請求的流程、瓶頸出現的位置，以及最佳化工作應著重的地方。

預期成果： 提供分散式系統請求流程的全面概覽，實現精確偵錯、最佳化效能，並改善使用者體驗。

常見的反模式：

- 不一致的檢測：並非所有分散式系統中的服務都經過檢測可進行追蹤。
- 忽略延遲：僅專注於錯誤，而未考慮延遲或效能逐漸降低的現象。

建立此最佳實務的優勢：

- 全方位的系統概觀：從進入到退出，徹底視覺化整個請求路徑。
- 強化偵錯：快速識別失敗或效能問題發生的位置。
- 改善使用者體驗：根據實際使用者資料進行監控與最佳化，確保系統符合實際需求。

未建立此最佳實務時的曝險等級： 高

實作指引

首先，識別工作負載中需要檢測的所有元素。將所有元件列入考量之後，就可以利用像是 AWS X-Ray 和 OpenTelemetry 等工具來收集追蹤資料，以便使用 X-Ray 和 Amazon CloudWatch ServiceLens Map 等工具進行分析。與開發人員一起進行定期檢閱，並在討論過程中利用 Amazon DevOps Guru、X-Ray Analytics 和 X-Ray Insights 等工具進行補充，以協助發掘更深入的調查結果。從追蹤資料建立警示，以便在工作負載監視計畫中定義的結果存在風險時發出通知。

實作步驟

若要有效實作分散式追蹤：

1. 採用 AWS X-Ray： 將 X-Ray 整合到您的應用程式中，以獲得深入其行為的洞見、了解效能，並且找出瓶頸的確切位置。利用 X-Ray Insights 進行自動化追蹤分析。
2. 檢測您的服務： 確認每一項服務 (從 AWS Lambda 函數到 EC2 執行個體)都會傳送追蹤資料。檢測的越多項服務，端對端檢視就越清楚。
3. 納入 CloudWatch 實際使用者監控 和 綜合監控： 將實際使用者監控 (RUM) 和綜合監控與 X-Ray 整合在一起。這樣就能擷取實際使用者體驗並模擬使用者互動，以從中找出潛在問題。
4. 使用 CloudWatch 代理程式： 代理程式可從 X-Ray 或 OpenTelemetry 傳送追蹤，進而獲得更深入的洞見。
5. 使用 Amazon DevOps Guru： DevOps Guru 使用來自 X-Ray、CloudWatch、AWS Config 和 AWS CloudTrail 的資料提供可付諸行動的建議。
6. 分析追蹤： 定期檢閱追蹤資料，以找出可能影響應用程式效能的模式、異常或瓶頸。
7. 設定警示： 在 CloudWatch 中設定警報來通報不尋常的模式或過久的延遲，以主動解決問題。
8. 持續改善： 隨著服務增加或修改重新檢視您的追蹤策略，以擷取所有相關資料點。

實作計劃的工作量： 中

資源

相關的最佳實務：

- OPS04-BP01 識別關鍵績效指標
- OPS04-BP02 實作應用程式遙測
- OPS04-BP03 實作使用者體驗遙測
- OPS04-BP04 實作相依性遙測

相關文件：

- AWS X-Ray 開發人員指南
- Amazon CloudWatch 代理程式使用者指南
- Amazon DevOps Guru 使用者指南

相關影片：

- 使用 AWS X-Ray Insights
- AWS on Air ft.可觀測性：Amazon CloudWatch 和 AWS X-Ray

相關範例：

- 使用 AWS X-Ray 檢測您的應用程式

營運設計

採用的方法需能夠改善變更發揮作用的流程，並有助於重構、快速提供品質意見回饋，以及修復錯誤。這些方法會加快有助益的改變發揮作用的速度、限制部署問題，並能快速識別和修復部署活動造成的問題。

在 AWS 中，您可以程式碼檢視您的整個工作負載 (應用程式、基礎設施、原則、管控和營運)。所有這些均可在其中予以定義並使用程式碼進行更新。這表示您可以在您堆疊的每個元素中套用與您應用程式程式碼所用相同的工程原則。

最佳實務

- OPS05-BP01 使用版本控制

- OPS05-BP02 測試並驗證變更
- OPS05-BP03 使用組態管理系統
- OPS05-BP04 使用建置和部署管理系統
- OPS05-BP05 執行修補程式管理
- OPS05-BP06 共用設計標準
- OPS05-BP07 實作用於提高程式碼品質的實務
- OPS05-BP08 使用多個環境
- OPS05-BP09 進行頻繁、細微和可逆的變更
- OPS05-BP10 完全自動化整合和部署

OPS05-BP01 使用版本控制

使用版本控制來追蹤變更和發佈。

許多 AWS 服務都提供版本控制功能。使用修訂版或原始程式碼控制系統 (例如 AWS CodeCommit)，管理程式碼和其他成品，例如基礎架構之版本控制的 AWS CloudFormation 範本。

預期成果： 您的團隊共同撰寫程式碼。合併後，程式碼會是一致的，且變更不會遺失。透過正確的版本控制就能輕鬆復原錯誤。

常見的反模式：

- 您已在工作站上開發和儲存程式碼。您的工作站發生無法復原的儲存錯誤，造成程式碼遺失。
- 變更覆寫現有的程式碼之後，您重新啟動應用程式卻無法運作。您無法還原變更。
- 您對其他人要編輯的報告檔案加上了寫入鎖定。他們會與您聯絡，要求您停止處理該檔案，以便完成任務。
- 您的研究團隊一直在進行詳細的分析，以塑造您未來的工作。某人意外地將自己的購物清單儲存在最終報告中。您無法還原變更，且必須重新建立報告。

建立此最佳實務的優勢： 透過使用版本控制功能，您可以輕鬆回復為已知的良好狀態和舊版本，並有效降低資產遺失的風險。

未建立此最佳實務時的曝險等級： 高

實作指引

在版本控制的儲存庫中維護資產。此舉可實現變更追蹤、新版本部署、對現有版本的變更偵測以及還原到先前的版本 (例如，在發生故障時復原到已知的良好狀態)。將組態管理系統的版本控制功能整合到您的程序中。

資源

相關的最佳實務：

- OPS05-BP04 使用建置和部署管理系統

相關文件：

- 什麼是 AWS CodeCommit？

相關影片：

- AWS CodeCommit 簡介

OPS05-BP02 測試並驗證變更

所部署的每項變更都必須經過測試，以避免在生產環境中發生錯誤。此一最佳實務著重於各種變更 (從版本控制到成品組建) 的測試。除了應用程式的程式碼變更以外，測試也應包含基礎設施、組態、安全控制和操作程序。測試採取多種形式，從單元測試到軟體元件分析 (SCA) 都包括在內。將測試進一步納入軟體整合和交付程序中，可進一步確保成品的品質。

您的組織必須制定所有軟體成品的測試標準。自動化測試可節省人力並避免手動測試錯誤。在某些情況下可能需進行手動測試。開發人員必須有權存取自動化測試結果，以建立可改善軟體品質的回饋迴圈。

預期成果： 您的軟體變更在交付前都經過測試。開發人員有權存取測試結果和驗證。您的組織具有適用於所有軟體變更的測試標準。

常見的反模式：

- 您在部署新軟體變更時未進行任何測試。軟體在生產環境中無法執行，因而導致中斷。
- 新的安全群組透過 AWS CloudFormation 進行部署，而未在生產前環境中測試。安全群組使您的客戶無法連線到應用程式。
- 方法已經過修改，但沒有單元測試。軟體部署至生產環境時失敗。

建立此最佳實務的優勢： 軟體部署的變更失敗率降低。軟體品質獲得改善。開發人員對於程式碼的可行性感知能力提高。可以安心推出安全政策，以支援組織的合規性。基礎設施變更 (例如自動化擴展政策更新) 會事先經過測試，以符合流量需求。

未建立此最佳實務時的曝險等級： 高

實作指引

在持續整合的實務過程中，會對所有變更執行測試，從應用程式程式碼到基礎設施都包含在內。會發佈測試結果，讓開發人員迅速獲得反饋。您的組織具有所有變更都必須通過的測試標準。

客戶範例

在其持續整合管道中，AnyCompany Retail 對所有軟體成品執行了數種類型的測試。他們實行了測試驅動的開發，因此所有軟體都有測試單元。在成品建置後，他們執行了端對端測試。這個第一輪測試完成後，他們執行了靜態應用程式安全掃描，以尋找已知漏洞。開發人員在每個測試門檻通過後均收到訊息。所有測試都完成後，軟體成品即儲存在成品儲存庫中。

實作步驟

1. 與組織中的利害關係人合作制定軟體成品的測試標準。所有成品均應通過的標準測試為何？ 是否有必須納入測試涵蓋範圍內的合規或管控要求？ 您是否需要執行程式碼品質測試？ 測試完成時，誰需要得知？
   a. 此 AWS 部署管道參考架構 包含可在整合管道中對軟體成品執行之測試類型的授權清單。
2. 根據您的軟體測試標準，以必要的測試檢測您的應用程式。每一組測試均應在十分鐘內完成。測試應執行為整合管道的一部分。
   a. Amazon CodeGuru Reviewer 可測試您的應用程式程式碼是否有缺陷。
   b. 您可以使用 AWS CodeBuild 對軟體成品執行測試。
   c. AWS CodePipeline 可將您的軟體測試安排到管道中。

資源

相關的最佳實務：

- OPS05-BP01 使用版本控制
- OPS05-BP06 共用設計標準
- OPS05-BP10 完全自動化整合和部署

相關文件：

- 採用測試驅動的開發方法
- 使用 TaskCat 和 CodePipeline 的自動化 AWS CloudFormation 測試管道
- 使用開放原始碼 SCA、SAST 和 DAST 工具建置端對端 AWS DevSecOps CI/CD 管道
- 開始測試無伺服器應用程式
- CI/CD 管道是我的 release captain
- 在 AWS 上實行持續整合和持續交付白皮書

相關影片：

- AWS re:Invent 2020：可測試的基礎設施：對 AWS 的整合測試
- AWS Summit ANZ 2021 - 透過 CDK 和測試驅動的開發施行測試優先策略
- 使用 AWS CDK 測試基礎設施即程式碼

相關資源：

- AWS 部署管道參考架構 - 應用程式
- AWS Kubernetes DevSecOps 管道
- 政策即程式碼研討會 – 測試驅動的開發
- 使用 AWS CodeBuild 為 GitHub 中的 Node.js 應用程式執行單元測試
- 使用 Serverspec 進行基礎設施程式碼的測試驅動開發

相關服務：

- Amazon CodeGuru Reviewer
- AWS CodeBuild
- AWS CodePipeline

OPS05-BP03 使用組態管理系統

使用組態管理系統進行和追蹤組態變更。這些系統可減少由手動程序引起的錯誤，並減少部署變更的工作量。

靜態組態管理會在初始化資源時設定值，這些值預期會在資源的整個生命週期內保持一致。部分範例包括在執行個體上設定 Web 或應用程式伺服器的組態，或定義 AWS 服務的組態 (在 AWS Management Console 內) 或透過 AWS CLI。

動態組態管理會在初始化時設定值，這些值可能或是預期會在資源的整個生命週期內保持一致。例如，您可以設定功能切換，透過組態變更啟動程式碼中的功能，或者在事故期間變更日誌詳細資訊等級以擷取更多資料，然後在事故後改回來，藉此消除目前不需要的日誌及相關費用。

在 AWS 上，您可以使用 AWS Config 跨帳戶和區域持續監控 AWS 資源組態。它可協助您追蹤其組態歷史記錄、了解組態變更如何影響其他資源、以及針對預期或所需的組態進行稽核，方法是使用 AWS Config 規則 和 AWS Config 合規套件。

如果您在 Amazon EC2 執行個體、AWS Lambda、容器、行動應用程式或 IoT 裝置上執行的應用程式中具有動態組態，則可以使用 AWS AppConfig 在您的環境中設定、驗證、部署和監控這些組態。

在 AWS 上，您可以使用 AWS 開發人員工具 例如：AWS CodeCommit、 AWS CodeBuild、 AWS CodePipeline、 AWS CodeDeploy和 AWS CodeStar) 來建置持續整合/持續部署 (CI/CD) 管道。

預期成果： 您會在持續整合、持續交付 (CI/CD) 管道中進行設定、驗證和部署。您會進行監控，以確認組態正確無誤。這會將終端使用者和客戶受到的任何負面影響降到最低。

常見的反模式：

- 您手動更新整個機群的 Web 伺服器組態，但由於更新錯誤，導致多部伺服器無法回應。
- 您在數小時內手動更新應用程式伺服器機群。變更期間的組態不一致會導致未預期的行為。
- 某人已更新您的安全群組，無法再存取您的 Web 伺服器。若不知道進行了哪些變更，您就需要花大量時間來調查問題，復原時間也會跟著拉長。
- 您可以透過 CI/CD 將生產前組態推送到生產環境中，而不需進行驗證。您讓使用者和客戶面臨使用不正確的資料和服務。

建立此最佳實務的優勢： 採用組態管理系統可減少進行和追蹤變更的工作量，以及手動程序造成的錯誤頻率。組態管理系統提供了管控、合規和法規需求方面的保證。

未建立此最佳實務時的曝險等級： 中

實作指引

組態管理系統可用來追蹤和實作應用程式與環境組態的變更。組態管理系統也可用來減少手動程序所造成的錯誤、讓組態變更可重複且可稽核，以及減少工作量。

實作步驟

1. 確定組態擁有者。
   a. 讓組態擁有者得知任何合規、管控或法規需求。
2. 確定組態項目與交付成果。
   a. 組態項目是指受到 CI/CD 管線內部署影響的所有應用程式和環境組態。
   b. 交付成果包括成功條件、驗證及監控對象。
3. 請根據您的業務需求和交付管道選取工具來進行組態管理。
4. 請考慮針對重大組態變更進行加權部署 (例如金絲雀部署)，以盡量減少錯誤組態造成的影響。
5. 將組態管理整合到 CI/CD 管道中。
6. 驗證所有推送的變更。

資源

相關的最佳實務：

- OPS06-BP01 為失敗變更進行規劃
- OPS06-BP02 測試部署
- OPS06-BP03 採用安全的部署策略
- OPS06-BP04 自動化測試和復原

相關文件：

- AWS Control Tower
- AWS 登陸區域加速器
- AWS Config
- 什麼是 AWS Config？
- AWS AppConfig
- 什麼是 AWS CloudFormation？
- AWS 開發人員工具

相關影片：

- AWS re:Invent 2022 - AWS 工作負載的主動管控與合規

- AWS re:Invent 2020：使用 AWS Config 實現合規即程式碼
- 使用 AWS AppConfig 管理和部署應用程式組態

OPS05-BP04 使用建置和部署管理系統

使用建置和部署管理系統。這些系統可減少由手動程序引起的錯誤，並減少部署變更的工作量。

在 AWS 中，您可以使用 AWS 開發人員工具 等服務 (例如，AWS CodeCommit、 AWS CodeBuild、AWS CodePipeline、 AWS CodeDeploy和 AWS CodeStar) 來建置持續整合/持續部署 (CI/CD) 管道。

預期成果： 您的建置和部署管理系統可支援組織的持續整合持續交付 (CI/CD) 系統，提供了使用正確組態自動化安全推展的功能。

常見的反模式：

- 在開發系統中編譯程式碼之後，您將可執行檔複製到生產系統中，卻無法啟動。本機日誌檔案指出其因缺少相依性而失敗。
- 您在開發環境中使用新功能成功建置應用程式，並提供程式碼以進行品質保證 (QA)。它未通過 QA，因為缺少靜態資產。
- 週五，在經過一番努力之後，您成功在開發環境中手動建置應用程式，包括新編碼的功能。到了週一，您卻無法重複成功建置應用程式的步驟。
- 您執行為新版本建立的測試。然後，您會在下週設定測試環境，並執行所有現有的整合測試，接著執行效能測試。新的程式碼具有無法接受的效能影響，必須重新開發及測試。

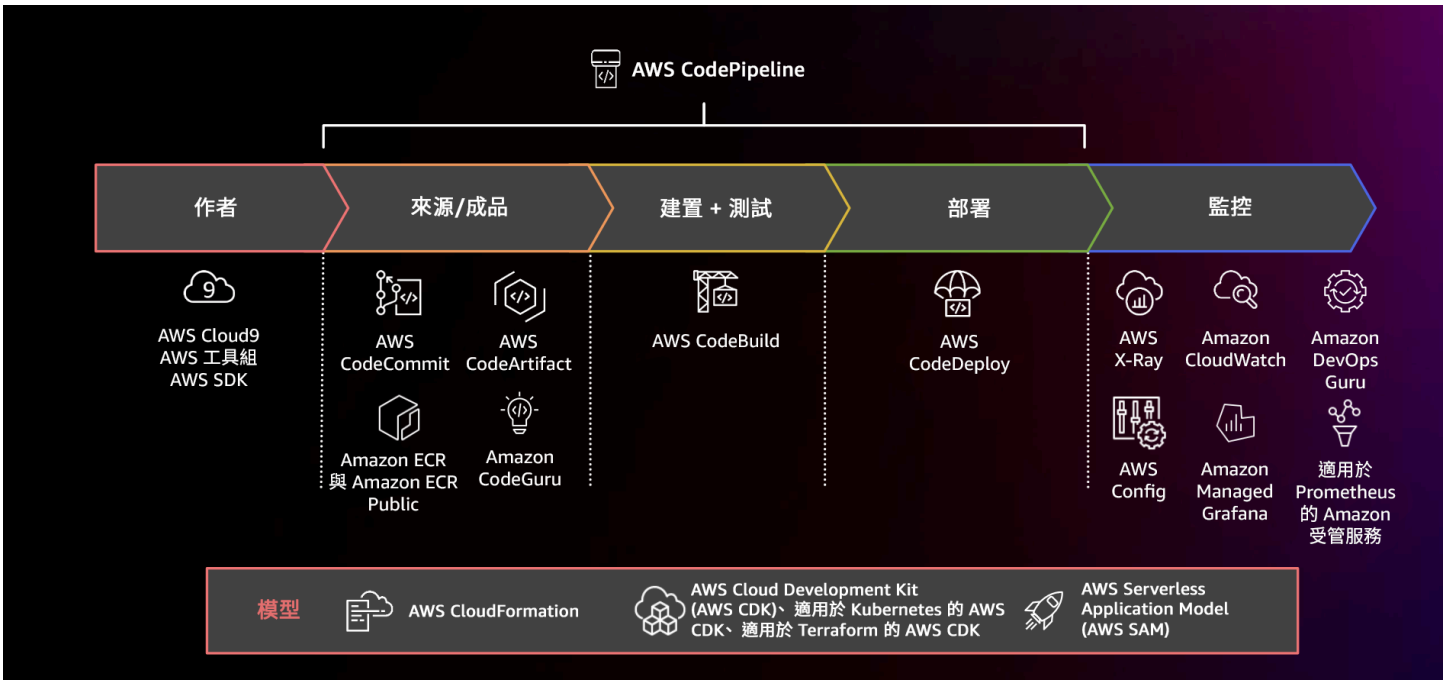
建立此最佳實務的優勢： 透過提供用於管理建置和部署活動的機制，您可以減少執行重複性任務的工作量，讓團隊成員專注於高價值的創意任務，並減少手動程序導致的錯誤。

未建立此最佳實務時的曝險等級： 中

實作指引

建置和部署管理系統可用來追蹤和實作變更、減少手動程序導致的錯誤，以及減少安全部署所需的工作量。從程式碼簽入到建置、測試、部署和驗證，完全自動化整合和部署管道。此舉可縮短前置時間、降低成本、促進增加變更頻率、減少工作量，並且增進協作。

實作步驟



圖中顯示使用 AWS CodePipeline 和相關服務的 CI/CD 管道

1. 使用 AWS CodeCommit 進行版本控制、儲存及管理資產 (例如文件、原始程式碼和二進位檔案)。
2. 使用 CodeBuild 編譯原始程式碼、執行單元測試，以及產生立即可部署的成品。
3. 使用 CodeDeploy 做為部署服務，將應用程式自動部署至 Amazon EC2 執行個體、內部部署執行個體、無伺服器 AWS Lambda 函數或 Amazon ECS。
4. 監控您的部署。

資源

相關的最佳實務：

- OPS06-BP04 自動化測試和復原

相關文件：

- AWS 開發人員工具
- 什麼是 AWS CodeCommit？
- 什麼是 AWS CodeBuild？

- AWS CodeBuild
- 什麼是 AWS CodeDeploy？

相關影片：

- AWS re:Invent 2022 - 適用 AWS 上 DevOps 的 AWS Well-Architected 最佳實務

OPS05-BP05 執行修補程式管理

執行修補程式管理以取得功能、解決問題並保持遵循管控。自動化修補程式管理，以減少由手動程序引起的錯誤、進行擴展，並減少修補工作量。

修補程式和漏洞管理屬於您利益和風險管理活動的一部分。最好擁有不可變的基礎設施，並在已驗證的已知良好狀態下部署工作負載。如果這種方法不可行，剩下的方法就是進行修補。

Amazon EC2 Image Builder 提供更新機器映像的管道。在修補程式管理的過程中，請考慮 Amazon Machine Image (AMI) (使用 AMI 影像管道)， 或容器映像 (使用 Docker 映像管道)，同時 AWS Lambda 會提供模式 讓自訂執行階段和其他程式庫 移除漏洞。

您應使用下列工具管理 Amazon Machine Image for Linux 或 Windows 伺服器映像的更新： Amazon EC2 Image Builder。您可以使用 Amazon Elastic Container Registry (Amazon ECR) 搭配現有的管道來管理 Amazon ECS 映像和管理 Amazon EKS 映像。Lambda 包括 版本管理功能。

若未先在安全環境中進行測試，就不應在生產系統上執行修補程式。只有在修補程式能夠支援營運或業務成果時，才應套用修補程式。在 AWS 上，您可以使用 AWS Systems Manager Patch Manager 自動化受管系統的修補程序，以及使用下列工具來排程活動： Systems Manager 維護時段。

預期成果： 您的 AMI 和容器映像已完成修補、處於最新狀態，並準備好啟動。您可以追蹤所有已部署映像的狀態，並了解修補程式的合規狀況。您可以通報目前狀態，並設立程序來滿足合規需求。

常見的反模式：

- 您必須在兩小時內套用所有新的安全修補程式，結果導致應用程式與修補程式不相容而發生多次停機。
- 未修補的程式庫導致意外後果發生，因為有不明對象利用其中的漏洞來存取您的工作負載。
- 您自動修補開發人員環境，而未通知開發人員。您收到來自開發人員的多次投訴，表示其環境如預期停止運作。
- 您尚未在持續執行的執行個體上修補商用現成軟體。當軟體發生問題而您聯絡廠商時，他們會通知您不支援該版本，您必須修補至特定程度才能獲得協助。

- 您使用的加密軟體近期發佈了修補程式，使效能獲得大幅改善。未修補的系統因未修補仍存在效能問題。
- 收到發生零時差漏洞的通知時，需緊急修正並手動修補所有環境。

建立此最佳實務的優勢： 透過建立修補程式管理程序 (包括修補準則和在各環境中散佈的方法)，您就能擴展和報告修補程度。這樣可保證修補過程安全無虞，並確保能清楚看見已知修正的狀態。如此可促進採用所需的功能、迅速消除問題，並持續遵循管控要求。實作修補程式管理系統和自動化，以減少部署修補程式的工作量，並限制手動程序引起的錯誤。

未建立此最佳實務時的曝險等級： 中

實作指引

修補系統以補救問題，獲得所需的功能，並保持符合管控政策和廠商支援需求。在不可變系統中，部署適當的修補程式集以實現所需的結果。自動化修補程式管理機制，以縮短修補時間、避免手動程序引起的錯誤，並減少修補工作量。

實作步驟

針對 Amazon EC2 Image Builder：

1. 使用 Amazon EC2 Image Builder 指定管道詳細資訊：
    a. 建立映像管道並命名
    b. 定義管道排程和時區
    c. 設定任何相依性
2. 選擇配方：
    a. 選取現有配方或建立新配方
    b. 選取映像類型
    c. 提供配方的名稱和版本
    d. 選取基礎映像
    e. 新增組建元件並新增至目標登錄檔
3. 選用 - 定義您的基礎設施組態。
4. 選用 - 定義組態設定。
5. 檢閱設定。
6. 定期維護配方乾淨度。

針對 Systems Manager Patch Manager：

1. 建立修補基準。
2. 選取路徑操作方法。
3. 啟用合規報告和掃描。

資源

相關的最佳實務：

- OPS06-BP04 自動化測試和復原

相關文件：

- 什麼是 Amazon EC2 Image Builder
- 使用 Amazon EC2 Image Builder 建立映像管道
- 建立容器映像管道
- AWS Systems Manager Patch Manager
- 使用 Patch Manager
- 使用修補程式合規報告
- AWS 開發人員工具

相關影片：

- AWS 上適用於無伺服器應用程式的 CI/CD
- 設計時考量 Ops

相關範例：

- Well-Architected 實驗室 - 庫存和修補程式管理
- AWS Systems Manager Patch Manager 教學課程

OPS05-BP06 共用設計標準

在團隊之間共用最佳實務，以提高認識並最大化開發工作的效益。記載它們並且隨著您的架構演進讓它們保持在最新狀態。如果您的組織中強制執行共用標準，則必須存在用於請求標準新增、變更及例外狀況的機制。如果沒有此選項，標準就會限制創新。

預期成果： 設計標準在貴組織的團隊之間共用。系統會記錄標準並且隨著最佳實務演進保持最新狀態。

常見的反模式：

- 兩個開發團隊各自建立了使用者身分驗證服務。您的使用者必須針對要存取的系統的每一部分，維護一組單獨的憑證。
- 每個團隊管理他們自己的基礎設施。新的合規要求會強制變更您的基礎設施，每個團隊會以不同的方式實作。

建立此最佳實務的優勢： 以共用的標準支援來實踐最佳實務，讓開發工作量發揮最大效益。記錄並且更新設計標準，讓貴組織的最佳實務和安全與合規要求保持在最新狀態。

未建立此最佳實務時的曝險等級： 中

實作指引

在團隊之間共用現有的最佳實務、設計標準、檢查清單、操作程序以及指導和管控要求。對於請求對設計標準進行變更、新增和例外設立程序，以支援改進和創新。讓團隊得知發佈的內容。設立機制讓設計標準隨著最佳實務發展而保持在最新狀態。

客戶範例

AnyCompany Retail 有跨部門架構團隊，該團隊會建立軟體架構模式。這個團隊會建置具有內建合規和管控的架構。採用這些共用標準的團隊會獲得具有內建合規和管控的優點。他們可以快速地在設計標準的基礎上建置。架構團隊每季開會一次，評估架構模式並且視需要更新。

實作步驟

1. 識別擁有開發和更新設計標準的跨部門團隊。這個團隊應與整個組織的利害關係人合作，共同開發設計標準、操作程序、檢查清單、指引和管控需求。記錄設計標準並且在組織內共用。
   a. AWS Service Catalog 可以用來建立套裝服務，代表使用基礎設施即程式碼的設計標準。您可以與所有帳戶共用套裝服務。
2. 設立機制讓設計標準隨著新的最佳實務出現而保持在最新狀態。

3. 如果設計標準是集中強制執行，設立程序來請求變更、更新和豁免。

實作計劃的工作量： 中。開發程序來建立和共用設計標準，即可與整個組織的利害關係人協調和合作。

資源

相關的最佳實務：

- OPS01-BP03 評估管控要求 - 管控需求會影響設計標準。
- OPS01-BP04 評估合規要求 - 合規是建立設計標準中的重要輸入。
- OPS07-BP02 確保對營運準備度進行一致的審查 - 營運準備度檢查清單是在設計您的工作負載時實作設計標準的機制。
- OPS11-BP01 建立持續改進程序 - 更新設計標準是持續改善的一部分。
- OPS11-BP04 執行知識管理 - 在您的知識管理實務中，記錄和共用設計標準。

相關文件：

- 使用 AWS Service Catalog 自動化 AWS Backup
- AWS Service Catalog Account Factory 增強
- Expedia Group 如何使用 AWS Service Catalog 建置資料庫即服務 (DBaaS) 方案
- 維護使用雲端架構模式的可見性
- 簡化在 AWS Organizations 設定中共用您的 AWS Service Catalog 套裝服務

相關影片：

- AWS Service Catalog – 入門
- AWS re:Invent 2020：像專家一樣管理您的 AWS Service Catalog 套裝服務

相關範例：

- AWS Service Catalog 參考架構
- AWS Service Catalog 研討會

相關服務：

- AWS Service Catalog

OPS05-BP07 實作用於提高程式碼品質的實務

實作實務以提高程式碼品質並將缺陷降至最少。部分範例包括測試驅動的開發、程式碼檢閱、標準採用和配對程式設計。將這些實務併入您的持續整合和交付程序。

預期成果： 貴組織使用例如程式碼檢閱或配對程式設計的最佳實務來改善程式碼品質。開發人員和操作人員在軟體開發生命週期過程中採用程式碼品質最佳實務。

常見的反模式：

- 您將程式碼遞交至應用程式的主要分支，而未進行程式碼檢閱。變更會自動部署到生產並且造成中斷。
- 新的應用程式在沒有任何單位、端對端或整合測試的情況下進行開發。無法在部署之前測試應用程式。
- 您的團隊在生產中進行手動變更以解決缺陷。變更不會經過測試或程式碼檢閱，而且不會在持續整合或交付程序中擷取或記錄。

建立此最佳實務的優勢： 透過採用實務來提高程式碼品質，就能協助盡量減少生產環境中引發的問題。使用像是配對程式設計和程式碼檢閱的最佳實務可提高程式碼品質。

未建立此最佳實務時的曝險等級： 中

實作指引

實作實務以提高程式碼品質，在程式碼部署之前將缺陷降至最低。使用像是測試驅動的開發、程式碼檢閱和配對程式設計等實務來提高開發的品質。

客戶範例

AnyCompany Retail 採用數個實務來改善程式碼品質。該公司採用了測試驅動的開發做為撰寫應用程式的標準。對於某些新功能，該公司會讓開發人員在衝刺期間一起進行配對程式設計。每個提取請求都會先經過資深開發人員的程式碼檢閱，然後再整合和部署。

實作步驟

1. 在您的持續整合和交付程序中，採用像是測試驅動的開發、程式碼檢閱和配對程式設計等程式碼品質實務。使用這些技術來改善軟體品質。

a. Amazon CodeGuru Reviewer 可以使用機器學習針對 Java 和 Python 程式碼提供程式設計建議。
b. 您可以使用 AWS Cloud9 建立共用的開發環境，並在其中共同開發程式碼。

實作計劃的工作量： 中。有許多方式可以實作此最佳實務，但是要讓整個組織採用可能會是一項挑戰。

資源

相關的最佳實務：

- OPS05-BP06 共用設計標準 - 您可以在程式碼品質實務中共用設計標準。

相關文件：

- Agile 軟體指南
- CI/CD 管道是我的 release captain
- 使用 Amazon CodeGuru Reviewer 自動進行程式碼檢閱
- 採用測試驅動的開發方法
- DevFactory 如何使用 Amazon CodeGuru 建置更佳的應用程式
- 關於配對程式設計
- RENGA Inc. 使用 Amazon CodeGuru 自動進行程式碼檢閱
- 敏捷開發的藝術：測試驅動的開發
- 程式碼檢閱為何重要 (而且確實可節省時間！)

相關影片：

- AWS re:Invent 2020：使用 Amazon CodeGuru 持續改善程式碼品質
- AWS Summit ANZ 2021 - 透過 CDK 和測試驅動的開發施行測試優先策略

相關服務：

- Amazon CodeGuru Reviewer
- Amazon CodeGuru Profiler
- AWS Cloud9

OPS05-BP08 使用多個環境

使用多個環境來試驗、開發和測試您的工作負載。當環境接近生產環境時提高控制層級，以確保您的工作負載在部署後依預期執行。

預期成果： 您有多個環境可反映您的合規和管控需求。您透過環境測試並推廣程式碼，以逐步實現生產環境。

常見的反模式：

- 您在共享開發環境中進行開發，而另一名開發人員覆寫您的程式碼變更。
- 對共享開發環境的限制性安全控制，讓您無法試驗新服務和功能。
- 您對生產系統執行負載測試，並給使用者造成停機。
- 在生產環境中發生導致資料遺失的嚴重錯誤。在您的生產環境中，您試圖重建導致資料遺失的條件，以便了解此情況如何發生，並防止再次發生。為防止更多資料在測試期間遺失，您必須讓使用者無法使用應用程式。
- 您正在操作多租用戶服務，且無法支援客戶對專用環境的要求。
- 您不一定會進行測試，但要測試時，您會在生產環境中進行。
- 您認為簡單的單一環境會覆寫環境內變更的影響範圍。

建立此最佳實務的優勢： 您可以支援多個同時開發、測試和生產的環境，而不會在開發人員或使用者社群之間產生衝突。

未建立此最佳實務時的曝險等級： 中

實作指引

使用多個環境，並且對開發人員沙盒環境實施最低限度的控制，以協助實驗。提供多個單獨的開發環境，以協助實現並行工作，進而提高開發敏捷性。在環境逐漸達到生產環境的條件時，實施更嚴格的控制，以允許開發人員創新。使用基礎設施即程式碼和組態管理系統來部署所設定控制條件與生產環境一致的環境，以確保系統在部署後依預期執行。當不使用環境時，關閉環境以避免產生與閒置資源相關的成本 (例如，在夜間和週末關閉開發系統)。進行負載測試時，部署與生產環境同等的環境，以改善有效的結果。

資源

相關文件：

- AWS 上的 Instance Scheduler
- 什麼是 AWS CloudFormation？

OPS05-BP09 進行頻繁、細微和可逆的變更

頻繁、細微和可逆的變更會縮小變更的範圍和影響。與變更管理系統、組態管理系統以及建置與交付系統搭配使用時，頻繁、細微和可逆的變更可縮小變更的範圍和影響。透過回復變更，可以更有效地進行疑難排解並加快修復速度。

常見的反模式：

- 您每季部署應用程式的新版本，這表示在這段變更期間，核心服務為關閉狀態。
- 您經常對資料庫結構描述進行變更，但未在您的管理系統中追蹤變更。
- 您執行手動就地更新，並覆寫現有的安裝和組態，但沒有明確的回復計畫。

建立此最佳實務的優勢： 透過經常部署小幅度的變更，加快了開發工作的速度。若變更幅度很小，就更容易了解變更是否會產生意外的後果，也更容易回復。如果變更可逆，由於復原過程較單純，因此實作變更的風險也會降低。變更程序的風險降低，而且變更失敗的影響也會降低。

未建立此最佳實務時的曝險等級： 低

實作指引

透過頻繁、細微和可逆的變更來縮小變更的範圍和影響。這樣可以簡化疑難排解，有助於加速修復，並提供回復變更的選項。另外還可以提高您為企業帶來價值的速度。

資源

相關的最佳實務：

- OPS05-BP03 使用組態管理系統
- OPS05-BP04 使用建置和部署管理系統
- OPS06-BP04 自動化測試和復原

相關文件：

- 實作 AWS 上的微型服務
- 微型服務 - 可觀測性

OPS05-BP10 完全自動化整合和部署

自動化工作負載的建置、部署和測試。此舉可減少由手動程序引起的錯誤，以及部署變更的工作量。

依照一致的標記策略，使用 資源標籤 和 AWS Resource Groups 來套用 中繼資料， 以協助識別您的資源。標記您的資源，以用於組織、成本會計、存取控制，以及將自動執行營運活動設為目標。

預期成果： 開發人員使用工具交付程式碼並推廣至生產環境。開發人員不必登入 AWS Management Console 就可以交付更新。有完整的變更與組態稽核記錄，可滿足管控和合規的需求。程序可在各團隊重複執行並且標準化。開發人員可全心專注於開發和程式碼推送，從而提高生產力。

常見的反模式：

- 週五，您完成了為功能分支編寫新程式碼。週一，執行程式碼品質測試指令碼和每個單位測試指令碼之後，請為下一排程版本檢查程式碼。
- 系統會指派您編寫修正程式碼，以解決影響生產環境中大量客戶的重大問題。測試修正後，您遞交程式碼和電子郵件變更管理內容，以請求核准將其部署到生產環境中。
- 您以開發人員身分登入 AWS Management Console，以使用非標準方法和系統來建立新的開發環境。

建立此最佳實務的優勢： 透過實作自動化建置和部署管理系統，您可以減少手動程序引起的錯誤，以及部署變更的工作量，協助您的團隊成員專注於提供商業價值。您在推廣至生產環境的同時，加快了交付速度。

未建立此最佳實務時的曝險等級： 低

實作指引

您使用建置和部署管理系統來追蹤和實作變更，以減少由手動程序引起的錯誤，並減少工作量。從程式碼簽入到建置、測試、部署和驗證，完全自動化整合和部署管道。此舉可縮短前置時間、促進增加變更頻率、減少工作量、加快上市速度、提高生產力，並且在您推廣到生產環境時提高程式碼的安全性。

資源

相關的最佳實務：

- OPS05-BP03 使用組態管理系統
- OPS05-BP04 使用建置和部署管理系統

相關文件：

- 什麼是 AWS CodeBuild？
- 什麼是 AWS CodeDeploy？

相關影片：

- AWS re\:Invent 2022 - 適用 AWS 上 DevOps 的 AWS Well-Architected 最佳實務

緩解部署風險

採用可快速提供品質意見回饋，並從成果不盡理想的改變中快速復原的方法。使用這些實務可緩解部署變更所帶來問題的影響。

工作負載的設計應包括如何部署、更新及操作。您希望實作符合減少缺陷和快速安全修復的工程實務。

最佳實務

- OPS06-BP01 為失敗變更進行規劃
- OPS06-BP02 測試部署
- OPS06-BP03 採用安全的部署策略
- OPS06-BP04 自動化測試和復原

OPS06-BP01 為失敗變更進行規劃

計劃在部署造成非預期成果時恢復到已知的良好狀態，或者在生產環境中進行修復。擁有制定這類計畫的政策可以協助所有團隊訂立政策，從失敗變更中恢復。一些範例策略包括部署和回復步驟、變更政策、功能旗標、流量隔離和流量轉移。單一版本可能包含多個相關元件變更。策略要能提供您承受或從任何失敗元件變更中恢復的能力。

預期成果： 您已經為失敗變更準備了周延的恢復計畫。此外，您也縮減了發行版本的大小，如此一來，對其他工作負載元件的潛在影響將降到最低。因此，您可以縮短因變更失敗而造成的可能停機時間，並提高回復時間的彈性和效率，進而降低對業務的影響。

常見的反模式：

- 您執行了部署，而您的應用程式變得不穩定，但系統中似乎有作用中使用者。您必須決定是否要復原變更並影響作用中使用者，或在知道使用者無論如何都會受到影響的情況下，等待復原變更。

- 在進行路由變更後，您可以存取新的環境，但其中一個子網路變成無法連線。您必須決定是否要復原所有項目，或嘗試修正無法存取的子網路。當您做出該決定時，子網路仍無法連線。
- 您的系統架構並不允許以較小版本進行更新。因此，在部署失敗期間，您無法回復這些大量變更。
- 您未使用基礎架構即程式碼 (IaC)，並且您以手動方式更新了基礎架構，而造成不理想的組態。您無法有效追蹤和還原手動變更。
- 由於您尚未測量部署的增加頻率，您的團隊不會想降低變更規模和改善每次變更的回復計畫，進而造成更高的風險和失敗率。
- 請不要測量因變更失敗而導致中斷的總持續時間。您的團隊無法排定優先順序並改善其部署程序和回復計畫的效能。

建立此最佳實務的優勢： 若制定失敗變更的回復計畫，可將平均復原時間 (MTTR) 降至最低，並減少業務影響。

未建立此最佳實務時的曝險等級： 高

實作指引

發行團隊採用的一致記錄政策和實務可讓組織規劃發生失敗變更時應採取的動作。該政策應允許在特定情況下向前修正。在任何情況下，向前修正或回復計畫都應該在部署到現場生產之前先經過詳細記錄和測試，以將回復變更所需的時間降到最低。

實作步驟

1. 記錄要求團隊有效計劃在特定期間內還原變更的政策。
    a. 政策應指明允許向前修正的情況。
    b. 要求所有參與者皆能存取記錄完善的回復計畫。
    c. 指定回復需求 (例如：發現部署未經授權的變更時)。
2. 分析工作負載每個元件相關所有變更的影響程度。
    a. 如果可重複的變更遵循強制執行變更政策的一致工作流程，則允許這些變更進行標準化、範本化和預先授權。
    b. 縮小變更規模以減少任何變更的潛在影響，進而降低回復時間和對業務的影響。
    c. 確保回復程序會將程式碼回復到已知的良好狀態，避免可能發生的意外。
3. 整合工具和工作流程，以程式設計方式執行政策。
4. 讓其他工作負載擁有者可以看到變更相關資料，以改善任何無法回復失敗變更的診斷速度。
    a. 使用可見的變更資料來衡量這項實務的成效，並識別出反覆改進方式。

5. 使用監控工具來驗證部署成敗，以加速回復的決策過程。
6. 測量失敗變更期間的中斷時間，以持續修正回復計畫。

實作計劃的工作量： 中

資源

相關的最佳實務：

- OPS06-BP04 自動化測試和復原

相關文件：

- AWS Builders Library | 確保部署期間的回復安全
- AWS 白皮書 | 雲端中的變更管理

相關影片：

- re:Invent 2019 | Amazon 的高可用部署方式

OPS06-BP02 測試部署

使用與生產環境相同的部署組態、安全控制、步驟和程序，在生產前測試發行程序。驗證所有部署的步驟均按照預期完成，例如檢查檔案、組態和服務。透過功能、整合和負載測試以及任何監控 (例如運作狀態檢查) 進一步測試所有變更。透過這些測試，您可以及早發現部署問題，有機會在生產前進行規劃和問題緩解。

您可以建立暫時的平行環境來測試每項變更。使用基礎設施即程式碼 (IaC) 來自動化測試環境的部署，協助減少涉及的工作量，並確保穩定性、一致性和更快的功能交付。

預期成果： 您的組織採用測試驅動的開發文化，其中包含測試部署。如此一來，便能確保團隊專注於交付商業價值，而非管理發行版本。團隊會及早找出部署風險，並訂定適當的緩解方案。

常見的反模式：

- 使用生產版本期間，因為未經測試的部署經常會導致問題，而需要疑難排解或升級處理。
- 您的版本包含更新現有資源的基礎設施即程式碼 (IaC)。您不確定 IaC 是否會成功執行，或對資源造成影響。

- 您為應用程式部署一個新功能。該功能無法按照您的預期運作，且在受影響的使用者回報之前無法預見問題。
- 您更新憑證。您不小心將憑證安裝到錯誤的元件，這些元件未被偵測並因為無法建立與網站的安全連線，而影響了網站訪客。

建立此最佳實務的優勢： 針對部署程序的生產前階段及其帶來的變更進行廣泛測試，將部署步驟對生產的潛在負面影響降到最低。這麼做能增加產品發行期間的信心，並盡可能減少操作支援，同時不影響交付變更的速度。

未建立此最佳實務時的曝險等級： 高

實作指引

測試部署程序與測試部署所產生的變更同樣重要。您可以在生產前環境中測試部署步驟，盡可能準確反映生產環境。諸如不完整或錯誤部署步驟，或者配置錯誤等常見問題都能在生產環境之前偵測。此外，您也可以測試回復步驟。

客戶範例

作為持續整合與持續交付 (CI/CD) 管道的一部分，AnyCompany Retail 在一個類似生產環境中，執行為客戶發行基礎設施和軟體更新所需的定義步驟。流程包含許多預先檢查程序，可以在部署之前偵測到資源偏移 (偵測 IaC 以外所執行的資源變更)，以及驗證 IaC 啟動時所採取的動作。這個程序會驗證部署步驟，例如確認特定檔案和組態已準備就緒，或服務處於執行狀態，並在向負載平衡器重新註冊之前，正確回應本機上的運作狀態檢查。此外，所有變更都標記了許多自動化測試，例如功能、安全性、迴歸、整合和負載測試。

實作步驟

1. 執行安裝前檢查，將生產前環境反映到生產環境。
   a. 使用 偏移偵測 來偵測 AWS CloudFormation 外部資源的變更時間。
   b. 使用 變更集 來確認堆疊變更的目的是否符合啟動變更集時 AWS CloudFormation 所採取的動作。
2. 這會觸發手動核准步驟 (AWS CodePipeline)，以授權生產前環境的部署。
3. 使用部署組態 (例如 AWS CodeDeploy AppSpec 檔案) 來定義部署和驗證步驟。
4. 在適用情況下，會整合 AWS CodeDeploy 和其他 AWS 服務 或 會整合 AWS CodeDeploy 和合作夥伴產品與服務。
5. 監控部署 時會使用 Amazon CloudWatch、AWS CloudTrail,和 Amazon SNS 事件通知。
6. 執行部署後自動化測試，包括功能、安全性、迴歸、整合和負載測試。

7. 故障排除 部署問題。
8. 成功驗證前述步驟後，應該會起始化手動核准工作流程，以授權部署到生產環境。

實作計劃的工作量： 高

資源

相關的最佳實務：

- OPS05-BP02 測試並驗證變更

相關文件：

- AWS 建置者資料中心 | 自動化安全、無人為介入的部署 | 測試部署
- AWS 白皮書 | 在 AWS 上實行持續整合和持續交付
- Apollo 的故事 - Amazon 部署引擎
- 在交付程式碼之前，如何在本機測試和偵錯 AWS CodeDeploy
- 整合網路連線能力測試和基礎設施部署

相關影片：

- re:Invent 2020 | 在 Amazon 測試軟體和系統

相關範例：

- 教學 | 具驗證測試的部署和 Amazon ECS 服務

OPS06-BP03 採用安全的部署策略

安全的生產上市可以控制正面變更的流程，目的是將這些變更對客戶造成的任何負面影響降到最低。安全控制提供檢查機制，以驗證預期結果，並限制變更所帶來的任何負面影響或部署失敗造成的影響範圍。安全上市可能包括諸如功能旗標、一體式、滾動式 (金絲雀版本)、不可變、流量拆分和藍/綠部署等策略。

預期成果： 您的組織使用持續整合與持續交付 CI/CD 系統，提供自動化安全上市功能的能力。團隊必須使用適當的安全上市策略。

常見的反模式：

- 您一次性將失敗的變更部署至所有生產環境。因此，所有客戶會同時受影響。
- 同時部署到所有系統的負面影響必須以緊急版本因應。需要數天時間為所有客戶進行錯誤修正。
- 管理生產發行需要多個團隊的規畫和參與。這會限制您為客戶積極提供更新功能的能力。
- 您透過修改現有系統來執行可變部署。發現變更失敗之後，您必須再次修改系統以還原舊版本，這會延長回復時間。

建立此最佳實務的優勢： 自動化部署持續為客戶在上市速度與交付正面變更之間取得平衡。限制影響範圍可以預防損失慘重的部署失敗，並盡可能提高團隊有效回應故障的能力。

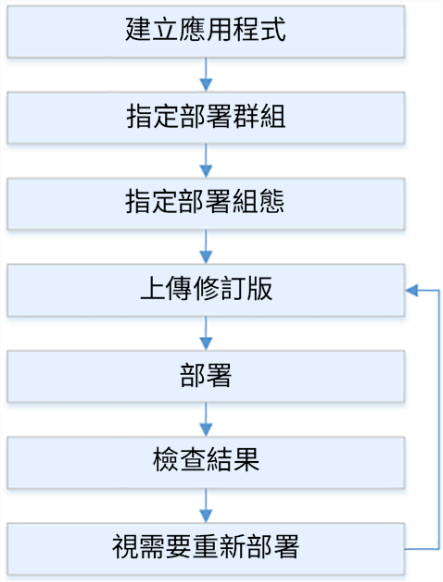
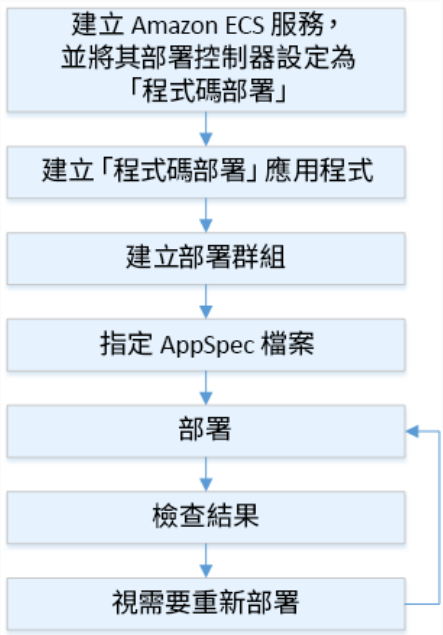
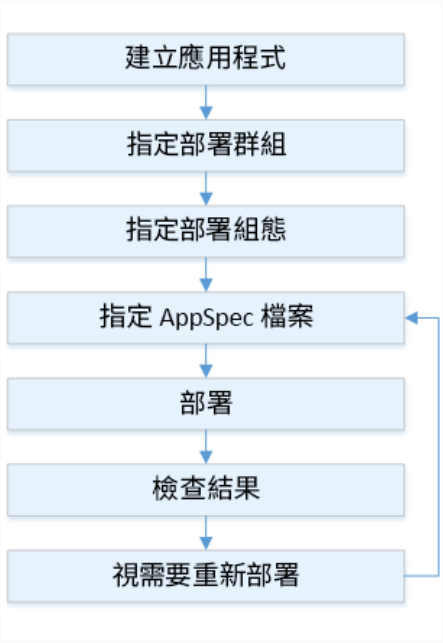
未建立此最佳實務時的曝險等級： 中

實作指引

持續交付的失敗可能導致服務可用性降低和糟糕的客戶體驗。為了盡可能提高部署成功率，請在端對端發行程序中實施安全控制，盡量減少部署錯誤，而目標則是實現零部署失敗。

客戶範例

AnyCompany Retail 的使命是實現最小至零停機時間的部署，這表示部署期間沒有對使用者產生任何可感知的負面影響。為了達成此目標，該企業已建立部署模式 (請參閱下方工作流程圖)，例如滾動部署和藍/綠部署。所有團隊都在其 CI/CD 管道中採用一個或多個模式。

Amazon EC2 的 CodeDeploy 工作流程	Amazon ECS 的 CodeDeploy 工作流程	Lambda 的 CodeDeploy 工作流程
		

實作步驟

1. 使用升級至生產環境的核准工作流程，觸發生產上市步驟的一系列動作。
2. 使用自動化部署系統，例如 AWS CodeDeploy。AWS CodeDeploy 部署選項 包含 EC2/內部部署的就地部署和藍/綠部署、AWS Lambda 以及 Amazon ECS (請參閱下方工作流程表)。
   a. 在適用情況下，整合 AWS CodeDeploy 和其他 AWS 服務 或 整合 AWS CodeDeploy 和合作夥伴產品與服務。
3. 針對資料庫使用藍/綠部署，例如 Amazon Aurora 和 Amazon RDS。
4. 監控部署 時會使用 Amazon CloudWatch、AWS CloudTrail,和 Amazon SNS 事件通知。
5. 執行部署後自動化測試，包括功能、安全性、迴歸、整合和任何負載測試。
6. 故障排除 部署問題。

實作計劃的工作量： 中

資源

相關的最佳實務：

- OPS05-BP02 測試並驗證變更
- OPS05-BP09 進行頻繁、細微和可逆的變更
- OPS05-BP10 完全自動化整合和部署

相關文件：

- AWS 建置者資料中心 | 自動化安全、無人為介入的部署 | 生產部署
- AWS 建置者資料中心 | CI/CD 管道是我的 release captain | 安全、自動化生產版本
- AWS 白皮書 | 在 AWS 上實行持續整合和持續交付 | 部署方式
- AWS CodeDeploy 使用者指南
- 在 AWS CodeDeploy 中使用部署組態
- 設定 API Gateway 金絲雀版本部署
- Amazon ECS 部署類型
- Amazon Aurora 和 Amazon RDS 中的全受管藍/綠部署
- 使用 AWS Elastic Beanstalk 的藍/綠部署

相關影片：

- re:Invent 2020 | 無人為介入：在 Amazon 的持續交付管道
- re:Invent 2019 | Amazon 的高可用部署方式

相關範例：

- 在 AWS CodeDeploy 中嘗試範例藍/綠部署
- Worlshop | 使用 AWS CDK 為 Lambda 金絲雀部署建置 CI/CD 管道
- Workshop | EKS 和 ECS 的藍/綠和金絲雀部署
- Workshop | 建置跨帳戶 CI/CD 管道

OPS06-BP04 自動化測試和復原

為了提高部署程序的速度和可靠性，請在生產前和生產環境中制定自動化測試和回復功能的策略。在部署到生產環境時自動化測試，以模擬人類與系統的互動，驗證部署的變更。自動回復以快速回復到之

前已知的良好狀態。回復應該在預先定義的條件下自動啟動，例如當未達到預期成果或自動化測試失敗時。將這兩項活動的自動化可以提高部署成功率，盡可能縮短回復時間，並減少對業務的潛在影響。

預期成果： 您的自動化測試和回復策略將整合至持續整合與持續交付 (CI/CD) 管道。您的監控能夠根據您的成功條件進行驗證，並在失敗時啟動自動回復。這會將終端使用者和客戶受到的任何負面影響降到最低。例如，當所有測試結果都達到標準時，您可以利用相同的測試案例，將程式碼提升至啟動自動迴歸測試的生產環境。若迴歸測試結果不符預期，則會在管線工作流程中啟動自動回復。

常見的反模式：

- 您的系統架構並不允許以較小版本進行更新。因此，在部署失敗期間，您無法回復這些大量變更。
- 您的部署程序包含一系列手動步驟。將變更部署到工作負載之後，即可開始部署後測試。測試之後，您會發現工作負載無法運作，且客戶中斷連線。然後您開始回復到之前的版本。所有這些手動步驟都會延遲整體系統回復，並對客戶造成長期影響。
- 您花時間為應用程式中不常使用的功能開發自動化測試案例，因而大幅降低了自動化測試功能的投資報酬率。
- 您的版本包含彼此獨立的應用程式、基礎設施、修補程式和組態更新。但是，您有一個 CI/CD 管道可以一次交付所有變更。一個元件故障會強迫您還原所有變更，進而使回復過程變得複雜且效率低下。
- 您的團隊在第一個衝刺階段完成編碼，並開始衝刺兩項工作，但直到第三個衝刺階段，計畫中都不包括測試。最終，自動化測試找出第一個衝刺階段的缺漏，必須在測試第二個衝刺階段前解決，才能啟動交付項目，因此整個版本延遲，進而降低您的自動化測試效率。
- 生產版本的自動迴歸測試案例已經完成，但您並未監控工作負載的運作狀況。由於無法查看是否已重啟服務，您不確定是否需要回復或已啟動回復。

建立此最佳實務的優勢： 自動化測試可以提高測試流程的透明度，以及您在更短時間內顧及更多功能的能力。在生產環境中測試和驗證變更，可以立即識別出問題。改善自動化測試工具的一致性可以更精確地偵測問題。透過自動回復至舊版本，將對客戶的影響降至最低。自動化回復最終可減少業務影響，讓您對部署功能更有信心。整體而言，這些功能可縮短交付時間，同時確保品質。

未建立此最佳實務時的曝險等級： 中

實作指引

自動測試已部署的環境，更快確認是否達到預期成果。當無法達成預先定義的結果時，自動還原到先前的良好狀態，以盡量縮短還原時間，並減少由手動程序引起的錯誤。將測試工具與管道工作流程整合，持續進行測試並減少手動輸入。優先處理自動化測試案例，例如減緩最高風險且需要在每次變更時經常測試的案例。此外，還可以根據測試計畫中預先定義的特定條件進行自動回復。

實作步驟

1. 為您的開發生命週期建立測試生命週期，定義需求規劃到測試案例開發、工具配置、自動化測試和測試案例結案等每個測試程序階段。
   a. 根據您的整體測試策略建立針對特定工作負載的測試方式。
   b. 在整個開發生命週期中，考慮適當的連續測試策略。
2. 根據您的業務需求和管道投資，選擇用於測試和回復的自動化工具。
3. 決定您應該分別自動化和手動執行哪些測試案例。這些內容皆可以根據受測功能的業務價值優先順序來決定。使所有團隊成員隨時接收計畫最新資訊，並確認執行手動測試的權責分配。
   a. 將自動化測試功能應用於對自動化有意義的特定測試案例，例如可重複或經常執行的案例、需要重複作業的案例，或跨多個組態所需的案例。
   b. 在自動化工具中定義測試自動化指令碼和成功條件，如此一來，當特定案例失敗時，可以啟動持續的工作流程自動化。
   c. 定義自動回復的特定失敗條件。
4. 測試案例其中複雜度和人工互動具較高的失敗風險，因此必須排定測試自動化的優先順序，透過詳盡的測試案例開發來產生一致的結果。
5. 將您的自動化測試和回復工具整合到 CI/CD 管道。
   a. 為變更制定明確的成功條件。
   b. 監控觀察以偵測這些條件，並在符合特定回復條件時自動回復變更。
6. 執行不同類型的自動化生產測試，例如：
   a. A/B 測試，以顯示結果比較兩個使用者測試組之間的當前版本。
   b. 金絲雀測試讓您能在將變更發佈給所有使用者之前，先將其發佈給一部分使用者。
   c. 功能旗標測試允許您從應用程序外部標記新版本的功能 (每次僅限一個)，進而使每個新功能皆能逐一進行驗證。
   d. 迴歸測試，以現有的關聯元件驗證新功能。
7. 透過其他應用程式和元件，監控應用程式、交易和互動的操作面向。開發報告，以便按照部銅工作負載顯示變更成功率，讓您得以識別出能夠進一步最佳化的自動化和工作流程部分。
   a. 開發測試結果報告，協助您快速決定是否應該調用回復程序。
   b. 實施策略，允許根據一個或多個測試方法導出的預定義失敗條件進行自動回復。
8. 開發自動化測試用例，以便在未來可重複的變更中重複使用。

實作計劃的工作量：中

資源

相關的最佳實務：

- OPS06-BP01 為失敗變更進行規劃
- OPS06-BP02 測試部署

相關文件：

- AWS Builders Library | 確保部署期間的回復安全
- 使用 AWS CodeDeploy 重新部署和回復部署
- 使用 AWS CloudFormation 自動化部署時的 8 個最佳實務

相關範例：

- 使用 Selenium、AWS Lambda、AWS Fargate (Fargate) 和 AWS 開發人員工具進行無伺服器 UI 測試

相關影片：

- re:Invent 2020 | 無人為介入：在 Amazon 的持續交付管道
- re:Invent 2019 | Amazon 的高可用部署方式

營運準備度和變更管理

評估工作負載、流程、程序及人員的營運準備度，以了解與工作負載相關的營運風險。管理環境的變更流程。

您應使用一致的程序 (包括手動或自動檢查清單) 來獲悉工作負載或變更執行就緒的時間。這樣也有助於尋找需要您制定計畫以解決問題的任何領域。您將擁有可記錄例行活動的執行手冊，以及可指引問題解決程序的程序手冊。使用一種機制來管理變更，此機制支援提供商業價值的並協助降低與變更相關的風險。

最佳實務

- OPS07-BP01 確保人員能力
- OPS07-BP02 確保對營運準備度進行一致的審查

- OPS07-BP03 使用執行手冊執行程序
- OPS07-BP04 使用程序手冊來調查問題
- OPS07-BP05 做出部署系統和變更的明智決策
- OPS07-BP06 啟用生產工作負載的支援計劃

OPS07-BP01 確保人員能力

建立一種機制，用於驗證您有適當數量受過培訓的人員來支援工作負載。他們必須受過組成您的工作負載的平台和服務的培訓。為他們提供操作工作負載所需的知識。您必須擁有足夠受過培訓的人員，才能支援工作負載的一般操作，並且針對會發生的任何事件進行疑難排解。擁有足夠的人員，以便您可以輪替待命和休假的人員，避免倦怠。

預期成果：

- 有足夠受過培訓的人員可以在工作負載可用時支援工作負載。
- 您為人員提供組成您的工作負載的軟體和服務的培訓。

常見的反模式：

- 在沒有受過培訓操作使用中平台和服務的團隊成員之情況下，部署工作負載。
- 沒有足夠的人員可以支援待命輪替或人員休假。

建立此最佳實務的優勢：

- 擁有熟練的團隊成員可有效支援您的工作負載。
- 具有足夠的團隊成員，您可以支援工作負載和待命輪替，同時降低倦怠風險。

未建立此最佳實務時的風險暴露等級：高

實作指引

驗證人員是否已經過充分培訓，可支援工作負載。確認擁有足夠且訓練有素的團隊成員，以妥善應對一般營運活動，包括待命輪替。

客戶範例

AnyCompany Retail 確保支援工作負載的團隊有適當的配備人員且訓練有素。他們有足夠的工程師可以支援待命輪替。人員會獲得工作負載建置基礎的軟體和平台的培訓，並且鼓勵他們考取認證。有足夠的人員讓員工可以休假，同時仍然支援工作負載和待命輪替。

實作步驟

1. 指派適當數量的人員來操作和支援您的工作負載，包括隨時待命。
2. 為您的人員提供組成您的工作負載的軟體和平台的培訓。
   a. AWS 培訓與認證有 AWS 的相關課程庫。它們提供免費和付費課程，線上或面授。
   b. AWS 主持活動和研討會，您可以向 AWS 專家學習。
3. 定期隨著操作條件和工作負載變更，評估團隊大小和技能。調整團隊大小和技能以符合操作要求。

實作計劃的工作量：高。招聘和培訓團隊來支援工作負載需要大量的努力，但是會有重大的長期優點。

資源

相關的最佳實務：

- OPS11-BP04 執行知識管理 - 團隊成員必須擁有操作和支援工作負載所需的資訊。知識管理是提供這項能力的關鍵。

相關文件：

- AWS 活動和研討會
- AWS 培訓與認證

OPS07-BP02 確保對營運準備度進行一致的審查

使用營運準備度審查 (ORR)，來確認您可以運行工作負載。 ORR 是在 Amazon 開發的機制，可確認團隊是否可放心地運行工作負載。ORR 是使用需求檢查清單的審查和檢查程序。ORR 是一種自助服務體驗，團隊會透過此體驗來進行工作負載的認證。ORR 包含的最佳實務皆汲取我們多年來建置軟體所獲得的經驗。

ORR 檢查清單包含架構建議、營運程序、事件管理和發行品質。錯誤糾正 (CoE) 程序是這些項目的主要驅動要素。您專屬的事件後分析應有助於專屬 ORR 的發展。ORR 不只是遵循最佳實務，還能防止先前發生過的事件再發。最後，ORR 中也能夠包含安全性、管控和合規需求。

在工作負載啟動以全面供應前，並在整個軟體開發生命週期執行 ORR。在啟動前執行 ORR 可改善安全運行工作負載的能力。定期針對工作負載重新執行 ORR 可捕捉最佳實務中的任何偏移。您可以為新服務的推出制定 ORR 檢查清單，並為定期審查制定 ORR。此可協助您掌握新出現的最佳實務最新狀態，並採納從事件後分析獲得的經驗。隨著您可以更熟練地使用雲端後，您就可以在架構中建置 ORR 需求作為預設值。

預期成果： 您制定 ORR 檢查清單，內含組織的最佳實務。ORR 會在工作負載啟動前執行。ORR 會在工作負載生命週期的過程中定期執行。

常見的反模式：

- 您啟動工作負載，但不知道自己是否能夠運行工作負載。
- 啟動工作負載的認證中未納入管控和安全性需求。
- 不會定期重新評估工作負載。
- 工作負載啟動，但不需設置必要的程序。
- 您可以在多個工作負載中看到重複出現的相同根本原因失敗。

建立此最佳實務的優勢：

- 工作負載包含架構、程序和管理最佳實務。
- 經驗已納入 ORR 程序中。
- 工作負載啟動時，已設置必要的程序。
- ORR 會在工作負載的整個軟體生命週期執行。

若未建立此最佳實務的風險等級： 高

實作指引

ORR 有兩個部分：程序和檢查清單。貴組織應採用 ORR 程序，並由執行主辦人支援此程序。至少，必須在工作負載啟動以全面供應前執行 ORR。在整個軟體開發生命週期執行 ORR，使其與最佳實務或新需求保持同步。ORR 檢查清單應包含組態項目、安全性和管控需求，以及來自貴組織的最佳實務。在經過一段時間後，您可以使用服務，例如 AWS Config、 AWS Security Hub，和 AWS Control Tower 防護機制，來將 ORR 中的最佳實務建置在防護機制中，以便自動偵測最佳實務。

客戶範例

在發生數個生產事件後，AnyCompany Retail 決定實作 ORR 程序。他們建立了一份檢查清單，其中由最佳實務、管控和合規需求，以及從中斷中汲取的經驗教訓所組成。在工作負載啟動前，新的工作負

載會執行 ORR。每個工作負載每年都會使用一部分的最佳實務來執行 ORR，以便納入在 ORR 檢查清單中新增的最佳實務和需求。經過一段時間後，AnyCompany Retail 使用 AWS Config 來偵測最佳實務，進而縮短 ORR 程序的時間。

實作步驟

若要進一步了解 ORR，請閱讀 「營運準備度審查 (ORR)」白皮書。其中提供詳細的資訊，說明 ORR 程序的歷史、如何建立您專屬的 ORR 實務，以及如何制定 ORR 檢查清單。以下步驟是該文件的精簡版本。如需深入了解 ORR 是什麼，以及如何建立您專屬的 ORR，我們建議閱讀該白皮書。

1. 召集關鍵利害關係人，包含安全性、營運和開發等團隊的代表人員。
2. 請每位利害關係人提供至少一個需求。對於第一次的反覆測試，請嘗試將項目數限制在三十個以下。
   - 附錄 B：來自「營運準備度審查 (ORR)」白皮書的 ORR 問題範例包含您可以開始使用的範例問題。
3. 將需求集中放在試算表中。
   - 您可以使用 在 AWS Well-Architected Tool 中 使用自訂聚焦 來制定 ORR 並在帳戶和 AWS 組織之間進行共用。
4. 找出要在其中執行 ORR 的一個工作負載。啟動前的工作負載或內部工作負載是理想的選擇。
5. 演練 ORR 檢查清單，並記下任何所探索的項目。如果採取緩解措施，那就可能無法進行探索。對於缺少緩解措施的任何探索，請將那些探索新增至項目的待辦清單中，然後在啟動前加以實作。
6. 隨著時間持續在 ORR 檢查清單中新增最佳實務和需求。

使用 Enterprise Support 的 AWS Support 客戶可請求 「營運準備度審查」研討會 (透過其技術客戶經理)。研討會是互動式的 逆向思維 課程，可讓您制定自己的 ORR 檢查清單。

實作計劃的工作量： 高。在組織中採用 ORR 實務需要高層和利害關係人的支持。使用貴組織提供的各方意見，來建立和更新檢查清單。

資源

相關的最佳實務：

- OPS01-BP03 評估管控要求 – ORR 檢查清單原本就很適合用來管控需求。
- OPS01-BP04 評估合規要求 – ORR 檢查清單中有時會包含合規需求。有些時候，它們會是獨立的程序。

- OPS03-BP07 適當地為團隊提供資源 – 團隊能力是 ORR 需求的絕佳候選項。
- OPS06-BP01 為失敗變更進行規劃 – 啟動工作負載前，必須先建立回復或向前回復計劃。
- OPS07-BP01 確保人員能力 – 若要支援工作負載，您必須具備所需的人員。
- SEC01-BP03 識別和驗證控制目標 – 安全性控制目標是絕佳的 ORR 需求。
- REL13-BP01 定義停機和資料遺失的復原目標 – 災難復原計劃是絕佳的 ORR 需求。
- COST02-BP01 根據貴組織的需求制定政策 – 將成本管理政策納入 ORR 檢查清單是很棒的做法。

相關文件：

- AWS Control Tower - AWS Control Tower 中的防護機制
- AWS Well-Architected Tool - 自訂聚焦
- Adrian Hornsby 提供的營運準備度審查範本
- 「營運準備度審查 (ORR)」白皮書

相關影片：

- AWS Support 為您提供支援 | 建立有效的營運準備度審查 (ORR)

相關範例：

- 營運準備度審查 (ORR) 聚焦範例

相關服務：

- AWS Config
- AWS Control Tower
- AWS Security Hub
- 使用自訂聚焦

OPS07-BP03 使用執行手冊執行程序

路由層 執行手冊 是為了實現特定結果而記錄的程序。執行手冊由一系列可供遵循以完成某項工作的步驟組成。早在航空器製造初期，操作過程中就會使用執行手冊。在雲端操作中，我們使用執行手冊來降低風險及達到預期成果。簡言之，執行手冊就是完成一項工作的檢查清單。

執行手冊是工作負載的運作不可或缺的部分。從新團隊成員的上線到部署主要版本，執行手冊無論由誰使用，都是可提供一致結果的編碼程序。執行手冊應在集中發佈，並隨著程序的演進而更新，因為更新執行手冊是變更管理程序的重要環節。其中也應包含關於問題發生時的錯誤處理、工具、許可、例外狀況和呈報的指引。

隨著組織的成熟，您可以開始將執行手冊自動化。請從簡短且常用的執行手冊開始著手。使用指令碼語言自動執行步驟，或使步驟較容易執行。前幾個執行手冊完成自動化後，您會專注於將較複雜的執行手冊自動化。經過一段時間後，您大多數的執行手冊應該都已做了某種程度的自動化。

預期成果： 您的團隊有一系列執行工作負載任務的逐步指南。執行手冊中包含預期成果、必要的工具和許可，以及錯誤處理指示。這些執行手冊會集中存放，並且經常更新。

常見的反模式：

- 憑藉記憶完成程序中的每個步驟。
- 手動部署變更而不使用檢查清單。
- 不同的團隊成員執行相同程序，但使用的步驟不同，或結果不同。
- 執行手冊失去與系統變更和自動化的同步。

建立此最佳實務的優勢：

- 降低手動工作的錯誤率。
- 以一致的方式執行操作：
- 新的團隊成員可更快開始執行工作。
- 可將執行手冊自動化以節省人力。

未建立此最佳實務時的曝險等級： 中

實作指引

根據組織的成熟度，執行手冊採取數種形式。其中至少應包含逐步說明文字文件。預期成果應明確指出。明確記載必要的特殊許可或工具。提供詳細指引，說明在發生狀況時應如何處理錯誤及呈報。列出執行手冊擁有者，並將其集中發佈。執行手冊列入文件後，應請團隊的其他成員加以執行，以進行驗證。隨著程序的演進，請根據您的變更管理程序更新執行手冊。

隨著組織逐漸成熟，您的文字執行手冊應該要自動化。使用諸如 AWS Systems Manager 自動化的服務，您可以將一般文字轉換為可對工作負載執行的自動化。這些自動化可作為事件的應變動作來執行，以降低您維持工作負載的操作負擔。

客戶範例

AnyCompany Retail 必須在軟體部署期間執行資料庫結構描述更新。雲端維運團隊與資料庫管理團隊共同建置用來手動部署這些變更的執行手冊。執行手冊以檢查清單格式列出了程序中的每個步驟。其中包含相關發生狀況時進行錯誤處理的章節。他們將執行手冊發佈於內部 Wiki，與其他執行手冊放在一起。雲端維運團隊規劃要在未來的衝刺期間將執行手冊自動化。

實作步驟

如果您沒有現有的文件儲存庫，版本控制儲存庫將是您開始建置執行手冊程式庫的絕佳選擇。您可以使用 Markdown 來建置執行手冊。我們提供了範例執行手冊範本，讓您用來開始建置執行手冊。

```
# ###### ## ###### | #### ID | ## | ##### | #### | ###### | ###### | ## POC |
 |-------|-------|-------|-------|-------|-------|-------| | RUN001 | ########## #####
## | ## | ## | #### | 2022-09-21 | #### | ## ## 1.### 2.###
```

1. 如果您沒有現有的文件儲存庫或 Wiki，請在您的版本控制系統中建立新的版本控制儲存庫。
2. 識別沒有執行手冊的程序。經常執行、步驟數較少，且失敗的影響程度不高的程序，就是理想的程序。
3. 在您的文件儲存庫中，使用範本建立新的草稿 Markdown 文件。填入 ######## 和必要欄位 (在 ## ########)。
4. 從第一個步驟開始，填入執行手冊的 #### 部分。
5. 將執行手冊提供給團隊成員。讓他們使用執行手冊來驗證步驟。如有任何事項缺漏或需要釐清，請更新執行手冊。
6. 將執行手冊發佈至您的內部文件存放區。發佈後，請告知團隊和其他利害關係人。
7. 一段時間後，您會建置執行手冊程式庫。隨著該程式庫的擴增，您應開始設法將執行手冊自動化。

實作計劃的工作量： 低。執行手冊的最低標準是逐步文字指南。將執行手冊自動化可能會增加實作工作量。

資源

相關的最佳實務：

- OPS02-BP02 已為流程和程序識別擁有者：執行手冊應有擁有者負責加以維護。
- OPS07-BP04 使用程序手冊來調查問題：執行手冊和程序手冊兩者相類似，但有一項顯著差異：執行手冊有預期成果。在許多情況下，當程序手冊識別出根本原因時，就會觸發執行手冊。

- OPS10-BP01 使用程序進行事件、事故和問題管理：執行手冊是良好事件、事故和問題管理實務的一部分。
- OPS10-BP02 每個提醒建立一個程序：執行手冊和程序手冊應用來回應提醒。一段時間後，這些因應動作應該要自動化。
- OPS11-BP04 執行知識管理：維護執行手冊是知識管理的重要環節。

相關文件：

- 使用自動化的執行手冊和程序手冊達成卓越營運
- AWS Systems Manager：使用執行手冊
- AWS 大型遷移的遷移程序手冊 - 任務 4：改進您的遷移執行手冊
- 使用 AWS Systems Manager 自動化執行手冊完成營運任務

相關影片：

- AWS re:Invent 2019：執行手冊、事故報告和事故應變的 DIY 指南 (SEC318-R1)
- 如何使用 AWS | Amazon Web Services 將 IT 營運自動化
- 將指令碼整合到 AWS Systems Manager 中

相關範例：

- AWS Systems Manager：自動化演練
- AWS Systems Manager：從最新的快照執行手冊還原根磁碟區
- 使用 Jupyter 筆記本和 CloudTrail Lake 建置 AWS 事故應變執行手冊
- Gitlab - 執行手冊
- Rubix - 用來在 Jupyter 筆記本中建置執行手冊的 Python 程式庫
- 使用 Document Builder 建立自訂執行手冊
- Well-Architected 實驗室：使用程序手冊和執行手冊將操作自動化

相關服務：

- AWS Systems Manager

OPS07-BP04 使用程序手冊來調查問題

程序手冊是用來調查事件的逐步指南。事件發生時，我們會使用程序手冊來調查、確認影響範圍和找出根本原因。程序手冊可用於各種情境，從部署失敗到安全性事件皆涵蓋在內。在許多案例中，程序手冊可釐清根本原因，而執行手冊則用來緩解該根本原因。程序手冊是組織事件應變計劃的關鍵要素。

優良的程序手冊有幾個重要的特點。它會透過探索的過程來逐步引導使用者。請試著從各種角度思考，我們應遵循哪些步驟來診斷事件？ 透過程序手冊明確定義，在程序手冊中是否需要特殊工具或提高權限。制定溝通計劃，向利害關係人告知調查的最新狀態是關鍵要素。在無法釐清根本原因的狀況下，程序手冊應具備呈報計劃。如果已確定根本原因，程序手冊應指向執行手冊，後者會描述如何解決該根本原因。程序手冊應集中存放並定期維護。如果您使用程序手冊來發出特定警示，請為團隊提供警示中該程序手冊的指標。

隨著組織逐漸成熟，將程序手冊自動化。從涵蓋低風險事件的程序手冊開始。使用指令碼來自動化探索步驟。確保您有配套執行手冊來緩解常見的根本原因。

預期成果： 您的組織具備常見事件的程序手冊。該程序手冊存放在集中的位置，可供團隊成員使用。程序手冊會頻繁更新。對於任何已知的根本原因，都已建立配套執行手冊。

常見的反模式：

- 調查事件並沒有標準的方法。
- 團隊成員依賴肌肉記憶或機構知識，來針對失敗的部署進行疑難排解。
- 新團隊成員會學習如何透過試錯來調查問題。
- 各個團隊間並未共用調查問題的最佳實務。

建立此最佳實務的優勢：

- 程序手冊可為您省下緩解事件所需的心力。
- 不同的團隊成員可以使用相同的程序手冊，以一致的方式找出根本原因。
- 您可以為已知的根本原因制定執行手冊，進而縮短復原時間。
- 程序手冊可協助團隊成員更快開始做出貢獻。
- 團隊可以透過可重複的程序手冊擴展其程序。

若未建立此最佳實務，暴露的風險等級： 中

實作指引

您如何根據組織的成熟度來建立和使用程序手冊。如果您剛接觸雲端，請在中央文件儲存庫中建立文字形式的程序手冊。隨著組織逐漸成熟，您就可以透過 Python 之類的指令碼語言將程序手冊半自動化。您可以在 Jupyter 筆記本中執行這些指令碼來加快探索速度。先進的組織具有全自動化的程序手冊，這些手冊適用於透過執行手冊自動修復的常見問題。

透過列出在您工作負載中發生的常見事件，來開始建立程序手冊。為低風險以及根本原因的範圍已縮減至幾個問題的事件選擇程序手冊，然後開始。在您為較簡單情境建立程序手冊後，請接著嘗試風險較高或尚未確定根本原因的情境。

隨著組織逐漸成熟，應將您的文字程序手冊自動化。使用諸如 AWS Systems Manager Automations 的服務，可以將一般文字轉換為自動化。您可以針對工作負載執行這些自動化來加快調查速度。您可以啟動這些自動化來回應事件、縮短事件探索和解決的平均時間。

客戶可使用 AWS Systems Manager Incident Manager 來回應事件。此服務提供單一介面，來分類事件、在探索和緩解期間通知利害關係人，並在整個事件期間進行合作。其使用 AWS Systems Manager Automations 來加快偵測和復原速度。

客戶範例

生產事件會影響 AnyCompany Retail。待命的工程師使用程序手冊來調查問題。隨著透過步驟取得進展時，該工程師會確保程序手冊中識別的重要利害關係人都能了解最新進展。他發現根本原因是後端服務中的一項競賽條件。該工程師使用執行手冊，重新啟動服務，使 AnyCompany Retail 重新上線。

實作步驟

如果您沒有現有的文件儲存庫，我們建議為程序手冊程式庫建立版本控制儲存庫。您可以使用 Markdown 建立程序手冊，Markdown 與多數程序手冊自動化系統都相容。如果您是從頭開始建立，請使用以下範例程序手冊範本。

```
# Playbook Title ## Playbook Info | Playbook ID | Description
 | Tools Used | Special Permissions | Playbook Author | Last
 Updated | Escalation POC | Stakeholders | Communication Plan |
 |-------|-------|-------|-------|-------|-------|-------|-------|-------| | RUN001
 | What is this playbook for? What incident is it used for? | Tools | Permissions |
 Your Name | 2022-09-21 | Escalation Name | Stakeholder Name | How will updates be
 communicated during the investigation? | ## Steps 1.Step one 2.Step two
```

1. 如果您沒有現有的文件儲存庫或 Wiki，請在版本控制系統中為程序手冊建立新的版本控制儲存庫。
2. 找出需要調查的常見問題。應存在根本原因僅限於幾個問題的情境，解決方案的風險很低。

3. 使用 Markdown 範本，填寫 ###### 區段以及程序手冊資訊 ####。
4. 填寫疑難排解步驟。盡可能清楚說明要執行哪些動作或應調查哪些地方。
5. 將程序手冊提供給團隊成員，讓成員透過該手冊來進行驗證。如果缺少任何資訊或內容不清楚，請更新程序手冊。
6. 在文件儲存庫中發佈程序手冊，並通知團隊和任何利害關係人。
7. 此程序手冊程式庫會隨著您新增更多程序手冊而成長。在您有數本程序手冊後，請開始使用 AWS Systems Manager Automations 之類的工具來進行自動化，進而確保自動化和程序手冊都能保持同步。

實作計劃的工作量： 低。程序手冊應為集中存放的文字文件。越來越多發展成熟的組織會開始自動化程序手冊。

資源

相關的最佳實務：

- OPS02-BP02 已為流程和程序識別擁有者：程序手冊應有擁有者，擁有者會負責維護這類手冊。
- OPS07-BP03 使用執行手冊執行程序：執行手冊和程序手冊兩者相類似，但有一項顯著差異：執行手冊有預期成果。在許多情況下，當程序手冊找出根本原因時，就會使用執行手冊。
- OPS10-BP01 使用程序進行事件、事故和問題管理：程序手冊是正常事件、事故和問題管理實務的一部分。
- OPS10-BP02 每個提醒建立一個程序：執行手冊和程序手冊應用來回應警示。一段時間後，應將這些因應措施自動化。
- OPS11-BP04 執行知識管理：維護程序手冊是知識管理的重要環節。

相關文件：

- 使用自動化的執行手冊和程序手冊達成卓越營運
- AWS Systems Manager：使用執行手冊
- 使用 AWS Systems Manager Automation 執行手冊解決營運任務

相關影片：

- AWS re:Invent 2019：執行手冊、事件報告和事件應變的 DIY 指南 (SEC318-R1)
- AWS Systems Manager Incident Manager - AWS 虛擬研討會

- 將指令碼整合到 AWS Systems Manager 中

相關範例：

- AWS 客戶程序手冊架構
- AWS Systems Manager：自動化演練
- 使用 Jupyter 筆記本和 CloudTrail Lake 建置 AWS 事件應變執行手冊
- Rubix - 用來在 Jupyter 筆記本中建置執行手冊的 Python 程式庫
- 使用 Document Builder 建立自訂執行手冊
- Well-Architected 實驗室：使用程序手冊和執行手冊將操作自動化
- Well-Architected 實驗室：使用 Jupyter 事件應變程序手冊

相關服務：

- AWS Systems Manager Automation
- AWS Systems Manager Incident Manager

OPS07-BP05 做出部署系統和變更的明智決策

為成功和失敗變更工作負載建立程序。事前剖析是一種演練，團隊可藉此模擬失敗，開發緩解策略。使用事前剖析可預測失敗並適時建立程序。評估將變更部署到您的工作負載的優點和風險。確認所有變更都符合管控。

預期成果：

- 您在將變更部署到您的工作負載時做出明智決策。
- 變更符合管控。

常見的反模式：

- 將變更部署到我們的工作負載，而沒有處理失敗部署的程序。
- 對不符合管控要求的生產環境進行變更。
- 部署新版本的工作負載，而未建立資源使用率的基準。

建立此最佳實務的優勢：

- 您對工作負載的失敗變更已做好準備。
- 變更您的工作負載符合管控政策。

未建立此最佳實務時的風險暴露等級：低

實作指引

使用事前剖析來開發失敗變更的程序。記載失敗變更的程序。確定所有變更都符合管控。評估將變更部署到您的工作負載的優點和風險。

客戶範例

AnyCompany Retail 定期執行事前剖析來驗證他們失敗變更的程序。他們在共用 Wiki 中記載程序並且頻繁更新。所有變更都符合管控要求。

實作步驟

1. 在將變更部署到您的工作負載時做出明智決策。建立及檢閱成功部署的準則。開發會觸發變更回復的情境或準則。權衡部署變更的優點與失敗變更的風險。
2. 確認所有變更都符合管控政策。
3. 使用事前剖析為失敗變更進行規劃並且記載緩解策略。執行桌上模擬演練來建立失敗變更的模型，並且驗證回復程序。

實作計劃的工作量：中。實作事前剖析的實務需要貴組織利害關係人的協調和努力

資源

相關的最佳實務：

- OPS01-BP03 評估管控要求 - 管控要求是判斷是否部署變更的關鍵因素。
- OPS06-BP01 為失敗變更進行規劃 - 建立計劃來緩解失敗的部署並且使用事前剖析來驗證它們。
- OPS06-BP02 測試部署 - 每個軟體變更都應該在部署之前先適當的進行測試，以便在生產中減少缺陷。
- OPS07-BP01 確保人員能力 - 擁有支援工作負載的足夠受過培訓的人員，對於為部署系統變更做出明智決策相當重要。

相關文件：

- Amazon Web Services：風險與合規
- AWS 共同責任模式
- AWS 雲端 中的管控：敏捷和安全之間的正確平衡。

OPS07-BP06 啟用生產工作負載的支援計劃

針對您的生產工作負載所依賴的任何軟體和服務啟用支援。根據您的生產服務層級需求選取適當的支援等級。必須要有這些相依性的支援計劃，以備發生服務中斷或軟體問題時使用。記錄支援計劃，以及如何要求所有服務和軟體供應商的支援。實作相關機制以確認支援的聯絡窗口是最新的。

預期成果：

- 為生產工作負載所依賴的軟體和服務實作支援計劃。
- 根據服務層級需求選擇適當的支援計劃。
- 記錄支援計劃、支援等級，以及如何要求支援。

常見的反模式：

- 您沒有主要軟體供應商的支援計劃。您的工作負載因此受到影響，且您無法加速進行修正，或及時獲得供應商提供的更新。
- 擔任軟體供應商主要聯絡窗口的開發人員已離開公司。您無法直接聯繫供應商支援人員。您必須花時間研究及瀏覽通用聯絡系統，因此必要時的回應將更為耗時。
- 軟體供應商發生生產中斷。目前沒有文件說明如何提出支援案例。

建立此最佳實務的優勢：

- 透過適當的支援等級，您將可在必要的時間範圍內獲得回應以滿足服務層級需求。
- 受支援的客戶可在遇到生產問題時加以呈報。
- 軟體和服務供應商可在事件發生期間協助進行疑難排解。

未建立此最佳實務時的風險暴露等級：低

實作指引

針對您的生產工作負載所依賴的任何軟體和服務供應商啟用支援計劃。設定適當的支援計劃以滿足服務層級需求。對 AWS 客戶而言，這意味著在任何有生產工作負載的帳戶上啟用 AWS Business Support

(或更高等級)。定期與支援供應商聯繫，取得關於支援優惠、程序和聯絡人的更新。記錄如何要求軟體和服務供應商的支援，包括如何在中斷發生時加以呈報。實作相關機制以保有最新的支援聯絡資料。

客戶範例

在 AnyCompany Retail，所有商業軟體和服務相依性都有支援計劃。例如，他們在所有具有生產工作負載的帳戶上啟用了 AWS Enterprise Support。任何開發人員都可在問題發生時提出支援案例。有 Wiki 頁面提供了相關資訊說明如何要求支援、應通知誰，以及加速處理案例的最佳實務為何。

實作步驟

1. 與組織中的利害關係人合作，識別您的工作負載所依賴的軟體和服務供應商。記錄這些相依性。
2. 確認工作負載的服務層級需求。選取相對應的支援計劃。
3. 針對商業軟體和服務，與供應商共同建立支援計劃。
    a. 為所有生產帳戶訂閱 AWS Business Support 或更高等級可獲得 AWS Support 更快的回應時間，極力建議這麼做。如果您沒有付費支援，則必須有處理問題的行動計劃，而這需要 AWS Support 的協助。AWS Support 提供了多種工具和技術、人員和方案，旨在主動協助您優化效能、降低成本和加快創新速度。AWS Business Support 提供了額外權益，包括能夠存取 AWS Trusted Advisor 和 AWS Personal Health Dashboard，以及更快速的回應時間。
4. 在您的知識管理工具中記錄支援計劃。納入如何要求支援、在提出支援案例時應通知誰，以及在事件發生時如何加以呈報等資訊。任何人在得知支援程序或聯絡資料有所變更時，都可以利用 Wiki 這項機制對文件進行必要的更新。

實作計劃的工作量：低。大部分的軟體和服務供應商都提供選擇加入支援計劃。在您的知識管理系統上記錄並分享支援最佳實務，可確保您的團隊知道在生產問題發生時應如何因應。

資源

相關的最佳實務：

- OPS02-BP02 已為流程和程序識別擁有者

相關文件：

- AWS Support Plans

相關服務：

- AWS Business Support
- AWS Enterprise Support

營運

成功是業務成果的實現，並根據您定義的指標來衡量。透過了解工作負載和營運運作狀態，您可以確定組織和業務成果何時可能處於風險狀態或目前正處於風險狀態，並適當地做出回應。

為取得成功，必須能夠：

主題

利用工作負載可觀測性

利用可觀測性確保最佳的工作負載運作狀況。利用相關指標、日誌和追蹤，全面掌握工作負載效能並有效解決問題。

可觀測性讓您能夠專注於有意義的資料，並了解工作負載的互動和結果。透過專注於基本洞見並消除不必要的資料，您就能持續使用簡單直接的方式來了解工作負載效能。

重點不只是收集資料，還要正確解譯資料。定義清楚的基準、設定適當的警示閾值，並主動監控任何偏差情況。一旦關鍵指標稍有變化，尤其是與其他資料相關時，就能精確指出特定問題所在。

有了可觀測性，您就具備更優異的預測能力，並且能應付潛在的挑戰，進而確保工作負載順利運行並滿足業務需求。

AWS 提供特定的工具，如 Amazon CloudWatch ，可用於監控和記錄，以及 AWS X-Ray ，可用於分散式追蹤。這些服務可與各種不同的 AWS 資源輕鬆整合，進而有效率地收集資料、根據預先定義的閾值設定警示，以及在儀表板上呈現資料以便進行解讀。利用這些洞見就能讓您做出明智的資料驅動決策，以滿足您的營運目標。

最佳實務

- OPS08-BP04 建立可付諸行動的警示
- OPS08-BP05 建立儀表板

OPS08-BP01 分析工作負載指標

實作應用程式遙測之後，請定期分析收集到的指標。雖然延遲、請求、錯誤和容量 (或配額) 可提供深入了解系統效能的洞見，但務必將檢閱業務成果指標視為優先事項。這樣做可確保您所做的資料驅動決策符合您的業務目標。

預期成果： 獲得深入工作負載效能的精確洞見，有助於做出資料驅動的決策，確保與業務目標保持一致。

常見的反模式：

- 單獨分析指標，未能考慮到其對業務目標的影響。
- 過度依賴技術指標，而輕忽業務指標。
- 未能時常檢閱指標，而錯失即時決策的機會。

建立此最佳實務的優勢：

- 增進對於技術表現與業務成果之間相互關聯的了解。
- 透過即時資料改善了決策過程。
- 主動識別並緩解問題，不讓問題影響業務成果。

未建立此最佳實務時的曝險等級： 中

實作指引

利用像是 Amazon CloudWatch 等工具進行指標分析。AWS 服務 (如 AWS Cost Anomaly Detection 和 Amazon DevOps Guru) 可用來偵測異常狀況，特別是在靜態閾值未知，或行為模式更適合異常偵測的情況下。

實作步驟

1. 分析與檢閱： 定期檢閱和解讀您的工作負載指標。
   a. 將業務成果指標視為優先於純粹技術指標的事項。
   b. 了解資料中峰值、下降或模式的重要性。

2. 利用 Amazon CloudWatch： 使用 Amazon CloudWatch 集中檢視並進行深入分析。
   a. 設定 CloudWatch 儀表板以視覺化您的指標，並長時間進行比較。
   b. 在 CloudWatch 中使用百分位數 以清楚了解指標的分佈情形，這有助於定義 SLA 和了解極端值。
   c. 設定 AWS Cost Anomaly Detection 以識別不尋常的模式，而不依賴靜態閾值。
   d. 實作 CloudWatch 跨帳戶可觀測性 以監控跨區域內多個帳戶的應用程式並進行疑難排解。
   e. 使用 CloudWatch Metric Insights 查詢和分析跨帳戶和區域的指標資料，以找出趨勢和異常狀況。
   f. 套用 CloudWatch Metric Math 來轉換、彙總或對您的指標執行計算，以獲得更深入的洞見。
3. 採用 Amazon DevOps Guru： 納入 Amazon DevOps Guru 以利用其機器學習強化的異常偵測功能，識別無伺服器應用程式操作問題的早期跡象，並矯正問題以免影響客戶。
4. 根據洞見最佳化： 根據您的指標分析做出明智的決策，以調整和改善您的工作負載。

實作計劃的工作量： 中

資源

相關的最佳實務：

- OPS04-BP01 識別關鍵績效指標
- OPS04-BP02 實作應用程式遙測

相關文件：

- The Wheel 部落格 - 強調持續檢閱指標的重要性
- 百分位數很重要
- 使用 AWS Cost Anomaly Detection
- CloudWatch 跨帳戶可觀測性
- 使用 CloudWatch Metrics Insights 查詢您的指標

相關影片：

- 在 Amazon CloudWatch 中啟用跨帳戶可觀測性
- Amazon DevOps Guru 簡介

- 使用 AWS Cost Anomaly Detection 持續分析指標

相關範例：

- One Observability 研討會
- 使用 Amazon DevOps Guru 獲得 AIOps 的運作洞見

OPS08-BP02 分析工作負載日誌

定期分析工作負載日誌相當重要，藉此能夠深入了解應用程式的各個操作層面。藉由有效率地篩選、視覺化和解讀日誌資料，可持續最佳化應用程式效能和安全。

預期成果： 從徹底的日誌分析中獲得深入應用程式行為和操作的豐富洞見，以確保主動偵測和緩解問題。

常見的反模式：

- 忽略日誌分析，直到出現嚴重問題。
- 未使用一套完整的工具進行日誌分析，而錯過了關鍵的洞見。
- 只倚賴手動檢閱日誌，而未利用自動化和查詢功能。

建立此最佳實務的優勢：

- 主動找出操作瓶頸、安全威脅及其他潛在問題。
- 有效利用日誌資料，以持續最佳化應用程式。
- 加強對應用程式行為的理解，幫助偵錯和疑難排解。

未建立此最佳實務時的曝險等級： 中

實作指引

Amazon CloudWatch Logs 是強大的日誌分析工具。像是 CloudWatch Logs Insights 和 Contributor Insights 這類整合式功能，可提供簡單直接且有效率的方式從日誌中產生有意義的資訊。

實作步驟

1. 設定 CloudWatch Logs： 設定應用程式和服務以將日誌傳送至 CloudWatch Logs。

2. 設定 CloudWatch Logs Insights： 使用 CloudWatch Logs Insights 進行互動式搜尋和分析日誌資料。

   a. 製作查詢以找出模式、視覺化日誌資料，並產生可付諸行動的洞見。

3. 利用 Contributor Insights 使用 CloudWatch Contributor Insights 識別高基數維度 (例如 IP 地址或使用者客服人員) 中最活躍的發言者。
4. 實作 CloudWatch Logs 指標篩選器： 設定 CloudWatch 日誌指標篩選器 將日誌資料轉換成可付諸行動的指標。如此您就能設定警報或進一步分析模式。
5. 定期檢閱和改進： 定期檢閱您的日誌分析策略，以擷取所有相關資訊並持續最佳化應用程式效能。

實作計劃的工作量： 中。

資源

相關的最佳實務：

- OPS04-BP01 識別關鍵績效指標
- OPS04-BP02 實作應用程式遙測
- OPS08-BP01 分析工作負載指標

相關文件：

- 使用 CloudWatch Logs Insights 分析日誌資料
- 使用 CloudWatch Contributor Insights
- 建立和管理 CloudWatch Logs 日誌指標篩選器

相關影片：

- 使用 CloudWatch Logs Insights 分析日誌資料
- 使用 CloudWatch Contributor Insights 分析高基數資料

相關範例：

- CloudWatch Logs 範例查詢
- One Observability 研討會

OPS08-BP03 分析工作負載追蹤

對於獲得應用程式操作之旅全面性的總覽來說，分析追蹤資料是相當重要的一環。透過視覺化和了解各種不同元件之間的互動，就能微調效能、找出瓶頸，並且增強使用者體驗。

預期成果： 清楚掌握應用程式的分散式操作，就能更快解決問題並增強使用者體驗。

常見的反模式：

- 忽略追蹤資料，只依賴日誌和指標。
- 未將追蹤資料與相關的日誌建立關聯。
- 忽略從追蹤產生的指標，如延遲和故障率。

建立此最佳實務的優勢：

- 改善疑難排解並縮短平均解決時間 (MTTR)。
- 獲得深入相依性及其影響的洞見。
- 快速找出並糾正效能問題。
- 利用追蹤產生的指標做出明智的決策。
- 透過最佳化元件互動改善使用者體驗。

未建立此最佳實務時的曝險等級： 中

實作指引

AWS X-Ray 提供了全方位的追蹤資料分析套件，能讓您深入了解服務互動的各個層面、監控使用者活動，以及偵測效能問題。像是 ServiceLens、X-Ray Insights、X-Ray Analytics 及 Amazon DevOps Guru 等功能可從追蹤資料產生更深入且可付諸行動的洞見。

實作步驟

下列步驟提供了結構化的方法，以使用 AWS 服務有效實作追蹤資料分析：

1. 整合 AWS X-Ray： 確保 X-Ray 與您的應用程式整合以擷取追蹤資料。
2. 分析 X-Ray 指標： 深入研究從 X-Ray 追蹤產生的指標，例如延遲、請求率、故障率和回應時間分佈情形，方法是使用 服務圖 來監控應用程式運作狀況。

3. 使用 ServiceLens： 利用 ServiceLens Map 加強服務和應用程式的可觀測性。如此就能將追蹤、指標、日誌、警報和其他運作狀況資訊整合在一起檢視。
4. 啟用 X-Ray Insights：
   a. 開啟 X-Ray Insights 以便自動偵測追蹤內的異常情況。
   b. 檢查洞見以找出明確的模式並確定根本原因，例如故障率或延遲增加。
   c. 請參考 Insights 時間軸，依時間順序查看所偵測到問題的分析。
5. 使用 X-Ray Analytics： X-Ray Analytics 可讓您徹底探索追蹤資料、找出明確的模式，以及擷取洞見。
6. 使用 X-Ray 中的群組： 在 X-Ray 中建立群組，即可根據如高延遲等條件篩選追蹤，以進行更針對性的分析。
7. 納入 Amazon DevOps Guru： 參與 Amazon DevOps Guru 以便利用機器學習模型的優勢，從追蹤中找出明確的操作異常狀況。
8. 使用 CloudWatch Synthetics： 使用 CloudWatch Synthetics 建立 Canary 來持續監控您的端點和工作流程。這些 Canary 可與 X-Ray 整合，以提供追蹤資料，用來對要測試的應用程式進行深入分析。
9. 使用實際使用者監控 (RUM)： 透過 AWS X-Ray 和 CloudWatch RUM，您可以分析請求路徑並進行偵測，從應用程式的最終使用者開始，一路到下游 AWS 受管服務。這樣做有助於找出影響使用者的延遲趨勢和錯誤。
10. 與日誌建立關聯： 將追蹤資料與 X-Ray 追蹤檢視內的相關日誌建立關聯，以透過更深入的觀點來了解應用程式行為。如此可讓您檢視與追蹤的交易直接相關的日誌事件。

實作計劃的工作量： 中。

資源

相關的最佳實務：

- OPS08-BP01 分析工作負載指標
- OPS08-BP02 分析工作負載日誌

相關文件：

- 使用 ServiceLens 監控應用程式運作狀況
- 使用 X-Ray Analytics 探索追蹤資料

- 使用 X-Ray Insights 偵測追蹤中的異常狀況
- 使用 CloudWatch Synthetics 持續監控

相關影片：

- 使用 Amazon CloudWatch Synthetics 和 AWS X-Ray 分析應用程式並進行偵錯
- 使用 AWS X-Ray Insights

相關範例：

- One Observability 研討會
- 使用 AWS Lambda 實作 X-Ray
- CloudWatch Synthetics Canary 範本

OPS08-BP04 建立可付諸行動的警示

及時偵測並回應應用程式行為偏差的情況，是相當重要的一環。尤其重要的是，能夠辨識以關鍵績效指標 (KPI) 為基礎的成果何時存在風險，或何時出現非預期的異常狀況。以 KPI 做為警示的基礎，可確保您收到的訊號與業務或營運影響直接相關。這種可付諸行動的警示可推動主動回應，且有助於維持系統效能和可靠性。

預期成果： 接收及時、相關且可付諸行動的警示，以便迅速找出並緩解潛在問題，尤其是 KPI 成果存在風險時。

常見的反模式：

- 設定太多非嚴重警示，導致警示疲勞。
- 未根據 KPI 排定警示的優先順序，因此難以了解問題對業務造成的影響。
- 忽略解決根本原因，導致一再出現相同問題的警示。

建立此最佳實務的優勢：

- 專注於可付諸行動且相關的警示，以減少警示疲勞的情況。
- 透過主動偵測和緩解問題，改善系統運作時間和可靠性。
- 透過整合熱門的警示和通訊工具，強化團隊協作並加快問題解決速度。

未建立此最佳實務時的曝險等級： 高

實作指引

若要建立有效的警示機制，則務必使用指標、日誌和追蹤資料，因為這些資料會在 KPI 為基礎的成果存在風險或偵測到異常時發出訊號。

實作步驟

1. 確定關鍵績效指標 (KPI)： 識別應用程式的 KPI。警示應與這些 KPI 密切相關，才能準確反映業務影響。
2. 實作異常偵測：
   - 使用 AWS Cost Anomaly Detection： 設定 AWS Cost Anomaly Detection 以自動偵測不尋常的模式，確保真正發生異常狀況時會產生警示。
   - 使用 X-Ray Insights：
     a. 設定 X-Ray Insights 以偵測追蹤資料中的異常情況。
     b. 設定 X-Ray Insights 的通知， 以便在偵測到問題時收到警示。
   - 與 DevOps Guru 整合：
     a. 利用 Amazon DevOps Guru 的機器學習功能來偵測現有資料中的操作異常狀況。
     b. 瀏覽至 通知設定 (DevOps Guru 中) 以設定異常警示。
3. 實作可付諸行動的警示： 設計警示，以提供足夠資訊來立即採取行動。
4. 減少警示疲勞： 盡量減少非嚴重警示。產生大量不重要的警示會使團隊疲於奔命，導致疏忽嚴重的問題，而降低警示機制的整體效用。
5. 設定複合警報： 使用 Amazon CloudWatch 複合警報 來合併多個警報。
6. 整合警示工具： 合併各種工具，如 Ops Genie 和 PagerDuty。
7. 參與 AWS Chatbot 整合 AWS Chatbot以將警示轉送至 Chime、Microsoft Teams 和 Slack。
8. 以日誌為基礎的警示： 使用 日誌指標篩選器 (CloudWatch 中)，以根據特定日誌事件建立警報。
9. 檢閱和反覆執行： 定期重新檢視和改進警示組態。

實作計劃的工作量： 中。

資源

相關的最佳實務：

- OPS04-BP01 識別關鍵績效指標
- OPS04-BP02 實作應用程式遙測
- OPS04-BP03 實作使用者體驗遙測
- OPS04-BP04 實作相依性遙測
- OPS04-BP05 實作分散式追蹤
- OPS08-BP01 分析工作負載指標
- OPS08-BP02 分析工作負載日誌
- OPS08-BP03 分析工作負載追蹤

相關文件：

- 使用 Amazon CloudWatch 警報
- 建立複合警報
- 根據異常偵測建立 CloudWatch 警報
- DevOps Guru 通知
- X-Ray Insights 通知
- 透過互動式 ChatOps 監控和操作您的 AWS 資源並進行疑難排解
- Amazon CloudWatch 整合指南 | PagerDuty
- 將 OpsGenie 與 Amazon CloudWatch 整合

相關影片：

- 在 Amazon CloudWatch 中建立複合警報
- AWS Chatbot 概觀
- AWS on Air ft.AWS Chatbot 中的變異命令

相關範例：

- 雲端中使用 Amazon CloudWatch 的警報、事件管理和矯正功能
- 教學課程：建立 Amazon EventBridge 規則將通知傳送至 AWS Chatbot
- One Observability 研討會

OPS08-BP05 建立儀表板

儀表板提供人性化的檢視方式，讓您深入了解工作負載的遙測資料。雖然儀表板是重要的視覺介面，但不應取代警示機制，而是相輔相成。經過精心打造的儀表板不僅能提供快速了解系統運作狀況和效能的洞見，還能對利害關係人呈現有關業務成果和問題影響層面的即時資訊。

預期成果： 使用視覺呈現的方式，提供清楚、深入系統與業務運作狀況且可付諸行動的洞見。

常見的反模式：

- 包含太多指標、過於複雜的儀表板。
- 依賴儀表板，卻沒有異常偵測警示。
- 儀表板未隨著工作負載發展而更新。

建立此最佳實務的優勢：

- 立即掌握關鍵系統指標和 KPI。
- 強化利害關係人的溝通與理解。
- 快速深入洞察操作問題的影響層面。

未建立此最佳實務時的曝險等級： 中

實作指引

以業務為中心的儀表板

專為業務 KPI 量身打造的儀表板，可與更廣泛的利害關係人進行互動。儘管這些人可能對系統指標不感興趣，但他們會急於了解這些數字對業務的影響。以業務為中心的儀表板可確保所有受監控且經過分析的技術和操作指標，都與總體業務目標保持同步。這種一致性確保每個人清楚了解目標，且對於重要性有共同的認知。此外，強調業務 KPI 的儀表板往往更能付諸行動。利害關係人能夠迅速了解營運狀況、需要關注的環節，以及可能對業務成果造成的影響。

了解這點之後，在建立儀表板時，請務必在技術指標與業務 KPI 之間取得平衡。兩者都至關重要，但要滿足的對象不同。在理想情況下，您應有能夠提供全方位視角儀表板，以便深入掌握系統運作狀況與效能，同時也要強調關鍵業務成果及其影響。

Amazon CloudWatch 儀表板是 CloudWatch 主控台中可自訂的首頁，可用來在單一檢視中監控您的資源，甚至能監控分散到不同 AWS 區域 和帳戶中的資源。

實作步驟

1. 建立基本儀表板： 在 CloudWatch 中建立新儀表板，為它提供描述性的名稱。
2. 使用 Markdown 小工具： 在深入研究指標之前，使用 Markdown 小工具 在儀表板頂端新增文字內容。此內容應說明儀表板涵蓋的內容、所呈現指標的重要性，還可以包含其他儀表板和疑難排解工具的連結。
3. 建立儀表板變數： 納入儀表板變數 以適時提供動態且彈性的儀表板檢視。
4. 建立指標小工具： 新增指標小工具 以便將應用程式產生的各種不同指標視覺化，並調整這些小工具以便有效呈現系統運作狀況和業務成果。
5. Log Insights 查詢： 利用 CloudWatch Logs Insights 從日誌中產生可付諸行動的指標，並且在儀表板上顯示這些洞見。
6. 設定警報： 將 CloudWatch 警報 整合到儀表板中，以便快速查看違反其閾值的任何指標。
7. 使用 Contributor Insights： 納入 CloudWatch Contributor Insights 以分析高基數欄位，並且更清楚了解資源的首要參與者。
8. 設計自訂小工具： 對於未能透過標準小工具滿足的特定需求，可考慮建立 自訂小工具。這些小工具可從各種資料來源中提取資料，或以獨特的方式呈現資料。
9. 反覆執行並改進： 隨著應用程式發展，請定期重新檢視您的儀表板，以確保其相關性。

資源

相關的最佳實務：

- OPS04-BP01 識別關鍵績效指標
- OPS08-BP01 分析工作負載指標
- OPS08-BP02 分析工作負載日誌
- OPS08-BP03 分析工作負載追蹤
- OPS08-BP04 建立可付諸行動的警示

相關文件：

- 建置用於檢視營運狀況的儀表板
- 使用 Amazon CloudWatch 儀表板

相關影片：

- 建立跨帳戶和跨區域 CloudWatch 儀表板
- AWS re:Invent 2021 - 透過 AWS 雲端 營運儀表板獲得企業能見度

相關範例：

- One Observability 研討會
- 使用 Amazon CloudWatch 進行應用程式監控

了解運作狀態

定義、擷取和分析營運指標，掌握營運團隊的活動，以便採取適當行動。

您的組織應該要能輕鬆了解營運狀況。您會希望訂出營運團隊的業務目標，確定反映這些目標的關鍵績效指標，然後使用指標並根據營運成果制定指標，以獲得有用的洞見。您應使用這些指標來實作涵蓋業務和技術觀點的儀表板和報告，進而協助主管和利害關係人做出明智的決策。

AWS 可輕鬆彙集和分析您的營運日誌，如此一來，您便能產生指標、了解自己的營運狀態並從長期的營運中獲得洞見。

最佳實務

- OPS09-BP01 使用指標衡量營運目標與 KPI
- OPS09-BP02 傳達狀態和趨勢以確實掌控營運狀況
- OPS09-BP03 檢閱營運指標並優先改進

OPS09-BP01 使用指標衡量營運目標與 KPI

從您的組織取得定義營運成功的目標和 KPI，並決定反映這些目標的指標。設定基準做為參考點，並定期重新評估。制定機制以便從團隊收集這些指標以進行評估。

預期成果：

- 已發佈並共用組織運營團隊的目標和 KPI。
- 已建立反映這些 KPI 的指標。範例包括：
  - 票證佇列深度或票證平均存留時間
  - 依問題類型分組的票證計數
  - 處理問題所花的時間，無論是否有標準作業程序 (SOP)

- 從失敗的程式碼推送復原所花的時間長度
- 通話量

常見的反模式：

- 錯過部署期限，因為開發人員須分心處理疑難排解工作。開發團隊要求更多人力，但無法提出確切需要的人力數量，因為無法衡量被佔用的時間。
- 設立了 1 級服務台來處理使用者通話。經過一段時間後，加入了更多工作負載，但並沒有分配更多人員給 1 級服務台。客戶滿意度受到通話時間增加及問題未解決的時間拉長影響而下降，但管理層看不到這些現象的指標，未能採取任何行動。
- 有問題的工作負載已交由另一個營運團隊進行維護。與其他工作負載不同的是，並未針對這個新工作負載提供適當的文件和執行手冊。因此，團隊花費更長的時間進行疑難排解和解決失敗情況。然而，沒有任何指標記載此情況，因此無法明確究責。

建立此最佳實務的優勢： 只要工作負載監控顯示我們應用程式和服務的狀態，監控營運團隊就可讓擁有者深入了解這些工作負載取用者之間的變化，例如業務需求轉變。藉由建立能夠反映營運狀態的指標來衡量這些團隊的效用，並依據業務目標進行評估。指標可突顯支援問題，或識別何時發生偏離服務層級目標的情形。

未建立此最佳實務時的曝險等級： 中

實作指引

安排時間與企業領導者和利害關係人一起確定服務的整體目標。確定各個不同營運團隊應負責的任務，以及能夠應對哪些挑戰。使用這些來集思廣益，找出能夠反映這些營運目標的關鍵績效指標 (KPI)。這些可能包括客戶滿意度、從形成功能概念到部署的時間、平均問題解決時間及其他方面。

從 KPI 中找出最能反映這些目標的資料指標和來源。客戶滿意度可能由各種不同的指標組合而成，例如通話等待或回應時間、滿意度分數，以及提出的問題類型。部署時間可能是測試和部署，加上任何需要新增的部署後修正所需時間的總和。顯示不同類型的問題所花費時間 (或是這些問題的計數) 的統計資料，可提供一個切入視角，以了解需要針對性處理的地方。

資源

相關文件：

- Amazon QuickSight - 使用 KPI

- Amazon CloudWatch - 使用指標
- 建置儀表板
- 如何使用 KPI 儀表板追蹤成本最佳化 KPI

OPS09-BP02 傳達狀態和趨勢以確實掌控營運狀況

您須了解營運狀態及趨勢方向，以確定成果何時可能存在風險、是否可支援新增的工作，或是變更對您的團隊造成的影響。營運事件發生時，有提供使用者和營運團隊參考資訊的狀態頁面，就能減輕溝通管道的壓力，並有效傳播資訊。

預期成果：

- 主管對團隊處理的通話量類型和正在進行的工作 (例如部署)可以一目瞭然。
- 發生影響擴及正常營運的情況時，利害關係人和使用者社群就會收到警示。
- 組織領導階層和利害關係人可查看狀態頁面以便回應警示或影響，並且獲得有關營運事件的資訊，例如聯絡窗口、票證資訊及預估的復原時間。
- 領導階層和其他利害關係人會收到報告，報告中會顯示營運統計資料，例如某一段時間內的通話量、使用者滿意度分數、待處理票證數量及其存留時間。

常見的反模式：

- 工作負載停擺，造成服務無法使用。通話量暴增，因為使用者要求得知發生什麼情況。主管也要求得知誰在處理問題，因而增加了通話量。不同的營運團隊重複投入嘗試調查的工作。
- 由於需要新功能，因而轉派數名人員進行工程工作。但未回補空缺，使得解決問題的時間大幅拉長。領導階層並未獲得這些資訊，而是在經過數週且收到使用者不滿意的意見回饋後才察覺此問題。

建立此最佳實務的優勢： 在業務受到影響的營運事件中，各個團隊可能會浪費大量時間和精力來查詢資訊，以試圖了解情況。透過建立廣泛傳播的狀態頁面和儀表板，利害關係人就能迅速獲得資訊，例如是否偵測到問題、誰負責處理問題，或是預計何時恢復正常營運。如此一來，團隊成員就不需花太多時間與其他人溝通狀態，因而有更多時間解決問題。

未建立此最佳實務時的曝險等級： 中

實作指引

建置儀表板，以顯示營運團隊目前的關鍵指標，並且讓營運主管和管理層隨時可存取這些資訊。

建置可快速更新的狀態頁面，以顯示事故或事件何時發生、負責人是誰，以及誰負責協調回應。在此頁面上分享使用者應考量的任何步驟或因應措施，並廣泛傳播位置。鼓勵使用者遇到未知的問題時，先查看此位置。

收集並提供報告，以顯示長時間的營運狀況，並將此資訊傳達給主管和決策者，以說明運營工作及挑戰和需求。

在團隊之間共用這些最能反映目標和 KPI 的指標和報告，以及這些資訊在推動變革方面的影響力。花時間進行這些活動，以在團隊內部和團隊之間提高營運的重要性。

資源

相關文件：

- 測量進度
- 建置用於檢視營運狀況的儀表板

相關解決方案：

- 資料操作

OPS09-BP03 檢閱營運指標並優先改進

預留檢閱營運狀態的專屬時間和資源，以確保依舊優先處理日常業務線所需的服務。召集營運主管和利害關係人定期檢閱指標、重新確認或修改各項目標，並優先改進。

預期成果：

- 營運主管和員工定期開會，以檢閱一段特定報告期間的指標。說明挑戰、一同慶祝成就，並分享學到的經驗。
- 利害關係人與企業領導者會定期收到營運狀態的簡報，並徵求有關目標、KPI 和未來計畫的意見。討論服務交付、營運和維護之間的權衡，並納入相關環境中。

常見的反模式：

- 新產品已推出，但 1 級和 2 級營運團隊未接受足夠的培訓來提供支援，或未配置額外的人員。領導者未看見指出支援單解決次數減少且事故量增加的指標。訂閱數量隨著不滿的使用者離開平台而開始減少，但數週後才採取行動。

- 長久以來一直採用手動程序來執行工作負載維護工作。雖然渴望自動化，但由於系統的重要性較低，因此優先順序較低。然而經過一段時間後，系統的重要性已提高，而現在這些手動程序佔用了大多數營運時間。未安排資源來提供更多營運工具，導致員工隨著工作負載增加而倦怠。等到有員工離職並加入其他競爭對手，領導階層才察覺到此情況。

建立此最佳實務的優勢： 在某些組織中，將相同的時間和注意力分配給服務交付和新產品或方案可能會是一項挑戰。發生這種情況時，業務線可能會因為預期的服務層級逐漸惡化而受到影響。這是因為營運未隨著業務成長而改變和發展，並且可能很快就會落後。假如未定期檢閱營運收集的洞見，那麼察覺到業務風險時，便可能為時已晚。透過分配時間與營運員工和領導階層一起檢閱指標和程序，就能持續掌握營運所扮演的重要角色，並且能夠在風險達到嚴重等級之前發現。營運團隊能夠更深入洞察即將發生的業務變化與計畫，進而採取積極的行動。領導階層對於營運指標的掌握程度，展現了這些團隊在內部和外部的客戶滿意度方面所扮演的角色，並讓他們在各種選擇當中權衡出更適當的優先順序，或確保營運團隊有時間和資源能夠隨著新的業務和工作負載計畫做出改變與發展。

未建立此最佳實務時的曝險等級： 中

實作指引

花時間與利害關係人和營運團隊一起檢閱營運指標，並檢閱報告資料。將這些報告與組織的目標相互比對，以確定是否符合這些目標。找出目標不明確，或者要求與付出之間存在衝突的模糊地帶來源。

找出時間、人員和工具能夠協助實現營運成果的地方。確定哪些 KPI 會受到影響，以及哪些應是成功的目標。定期重新檢視，以確保營運資源充足，可支援業務線。

資源

相關文件：

- Amazon Athena
- Amazon CloudWatch 指標和維度參考
- Amazon QuickSight
- AWS Glue
- AWS Glue Data Catalog
- 使用 Amazon CloudWatch Agent 從 Amazon EC2 執行個體和內部部署伺服器收集指標和日誌
- 使用 Amazon CloudWatch 指標

回應事件

您應可以預測營運事件，不論是計劃 (如銷售促銷、部署和故障測試) 或非計劃 (如使用率突增和元件故障) 中的事件。回應提醒時，應使用現有的執行手冊和程序手冊以實現一致的結果。定義的提醒應由負責回應和向上呈報的角色或團隊擁有。您還將希望了解系統元件的業務影響，並在需要時使用它來確定工作目標。您應在事件發生後執行根本原因分析 (RCA)，然後防止再次發生失敗或文件因應措施。

AWS 提供的工具可以程式碼支援您工作負載及營運的各個方面，進而簡化您的事件回應。這些工具可讓您編寫營運事件回應的指令碼，並進行初始化以回應監控資料。

在 AWS 中，您可透過將失敗的元件取代為已知良好的版本來縮短復原時間，而不是嘗試進行修復。然後，您可以對失敗的頻外資源執行分析。

最佳實務

- OPS10-BP01 使用程序進行事件、事故和問題管理
- OPS10-BP02 每個提醒建立一個程序
- OPS10-BP03 根據業務影響確定營運事件的優先順序
- OPS10-BP04 定義向上呈報路徑
- OPS10-BP05 定義因應中斷的客戶通訊計劃
- OPS10-BP06 透過儀表板傳達狀態
- OPS10-BP07 自動回應事件

OPS10-BP01 使用程序進行事件、事故和問題管理

您的組織具有處理事件、事故和問題的程序。事件 是發生於工作負載、但可能無需由您介入的事項。事故 是需要介入的事件。 問題 是重複發生而需要介入或無法解決的事件。您需要相關程序來減輕這些事件對業務的影響，並確保您能夠適當因應。

當工作負載發生事故和問題時，您需要有相關程序來加以處理。您如何讓利害關係人得知事件的狀態？應變由誰監控？ 您使用哪些工具來減輕事件的影響? 在此舉例說明一些您為了獲得可靠的應變程序而有待解答的問題。

程序必須集中記載，並且提供給涉及工作負載的每個人使用。如果您沒有集中的 Wiki 或文件存放區，可以使用版本控制儲存庫。您將隨著程序的演進而保有最新計劃。

問題是可以自動化的。這些事件佔據的時間會影響到您的創新能力。請開始建置可重複的程序，以減輕問題。經過一段時間後，您將著重於緩解措施的自動化或修正基礎問題。如此您即有時間投入於工作負載的改進。

預期成果： 您的組織具有處理事件、事故和問題的程序。這些程序會集中記載並存放。這些文件會隨著程序的變更而更新。

常見的反模式：

- 週末發生了事故，而值班工程師不知該如何處理。
- 客戶傳送電子郵件給您，指出應用程式已關閉。您將伺服器重新開機，試著修正問題。此狀況頻繁地發生。
- 有一項事故讓多個團隊各自獨立試著加以解決。
- 您的工作負載中發生了部署，但並未記錄。

建立此最佳實務的優勢：

- 您的工作負載中有事件的稽核軌跡。
- 您的事故中復原的時間減少了。
- 團隊成員可用一致的方式解決事故和問題。
- 調查事故的人力會更加整合。

未建立此最佳實務時的曝險等級： 高

實作指引

實作此最佳實務，意味著您會追蹤工作負載事件。您具有處理事故和問題的程序。這些程序會經常記載、共用及更新。問題經識別後會定出優先順序，然後獲得修正。

客戶範例

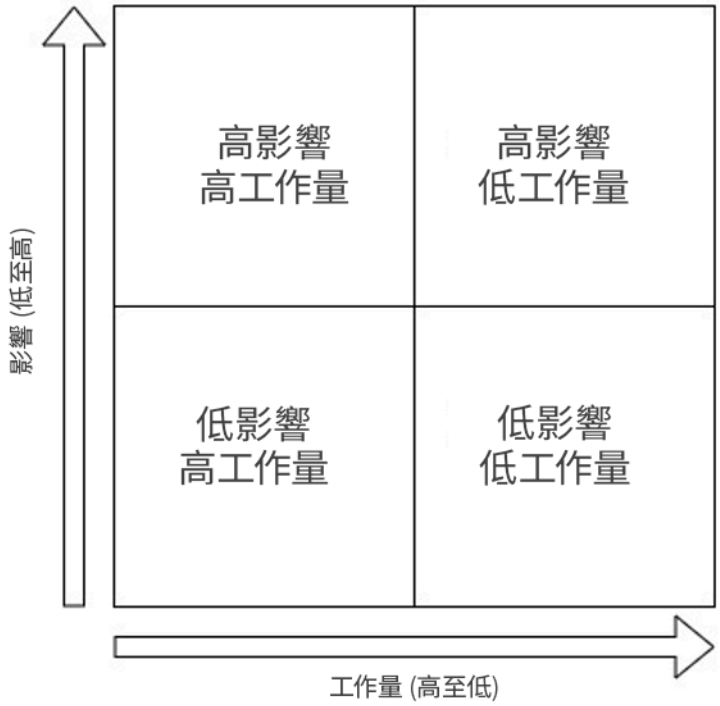
AnyCompany Retail 有某部分的內部 Wiki 專門用來處理事件、事故和問題管理。所有事件都會傳送至 Amazon EventBridge。問題會在 AWS Systems Manager OpsCenter 中識別為 OpsItems，並定出修正的優先順序，以減少無特殊專長人力。程序變更後，會隨即在其內部 Wiki 中更新。他們使用 AWS Systems Manager Incident Manager 來管理事故及協調緩解工作。

實作步驟

1. 事件
    - 追蹤發生在工作負載中的事件，即使無需人為介入亦然。
    - 與工作負載利害關係人共同擬定應追蹤的事件清單。範例包括已完成的部署或成功的修補。

- 您可以使用諸如 Amazon EventBridge 或者 Amazon Simple Notification Service 等服務來產生要追蹤的自訂事件。

2. 事故
   - 首先請定義事故的溝通計劃。哪些利害關係人必須獲得通知？ 您如何維繫其參與度? 協調工作由誰監控？ 我們建議建立內部交談管道，以利溝通和協調。
   - 為支援工作負載的團隊定義呈報路徑，尤其是團隊未設置當班輪值時。根據您的支援等級，您也可以向 AWS Support 申請立案。
   - 建立用來調查事件的程序手冊。其中應包含溝通計劃和詳細的調查步驟。在您的調查中納入對 AWS Health Dashboard 的檢查。
   - 記載您的事故應變計劃。傳達事故管理計劃，讓內部與外部客戶都了解互動的規則及其應有的預期。對您的團隊成員進行其使用訓練。
   - 客戶可使用 Incident Manager 來設定及管理其事故應變計劃。
   - Enterprise Support 客戶可以要求 事件管理研討會 (透過其技術客戶經理)。這個指導研討會將測試您現有的事故應變計劃，並協助您識別改善的領域。
3. 問題
   - 問題必須在 ITSM 系統中受到識別及追蹤。
   - 識別所有已知問題，並按照修正的工作量以及對工作負載的影響定出優先順序。



   - 先解決高影響、低工作量的問題。這些問題解決後，再接著解決位於低影響、低工作量象限的問題。

- 您可以使用 Systems Manager OpsCenter 來識別這些問題、將執行手冊連結至問題，並加以追蹤。

實作計劃的工作量： 中。必須同時具備程序和工具，才能實作此最佳實務。記載您的程序，並且讓工作負載的任何相關人員都可加以使用。經常加以更新。您具有管理問題和加以緩解或修正的程序。

資源

相關的最佳實務：

- OPS07-BP03 使用執行手冊執行程序：已知問題需要相關聯的執行手冊，讓緩解工作保有一致性。
- OPS07-BP04 使用程序手冊來調查問題：事件需使用程序手冊來調查。
- OPS11-BP02 執行事故後分析：從事故復原後務必要執行事後檢討。

相關文件：

- Atlassian - DevOps 時代的事故管理
- AWS 安全事故應變指南
- DevOps 和 SRE 時代的事故管理
- PagerDuty - 什麼是事故管理？

相關影片：

- AWS re:Invent 2020：分散式組織中的事故管理
- AWS re:Invent 2021 - 使用事件驅動架構建置新一代的應用程式
- AWS 為您提供支援 | 探索事故管理桌上模擬演練
- AWS Systems Manager Incident Manager - AWS 虛擬研討會
- AWS 下一步 ft. Incident Manager | AWS 事件

相關範例：

- AWS 管理與管控工具研討會 - OpsCenter
- AWS 主動服務 – 事故管理研討會
- 使用 Amazon EventBridge 建置事件驅動應用程式
- 在 AWS 上建置事件驅動架構

相關服務：

- Amazon EventBridge
- Amazon SNS
- AWS Health Dashboard
- AWS Systems Manager Incident Manager
- AWS Systems Manager OpsCenter

OPS10-BP02 每個提醒建立一個程序

對於引發提醒的任何事件，建立明確定義的回應 (執行手冊或程序手冊)，並指明。此舉可確保對營運事件的有效而迅速的回應，並防止需採取動作的事件被無價值的通知所淹沒。

常用的反模式：

- 您的監控系統會將核准的連線串流以及其他訊息一起提供給您。訊息數量如此龐大，以至於您錯過需要您介入的定期錯誤訊息。
- 您收到提醒，指出網站運作中斷。發生這種情況時沒有已定義的程序。您必須採取臨機操作方法來診斷和解決問題。隨需開發此程序會延長復原時間。

建立此最佳實務的優勢： 只有在需要採取動作時才發出提醒，可防止低值提醒隱藏高值提醒。透過讓每個可採取動作的提醒都具有一個程序，您可針對環境中的事件實現一致且迅速的回應。

若未建立此最佳實務，暴露的風險等級為： 高

實作指引

- 每個提醒建立一個程序：對於引發提醒的任何事件，都應建立明確定義的回應 (執行手冊或程序手冊)，並指明負責人 (例如，個人、團隊或角色) 來對成功完成的程序負責。回應的執行可以自動化，也可以由另一個團隊完成，但負責人要對確保流程交付預期結果負責。透過建立這些程序，您可以確保對營運事件做出迅速有效的回應，並防止需採取行動的事件被無價值的通知所淹沒。例如，自動調整規模功能可能應用於調整 Web 前端規模，但營運團隊可能需負責確保自動調整規模規則和限制符合工作負載需求。

資源

相關文件：

- Amazon CloudWatch 功能
- 什麼是 Amazon CloudWatch Events？

相關影片：

- 制定監控計劃

OPS10-BP03 根據業務影響確定營運事件的優先順序

確保在有多個事件需要介入時，首先解決對業務最重要的事件。影響可能包括人員傷亡、經濟損失或聲譽或信用受損。

常用的反模式：

- 您收到為使用者新增印表機組態的支援請求。處理此問題時，您收到支援請求，而其指出您的零售網站運作中斷。為使用者完成印表機組態後，您便開始處理網站問題。
- 您收到零售網站和薪資系統運作中斷的通知。您不知道應該優先處理哪一個。

建立此最佳實務的優勢： 將對業務影響最大的事件回應排定優先順序，讓您能夠管理該影響。

若未建立此最佳實務，暴露的風險等級為： 中

實作指引

- 根據業務影響，排定操作事件的優先順序：確保在有多個事件需要介入時，首先解決對業務最重要的事件。影響可能包括人員傷亡、經濟損失、違反法規或聲譽或信用受損。

OPS10-BP04 定義向上呈報路徑

在您的執行手冊和程序手冊中定義向上呈報路徑，包括觸發向上呈報的條件以及向上呈報的程序。明確確定每個動作的擁有者，以確保對營運事件做出迅速有效的回應。

在採取行動之前，確定何時需要人為決策。與決策者合作，事先做出該決策，並預先核准行動，如此就不會延長 MTTR 等待回應的時間。

常用的反模式：

- 您的零售網站已運作中斷。您不了解用於恢復網站的執行手冊。您開始打電話給同事，希望有人能夠幫助您。

- 您收到應用程式無法連線的支援案例。您沒有管理系統的許可。您不知道誰有這個許可。您嘗試聯絡開立此案例的系統擁有者，但沒有回應。您沒有此系統的聯絡人，而且您的同事對此不熟悉。

建立此最佳實務的優勢： 透過定義向上呈報、向上呈報觸發條件和向上呈報程序，您可以針對影響以適當的速率將資源系統性地新增到事件中。

若未建立此最佳實務，暴露的風險等級： 中

實作指引

- 定義向上呈報路徑：在您的執行手冊和程序手冊中定義向上呈報路徑，包括觸發向上呈報的條件以及向上呈報的程序。例如，當執行手冊無法解決問題或經過預定時間，將問題從支援工程師向上呈報給資深支援工程師。適當的向上呈報途徑還有，當程序手冊無法確定工作負載的補救途徑或經過預定時間，從高級支援工程師向上呈報給開發團隊。明確確定每個動作的擁有者，以確保對營運事件做出迅速有效的回應。向上呈報可以包括第三方。例如，網路連接提供商或軟體供應商。向上呈報可以包括受影響系統的指定授權決策者。

OPS10-BP05 定義因應中斷的客戶通訊計劃

定義和測試系統中斷時您可以仰賴的通訊計劃，讓您的客戶和利害關係人在中斷期間持續獲得通知。使用者所使用的服務受到影響以及服務回到正常狀態時，直接與使用者通訊。

預期成果：

- 您對於從排程維護到大型非預期失敗範圍的情況具有通訊計劃，包括叫用災難復原計劃。
- 在您的通訊中，您提供清楚、透明的系統問題相關資訊，協助客戶避免第二次猜測他們系統的效能。
- 您使用自訂錯誤訊息和狀態頁面，減少服務台請求的峰值，並且讓使用者了解情況。
- 定期測試通訊計劃，以確認在實際發生中斷時，計劃會如預期執行。

常見的反模式：

- 發生工作負載中斷，但是您沒有任何通訊計劃。使用者的請求淹沒您的故障票證系統，因為他們沒有任何有關中斷的資訊。
- 您在中斷期間傳送電子郵件通知給您的使用者。通知不包含恢復服務的時間表，所以使用者無法針對中斷進行計劃。
- 有中斷的通訊計劃，但是從未測試過。發生中斷且通訊計劃失敗，因為遺漏可以在測試中攔截的關鍵步驟。

- 在中斷期間，您傳送給使用者的通知中包含太多 AWS NDA 之下的技術詳細資料和資訊。

建立此最佳實務的優勢：

- 在中斷期間維護通訊可確保為客戶提供問題進度和解決方式預估時間的可見性。
- 開發定義明確的通訊計劃，確認您的客戶和最終使用者了解情況，讓他們可以採取必要的額外步驟來緩解中斷的影響。
- 透過適當的通訊和對於預定和意外中斷提高的感知，您可以改善客戶滿意度、限制意外的反應並且留住客戶。
- 即時且透明的系統中斷通訊可建立信心，並且建立維護您與客戶之間關係所需的信任。
- 中斷或危機期間經實證的通訊策略可減少可能會阻礙您的復原能力的推測和流言。

未建立此最佳實務時的風險暴露等級：中

實作指引

在中斷期間讓您的客戶了解狀況的通訊計劃是全方位的，並且涵蓋多個界面，包括面對客戶的錯誤頁面、自訂 API 錯誤訊息、系統狀態橫幅以及運作狀態頁面。如果您的系統包含已註冊的使用者，您可以透過例如電子郵件、簡訊或推送通知的傳訊通道進行通訊，將個人化訊息內容傳送給您的客戶。

客戶通訊工具

做為防禦的第一線，Web 和行動應用程式應該在中斷期間提供易讀且內容豐富的錯誤訊息，並且能夠將流量重新導向至狀態頁面。Amazon CloudFront 是全受管內容交付網路 (CDN)，其中包括定義和提供自訂錯誤內容的功能。CloudFront 中的自訂錯誤頁面是針對元件層級中斷的客戶傳訊良好第一層。CloudFront 也可以簡化管理和啟動狀態頁面，在預定和意外中斷期間攔截所有請求。

中斷隔離以離散服務時，自訂 API 錯誤訊息可以協助偵測和減少影響。Amazon API Gateway 可讓您為您的 REST API 設定自訂回應。這可讓您在 API Gateway 無法連線後端服務時，對 API 取用者提供清楚且有意義的訊息。特定系統功能由於服務層中斷而降級時，自訂訊息也可以用來支援中斷橫幅內容和通知。

直接傳訊是客戶傳訊最個人化的類型。Amazon Pinpoint 是可擴展多通道通訊的受管服務。Amazon Pinpoint 可讓您建置行銷活動，透過簡訊、電子郵件、語音、推送通知或您定義的自訂通道，廣泛地在您受影響的客戶之間廣播訊息。使用 Amazon Pinpoint 管理傳訊時，訊息行銷活動是明確定義、可測試，並且可以智慧化套用到目標客戶區段。一旦建立，可以依據活動排程或觸發行銷活動，並且可以輕易進行測試。

客戶範例

工作負載受損時，AnyCompany Retail 會將電子郵件通知傳送給他們的使用者。電子郵件說明哪個業務功能受損，並且提供服務何時還原的實際預估。此外，他們有狀態頁面，顯示有關工作負載運作狀態的即時資訊。通訊計劃每年兩次在開發環境中進行測試，驗證其有效性。

實作步驟

1. 決定您的傳訊策略的通訊通道。考慮您的應用程式的架構層面，並且判斷最佳策略，將意見回饋交付給您的客戶。這可能包括一或多個概述的指引策略，包括錯誤和狀態頁面、自訂 API 錯誤回應或直接傳訊。
2. 設計應用程式的狀態頁面。如果您判斷狀態或自訂錯誤頁面適合您的客戶，您需要為這些頁面設計您的內容和傳訊。錯誤頁面會向使用者說明為何應用程式無法使用、何時可再次變成可用，以及在此同時他們可以做什麼。如果您的應用程式使用 Amazon CloudFront，您可以提供自訂錯誤回應或在邊緣使用 Lambda 以轉譯錯誤並且重新撰寫頁面內容。CloudFront 也可讓您從應用程式內容切換到靜態 Amazon S3 內容來源這樣的目的地，其中包含您的維護或中斷狀態頁面。
3. 為您的服務設計正確的一組 API 錯誤狀態。當 API Gateway 無法連線後端服務以及服務層例外狀況時所產生的錯誤訊息，可能不包含適合顯示給最終使用者的易讀訊息。不需要對您的後端服務進行程式碼變更，您可以設定 API Gateway 自訂錯誤回應將 HTTP 回應代碼對應至策劃 API 錯誤訊息。
4. 從企業觀點設計訊息，讓它與系統的最終使用者相關，而且不包含技術詳細資料。請考慮您的對象並且調整您的訊息。例如，您可能會讓內部使用者採用利用替代系統的因應措施或手動程序。可能會要求外部使用者等候直到系統還原，或者訂閱更新以在系統還原時收到通知。針對多個情境定義已核准的傳訊，包括非預期中斷、預定維護和部分系統失敗，其中特定功能可能會降級或無法使用。
5. 範本化及自動化您的客戶傳訊。一旦您建立您的訊息內容，您可以使用 Amazon Pinpoint 或其他工具來自動化您的傳訊行銷活動。使用 Amazon Pinpoint，您可以針對特定受影響的使用者建立客戶目標區段，並且將訊息轉換為範本。請參閱 Amazon Pinpoint 教學以了解如何設定傳訊行銷活動。
6. 避免將傳訊功能緊密結合到您的面對客戶系統。您的傳訊策略不應該有與系統資料放區或服務的硬性相依性，以便確認您可以在遇到中斷時成功傳送訊息。請針對傳訊可用性考慮從多個可用區域或區域傳送訊息的能力。如果您使用 AWS 服務來傳送訊息，請透過控制平面操作利用資料平面來叫用您的傳訊。

實作計劃的工作量：高。開發通訊計劃以及傳送訊息的機制，需要大量工作量。

資源

相關的最佳實務：

- OPS07-BP03 使用執行手冊執行程序 - 您的通訊計劃應該具有相關聯的執行手冊，讓您的人員知道如何回應。
- OPS11-BP02 執行事故後分析 - 在中斷之後，執行事件後分析來識別機制以防止其他中斷。

相關文件：

- Amazon API Gateway 和 AWS Lambda 中的錯誤處理模式
- Amazon API Gateway 回應

相關範例：

- AWS Health 儀表板
- 維吉尼亞北部 (US-EAST-1) 區域中 AWS 服務的摘要

相關服務：

- AWS Support
- AWS 客戶協議
- Amazon CloudFront
- Amazon API Gateway
- Amazon Pinpoint
- Amazon S3

OPS10-BP06 透過儀表板傳達狀態

提供針對其目標受眾 (例如，內部技術團隊、領導和客戶) 量身定制的儀表板，以傳達業務的當前營運狀態，並提供感興趣的指標。

您可以使用 Amazon CloudWatch 儀表板 建立儀表板，它位於 CloudWatch 主控台中的自訂首頁上。您可以使用 Amazon QuickSight 這類商業智慧服務，建立和發佈工作負載和營運運作狀態 (例如，下單率、連線的使用者和交易時間) 的互動式儀表板。建立儀表板，以顯示指標的系統和業務等級檢視。

常用的反模式：

- 根據要求，您執行應用程式目前使用率的報告來進行管理。
- 在事故期間，相關系統擁有者每 20 分鐘就會聯絡您一次，想知道問題是否已修正。

建立此最佳實務的優勢： 透過建立儀表板，您可以自助存取資訊，讓您的客戶能夠自行獲得相關資訊並自行判斷是否需要採取動作。

若未建立此最佳實務，暴露的風險等級： 中

實作指引

- 透過儀表板溝通狀態：提供針對其目標受眾 (例如，內部技術團隊、領導階層和客戶) 量身定制的儀表板，以傳達企業的當前營運狀況，並提供感興趣的指標。提供自助獲取狀態資訊選項，減少因回應營運團隊狀態請求而造成的干擾。範例包括 Amazon CloudWatch 儀表板和 AWS Health Dashboard。
  - CloudWatch 儀表板建立和使用自訂指標檢視

資源

相關文件：

- Amazon QuickSight
- CloudWatch 儀表板建立和使用自訂指標檢視

OPS10-BP07 自動回應事件

自動對事件進行回應，以減少由手動程序引起的錯誤，並確保快速一致的回應。

有多種方式可以在 AWS 上將執行手冊和程序手冊動作自動化。若要回應來自 AWS 資源狀態變更的事件，或您自己的自訂事件，您應建立 CloudWatch Events 規則 透過 CloudWatch 目標觸發回應 (例如，Lambda 函數、Amazon Simple Notification Service (Amazon SNS) 主題、Amazon ECS 任務，以及 AWS Systems Manager Automation)。

要回應超過資源臨界值的指標 (例如，等待時間)，您應使用建立 CloudWatch 警示， 來執行一個或多個動作，方法為使用 Amazon EC2 動作、Auto Scaling 動作，或將通知傳送至 Amazon SNS 主題。如果您需要執行自訂動作來回應警示，則請透過 Amazon SNS 通知叫用 Lambda。使用 Amazon SNS 發佈事件通知和向上呈報訊息，以使人們了解情況。

AWS 還可透過 AWS 服務 API 和 SDK 支援第三方系統。AWS 合作夥伴和第三方提供了許多監控工具，可用於監控、通知和回應。其中一些工具包含 New Relic、Splunk、Loggly、SumoLogic 和 Datadog。

當自動化程序失敗時，您應保留重要的手動程序以供使用

常用的反模式：

- 開發人員檢查其程式碼。此事件原本可能用於啟動建置，然後執行測試，不過沒有發生任何情況。
- 您的應用程式會在停止運作之前記錄特定錯誤。您應非常了解重新啟動應用程式的程序，且可以編寫此程序的指令碼。您可以使用日誌事件來叫用指令碼，並重新啟動應用程式。相反地，當星期日凌晨 3 點發生錯誤時，您做為負責修正系統的待命資源將被喚醒。

建立此最佳實務的優勢： 透過對事件使用自動回應，您可以縮短回應時間，並限制手動活動引入錯誤。

若未建立此最佳實務，暴露的風險等級為： 低

實作指引

- 將事件的回應自動化：自動對事件進行回應，以減少由手動流程引起的錯誤，並確保快速、一致的回應。
  - 什麼是 Amazon CloudWatch Events？
  - 建立隨事件觸發的 CloudWatch Events 規則
  - 使用 AWS CloudTrail 建立隨 AWS API 呼叫觸發的 CloudWatch Events 規則
  - 來自所支援服務的 CloudWatch Events 事件範例

資源

相關文件：

- Amazon CloudWatch 功能
- 來自所支援服務的 CloudWatch Events 事件範例
- 使用 AWS CloudTrail 建立隨 AWS API 呼叫觸發的 CloudWatch Events 規則
- 建立隨事件觸發的 CloudWatch Events 規則
- 什麼是 Amazon CloudWatch Events？

相關影片：

- 制定監控計劃

相關範例：

演進

演進是隨著時間的推移持續改善的過程。根據從營運活動中獲得的經驗，實作頻繁的小量循序漸進的變更，並評估其是否成功帶來改善。

若要讓營運隨著時間演進，您必須能夠：

主題

學習、分享和改進

您必須定期留有時間，以進行營運活動分析、失敗情境分析、試驗及作出改善。當事情失敗時，您將要確保您的團隊以及整個工程師社群都能從這些失敗中獲得經驗。您應分析失敗以識別獲得的經驗並規劃改善。您希望與其他團隊定期審查獲得的經驗，以驗證您的洞見。

最佳實務

OPS11-BP01 建立持續改進程序

根據內部和外部架構最佳實務評估您的工作負載。至少每年執行一次工作負載審查。根據您的軟體開發步調制定改進機會的優先順序。

預期成果：

- 您根據架構最佳實務以至少一年的間隔分析工作負載。
- 改進機會在您的軟體開發程序中獲得了均等的優先順序。

常見的反模式：

- 您在數年前部署工作負載後，即未對其執行過架構審查。
- 改進機會獲得了較低的優先順序，並保留在積存中。
- 沒有對組織的最佳實務實作修改的標準。

建立此最佳實務的優勢：

- 您的工作負載依據架構最佳實務保持在最新狀態。
- 您的工作負載演進以審慎的方式執行。
- 您可以利用組織最佳實務來改進所有工作負載。

未建立此最佳實務時的風險暴露等級：高

實作指引

您以至少一年的間隔執行工作負載的架構審查。使用內部和外部最佳實務，評估您的工作負載並識別改進機會。根據您的軟體開發步調制定改進機會的優先順序。

客戶範例

AnyCompany Retail 的所有工作負載均經過每年一次的架構審查處理。他們自行制定了適用於所有工作負載的最佳實務檢查清單。他們使用 AWS Well-Architected Tool 的自訂聚焦功能，利用最佳實務的工具和自訂聚焦執行審查。從審查產生的改進機會在軟體衝刺中獲得了優先順序。

實作步驟

1. 以至少一年的間隔，執行生產工作負載的定期架構審查。使用包含 AWS 特定最佳實務的已記載架構標準。
   a. 建議您使用自己的內部定義標準進行這些審查。如果您沒有內部標準，建議您使用 AWS Well-Architected Framework。
   b. 您可以使用 AWS Well-Architected Tool 來建立內部最佳實務的自訂聚焦，並執行架構審查。
   c. 客戶可聯絡其 AWS 解決方案架構師，在引導下執行其工作負載的 Well-Architected Framework 審查。

2. 在您的軟體開發程序中，為在審查期間找出的改進機會制定優先順序。

實作計劃的工作量：低。您可以使用 AWS Well-Architected Framework 執行年度架構審查。

資源

相關的最佳實務：

- OPS11-BP02 執行事故後分析 - 事件後分析是改進項目的另一個產生來源。將獲得的經驗饋送到架構最佳實務的內部清單中。
- OPS11-BP08 記錄和分享獲得的經驗 - 自行制定架構最佳實務時，請在您的組織中予以共享。

相關文件：

- AWS Well-Architected Tool - 自訂聚焦
- AWS Well-Architected 白皮書 - 審查程序
- 使用自訂聚焦和 AWS Well-Architected Tool 自訂 Well-Architected 審查
- 在您的組織中實作 AWS Well-Architected Custom Lens 生命週期

相關影片：

- Well-Architected 實驗室 - Level 100：AWS Well-Architected Tool 上的自訂聚焦

相關範例：

- AWS Well-Architected Tool

OPS11-BP02 執行事故後分析

審查影響客戶的事件，並識別造成問題的因素和預防性措施。使用此資訊來開發緩解措施，以限制或防止事件再次發生。制定可快速有效回應的程序。適當地傳達成因和為目標受眾量身打造的糾正措施。

常用的反模式：

- 您管理應用程式伺服器。大約每 23 小時 55 分鐘，所有作用中工作階段都會終止。您已嘗試識別應用程式伺服器上發生了什麼問題。您懷疑這反而可能是網路問題，但無法與網路團隊合作，因為他們

太忙而無法為您提供支援。您缺少可遵循的預先定義程序來取得支援與收集必要資訊，以判斷發生的情況。

- 您的工作負載內發生資料遺失問題。這是第一次發生，原因尚不確定。您確定它並不重要，因為您可以重新建立資料。資料遺失以影響客戶的較高頻率開始發生。當您還原遺失的資料時，這也會為您帶來額外的操作負擔。

建立此最佳實務的優勢： 透過預先定義的程序來判斷造成事件的元件、條件、動作和事件，讓您能夠找出改進機會。

若未建立此最佳實務，暴露的風險等級為： 高

實作指引

- 使用程序判斷成因：審查所有影響客戶的事故。建立程序來識別和記錄事件的成因，以便您可以制定緩解措施來限制或防止事件再次發生。另外，您還可以制定快速有效地做出回應的程序。根據目標受眾的不同以適當的方式告知根本原因。

OPS11-BP03 實作回饋迴圈

回饋迴圈提供可推動決策的可行洞察。在程序和工作負載中建立回饋迴圈。此可協助您找出問題和需要改善的地方。回饋迴圈也會驗證在改善中所做的投資。這些回饋迴圈是持續改善工作負載的基礎。

回饋迴圈分為兩種： 即時回饋 和 追溯性分析。透過審查營運活動的績效和成果來收集即時的回饋。此回饋來自團隊成員、客戶或活動的自動化輸出。接收 A/B 測試和交付新功能等方面的即時回饋，對於快速檢錯非常重要。

定期進行追溯性分析，以從對營運成果和指標的審查中獲取回饋。這些追溯性分析會在衝刺結束，按規律或在主要版本或事件後發生。這類回饋迴圈會驗證對營運或工作負載所做的投資。其可協助您衡量成功並驗證策略。

預期成果： 您使用即時回饋和追溯性分析來推動改善。存在可擷取使用者和團隊成員回饋的機制。追溯性分析會用來找出可推動改善的趨勢。

常見的反模式：

- 您推出新功能，但沒有辦法收到客戶對該功能的回饋。
- 針對營運改善投入資源和時間後，您無法執行追溯性分析來進行驗證。
- 您收集客戶的回饋，但未能定期審查回饋。

- 回饋迴圈讓我們得以提議行動項目，但軟體開發程序中未納入這些項目。
- 客戶沒有收到他們提議之改善的回饋。

建立此最佳實務的優勢：

- 您可以反過來與客戶合作來推動新功能。
- 您的組織文化可以更快速地應對變化。
- 趨勢會用來找出改善的機會。
- 追溯性分析可驗證對工作負載和營運所做的投資。

若未建立此最佳實務，暴露的風險等級： 高

實作指引

實作此最佳實務表示您同時使用即時回饋和追溯性分析。這些回饋迴圈可推動改善。有許多機制可用來處理即時回饋，包含調查、客戶投票和回饋表單。組織也會使用追溯性分析來找出改善的機會並驗證計劃。

客戶範例

AnyCompany Retail 建立網頁表單，客戶可在其中提供回饋或回報問題。在每週 Scrum 期間，軟體開發團隊會評估使用者回饋。該團隊會定期使用回饋來為其平台的發展釐清方向。他們會在每次衝刺結束時執行追溯性分析，來找出他們想要改善的項目。

實作步驟

1. 即時回饋
   - 您需要制定機制來接收來自客戶和團隊成員的回饋。您也可以設定營運活動來提供自動化的回饋。
   - 組織需要制定程序來審查此回饋、判斷需要改善的項目，並安排改善項目。
   - 您必須將回饋新增至軟體開發程序。
   - 在您著手改善後，請與回饋提交者追蹤後續進展。
     - 您可以使用 AWS Systems Manager OpsCenter， 以 OpsItems 的形式 建立和追蹤這些改善。
2. 追溯性分析
   - 在開發週期結束時，以固定的規律或在主要版本之後，執行追溯性分析。

- 召集工作負載中參與的利害關係人，進行回顧會議。
- 在白板或試算表建立三個欄位：停止、開始和持續。
  - 停止 是您希望團隊停止做的任何事。
  - 開始 是您希望開始執行的想法。
  - 持續 是您希望持續執行的項目。
- 詢問在場人士的想法，收集利害關係人的回饋。
- 排列回饋的優先順序。將動作和利害關係人指派至任何「開始」或「持續」項目。
- 將動作新增至軟體開發程序中，並在您執行改善項目時向利害關係人告知最新的狀態。

實作計劃的工作量： 中。若要實作此最佳實務，您需要找到方法來擷取即時回饋並進行分析。此外，您需要建立追溯性分析程序。

資源

相關的最佳實務：

- OPS01-BP01 評估外部客戶需求：回饋迴圈是一種機制，可收集外部客戶的需求。
- OPS01-BP02 評估內部客戶需求：內部利害關係人可以使用回饋迴圈來表達需要和需求。
- OPS11-BP02 執行事故後分析：事件後分析是在事件後執行的追溯性分析的一種重要形式。
- OPS11-BP07 執行營運指標審查：營運指標審查會找出趨勢和待改善的地方。

相關文件：

- 建置 CCOE 時應避開的 7 大陷阱
- Atlassian 團隊程序手冊 - 追溯性
- 電子郵件定義：回饋迴圈
- 根據 AWS Well-Architected Framework 審查建立回饋迴圈
- IBM Garage Methodology - 進行回顧
- Investopedia – PDCS 週期
- 最大化開發人員的效能 (作者：Tim Cochran)
- 營運準備度審查 (ORR) 白皮書 - 反覆執行
- TIL CSI - 持續服務改善
- 當 Toyota 遇見電子商務：Amazon 的精實原則

相關影片：

- 建立有效的客戶回饋迴圈

相關範例：

- Astuto - 開放原始碼客戶回饋工具
- AWS 解決方案 - AWS 上的 QnABot
- Fider - 整理客戶回饋的平台

相關服務：

- AWS Systems Manager OpsCenter

OPS11-BP04 執行知識管理

知識管理可協助團隊成員尋找資訊以執行其作業。在學習組織中，資訊是任意共用的，助個人一臂之力。資訊可以探索和搜尋。資訊是準確且最新的。存在機制以建立新資訊、更新現有資訊，以及封存過時資訊。最常見的知識管理平台範例是內容管理系統，例如 Wiki。

預期成果：

- 團隊成員可以存取及時、準確的資訊。
- 資訊是可搜尋的。
- 存在機制以新增、更新和封存資訊。

常見的反模式：

- 沒有集中式知識儲存。團隊成員會在他們的本機電腦上管理他們自己的備註。
- 您有自我託管的 Wiki，但是沒有管理資訊的機制，導致資訊過時。
- 某人識別遺漏的資訊，但是沒有要求在團隊 Wiki 中新增它的程序。他們自行新增，但是遺漏關鍵步驟，導致中斷。

建立此最佳實務的優勢：

- 因為資訊任意共用，所以團隊成員握有能力。

- 因為文件是最新的且可搜尋，所以新的團隊成員可以更快上線。
- 資訊是及時、準確且可行的。

未建立此最佳實務時的風險暴露等級：高

實作指引

知識管理是學習組織的重要面向。若要開始，您需要集中儲存庫來存放您的知識 (常見的範例是自我託管的 Wiki)。您必須開發新增、更新和封存知識的程序。開發應該記載哪些項目的標準，並且讓所有人做出貢獻。

客戶範例

AnyCompany Retail 託管內部 Wiki，在其中存放所有知識。團隊成員受到鼓勵在他們執行每日職責時新增至知識庫。跨功能團隊每季會評估哪些頁面最少更新，並且判斷它們是否應該封存或更新。

實作步驟

1. 從識別存放知識所在的內容管理系統開始。跨組織取得利害關係人的協議。
   a. 如果您沒有現有內容管理系統，請考慮執行自我託管 Wiki 或使用版本控制儲存庫做為起點。
2. 開發新增、更新和封存資訊的執行手冊。向您的團隊教育這些程序。
3. 識別哪些知識應該存放在內容管理系統中。從團隊成員執行的每日活動 (執行手冊和程序手冊) 開始。與利害關係人合作來排列新增知識的優先順序。
4. 定期與利害關係人合作來識別過時資訊並且將它封存或更新。

實作計劃的工作量：中。如果您沒有現有內容管理系統，您可以設定自我託管 Wiki 或版本控制文件儲存庫。

資源

相關的最佳實務：

- OPS11-BP08 記錄和分享獲得的經驗 - 知識管理可促進所學習課程的資訊共用。

相關文件：

- Atlassian - 知識管理

相關範例：

- DokuWiki
- Gollum
- MediaWiki
- Wiki.js

OPS11-BP05 定義改進驅動因素

確定改進驅動因素，以幫助您評估改進機會並排定其優先順序。

在 AWS 上，您可以彙整所有營運活動、工作負載和基礎設施的日誌，以建立詳細的活動歷史記錄。然後，您可以使用 AWS 工具，分析某段時間內的營運和工作負載運作狀態 (例如，識別趨勢、將事件和活動與成果關聯，以及在環境間和跨系統進行比較和對比)，根據驅動因素來發現改善機會。

您應使用 CloudTrail 追蹤 API 活動 (透過 AWS Management Console、CLI、SDK 和 API)，以了解整個帳戶中發生的情況。使用 CloudTrail 和 CloudWatch 追蹤您的 AWS 開發人員工具部署活動。這樣會將部署及其成果的詳細活動歷史記錄新增至 CloudWatch Logs 日誌資料中。

將日誌資料匯出至 Amazon S3 以進行長期儲存。您可以使用 AWS Glue，您可以探索和準備 Amazon S3 中的日誌資料以進行分析。使用 Amazon Athena，透過與 AWS Glue 原生整合來分析日誌資料。使用 Amazon QuickSight 這類商業智慧工具來視覺化、探索和分析您的資料

常用的反模式：

- 您有一個可運作但不巧妙的指令碼。您投入時間來重新撰寫它。現在該指令碼相當出色。
- 您的新創公司正嘗試從創投家獲得另一批資金。他們希望您證明 PCI DSS 的合規性。您想要讓客戶滿意，因此您以文件記錄合規情況，但錯過提供給客戶的交付日期，而失去該客戶。做這件事沒有錯，但現在您不知道這是否是對的。

建立此最佳實務的優勢： 藉由決定您想要用於改進的條件，您可以將事件型動機或情緒投資的影響降到最低。

若未建立此最佳實務，暴露的風險等級為： 中

實作指引

- 了解改進驅動因素：僅在支援理想結果時才對系統進行變更。

- 所需能力：在評估改進機會時，評估所需的功能和能力。
  - AWS 最新消息
- 不可接受的問題：在評估改進機會時，評估不可接受的問題、錯誤和弱點。
  - AWS 最新安全公告
  - AWS Trusted Advisor
- 合規要求：在審查改進機會時，評估保持法規、政策的遵從性或保持受到第三方支援所需的更新和變更。
  - AWS 合規
  - AWS 合規計劃
  - AWS 合規最新資訊

資源

相關文件：

- Amazon Athena
- Amazon QuickSight
- AWS 合規
- AWS 合規最新資訊
- AWS 合規計劃
- AWS Glue
- AWS 最新安全公告
- AWS Trusted Advisor
- 將日誌資料匯出至 Amazon S3
- AWS 最新消息

OPS11-BP06 驗證洞見

與跨職能團隊和企業擁有者一起審查您的分析結果和回應。透過這些審查建立共識，確定其他影響並確定行動方案。適當調整回應。

常用的反模式：

- 您看到系統上的 CPU 使用率為 95%，並優先找出降低系統負載的方法。您確定最佳行動方案是擴充。該系統是轉碼器，系統會擴展到一直以 95% 的 CPU 使用率執行。系統擁有者可能向您解釋情況，並讓您聯絡他們。您的時間被浪費了。
- 系統擁有者堅稱系統是任務關鍵性系統。系統未放置在高安全性的環境中。為改善安全性，您實作任務關鍵性系統所需的額外偵測和預防性控制措施。您通知系統擁有者工作已完成，且其需為其他資源支付相應費用。在此通知之後的討論中，系統擁有者了解了其系統不符合的任務關鍵系統的正式定義。

建立此最佳實務的優勢： 透過與企業擁有者和領域專家驗證洞見，您可以建立共識並更有效地引導改進。

若未建立此最佳實務，暴露的風險等級： 中

實作指引

- 驗證洞見：與企業擁有者和領域專家互動，確保您收集資料的意義得到眾人理解和同意。識別其他疑慮、潛在影響，並確定行動方案。

OPS11-BP07 執行營運指標審查

與來自不同業務領域的跨團隊參與者定期進行營運指標的追溯性分析。透過這些審查確定改進機會、可能的行動方案並分享獲得的經驗。

尋找所有環境 (例如開發、測試和生產) 中的改善機會。

常用的反模式：

- 您的維護時段中斷了重要的零售促銷。如果還有其他影響企業的事件，企業仍然不知道是否有可能會延遲的標準維護時段。
- 您使用組織中常用的錯誤程式庫，因此您經歷了長時間的中斷。之後您已遷移到可靠的程式庫。組織中的其他團隊不知道他們正面臨風險。如果你們定期會面並審查此事故，他們就會注意風險。
- 轉碼器的效能一直在穩定地下降，並影響媒體團隊。這還不是很令人震驚。除非該情況嚴重到足以造成事故，否則您將無法查明該情況。如果您與媒體團隊審查營運指標，則有機會識別指標及其體驗的變更並解決問題。
- 您沒有審查對客戶 SLA 的滿意度。您有不符合客戶 SLA 的趨勢。不符合客戶 SLA，會產生相關的財務處罰。如果你們定期會面，並審查這些 SLA 的指標，您將有機會識別並解決問題。

建立此最佳實務的優勢： 透過定期會議以審查營運指標、事件和事故，您可以在團隊間維持共識、分享獲得的經驗，並可以排定改進項目的優先順序並鎖定改進目標。

若未建立此最佳實務，暴露的風險等級： 中

實作指引

- 營運指標審查：與來自不同業務領域的跨團隊參與者定期進行營運指標的追溯性分析。與包括業務、開發和營運團隊在內的利害關係人進行互動，以驗證您從即時回饋和追溯性分析獲得的發現，並分享經驗教訓。利用這些洞見確定改進機會和可能的行動方案。
  - Amazon CloudWatch
  - 使用 Amazon CloudWatch 指標
  - 發佈自訂指標
  - Amazon CloudWatch 指標和維度參考

資源

相關文件：

- Amazon CloudWatch
- Amazon CloudWatch 指標和維度參考
- 發佈自訂指標
- 使用 Amazon CloudWatch 指標

OPS11-BP08 記錄和分享獲得的經驗

記錄並分享從營運活動中獲得的經驗，以便您可以在內部以及跨團隊使用它們。

您應分享您的團隊獲得的經驗，以提高整個組織的效益。您希望分享資訊和資源，以防止可避免的錯誤並簡化開發工作。如此可讓您聚焦於提供所需的功能。

使用 AWS Identity and Access Management (IAM) 定義權限，從而實現對您希望在帳戶內及帳戶間分享的資源的受控存取。然後，您使用版本控制的 AWS CodeCommit 儲存器來分享應用程式程式庫、執行指令碼的程序、程序文件及其他系統文件。透過分享對 AMI 的存取以及授權跨帳戶使用 Lambda 函數，進而分享您的運算標準。您還應將基礎設施標準作為 AWS CloudFormation 範本進行分享。

透過 AWS API 和 SDK，您可以整合外部和第三方工具及儲存器 (例如 GitHub、BitBucket 和 SourceForge)。分享您獲得的經驗和開發的知識時，請小心建構權限，以確保分享的儲存器的完整性。

常用的反模式：

- 您使用組織中常用的錯誤程式庫，因此您經歷了長時間的中斷。之後您已遷移到可靠的程式庫。組織中的其他團隊不知道他們正面臨風險。如果您在此程式庫中記錄和分享您的經驗，他們會注意風險。
- 您已在內部共用的微型服務中找出導致工作階段終止的邊緣案例。您已更新對服務的呼叫，以避免此邊緣案例。組織中的其他團隊不知道他們正面臨風險。如果您在此程式庫中記錄和分享您的經驗，他們會注意風險。
- 您已找到一個方法，可大幅降低其中一個微型服務所需的 CPU 使用率。您不知道是否有任何其他團隊可以利用此技術。如果您在此程式庫中記錄和分享您的經驗，其他團隊將有機會這樣做。

建立此最佳實務的優勢： 分享獲得的經驗以協助改進並將經驗的好處發揮到最大。

若未建立此最佳實務，暴露的風險等級： 低

實作指引

- 記錄和分享獲得的經驗：制定程序來記錄從執行營運活動和追溯性分析中學到的經驗教訓，以便其他團隊可以使用。
  - 分享經驗：制定程序來在團隊之間分享經驗教訓和相關成品。例如，透過可存取的 Wiki 分享更新的程序、指引、管控和最佳實務。透過公共儲存庫共用指令碼、程式碼和程式庫。
    - 委託存取您的 AWS 環境
    - 共用 AWS CodeCommit 儲存庫
    - 輕鬆授權 AWS Lambda 函數
    - 與特定 AWS 帳戶共用 AMI
    - 使用 AWS CloudFormation Designer URL 加速範本共用
    - 搭配 Amazon SNS 使用 AWS Lambda

資源

相關文件：

- 輕鬆授權 AWS Lambda 函數

- 共用 AWS CodeCommit 儲存庫
- 與特定 AWS 帳戶共用 AMI
- 使用 AWS CloudFormation Designer URL 加速範本共用
- 搭配 Amazon SNS 使用 AWS Lambda

相關影片：

- 委託存取您的 AWS 環境

OPS11-BP09 分配改進時間

在流程中投入時間和資源，以持續逐漸改善。

在 AWS 上，您可以建立臨時環境複本，從而降低試驗和測試的風險、工作量及成本。這些重複的環境可用於測試從您的分析、試驗和開發得出的結論，以及測試計劃的改善。

常用的反模式：

- 您的應用程式伺服器存在已知的效能問題。它會新增到每個計劃功能實作的待辦項目中。如果計劃功能的新增速率保持不變，則效能問題永遠不會解決。
- 為協助持續改進，您核准管理員和開發人員使用他們額外的時間來選取和實作改進項目。進改永遠不會有完成的一天。

建立此最佳實務的優勢： 透過在程序中投入時間和資源，您可以實現持續逐漸改善。

若未建立此最佳實務，暴露的風險等級： 低

實作指引

- 分配改進時間：在流程中投入時間和資源，以持續逐漸改善。實作變更以改進和評估結果，從而確定成功與否。如果結果未能達到目標，並且改進仍然是優先事項，則應採取替代行動方案。

結論

卓越營運需要堅持不懈的努力方能達成。

透過共同的目標來讓組織做好準備，邁向成功。確保每個人都了解自己在達成業務成果方面的角色，以及他們如何影響他人成功的能力。為團隊成員提供支援，讓他們能夠協助達成業務成果。

每個營運事件和失敗均應被視為改善您架構營運的機會。透過了解您的工作負載需求，預定義例行活動的執行手冊，和可指引問題解決方案的程序手冊，在 AWS 中將營運用作程式碼功能以及維持狀況認知，則在發生事件時，您的營運團隊將可做好更完善的準備並能夠更有效回應。

在優先事項變更時，透過專注於以優先事項為基礎的增量改善，以及從事件回應和追溯性分析中獲得的經驗，您將能加強活動的效率和效果，進而實現業務成功。

AWS 致力於協助您建置和營運架構，以便在您建置快速回應且適應性高的部署時能發揮最大效率。若要提高工作負載的卓越營運，您應該使用本白皮書中所述的最佳實務。

作者群

- Rich Boyd，Amazon Web Services Well-Architected 卓越營運支柱主管
- Jon Steele，Amazon Web Services Well-Architected 解決方案架構師
- Ryan King，Amazon Web Services 資深技術計劃經理
- Chris Kunselman，Amazon Web Services 諮詢顧問
- Peter Mullen，Amazon Web Services 諮詢顧問
- Brian Quinn，Amazon Web Services 資深諮詢顧問
- David Stanley，Amazon Web Services 雲端營運模式主管
- Chris Kozlowski，Amazon Web Services Enterprise Support 資深專家技術客戶經理
- Alex Livingstone，Amazon Web Services Cloud Operations 首席專家解決方案架構師
- Paul Moran，Amazon Web Services Enterprise Support 首席技術專家
- Peter Mullen，Amazon Web Services Professional Services 諮詢顧問
- Chris Pates，Amazon Web Services Enterprise Support 資深專家技術客戶經理
- Arvind Raghunathan，Amazon Web Services Enterprise Support 首席專家技術客戶經理
- Ben Mergen，Amazon Web Services 資深解決方案架構師

深入閱讀

如需其他指引，請參考以下資源：

- AWS Well-Architected Framework
- AWS 架構中心

文件修訂

若要收到此白皮書更新的通知，請訂閱 RSS 摘要。

變更	描述	日期
主要內容更新與整合	有多個最佳實務領域的內容已更新並整合。有兩個最佳實務領域 (OPS 04 和 OPS 08) 已重新撰寫，納入了新的內容和重點。 以下領域的最佳實務已更新並整合：營運設計、緩解部署風險和 了解運作狀態。最佳實務領域 OPS 04 已更新至 實作可觀測性。最佳實務領域 OPS 08 已更新至 利用工作負載可觀測性。	October 3, 2023
新框架的更新	最佳實務已更新，納入了規範性指引，並增加了新的最佳實務。	April 10, 2023
白皮書已更新	最佳實務更新了新的實作指引。	December 15, 2022
白皮書已更新	已擴充最佳實務並新增了改善計劃。	October 20, 2022
小幅度更新	小幅度編輯更新。	August 8, 2022
白皮書已更新	更新以反映新的 AWS 服務和功能以及最新的最佳實務。	February 2, 2022
小幅度更新	已將永續性支柱新增至簡介。	December 2, 2021

新框架的更新	更新以反映新的 AWS 服務和功能以及最新的最佳實務。	July 8, 2020
白皮書已更新	更新以反映新的 AWS 服務和功能以及更新的參考。	July 1, 2018
初版	卓越營運支柱 – AWS Well-Architected Framework 已發佈。	November 1, 2017