CI/CD 貼圖測試：您的管道是完全 CI/CD 嗎？

AWS 方案指引

AWS 方案指引: CI/CD 貼圖測試：您的管道是完全 CI/CD 嗎？

Table of Contents

CI/CD 貼圖測試：您的管道是完全 CI/CD 嗎？

Steven Guggenheimer 和 Ananya Koduri，Amazon Web Services (AWS)

2023 年 8 月 (文件歷史記錄)

您的管道是否自動化？ 這是一個簡單的問題，但許多組織太簡單地處理答案。答案比是或否更複雜。

技術創新不斷發生，有時候組織可能很難跟上進度。這個新事物是淡入的，還是下一個大事？ 我應該大修目前的實務，還是應該等待？ 通常，在很明顯的時候，有些事情確實是下一個大事，您可以發現自己正在玩補上。持續整合和持續交付 (CI/CD) 即將保留，但並非總是如此。許多人花了很長的時間來信服，而某些人仍然需要更信服。

CI/CD 是將軟體版本程序的來源、建置、測試、預備和生產階段自動化的程序，通常描述為管道。如今，CI/CD 自動化的成本節省和速度已讓大多數組織相信其價值。但轉換到這個新方法並不容易。您需要確保您的員工有適當的訓練，您需要升級一些資源，然後您需要測試、測試、測試。要做的有很多事。在大多數情況下，您想要逐步進行這些變更，以協助組織適應。

本文件的目的是定義擁有完整 CI/CD 程序的意義。它提供工具來評估您自己的程序，並為尚未存在的程序提供前進路徑。此路徑轉送很少是隔夜轉換。這些程序很複雜，取決於許多因素，包括目前的員工技能集和目前的基礎設施需求。我們建議您排定優先順序並進行小型的增量變更。

目標

以下是實作本指南中建議的潛在好處：

- 效率 – 完整的 CI/CD 部署程序可以降低複雜性、工作負載和花費數小時的除錯、執行手動程序和維護。如需詳細資訊，請參閱持續交付的優點。根據 TechAhead 部落格文章，CI/CD 程序的實作可預估節省 20% 的時間、精力和資源。
- 成本降低 – 根據 Forbes Insight 報告，「四分之三的高階主管同意，持續維護和管理所花費的時間、金錢和資源，而不是新的專案開發或新措施，會影響組織的整體競爭力。」 開發週期越短，您的組織就越有可能符合雄心勃勃time-to-market目標，並在正確的時間掌握正確的機會。
- 速度 – 一般而言，完全 CI/CD 管道可以在幾個小時內向客戶發佈軟體變更。特別是在快速故障隔離和小型修補程式推送的情況下，CI/CD 管道有助於改善平均復原時間 (MTTR)。如需詳細資訊，請參閱降低 MTTR。
- 安全 – 完全 CI/CD 管道也透過減少可能的攻擊進入點並降低人為錯誤的風險，來保護發佈程序。全自動化 CI/CD 管道帶來的安全性優勢有助於避免資料外洩、服務中斷等代價高昂的後果。

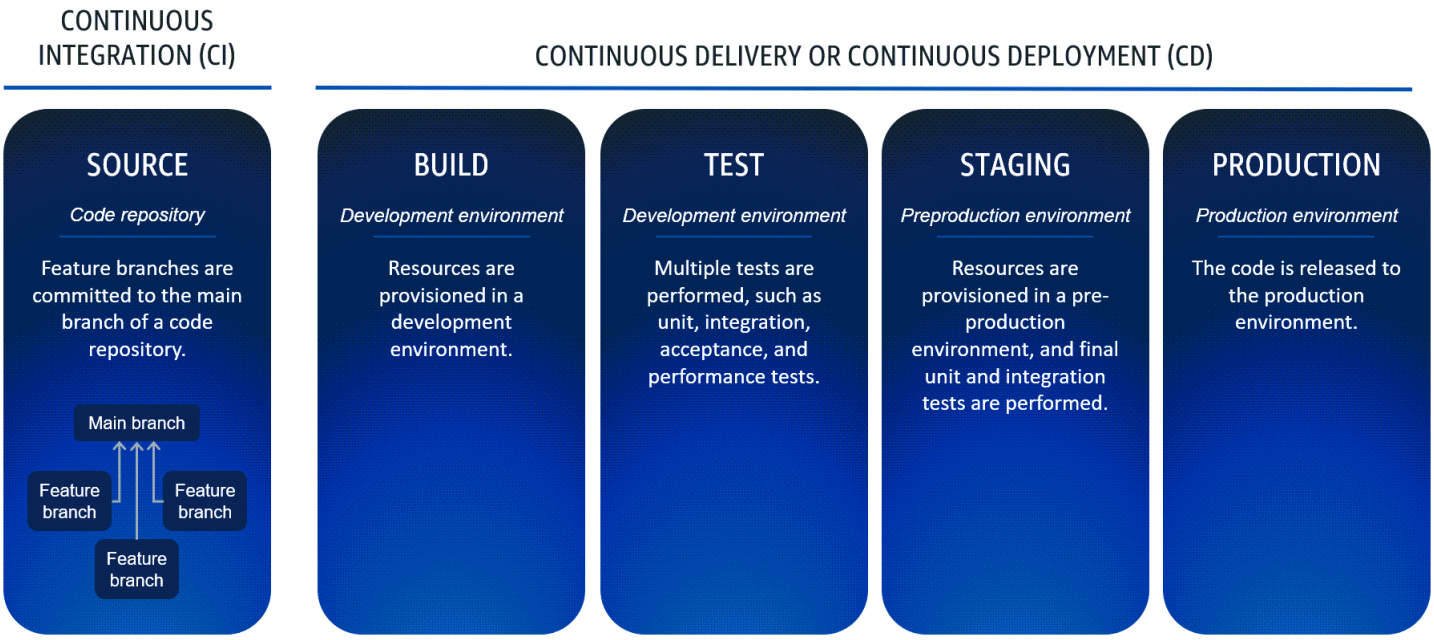
- 減少耗損 – 當開發人員可以花更多時間建立絕佳的功能，並減少在維護和偵錯的無限循環中模仿的時間時，他們會更滿意。對於組織，這表示取得和保留頂尖人才更長一段時間。
- 卓越品質程式碼 – 開發人員以小型批次將程式碼發佈到共用儲存庫，這可讓他們執行平行測試 (BrowserStack 部落格文章）。與其獨立工作，他們不常與團隊共用其組建，而是協同合作以識別重大錯誤。這為開發人員提供支援，這有助於防止錯誤程式碼進入生產環境。開發人員對等的支援有助於高品質版本並推動組織成長。
- 維護 – 維護和更新是打造優質產品的重要部分。不過，請勿在尖峰流量時間關閉系統。您可以使用 CI/CD 管道，在低使用時間執行維護，將停機時間和效能影響降至最低。

了解 CI/CD

持續整合和持續交付 (CI/CD) 是自動化軟體版本生命週期的程序。在某些情況下，CI/CD 中的 D 也可以表示部署。當您將變更發佈至生產環境時，會發生持續交付與持續部署之間的差異。持續交付時，在提升生產變更之前，需要手動核准。持續部署具有整個管道的不間斷流程，不需要明確核准。由於此策略討論一般 CI/CD 概念，因此提供的建議和資訊適用於持續交付和持續部署方法。

CI/CD 會自動化傳統上從遞交取得新程式碼到生產所需的許多或所有手動程序。CI/CD 管道包含來源、建置、測試、預備和生產階段。在每個階段中，CI/CD 管道會佈建部署或測試程式碼所需的任何基礎設施。透過使用 CI/CD 管道，開發團隊可以對程式碼進行變更，然後自動測試並推送到部署。

讓我們先檢閱基本 CI/CD 程序，然後再討論您可以知情或無意中偏離完全 CI/CD 的一些方式。下圖顯示每個階段中的 CI/CD 階段和活動。



關於持續整合

持續整合發生在程式碼儲存庫中，例如 中的 Git 儲存庫GitHub。您會將單一主分支視為程式碼基礎的真實來源，並建立短期分支以進行功能開發。當您準備好將功能部署到上層環境時，您可以將功能分支整合到主分支。功能分支永遠不會直接部署到上層環境。如需詳細資訊，請參閱本指南中的 以中繼線為基礎的方法。

持續整合程序

1. 開發人員會從主分支建立新的分支。
2. 開發人員會在本機進行變更、建置和測試。
3. 當變更準備就緒時，開發人員會建立提取請求 (GitHub 文件），以主分支做為目的地。
4. 系統會檢閱程式碼。
5. 當程式碼獲得核准時，它會合併到主分支。

關於持續交付

持續交付發生在隔離環境中，例如開發環境和生產環境。每個環境中發生的動作可能有所不同。通常，其中一個前階段用於在繼續之前更新管道本身。部署的最終結果是，每個環境都會以最新的變更進行更新。用於建置和測試的開發環境數量也有所不同，但我們建議您至少使用兩個。在管道中，每個環境都會依其重要性進行更新，並以最重要的環境做為結尾，也就是生產環境。

持續交付程序

管道的持續交付部分會透過從來源儲存庫的主分支提取程式碼並將其傳遞至建置階段來啟動。儲存庫的基礎設施做為程式碼 (IaC) 文件概述了在每個階段中執行的任務。雖然使用 IaC 文件不是強制性的，但AWS Cloud Development Kit (AWS CDK)強烈建議使用 IaC 服務或工具，例如 AWS CloudFormation或 。最常見的步驟包括：

1. 單位測試
2. 程式碼建置
3. 資源佈建
4. 整合測試

如果管道中的任何階段發生任何錯誤或任何測試失敗，目前階段會回復到先前的狀態，而管道會終止。後續變更必須在程式碼儲存庫中開始，並執行完整 CI/CD 程序。

CI/CD 管道的測試

部署管道中常用的兩種自動化測試類型是單元測試和整合測試。不過，您可以在程式碼基礎和開發環境上執行許多類型的測試。AWS 部署管道參考架構定義了下列類型的測試：

- 單元測試 – 這些測試會建置並執行應用程式程式碼，以確認其是否根據預期執行。它們模擬程式碼基礎中使用的所有外部相依性。單位測試工具的範例包括 JUnit、Jest 和 pytest。

- 整合測試 – 這些測試會驗證應用程式是否滿足技術需求，方法是針對佈建的測試環境進行測試。整合測試工具的範例包括 Cucumber、vRest NG 和 integ-tests （適用於 AWS CDK)。
- 接受測試 – 這些測試會針對佈建的測試環境進行測試，驗證應用程式是否符合使用者要求。接受測試工具的範例包括 Cypress 和 Selenium。
- 合成測試 – 這些測試會在背景中持續執行，以產生流量並驗證系統是否正常運作。合成測試工具的範例包括 Amazon CloudWatch Synthetics 和 Dynatrace Synthetic Monitoring。
- 效能測試 – 這些測試模擬生產容量。他們會判斷應用程式是否符合效能需求，並將指標與過去的效能進行比較。效能測試工具的範例包括 Apache JMeter、Locust 和 Gatling。
- 彈性測試 – 這些測試也稱為混亂測試，會將失敗注入環境，以識別風險區域。接著會將注入失敗的期間與沒有失敗的期間進行比較。彈性測試工具的範例包括 AWS Fault Injection Service和 Gremlin。
- 靜態應用程式安全測試 (SAST) – 這些測試會分析安全性違規的程式碼，例如 SQL 注入或跨網站指令碼 (XSS)。SAST 工具的範例包括 Amazon CodeGuru、SonarQube 和 Checkmarx。
- 動態應用程式安全測試 (DAST) – 這些測試也稱為滲透測試或筆測試。它們可識別佈建測試環境中的漏洞，例如 SQL 注入或 XSS。DAST 工具的範例包括 Zed Attack Proxy (ZAP) 和 HCL AppScan。如需詳細資訊，請參閱滲透測試。

並非所有完整 CI/CD 管道都會執行所有這些測試。不過，管道至少應該在程式碼基礎上執行單位測試和 SAST 測試，以及在測試環境中進行整合和接受測試。

CI/CD 管道的指標

根據AWS 部署管道參考架構，您至少應該追蹤 CI/CD 管道的下列四個指標：

- 前置時間 – 單一遞交完全投入生產所需的平均時間。我們建議您根據使用案例，以 1 小時到 1 天之間的前置時間為目標。
- 部署頻率 – 指定期間內的生產部署數量。我們建議您根據使用案例，將部署頻率設定為每天多次到每週兩次。
- 平均故障間隔時間 (MTBF) – 從成功管道開始到失敗管道開始的平均時間。我們建議以盡可能高的 MTBF 為目標。如需詳細資訊，請參閱增加 MTBF。
- 復原的平均時間 (MTTR) – 從失敗管道開始到下一個成功管道開始的平均時間。我們建議以盡可能低的 MTTR 為目標。如需詳細資訊，請參閱降低 MTTR。

這些指標可協助團隊追蹤其成為完整 CI/CD 的進度。團隊應與組織的利益相關者針對最佳目標進行公開討論。情況和需求因組織而異，甚至因團隊而異。

請務必記住，快速、劇烈的變更通常會增加發生問題的風險。設定目標以瞄準小型增量改進。完整 CI/CD 管道的常見最佳前置時間不到 3 小時。以 5.2 天的前置時間為起點的團隊，應目標每幾週減少一天。此團隊達到一天或更短的前置時間後，他們可以留在那裡數個月，並僅在團隊和組織利益相關者認為有必要時，才移至更積極的前置時間。

完整 CI/CD 程序有何不同

CI/CD 管道使用現代中繼線型工作流程，開發人員會將小型、頻繁的更新合併到透過 CI/CD 管道的 CD 部分建置和測試的主分支（或中繼)。此工作流程已取代 Gitflow 工作流程，其中開發和發行分支以發行排程分隔。在許多組織中，Gitflow 仍然是熱門的版本控制和部署方法。不過，現在會被視為舊版，而且整合到 CI/CD 管道可能很困難。

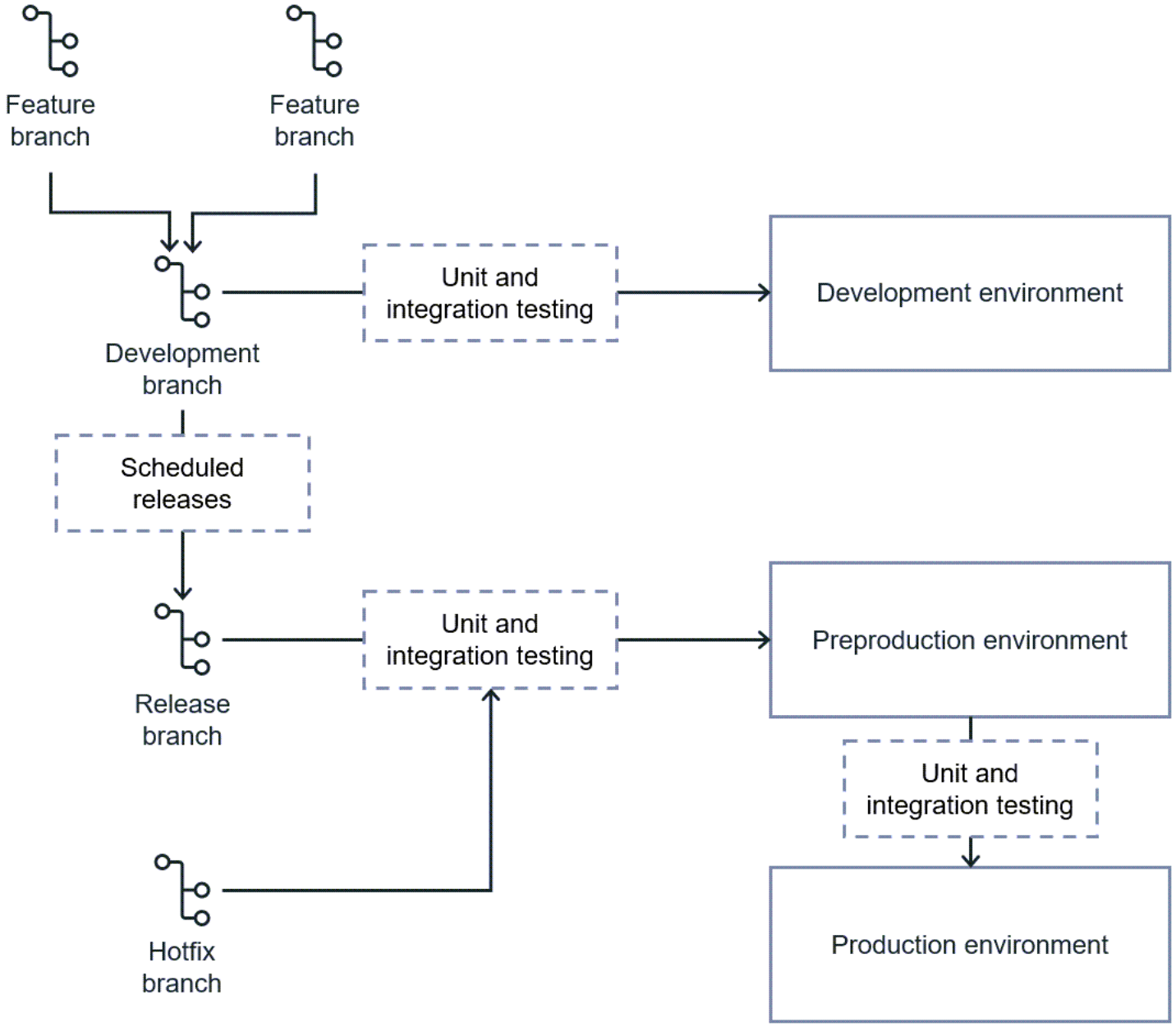
對於許多組織而言，從 Gitflow 工作流程到以中繼線為基礎的工作流程的轉換尚未完成，結果是它們在過程中卡在某個位置，永遠不會完全遷移到 CI/CD。以某種方式來說，他們的管道最終會追蹤到舊版工作流程的特定遺體，並卡在過去和現在之間的過渡狀態。檢閱 Git 工作流程中的差異，然後了解使用舊版工作流程如何影響下列項目：

- 環境完整性
- 推出
- 安全

為了更輕鬆地識別現代組態中舊版 Git 工作流程的剩餘，讓我們比較 Gitflow 與現代、以中繼為基礎的方法。

Gitflow 方法

下圖顯示 Gitflow 工作流程。Gitflow 方法使用多個分支來同時追蹤多個不同版本的程式碼。您可以在開發人員仍在處理目前版本的程式碼時，將應用程式的更新版本安排在未來某個時間點。以主體為基礎的儲存庫可以使用特徵標記來完成此操作，但預設會內建於 Gitflow。

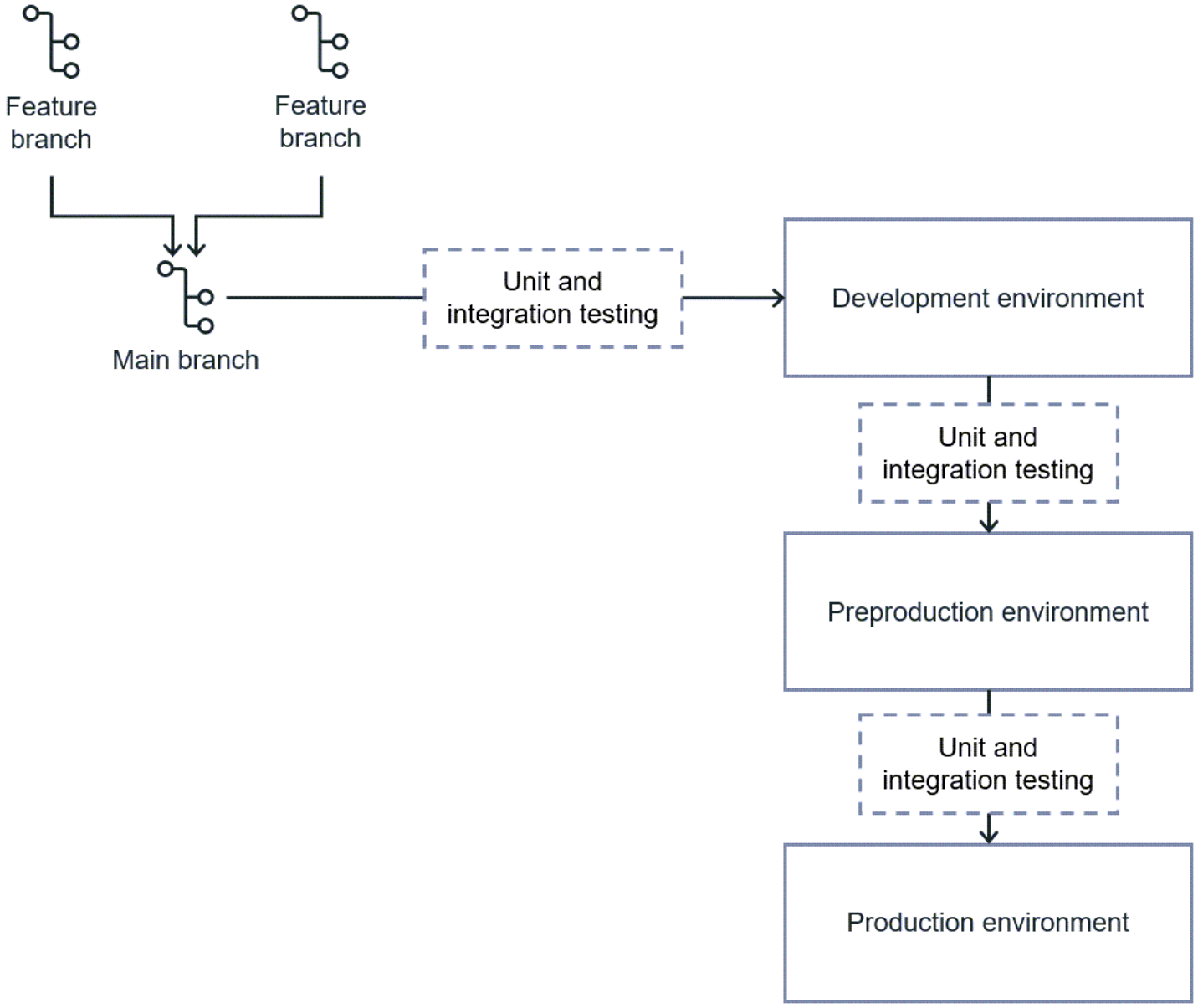


Gitflow 方法的一個結果是應用程式環境通常不同步。在標準 Gitflow 實作中，開發環境會反映程式碼的目前狀態，而生產前和生產環境仍會凍結在最新版本的程式碼基礎狀態上。

當瑕疵出現在生產環境中時，這會讓情況更為複雜，因為開發人員在未公開未發行的功能的情況下，無法將執行的程式碼基底合併到生產環境中。Gitflow 處理這種情況的方式是使用修正程式。從發行分支建立 Hotfix 分支，然後直接部署到上層環境。然後，Hotfix 分支會合併到開發分支中，以保持程式碼為最新狀態。

以中繼線為基礎的方法

下圖顯示以中繼線為基礎的工作流程。在以中繼為基礎的工作流程中，開發人員會在本機的特徵分支中建置和測試功能，然後將這些變更合併到主分支中。然後，主要分支會依序建置到開發環境、生產前環境和生產環境中。單元和整合測試會在每個環境之間進行。



使用此工作流程，所有環境都以相同的程式碼為基礎運作。不需要上層環境的 Hotfix 分支，因為您可以在主要分支中實作變更，而不會公開未發行的功能。主分支一律假設穩定、沒有瑕疵，並準備好發行。這可協助您將其整合為 CI/CD 管道的來源，其可透過管道中的所有環境自動測試和部署您的程式碼基礎。

以中繼線為基礎的方法的環境完整性優勢

如同許多開發人員所知，程式碼中的一個變更有時可以建立浮水印效果（美國科學家文章），其中看似無關的小偏差會引發鏈式反應，進而產生非預期的結果。開發人員接著必須完整調查，以探索根本原因。

當科學家進行實驗時，他們會將測試對象分成兩個群組：實驗群組和 控制群組。目的是讓實驗群組和控制群組完全相同，但實驗中測試的物件除外。當實驗群組中發生未發生在控制群組中的事件時，唯一原因可能是要測試的物件。

將部署中的變更視為實驗群組，並將每個環境視為不同的控制群組。只有在控制項與上一個環境相同時，在下一個環境中進行測試的結果才可靠。環境越偏離，在上層環境中探索瑕疵的機率就越高。換言之，如果程式碼變更在生產中會失敗，我們寧願它們先在 Beta 版中失敗，這樣他們就永遠不會進入生產環境。因此，應盡一切努力讓每個環境從最低測試環境到生產本身保持同步。這稱為環境完整性。

任何完整 CI/CD 程序的目標是盡早發現問題。使用以中繼線為基礎的方法來保持環境完整性，實際上可以消除對 Hotfix 的需求。在以中繼線為基礎的工作流程中，問題很少會先出現在生產環境中。

在 Gitflow 方法中，在 Hotfix 直接部署到上層環境之後，它會新增至開發分支。這會保留未來版本的修正。不過，Hotfix 是直接在應用程式的目前狀態之外開發和測試。即使 Hotfix 在生產中完美運作，當它與開發分支中較新的功能互動時，仍可能會出現問題。由於部署修正程式通常不理想，這會導致開發人員花額外的時間嘗試將修正程式改造到開發環境中。在許多情況下，這可能會導致重大的技術債務，並減少開發環境的整體穩定性。

當 環境中發生故障時，所有變更都會復原，讓環境回到其先前的狀態。程式碼庫的任何變更都應該從第一個階段再次開始管道。當生產環境中確實發生問題時，修正也應通過整個管道。與使用此方法所避免的問題相比，通過較低環境所需的額外時間通常可忽略。由於較低環境的整個目的是在錯誤到達生產環境之前攔截錯誤，因此透過 Gitflow 方法繞過這些環境是效率低且不必要的風險。

以中繼線為基礎的方法的發行優點

通常需要修正修正的問題之一是，在舊版工作流程中，開發人員正在處理的應用程式狀態可能包含數個尚未發行的功能，這些功能尚未在生產環境中運作。生產環境和開發環境只會在排程版本發生時同步，然後立即開始分歧，直到下一個排程版本為止。

在完全 CI/CD 程序中，可以進行排程版本。您可以使用功能旗標來延遲發佈程式碼至生產環境。不過，完全 CI/CD 程序可讓排程發行變得不必要的，從而允許更多彈性。畢竟，連續性是 CI/CD 中的關鍵字，這表示變更會在準備好時發佈。避免維護與較低測試環境幾乎不同步的個別發行環境。

如果管道不是完全 CI/CD，上、下環境之間的分歧通常發生在分支層級。開發人員會在開發分支中運作，並維護單獨的發行分支，只有在排程發行的時間才會更新。作為發行分支和開發分支分歧，可能會出現其他複雜性。

除了環境不同步之外，隨著開發人員在開發分支上工作，並習慣於遠遠比生產環境更早的應用程式狀態，每次出現問題時，他們都必須重新調整為生產狀態。開發分支的狀態可能是生產前的許多功能。當開發人員每天在該分支中工作時，很難記住什麼是和不向生產部門發佈。這會增加在修復其他錯誤的過程中引入新錯誤的風險。此結果似乎是修復的無限循環，可將時間軸和延遲功能版本延長數週、數月甚至數年。

以中繼線為基礎的方法的安全優勢

全 CI/CD 程序提供全自動化的單一事實來源部署方法。管道具有單一進入點。軟體更新會在一開始進入管道，並依原狀從一個環境傳遞到下一個環境。如果在管道的任何階段發現問題，修正問題的程式碼會變更，必須經過相同的程序，並在第一個階段開始。減少管道中的進入點也會減少漏洞可能引入管道的方式。

此外，由於進入點是生產環境最接近的可能點，因此大幅降低漏洞到達生產的可能性。如果您在完全 CI/CD 管道中實作手動核准程序，您仍然可以允許對是否將變更提升到下一個環境進行直接或直接決策。決策者不一定是部署變更的同一個人。這會區隔程式碼變更部署者的責任，以及這些變更的核准者。它也讓技術較少的組織領導者更能夠執行核准者的角色。

最後，單一進入點可協助您將對生產環境使用者介面 (UI) 主控台的寫入存取權限制為少數甚至零位使用者。透過減少可在 主控台中手動變更的使用者數量，您可以降低安全事件的風險。在舊版工作流程中，與 CI/CD 自動化方法相比，在生產環境中手動管理主控台的能力更為必要。這些手動變更較難追蹤、檢閱和測試。它們通常是為了節省時間而執行，但從長遠來看，它們會為專案增加重大的技術債務。

主控台安全問題不一定是由不法份子造成。主控台中發生的許多問題都是意外的。意外安全暴露非常常見，並導致零信任安全模型的增加。此模型儲存庫部分而言，當即使內部員工具有盡可能少的存取權，也稱為最低權限許可時，安全性意外的可能性也會降低。將所有程序限制為自動化管道，以保持生產環境的完整性，實際上可消除主控台相關安全問題的風險。

CI/CD 管道的 Litmus 測試

在化學中，貼圖紙是薄紙條，以特殊紅色或藍色染色處理，用於判斷物質的酸性。一種酸會變成藍色光蕩紙紅色，一種基底會變成紅色光蕩紙藍色，而中性物質完全不會影響紙張的顏色。

貼圖紙判斷酸性的方式是測量物質的 pH 值。如果 pH 值高於 8，則為酸性；如果低於 5，則為基本；如果介於 5 到 8 之間，則為中性。同樣地，CI/CD 貼圖測試可協助您測量管道的 CI/CD 層級。

測試您的管道是否為完全 CI/CD

1. 從 0 分開始。
2. 回答下列每個問題，並在每次回答是時為您的分數加上 1：
    - 我們的儲存庫是否每個儲存庫都有一個用來部署到環境的主分支？
    - 我們是否經常將程式碼遞交給主分支，並避免擁有長時間執行的功能分支？
    - 我們的管道是否具有單一進入點？ 換句話說，我們的管道是否確實從每個儲存庫提取程式碼一次？
    - 我們是否有多個部署環境？
    - 當管道未執行時，我們的上下環境是否通常同步？
    - 在部署之前，我們是否對程式碼執行測試？
    - 在提升到下一個環境之前，我們是否在環境上執行測試？
    - 我們的管道是否在故障後進行完整復原和結束？
    - 從故障復原時，我們的管道是否從第一步重新啟動？
    - 我們是否遵循相同的程序來修正生產環境中的錯誤，以將功能發佈至生產環境？
    - 我們是否使用某種形式的基礎設施做為程式碼 (IaC) 範本來部署程式碼？
3. 回答下列每個問題，並在每次回答否時為您的分數加上 1：
    - 我們是否曾經從主分支以外的分支直接部署到部署環境？
    - 我們是否曾經從任何分支直接部署到上層或生產環境？
    - 我們是否經常在上層環境中發現不存在的錯誤？
    - 在部署期間，我們是否曾經繞過較低的環境？
    - 我們是否等到排定的發行時間才能部署到生產環境？
    - 我們是否定期在生產環境的主控台進行更新？
    - 是否有任何手動部署步驟必須在生產環境的主控台中完成，才能完成部署？

- 是否有多個人員具有生產環境的寫入存取權？
- 是否超過五個人具有生產環境的寫入存取權？

4. 將您的分數除以 2。這是管道的 CI/CD 分數。
5. 將管道的 CI/CD 分數與下表進行比較，以判斷管道的 CI/CD 層級。

CI/CD 分數	CI/CD 層級
9.5 或更新版本	完全 CI/CD
8–9	主要是 CI/CD
5–7	Neutral
低於 5	非 CI/CD

如果您得分低於 8，我們建議您設定目標，逐步邁向下一個關卡。當實現該目標時，產品利益相關者應評估是否以及何時設定新目標。此練習的意圖不一定會擁護您管道的變更，而是讓您了解完整 CI/CD 部署程序的外觀，以及管道目前位於該頻譜上的位置。

CI/CD 管道的最佳實務

以下是完整 CI/CD 管道的最佳實務：

- 保護生產環境 – 由於使用 IaC 幾乎可以完成帳戶和環境維護所需的一切，因此請務必透過限制主控台和程式設計存取，盡一切努力保護生產環境。我們建議僅限制對少數或甚至零使用者的存取。當您透過 部署 IaC 時 AWS CloudFormation，使用者需要有限的許可。大多數許可會透過服務角色指派給 CloudFormation 服務。如需詳細資訊，請參閱 CloudFormation 文件中的服務角色和實作 最低權限許可的政策 AWS CloudFormation。
- 為每個環境建立個別帳戶 – 透過將個別帳戶指定給每個環境，您可以簡化部署程序，並在帳戶層級建立精細的存取控制。當多個環境共用資源時，它會降低環境的完整性，做為隔離單元。最好讓環境保持同步和獨特。這對生產環境來說更為重要，因為該帳戶中的所有內容都應視為生產資源。
- 限制個人身分識別資訊 (PII) 至生產環境 – 為安全及保護免於責任風險，請盡可能保護 PII。在較低的環境中，盡可能使用匿名或範例資料，而不是從生產環境複製潛在的敏感資料。
- 檢閱儲存庫中的程式碼 – 完整 CI/CD 程序會將管道的進入點減少至單一點，且應保護該單一點。因此，建議您在將特徵分支合併到主分支之前，先要求多個程式碼檢閱。這些程式碼檢閱可由任何合格的團隊成員執行，但至少應該有一個資深成員進行檢閱。程式碼應由檢閱者嚴格測試。畢竟，修正管道中問題的最佳方法是避免將問題導入管道。此外，在合併之前，解決任何檢閱者所做的所有評論也很重要。此解決方案可能只是解釋為什麼不需要變更，但解決所有評論是重要的額外檢查，有助於防止將問題引入管道。
- 進行小型且頻繁的合併 – 為了充分利用持續整合，建議也持續將本機變更推送到管道中。畢竟，如果本機環境也跟上，則讓開發環境保持同步會更有利。

如需 CI/CD 管道的更多最佳實務，請參閱實作持續整合和持續交付 AWS中的最佳實務摘要。

常見問答集

我的部署程序不是完全 CI/CD 的一些關鍵指標是什麼？

最常見的指標是，當有多個儲存庫分支代表管道中的不同環境時。完全 CI/CD 程序中的儲存庫使用以中繼為基礎的工作流程，其中一個分支充當該儲存庫部署的單一事實來源。如需詳細資訊，請參閱以中繼線為基礎的方法。其他指標包括簡單啟動或停止決策以外的手動部署步驟、使用修正程式和排程版本。

如果我想要使用完整 CI/CD 程序，但仍想要針對特定時間點排程特定功能的發行，該怎麼辦？

這通常使用功能旗標完成。在此程序中，部署仍會持續進行，但某些功能會在程式碼中使用條件式關閉來隱藏，直到需要釋出為止。

如果部署程序中的某些步驟無法自動化，該怎麼辦？

全 CI/CD 管道的其中一個目標是將手動程序的需求降至最低，但確實有可能需要手動程序的潛在使用案例。事實上，唯讀程序，例如諮詢應用程式日誌，通常可以在風險最低的生產環境中完成。不過，強烈建議您將生產中的手動寫入動作視為絕對最後一個補充。

如果我的技術人員比完全 CI/CD 程序更熟悉舊版工作流程，該怎麼辦？

技術人員通常會對重大變更產生抗拒，尤其是以前的最佳實務被較新的事物取代時。技術快速移動，並持續探索改善項目。雖然一定程度的懷疑對技術人員來說是良好的品質，但對他們來說，開放變革也同樣重要。請勿對懷疑員工太快移動，因為他們在實作系統變更之前需要管理變更。關鍵是防止懷疑者永遠保持靜態。

如果我的環境位於多個帳戶中，該怎麼辦？ 我是否仍然可以使用完整 CI/CD 程序？

是，事實上，建議為每個環境使用單獨的 帳戶。如需在不同帳戶中啟用階段的管道的詳細資訊，請參閱在 CodePipeline 中建立使用其他資源的管道 AWS 帳戶。

後續步驟

使用 CI/CD 管道的 Litmus 測試區段來評估組織中的 DevOps 程序。判斷您的程序是否為完全 CI/CD。如果不是，請決定他們是否需要改進，才能充分利用 CI/CD 部署的優勢。

如何知道何時完成？ 答案是，許多組織從未真正完成。他們會在路上某處停下來，在適合其使用案例的位置。雖然完全 CI/CD 管道是最佳案例，但它很大程度上取決於組織情況和決策背後的利益相關者。利益相關者必須決定 CI/CD 實作的哪個階段最適合其使用案例，以及如何將進度映射到下一個階段。

如需設計和建置 CI/CD 管道的詳細資訊，請參閱資源。

資源

AWS 文件和參考

- AWS 部署管道參考架構
- 什麼是持續交付？
- 在 上練習持續整合和持續交付 AWS (AWS 白皮書）
- 在 上設定 CI/CD 管道 AWS (AWS 實作教學課程）
- 在 中建立不支援 AWS 區域 的管道 AWS CodePipeline (AWS 規範性指導）
- 部署管道參考架構和參考實作 (AWS 部落格文章）

服務和工具

- CI/CD Litmus 測試
- AWS Cloud Development Kit (AWS CDK)
- AWS CloudFormation
- AWS CodePipeline

文件歷史紀錄

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

變更	描述	日期
初次出版	—	2023 年 8 月 25 日

AWS 規範性指引詞彙表

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

數字

7 R

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

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

A

ABAC

請參閱屬性型存取控制。

抽象服務

請參閱 受管服務。

ACID

請參閱原子性、一致性、隔離性、耐久性。

主動-主動式遷移

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

主動-被動式遷移

一種資料庫遷移方法，其中來源和目標資料庫保持同步，但只有來源資料庫處理來自連接應用程式的交易，同時將資料複寫至目標資料庫。目標資料庫在遷移期間不接受任何交易。

彙總函數

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

AI

請參閱人工智慧。

AIOps

請參閱人工智慧操作。

匿名化

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

反模式

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

應用程式控制

一種安全方法，僅允許使用核准的應用程式，以協助保護系統免受惡意軟體攻擊。

應用程式組合

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

人工智慧 (AI)

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

人工智慧操作 (AIOps)

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

非對稱加密

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

原子性、一致性、隔離性、耐久性 (ACID)

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

屬性型存取控制 (ABAC)

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

授權資料來源

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

可用區域

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

AWS 雲端採用架構 (AWS CAF)

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

AWS 工作負載資格架構 (AWS WQF)

一種工具，可評估資料庫遷移工作負載、建議遷移策略，並提供工作預估值。 AWS WQF 隨附於 AWS Schema Conversion Tool (AWS SCT)。它會分析資料庫結構描述和程式碼物件、應用程式程式碼、相依性和效能特性，並提供評估報告。

B

錯誤的機器人

旨在中斷或傷害個人或組織的機器人。

BCP

請參閱業務持續性規劃。

行為圖

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

大端序系統

首先儲存最高有效位元組的系統。另請參閱 Endianness。

二進制分類

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

Bloom 篩選條件

一種機率性、記憶體高效的資料結構，用於測試元素是否為集的成員。

藍/綠部署

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

機器人

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

殭屍網路

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

分支

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

碎片存取

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

棕地策略

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

緩衝快取

儲存最常存取資料的記憶體區域。

業務能力

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

業務連續性規劃 (BCP)

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

C

CAF

請參閱AWS 雲端採用架構。

Canary 部署

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

CCoE

請參閱 Cloud Center of Excellence。

CDC

請參閱變更資料擷取。

變更資料擷取 (CDC)

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

混沌工程

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

CI/CD

請參閱持續整合和持續交付。

分類

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

用戶端加密

在目標 AWS 服務 接收資料之前，在本機加密資料。

雲端卓越中心 (CCoE)

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

雲端運算

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

雲端操作模型

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

採用雲端階段

組織在遷移至 時通常會經歷的四個階段 AWS 雲端：

- 專案 – 執行一些與雲端相關的專案以進行概念驗證和學習用途
- 基礎 – 進行基礎投資以擴展雲端採用 (例如，建立登陸區域、定義 CCoE、建立營運模型)
- 遷移 – 遷移個別應用程式
- 重塑 – 優化產品和服務，並在雲端中創新

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

CMDB

請參閱組態管理資料庫。

程式碼儲存庫

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

冷快取

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

冷資料

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

電腦視覺 (CV)

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

組態偏離

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

組態管理資料庫 (CMDB)

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

一致性套件

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

持續整合和持續交付 (CI/CD)

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

CV

請參閱電腦視覺。

D

靜態資料

網路中靜止的資料，例如儲存中的資料。

資料分類

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

資料偏離

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

傳輸中的資料

在您的網路中主動移動的資料，例如在網路資源之間移動。

資料網格

架構架構，提供分散式、分散式資料擁有權與集中式管理。

資料最小化

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

資料周邊

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

資料預先處理

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

資料來源

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

資料主體

正在收集和處理其資料的個人。

資料倉儲

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

資料庫定義語言 (DDL)

用於建立或修改資料庫中資料表和物件之結構的陳述式或命令。

資料庫處理語言 (DML)

用於修改 (插入、更新和刪除) 資料庫中資訊的陳述式或命令。

DDL

請參閱資料庫定義語言。

深度整體

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

深度學習

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

深度防禦

這是一種資訊安全方法，其中一系列的安全機制和控制項會在整個電腦網路中精心分層，以保護網路和其中資料的機密性、完整性和可用性。當您在 上採用此策略時 AWS，您可以在 AWS

Organizations 結構的不同層新增多個控制項，以協助保護資源。例如，defense-in-depth方法可能會結合多重要素驗證、網路分割和加密。

委派的管理員

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

部署

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

開發環境

請參閱 環境。

偵測性控制

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

開發值串流映射 (DVSM)

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

數位分身

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

維度資料表

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

災難

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

災難復原 (DR)

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

DML

請參閱資料庫處理語言。

領域驅動的設計

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

DR

請參閱災難復原。

偏離偵測

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

DVSM

請參閱開發值串流映射。

E

EDA

請參閱探索性資料分析。

EDI

請參閱電子資料交換。

邊緣運算

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

電子資料交換 (EDI)

在組織之間自動交換商業文件。如需詳細資訊，請參閱什麼是電子資料交換。

加密

一種運算程序，可將人類可讀取的純文字資料轉換為加密文字。

加密金鑰

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

端序

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

端點

請參閱 服務端點。

端點服務

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

企業資源規劃 (ERP)

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

信封加密

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

環境

執行中應用程式的執行個體。以下是雲端運算中常見的環境類型：

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

epic

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

ERP

請參閱企業資源規劃。

探索性資料分析 (EDA)

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

F

事實資料表

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

快速失敗

一種使用頻繁和增量測試來縮短開發生命週期的理念。這是敏捷方法的關鍵部分。

故障隔離界限

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

功能分支

請參閱分支。

特徵

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

功能重要性

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

特徵轉換

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

少量擷取提示

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

FGAC

請參閱精細存取控制。

精細存取控制 (FGAC)

使用多個條件來允許或拒絕存取請求。

閃切遷移

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

FM

請參閱基礎模型。

基礎模型 (FM)

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

G

生成式 AI

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

地理封鎖

請參閱地理限制。

地理限制 (地理封鎖)

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

Gitflow 工作流程

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

黃金影像

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

綠地策略

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

防護機制

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

H

HA

請參閱高可用性。

異質資料庫遷移

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

高可用性 (HA)

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

歷史現代化

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

保留資料

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

異質資料庫遷移

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

熱資料

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

修補程序

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

超級護理期間

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

I

IaC

將基礎設施視為程式碼。

身分型政策

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

閒置應用程式

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

IIoT

請參閱工業物聯網。

不可變的基礎設施

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

傳入 (輸入) VPC

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

增量遷移

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

工業 4.0

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

基礎設施

應用程式環境中包含的所有資源和資產。

基礎設施即程式碼 (IaC)

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

工業物聯網 (IIoT)

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

檢查 VPC

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

物聯網 (IoT)

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

可解釋性

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

IoT

請參閱物聯網。

IT 資訊庫 (ITIL)

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

IT 服務管理 (ITSM)

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

ITIL

請參閱 IT 資訊庫。

ITSM

請參閱 IT 服務管理。

L

標籤型存取控制 (LBAC)

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

登陸區域

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

大型語言模型 (LLM)

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

大型遷移

遷移 300 部或更多伺服器。

LBAC

請參閱標籤型存取控制。

最低權限

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

隨即轉移

請參閱 7 個 R。

小端序系統

首先儲存最低有效位元組的系統。另請參閱 Endianness。

LLM

請參閱大型語言模型。

較低的環境

請參閱 環境。

M

機器學習 (ML)

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

主要分支

請參閱分支。

惡意軟體

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

受管服務

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

製造執行系統 (MES)

一種軟體系統，用於追蹤、監控、記錄和控制生產程序，將原物料轉換為現場成品。

MAP

請參閱遷移加速計劃。

機制

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

成員帳戶

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

製造執行系統

請參閱製造執行系統。

訊息佇列遙測傳輸 (MQTT)

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

微服務

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

微服務架構

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

Migration Acceleration Program (MAP)

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

大規模遷移

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

遷移工廠

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

遷移中繼資料

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

遷移模式

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

遷移組合評定 (MPA)

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

遷移準備程度評定 (MRA)

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

遷移策略

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

機器學習 (ML)

請參閱機器學習。

現代化

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

現代化準備程度評定

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

單一應用程式 (單一)

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

MPA

請參閱遷移產品組合評估。

MQTT

請參閱訊息佇列遙測傳輸。

多類別分類

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

可變基礎設施

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

O

OAC

請參閱原始存取控制。

OAI

請參閱原始存取身分。

OCM

請參閱組織變更管理。

離線遷移

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

OI

請參閱 操作整合。

OLA

請參閱操作層級協議。

線上遷移

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

OPC-UA

請參閱開啟程序通訊 - 統一架構。

開放程序通訊 - 統一架構 (OPC-UA)

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

操作水準協議 (OLA)

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

操作整備審查 (ORR)

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

操作技術 (OT)

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

操作整合 (OI)

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

組織追蹤

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

組織變更管理 (OCM)

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

原始存取控制 (OAC)

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

原始存取身分 (OAI)

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

ORR

請參閱操作整備審核。

OT

請參閱操作技術。

傳出 (輸出) VPC

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

P

許可界限

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

個人身分識別資訊 (PII)

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

PII

請參閱個人身分識別資訊。

手冊

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

PLC

請參閱可程式設計邏輯控制器。

PLM

請參閱產品生命週期管理。

政策

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

混合持久性

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

組合評定

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

述詞

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

述詞下推

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

預防性控制

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

委託人

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

設計隱私權

透過整個開發程序將隱私權納入考量的系統工程方法。

私有託管區域

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

主動控制

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

產品生命週期管理 (PLM)

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

生產環境

請參閱 環境。

可程式邏輯控制器 (PLC)

在製造中，高度可靠、可調整的電腦，可監控機器並自動化製造程序。

提示鏈結

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

擬匿名化

將資料集中的個人識別符取代為預留位置值的程序。假名化有助於保護個人隱私權。假名化資料仍被視為個人資料。

發佈/訂閱 (pub/sub)

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

Q

查詢計劃

一系列步驟，如指示，用於存取 SQL 關聯式資料庫系統中的資料。

查詢計劃迴歸

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

R

RACI 矩陣

請參閱負責、負責、諮詢、告知 (RACI)。

RAG

請參閱擷取增強生成。

勒索軟體

一種惡意軟體，旨在阻止對計算機系統或資料的存取，直到付款為止。

RASCI 矩陣

請參閱負責、負責、諮詢、告知 (RACI)。

RCAC

請參閱資料列和資料欄存取控制。

僅供讀取複本

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

重新架構師

請參閱 7 Rs。

復原點目標 (RPO)

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

復原時間目標 (RTO)

服務中斷和服務還原之間的可接受延遲上限。

重構

請參閱 7 個 R。

區域

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

迴歸

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

重新託管

請參閱 7 個 R。

版本

在部署程序中，它是將變更提升至生產環境的動作。

重新放置

請參閱 7 Rs。

Replatform

請參閱 7 Rs。

回購

請參閱 7 Rs。

彈性

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

資源型政策

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

負責者、當責者、事先諮詢者和事後告知者 (RACI) 矩陣

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

回應性控制

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

保留

請參閱 7 Rs。

淘汰

請參閱 7 個 R。

檢索增強生成 (RAG)

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

輪換

定期更新秘密的程序，讓攻擊者更難存取登入資料。

資料列和資料欄存取控制 (RCAC)

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

RPO

請參閱復原點目標。

RTO

請參閱復原時間目標。

執行手冊

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

S

SAML 2.0

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

SCADA

請參閱監督控制和資料擷取。

SCP

請參閱服務控制政策。

秘密

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

依設計的安全性

透過整個開發程序將安全性納入考量的系統工程方法。

安全控制

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

安全強化

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

安全資訊與事件管理 (SIEM) 系統

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

安全回應自動化

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

伺服器端加密

由 AWS 服務 接收資料的 在其目的地加密資料。

服務控制政策 (SCP)

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

服務端點

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

服務水準協議 (SLA)

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

服務層級指標 (SLI)

服務效能方面的測量，例如其錯誤率、可用性或輸送量。

服務層級目標 (SLO)

代表服務運作狀態的目標指標，由服務層級指標測量。

共同責任模式

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

SIEM

請參閱安全資訊和事件管理系統。

單點故障 (SPOF)

應用程式的單一關鍵元件故障，可能會中斷系統。

SLA

請參閱服務層級協議。

SLI

請參閱服務層級指標。

SLO

請參閱服務層級目標。

先拆分後播種模型

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

SPOF

請參閱單一故障點。

星狀結構描述

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

Strangler Fig 模式

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

子網

您 VPC 中的 IP 地址範圍。子網必須位於單一可用區域。

監控控制和資料擷取 (SCADA)

在製造中，使用硬體和軟體來監控實體資產和生產操作的系統。

對稱加密

使用相同金鑰來加密及解密資料的加密演算法。

合成測試

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

系統提示

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

T

標籤

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

目標變數

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

任務清單

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

測試環境

請參閱 環境。

訓練

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

傳輸閘道

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

主幹型工作流程

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

受信任的存取權

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

調校

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

雙比薩團隊

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

U

不確定性

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

未區分的任務

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

較高的環境

請參閱 環境。

V

清空

一種資料庫維護操作，涉及增量更新後的清理工作，以回收儲存並提升效能。

版本控制

追蹤變更的程序和工具，例如儲存庫中原始程式碼的變更。

VPC 對等互連

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

漏洞

危及系統安全性的軟體或硬體瑕疵。

W

暖快取

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

暖資料

不常存取的資料。查詢這類資料時，通常可接受中等緩慢的查詢。

視窗函數

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

工作負載

提供商業價值的資源和程式碼集合，例如面向客戶的應用程式或後端流程。

工作串流

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

WORM

請參閱寫入一次，讀取許多。

WQF

請參閱AWS 工作負載資格架構。

寫入一次，讀取許多 (WORM)

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

Z

零時差入侵

利用零時差漏洞的攻擊，通常是惡意軟體。

零時差漏洞

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

零鏡頭提示

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

殭屍應用程式

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

本文為英文版的機器翻譯版本，如內容有任何歧義或不一致之處，概以英文版為準。