

AWS Academy Cloud Architecting

# 模組 12：構建解耦架構



歡迎學習模組 12：構建解耦架構。

## 小節目錄

1. 架構需求
2. 解耦架構
3. 使用 Amazon Simple Queue Service (Amazon SQS) 進行解耦
4. 使用 Amazon Simple Notification Service (Amazon SNS) 進行解耦
5. 使用 Amazon MQ 在雲應用程式和本地之間發送消息



本模組包含以下章節：

1. 架構需求
2. 解耦架構
3. 使用 Amazon Simple Queue Service (Amazon SQS) 進行解耦
4. 使用 Amazon Simple Notification Service (Amazon SNS) 進行解耦
5. 使用 Amazon MQ 在雲應用程式和本地之間發送消息

在本模組結束時，您需要完成一個知識測驗，以測試您對本模組中涵蓋的關鍵概念的理解程度。

## 模組目標



學完本模組後，您應該能夠：

- 區分緊耦合架構和松耦合架構
- 確定 Amazon SQS 的工作原理以及使用時間
- 確定 Amazon SNS 的工作原理以及使用時間
- 描述 Amazon MQ

學完本模組後，您應該能夠：

- 區分緊耦合架構和松耦合架構
- 確定 Amazon SQS 的工作原理以及使用時間
- 確定 Amazon SNS 的工作原理以及使用時間
- 描述 Amazon MQ

模組 12：構建解耦架構

## 第 1 節：架構需求



介紹第 1 節：架構需求。

## 咖啡館業務要求



咖啡館的架構現在支援成千上萬的用戶。但是，很難對應用程式的一個層進行更改而不影響其他層。



咖啡館的架構現在支援成千上萬的用戶。但是，咖啡館的系統耦合過於緊密。很難對應用程式的一個層進行更改而不影響其他層。例如，每日訂購報告是從同一個 Web 伺服器生成的，該伺服器也為客戶提供咖啡館的網站。

此外，Frank 提到，他沒有收到週五 17:00 的定期報告。經過一系列調查，Sofia 和 Nikhil 發現，計劃維護時段與報告系統嘗試生成報告的時間相吻合。

他們與 Olivia 交談，Olivia 建議他們解耦架構。通過將報告流程移動到另一個系統，即使 Web 伺服器暫時不可用，報告資料也不會丟失。此外，生成報告的請求將排隊並進行處理。

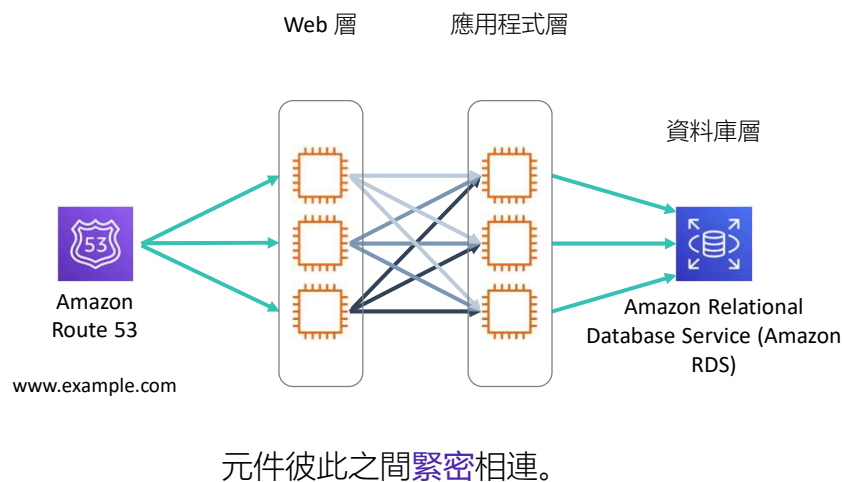
模組 12：構建解耦架構

## 第 2 節：解耦架構



介紹第 2 節：解耦架構。

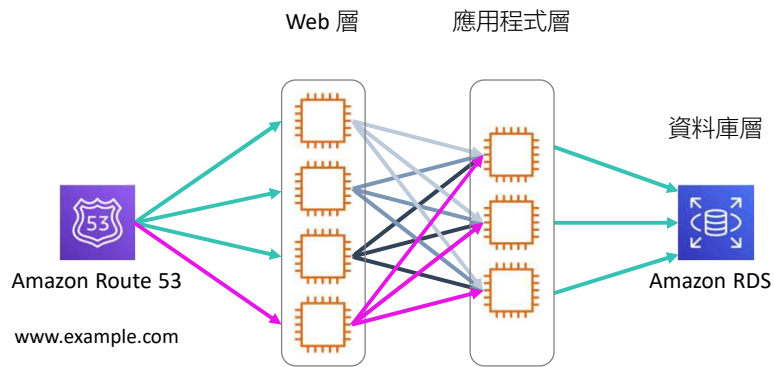
# 緊耦合的架構



傳統基礎設施具有緊密集成的元件鏈。每個元件都有特定的目的。當一個元件出現故障時，系統中斷可能是致命的。

考慮這個處理客戶訂單的 Web 應用程式的三層架構示例。Web 層中的每個實例都與應用程式層中的每個實例進行通信。應用程式層中的每個實例都將資料保存到後端資料庫中。Web 或應用程式層中的實例故障還會導致某些客戶訂單資料的持久性失敗。

## 緊耦合的架構阻礙擴展



添加資源會增加複雜性並阻礙擴展。

在緊耦合的系統中，擴展也會受到阻礙：如果在一個層添加伺服器，則還必須將其連接到每個連接層中的伺服器。繼續三層架構示例，添加到 Web 層的實例必須連接到應用程式層中的每個實例。



# 系統耦合的形式



**應用程式級耦合：**  
與管理傳入和傳出依賴項相關

**平臺耦合：**  
與異構系統組件的互通性相關

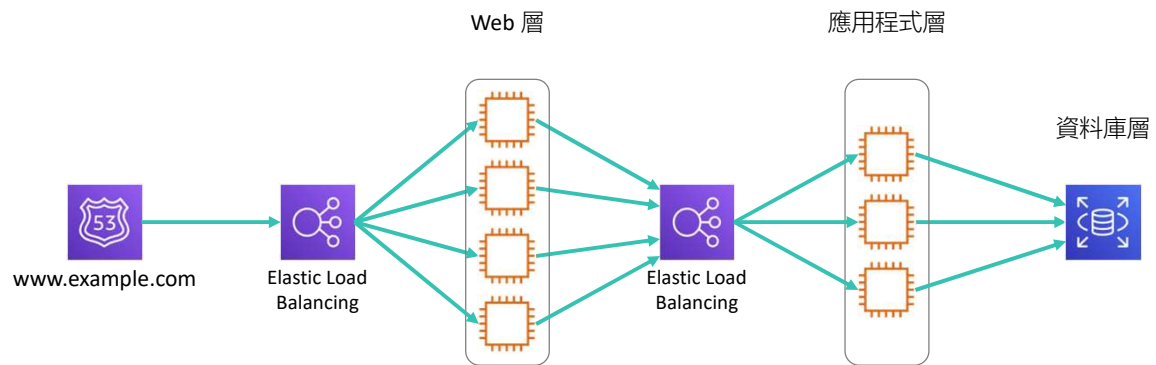
**空間耦合：**  
與在網路拓撲級別或協定級別管理元件相關

**時間（運行時）耦合：**  
指系統元件在執行同步阻塞操作時執行有意義工作的能力

系統可以通過多種方式進行耦合，在雲中構建分散式應用程式時，應考慮這些因素：

- 應用程式級耦合與管理傳入和傳出依賴項相關
- 平臺耦合與異構系統元件的互通性相關
- 空間耦合與在網路拓撲級別或協定級別管理元件相關
- 時間（或運行時）耦合指系統元件在執行同步阻塞操作時執行有意義工作的能力

# 松耦合架構

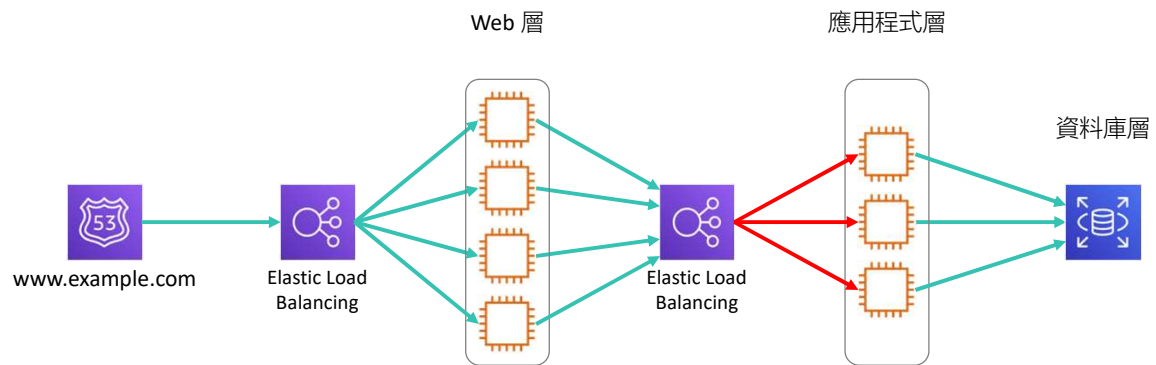


您可以使用託管解決方案作為各層之間的中間件。

為了說明確保應用程式隨著負載的增加而擴展，並確保系統不存在瓶頸或單點故障，請實施松耦合。通過松耦合，您可以通過將託管解決方案作為系統各層之間的中介軟體來減少系統中的依賴項。這樣一來，中介軟體會自動處理元件或層發生的故障和擴展。

再次考慮三層 Web 應用程式架構。您可以通過在 Web 和應用程式層前添加負載等化器來實現松耦合，以便在每層分配流量。如果一台伺服器出現故障，負載等化器將自動將流量定向到運行良好的實例。

## 松耦合架構中的注意事項



在訂單處理工作流的業務使用案例中，一個潛在的漏洞點是將訂單資料保存到資料庫。如果業務要求訂單資料必須保留在資料庫中，那麼各種情況（例如潛在的鎖死、競爭條件或網路問題）都可能會導致訂單持久性失敗。在這種情況下，訂單將丟失，您無法恢復。

## 第 2 節要點



- 緊耦合的系統具有緊密集成的元件鏈，會阻礙擴展
- 您可以使用託管解決方案（例如 Elastic Load Balancing）各層之間的中介軟體，在系統中實施松耦合

本模組中這節內容的要點包括：

- 緊耦合的系統具有緊密集成的元件鏈，會阻礙擴展
- 您可以使用託管解決方案（例如 Elastic Load Balancing）各層之間的中介軟體，在系統中實施松耦合

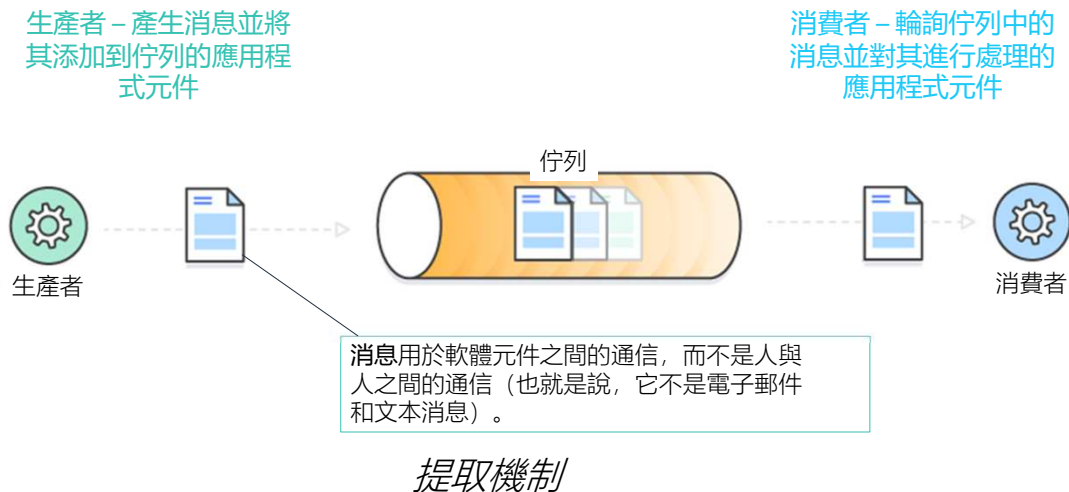
模組 12：構建解耦架構

## 第 3 節：使用 Amazon SQS 進行解耦



介紹第 3 節：使用 Amazon SQS 進行解耦。

# 解耦架構的訊息佇列



訊息佇列是幫助您實現解耦架構的另一個組件。訊息佇列可為這些分散式應用程式提供通信和協調。

訊息佇列是等待處理的消息的臨時儲存庫。這些消息通常較小，可以是請求、回復、錯誤消息或明文資訊等。消息示例包括客戶記錄、產品訂單、發票、患者記錄等。

要發送消息時，一個名為“生產者”的元件會將消息添加到佇列。消息將存儲在佇列中，直至名為“消費者”的另一個元件檢索並處理該消息。



Amazon Simple  
Queue Service  
(Amazon SQS)

- 完全託管的**消息佇列**服務
- 使用**提取**機制
- 加密和存儲消息，直至消息得到處理或被刪除
- 用作生產者和消費者之間的緩衝區

Amazon Simple Queue Service (Amazon SQS) 是一項完全託管的訊息佇列服務，使您能夠解耦應用程式元件，以便它們獨立運行。它可以讓 Web 服務應用程式對一個應用程式元件生成、由另一個元件使用的消息進行排隊。

佇列是等待處理的消息的臨時儲存庫。存儲消息，直至消息得到處理或被刪除（從 1 到 14 天；默認為 4 天）。消息可包含最多 256KB 的任何格式的文本。Amazon SQS 可以大規模處理工作，每天處理數十億條消息。它將所有訊息佇列和消息存儲在具有多個冗余可用區的單個高度可用的 AWS 區域中。任何一台電腦、網路或可用區出現故障都不會使消息無法訪問。可以同時發送和讀取消息。

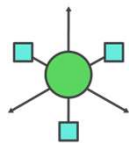
您可以安全地以匿名方式或與特定 AWS 帳戶共用 Amazon SQS 佇列。您也可以通過 IP 位址和時間限制佇列共用。SQS 佇列中的消息通過 AWS Key Management Service (AWS KMS) 中託管的金鑰使用伺服器端加密 (SSE) 進行加密。Amazon SQS 僅在將消息發送給授權消費者時才會對消息進行解密。

Amazon SQS 支援多個生產者和消費者與同一佇列進行交互。Amazon SQS 可與多種 AWS 服務一起使用，包括：Amazon Elastic Compute Cloud (Amazon EC2)、Amazon Simple Storage Service (Amazon S3)、Amazon Elastic Container Service (Amazon ECS)、AWS Lambda 和 Amazon DynamoDB。

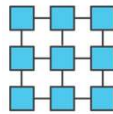
# 使用 Amazon SQS 實現松耦合



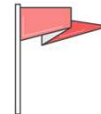
使用 Amazon SQS，您可以：



使用**非同步處理**  
快速獲取每個步  
驟的回應



通過增加作業實  
例的數量來處理  
**性能和服務要求**



輕鬆**從失敗的步驟中**  
**恢復**（因為消息將保  
留在佇列中）

通過 Amazon SQS，您可以在架構中實現松耦合。

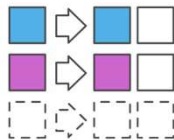
- 它執行非同步處理，以便您可以快速獲得每個步驟的回應
- 它可以通過增加作業實例的數量來應對性能和服務要求
- 您的應用程式能輕鬆從失敗的步驟中恢復（因為消息將保留在佇列中）



## Amazon SQS 一般使用案例



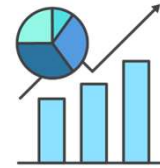
工作隊列



緩衝批次  
處理操作



請求卸載

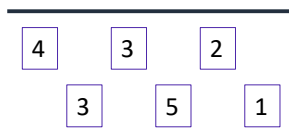


觸發 Amazon  
EC2 Auto Scaling

Amazon SQS 有以下多種使用方式：

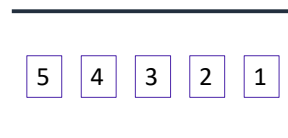
- 工作隊列 – 對可能無法同時全部處理相同工作量的分散式應用程式的元件進行解耦。
- 緩沖批處理操作 – 為您的架構添加可擴展性和可靠性，並可消除臨時卷峰值，而不會丟失消息或增加延遲。
- 請求卸載 – 將請求排入佇列，從互動式請求路徑中移出緩慢操作。
- 觸發 Amazon EC2 Auto Scaling – 使用 SQS 佇列說明確定應用程式的負載。當它們與 Amazon EC2 Auto Scaling 結合使用時，您可以根據流量規模向外或向內擴展 EC2 實例的數量。

## 標準佇列



- 至少一次傳遞
- 最大努力排序
- 接近無限輸送量

## 先進先出 (FIFO) 佇列



- 先進先出交付
- 恰好一次處理
- 高輸送量

有兩種 SQS 佇列類型：

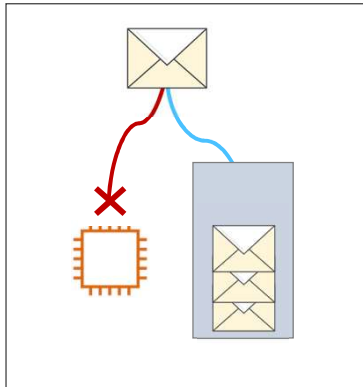
### 標準佇列提供：

- 至少一次傳遞 – 消息至少傳送一次，但偶爾會傳送消息的多個副本。
- 最大努力排序 – 消息偶爾可能按不同於其發送時的順序傳遞。
- 接近無限輸送量 – 標準佇列支援每個 API 操作幾乎無限數量的每秒事務數 (TPS)。

### 先進先出 (FIFO) 佇列：

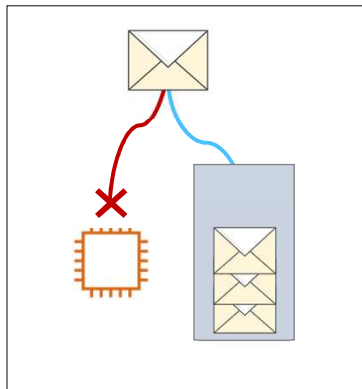
- 旨在確保按照消息的發送和接收順序對消息進行嚴格一次處理。
- 提供高輸送量 – FIFO 佇列每秒支援最多 300 條消息（每秒 300 次發送、接收或刪除操作）。如果您每次操作批量處理 10 條消息（最多），那麼 FIFO 佇列每秒最多可支援 3000 條消息。

## 無效信件佇列支持

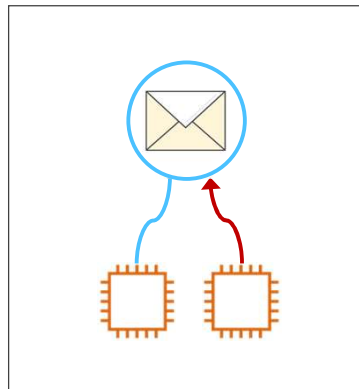


Amazon SQS 提供無效信件佇列支援。無效信件佇列(DLQ) 指的是無法處理的訊息佇列。它會在處理嘗試次數達到最大值後接收消息。DLQ 跟任何其他 SQS 佇列一樣：可向其發送消息和從其中接收消息。您可以使用 Amazon SQS API 和控制台創建 DLQ。

無效信件佇列支持



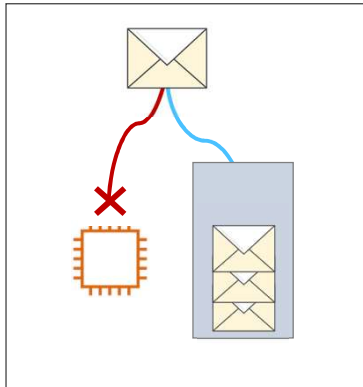
可見性超時



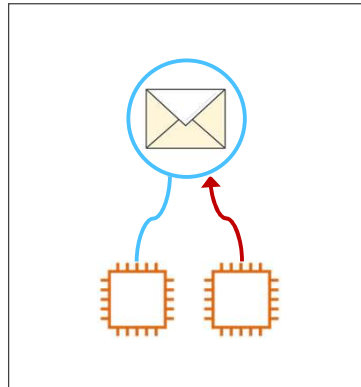
Amazon SQS 的另一個功能是可見性超時。*可見性超時*是 Amazon SQS 阻止其他消費者接收並處理相同消息的一段時間。超時有助於確保作業不會多次處理而導致重複。在可見性超時期間，收到消息的元件會先處理消息，然後將其從佇列中刪除。消息的默認可見性超時為 30 秒，最長為 12 小時。

如果消費者處理失敗，沒有在可見性超時到期之前刪除消息，則該消息將對其他消費者可見，並且可以進行再次處理。通常情況下，您應將可見性超時設置為應用程式處理消息並將其從佇列中刪除所花費的最長時間。

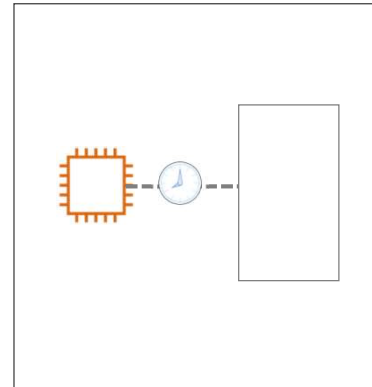
無效信件佇列支持



可見性超時



長輪詢



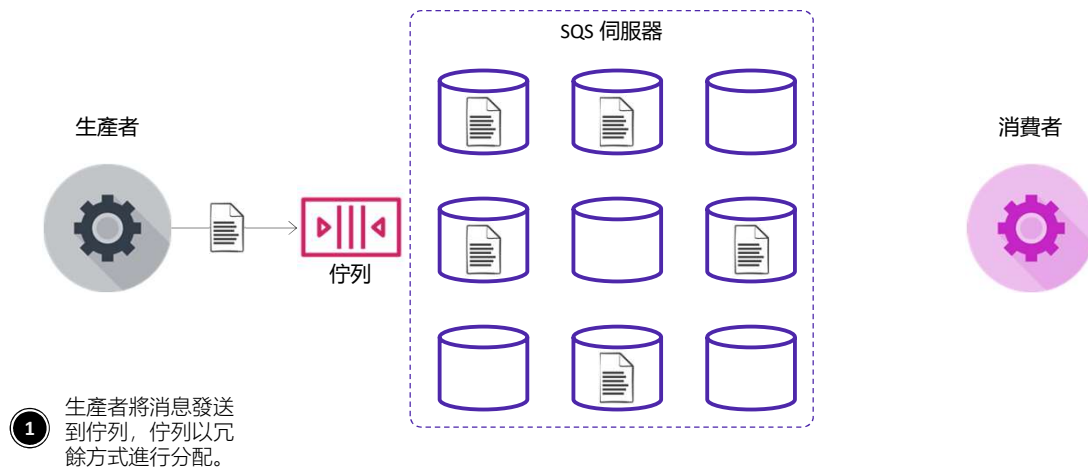
最後，Amazon SQS 支持短輪詢和長輪詢以從 SQS 佇列檢索消息。預設情況下，佇列使用短輪詢。

短輪詢查詢只能在回應中包含用於查找消息的伺服器的子集（基於加權隨機分佈）。即使查詢未發現消息，Amazon SQS 也會立即發送回應。

相比之下，長輪詢會在所有伺服器中查詢消息。Amazon SQS 會在達到收集回應的最大消息數或輪詢等待時間到期後發送回應。

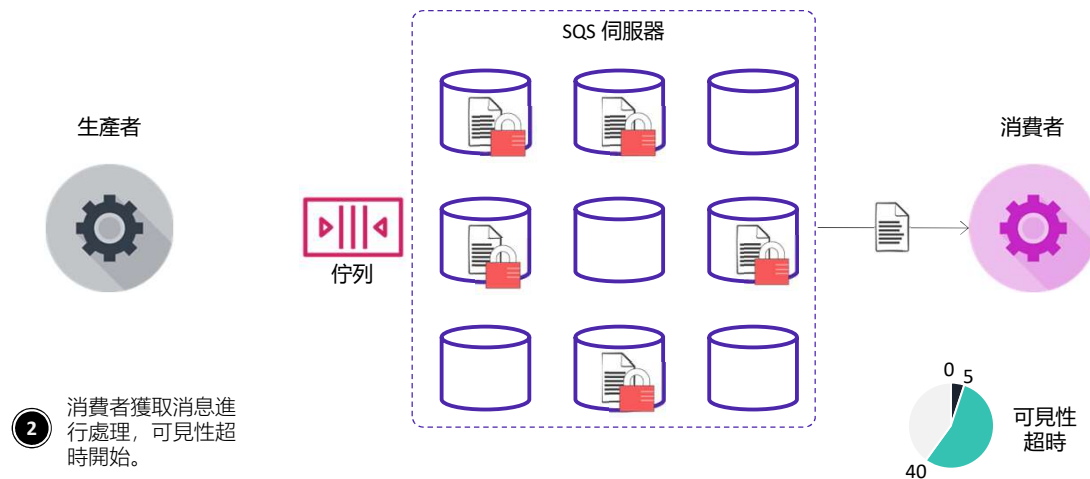
如果要在消息可用後立即從 SQS 佇列中檢索消息，那麼長輪詢是一種成本較低的方式。因為您可以減少接收空消息的次數，所以長輪詢可以降低 Amazon SQS 的使用成本。

# Amazon SQS 消息生命週期：創建



以下場景可以說明 sqs 佇列中消息的生命週期。首先，有一個生產者向佇列發送一條消息，消息以冗餘方式跨 sqs 伺服器分佈。

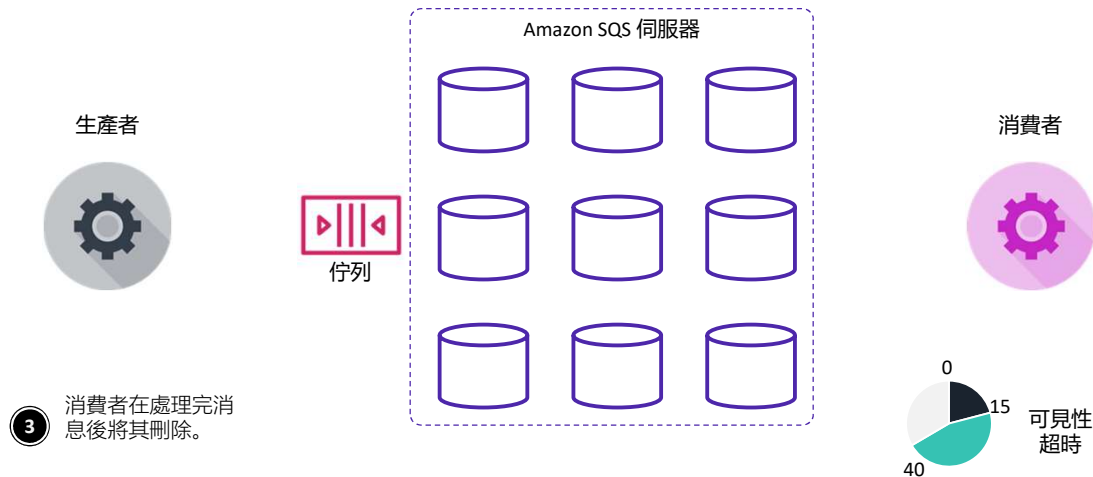
## Amazon SQS 消息生命週期：流程



當消費者準備好處理消息時，會從佇列中檢索消息。消息接受處理期間仍將保留在佇列中。

在可見性超時期間，其他消費者無法處理該消息。在此示例中，可見性超時為 40 秒。

# Amazon SQS 消息生命週期：刪除

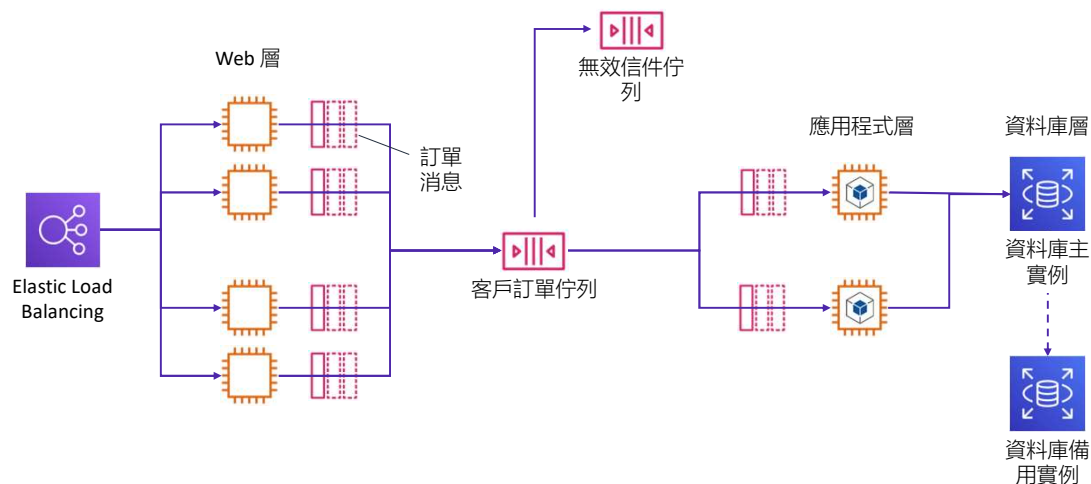


處理消息後，消費者將從佇列中刪除該消息。此操作可阻止消息在可見性超時過期後被再次接收和處理。

Amazon SQS 不會自動刪除消息。由於 Amazon SQS 是分散式系統，因此無法保證消費者實際收到消息（例如，由於連接問題或消費者應用程式中的問題而導致實際沒有收到消息）。因此，消費者在接收和處理消息後必須將其從佇列中刪除。



## 解耦示例：使用 Amazon SQS



再次考慮您在上一節中學到的訂購處理應用程式。您可以通過引入 sqs 佇列來解耦此應用程式的架構。您可以使用佇列將處理邏輯隔離到其元件中，以便在一個獨立于 Web 應用程式的進程中運行它。

這種設計使系統能夠更好地應對流量激增。此外，它還使工作只能在管理成本所需的速度下進行。

此外，您現在有了將訂單作為消息持久存儲的機制（佇列充當臨時資料庫）。您還將資料庫中的事務範圍進一步向下移動到堆疊。如果發生應用程式異常或事務失敗，此設計有助於確保將訂單處理重試或重定向到無效信件佇列，以便日後進行重新處理。

有關此使用案例的更多資訊，請參閱[此 AWS 計算博客文章](#)。

## 訊息佇列使用案例



✓ 服務到服務通信

✓ 非同步工作項






✓ 狀態更改通知

這些常見使用案例演示了訊息佇列最適合的場景：

- **服務到服務通信** – 例如，假設前端網站必須在後端客戶關係管理 (CRM) 服務中更新客戶的傳遞位址。您可以讓前端網站的代碼向 SQS 佇列發送消息，然後讓後端 CRM 服務使用這些消息。
- **異步工作專案** – 例如，假設酒店預訂系統必須取消預留，這是一個需要很長時間的過程。您可以向 SQS 佇列發送消息，讓該酒店預訂系統使用這些消息並執行非同步取消。
- **狀態變更通知** – 例如，假設您有管理某些資源的服務。您希望其他服務能夠接收有關該資源更改的更新。庫存系統可能會在某個物品庫存量較低且需要訂購時發佈通知。

## 訊息佇列使用案例



 服務到服務通信	 選擇特定消息
 非同步工作項	 大型消息
 狀態更改通知	

此外，瞭解特定技術何時不適合您的使用案例，也很重要。消息收發有其自己的一組常見反模式。

- **選擇特定消息** – 您可能希望有選擇地接收來自與特定屬性集或與臨時邏輯查詢匹配的佇列中的消息。例如，服務請求了一條具有特定屬性的消息，因為它包含對服務發出的另一消息的回應。在這種情況下，佇列中可能存在沒有人輪詢且沒有人使用的消息。
- **大型消息** – 大多數消息收發協定和實施都最適用於合理大小的消息（數十或數百 KB）。隨著消息大小的增加，您最好使用專用存儲系統（例如 Amazon S3），並在消息本身中傳遞對該存儲中物件的引用。

## 第 3 節要點



- Amazon SQS 是一項完全託管的訊息佇列服務，使您能夠解耦應用程式元件，以便它們獨立運行。
- Amazon SQS 支援標準佇列和 FIFO 佇列。
- 生產者向佇列發送消息。在可見性超時期間，消費者可以處理和刪除消息。
- 無法處理的消息可以發送到無效信件佇列。
- 長輪詢是從 SQS 佇列中檢索大量消息的方法。

本模組中這節內容的要點包括：

- Amazon SQS 是一項完全託管的訊息佇列服務，使您能夠解耦應用程式元件，以便它們獨立運行。
- Amazon SQS 支援標準佇列和 FIFO 佇列。
- 生產者向佇列發送消息。在可見性超時期間，消費者可以處理和刪除消息。
- 無法處理的消息可以發送到無效信件佇列。
- 長輪詢是從 SQS 佇列中檢索大量消息的方法。

模組 12：構建解耦架構

## 第 4 節：使用 Amazon SNS 進行解耦

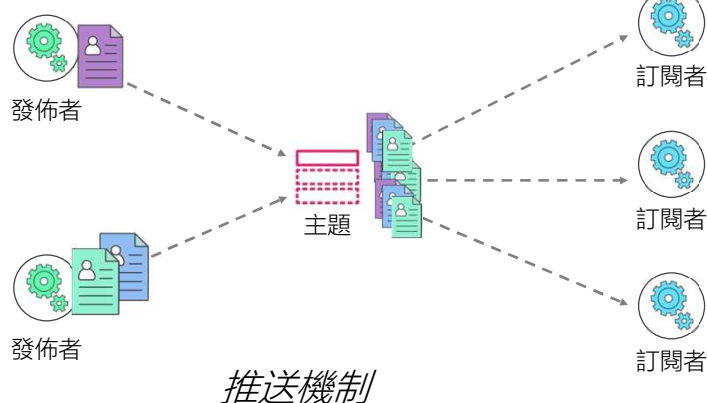


介紹第 4 節：使用 Amazon SNS 進行解耦。

# 發佈/訂閱消息收發

發佈者 – 將消息推送到主題的元件

訂閱者 – 訂閱主題的元件



在現代雲架構中，應用程式被解耦為多個規模較小且更易於開發、部署和維護的獨立構建塊。發佈/訂閱 (pub/sub) 消息收發可以為這些分散式應用程式提供即時事件通知。

發佈/訂閱模式讓消息能夠非同步廣播到系統中的不同部分。消息主題提供了廣播非同步事件通知的羽量級機制。它還提供了軟體元件能夠連接到主題的終端節點，以便它們能夠發送和接收這些消息。

在廣播消息時，一個叫做“發佈者”的元件會將消息推送到主題。與在消息被檢索前批量處理消息的訊息佇列不同的是，消息主題無需或使用極少訊息佇列即可傳輸消息，並將消息立即推送給所有訂閱者。除非設置了消息篩選策略，否則訂閱者將收到廣播的每條消息。訂閱者的示例包括 Web 伺服器、電子郵件地址、Amazon SQS 佇列和 AWS Lambda 函數。

消息主題的訂閱者通常執行不同的函數，並可以同時對消息執行不同的操作。發佈者無需知道誰在使用廣播的資訊，而訂閱者也無需知道消息來自哪裡。這種消息收發模式與訊息佇列稍有不同，在訊息佇列中，發送消息的元件通常知道發送的目的地。

在“發佈/訂閱” (pub-sub) 消息收發範式中，使用推送機制將通知傳輸到用戶端，無需定期檢查或輪詢新資訊和更新。



Amazon Simple  
Notification Service  
(Amazon SNS)

- 是具有高可用性、持久性、安全性和完全託管的**發布/訂閱消息收發**服務
- 使用**推送**機制
- 支援使用客戶主金鑰 (CMK) **加密主題**

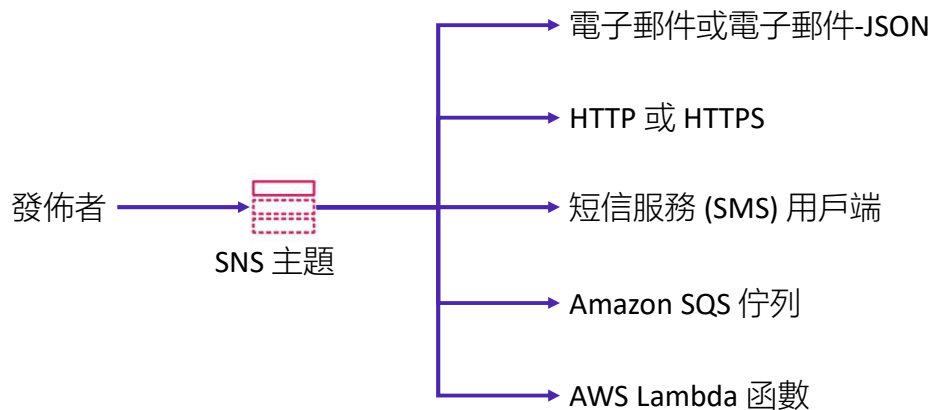
Amazon Simple Notification Service (Amazon SNS) 是一種 Web 服務，您可以輕鬆地在雲中設置、操作和發送通知。該服務遵循“發佈/訂閱” (*pub-sub*) 消息收發範例，使用“推送”機制將通知傳遞給用戶端。Amazon SNS 旨在滿足最大型和要求最嚴格的應用程式的需求，使應用程式可以隨時發佈無限量的消息。

使用 Amazon SNS 時，您可以創建**主題**並設置策略來限制誰可以發佈或訂閱該主題。發佈者會發送消息至他們創建的主題或他們有權發佈的主題。然後，Amazon SNS 會將主題與該主題的訂閱者清單進行匹配，並將消息傳遞給每個訂閱者。每個主題都有一個唯一的名稱，為發佈者和訂閱者標識 Amazon SNS 終端節點，以便他們發佈消息和訂閱註冊通知。訂閱者會收到發佈至他們所訂閱主題的所有消息，且一個主題的所有訂閱者收到的消息都相同。

Amazon SNS 支援加密主題。當您將消息發佈到加密主題後，Amazon SNS 會使用客戶主金鑰 (CMK) 來加密您的消息。CMK 是 AWS KMS 中的主要資源。Amazon SNS 支援客戶託管的 CMK，也支援 AWS 管理的 CMK。

在 Amazon SNS 收到您的消息後，它們會使用 256 位高級加密標準伽羅瓦/計數器模式 (AES-GCM) 演算法進行加密。加密後的消息會以冗餘方式存儲在多個伺服器和資料中心之間，並跨多個可用區以實現持久性。消息在傳送到已訂閱的終端節點之前會進行解密。如需瞭解有關加密發佈到 Amazon SNS 的消息的資訊，請閱讀此 [AWS 計算博客文章](#)。

## 支援的傳輸協定



Amazon SNS 支援以下傳輸協定來傳送消息：

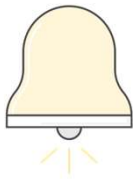
- **電子郵件或電子郵件 JSON** – 消息以電子郵件的形式發送到註冊地址。電子郵件 JSON 以 JavaScript 物件標記法 (JSON) 物件的形式發送通知，電子郵件則發送基於文本的電子郵件。
- **超文本傳輸協定 (HTTP) 或安全 HTTP (HTTPS)** – 在訂閱註冊期間，訂閱者將指定 URL。消息通過 HTTP POST 請求傳送到指定的 URL。
- **短信服務 (SMS)** – 消息以 SMS 文本消息的形式發送到註冊的電話號碼。
- **Amazon SQS 佇列** – 使用者將 SQS 標準佇列指定為終端節點。Amazon SNS 會將通知消息排隊到指定佇列。FIFO 佇列目前不受支援。
- **AWS Lambda 函數** – 消息傳遞到 AWS Lambda 函數，用於處理消息自訂項、維持消息持久性或與其他 AWS 服務進行通信。



# Amazon SNS 的一般使用案例



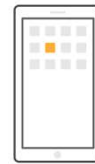
應用程式和系統警報



推送電子郵件和文本消息



移動推送通知



有多種方法可以使用 Amazon SNS：

- **應用程式和系統警報** – 您可以使用 Amazon SNS 在事件發生時接收即時通知，例如對 Auto Scaling 組進行更改。
- **推送電子郵件和文本消息** – 您可以使用 Amazon SNS 通過電子郵件或 SMS 將目標新聞標題推送給訂閱者。
- **移動推送通知** – 您可以使用 Amazon SNS 向應用程式發送通知，表明有可用的更新。通知消息可以包含下載和安裝更新的連結。

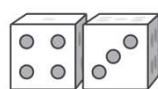
## Amazon SNS 考慮因素



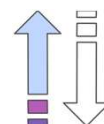
單個發佈  
消息



沒有召回  
選項



無法保證順  
序和傳遞

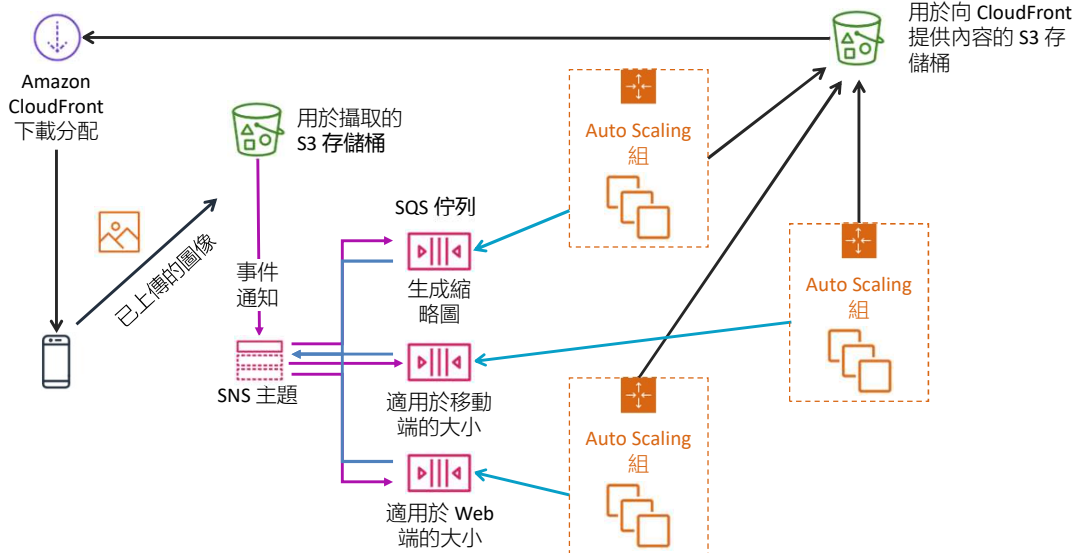


為每個傳輸協定  
重試策略

如果您計畫使用 Amazon SNS，請考慮以下幾點：

- 每個通知消息都包含一條已發佈的消息。
- 消息成功傳遞後便無法再重新調用。
- Amazon SNS 會嘗試按發佈到主題中的順序來傳遞發佈者的消息。但是，網路問題可能會導致訂閱者端的消息順序錯亂。
- Amazon SNS 為每個傳輸協議定義了一個傳輸策略。傳輸策略定義了在發生伺服器端錯誤時（即當託管已訂閱終端節點的系統變得不可用時），Amazon SNS 如何重試消息傳輸。如果第一次嘗試無法成功傳遞消息，則 Amazon SNS 將使用四階段重試策略：
  1. 在兩次嘗試之間沒有延遲地重試；
  2. 在兩次嘗試之間進行延遲最小的重試；
  3. 根據退避模型進行重試；
  4. 在兩次嘗試之間進行延遲最大的重試；當消息傳遞重試策略用盡時，Amazon SNS 可以將消息移動至 DLQ。

## 解耦示例：將 Amazon S3 與 Amazon SNS 結合使用



通過 Amazon SNS，您可以使用主題將消息發佈者與訂閱者解耦，將消息同時群發給多個收件人，並消除應用程式中的輪詢。

您可以使用 Amazon SNS 在單個帳戶中發送消息或向不同帳戶中的資源發送消息。

AWS 服務（例如 Amazon EC2、Amazon S3 和 Amazon CloudWatch）可以將消息發佈到您的 SNS 主題，以觸發事件驅動型計算和工作流。在此示例中，將映射上傳到 S3 存儲桶後，Amazon S3 會觸發事件通知，此操作會自動將消息發送到 SNS 主題。然後，Amazon SNS 將 S3 事件通知傳遞給 SQS 佇列訂閱者。EC2 實例的 Auto Scaling 組處理 SQS 佇列中的消息，然後將處理後的映射發佈到 S3 存儲桶，該存儲桶將內容提供給 Amazon CloudFront。

# Amazon SNS 與 Amazon SQS 的對比



特徵	Amazon SNS (發佈者/訂閱者)	Amazon SQS (生產者/消費者)
(生產者/消費者)	發佈/訂閱	發送/接收
交付機制	推送 (被動)	輪詢 (主動)
分配模型	多對多	一對一
消息持久性	否	是

- Amazon SNS 使用發佈/訂閱消息收發范式支援應用程式通過推送機制向多個訂閱者發送對時間有嚴格要求的消息。
- Amazon SQS 使用發佈/訂閱消息收發範式並通過輪詢模式交換消息 – 發送元件和接收元件是解耦的。
- Amazon SQS 為應用程式的分散式元件提供了靈活性 – 無需每個元件同時可用即可發送和接收消息。

## 第 4 節要點



- Amazon SNS 是一種 Web 服務，以便您設置、運行及從雲中發送通知
- Amazon SNS 遵循發佈/訂閱消息收發模式
- 使用 Amazon SNS 時，您可以創建主題並設置策略來限制誰可以發佈或訂閱該主題
- 您可以使用主題將消息發佈者與訂閱者解耦，將消息同時群發給多個收件人，並消除應用程式中的輪詢
- AWS 服務可以將消息發佈到您的 SNS 主題，以觸發事件驅動型計算和工作流

本模組中這節內容的要點包括：

- Amazon SNS 是一種 Web 服務，以便您設置、運行及從雲中發送通知
- Amazon SNS 遵循發佈/訂閱消息收發模式
- 使用 Amazon SNS 時，您可以創建主題並設置策略來限制誰可以發佈或訂閱該主題
- 您可以使用主題將消息發佈者與訂閱者解耦，將消息同時群發給多個收件人，並消除應用程式中的輪詢
- AWS 服務可以將消息發佈到您的 SNS 主題，以觸發事件驅動型計算和工作流

模組 12：構建解耦架構

## 第 5 節：使用 Amazon MQ 在雲應用程式和本地之間發送消息



介紹第 5 節：使用 Amazon MQ 在雲應用程式和本地之間發送消息。



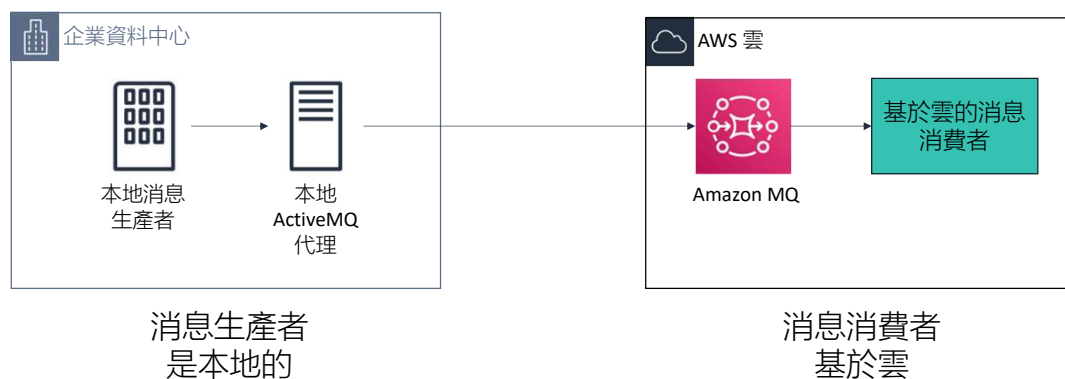
Amazon  
MQ

- 是 Apache ActiveMQ 的託管消息代理服務
- 管理 ActiveMQ 的預置、設置和維護
- 簡化消息到雲的遷移
- 與各種開放標準 API 和協定相容
  - JMS、NMS、AMQP、STOMP、MQTT 和 WebSocket

Amazon MQ 是一種適用於 Apache ActiveMQ 的託管消息代理服務，讓您能夠在雲中設置和操作消息代理。利用消息代理，不同的軟體系統（通常使用不同程式設計語言且在不同平臺上運行）能夠進行通信並交換資訊。Amazon MQ 可以管理常用開源消息代理 ActiveMQ 的預置、設置和維護，從而減輕您的運營負荷。

通過 Amazon MQ，您可以輕鬆將消息收發遷移至雲，同時保留應用程式間的現有連接。它支援用於消息收發的開放標準 API 和協定，包括 Java 消息服務 (JMS)、.NET 消息服務 (NMS)、高級訊息佇列協議 (AMQP)、面向流文本的消息傳輸協定 (STOMP)、訊息佇列遙測傳輸 (MQTT) 和 WebSocket。您可以輕鬆從使用這些標準的任意消息代理遷移至 Amazon MQ，通常無需重新編寫任何消息收發代碼。大多數情況下，您可以更新應用程式的終端節點來連接到 Amazon MQ，然後就可以開始發送消息。

## Amazon MQ 使用案例：混合雲環境



許多組織，尤其是企業，都依賴消息代理來連接和協調不同的系統。消息代理使分散式應用程式可以相互通信。他們是 IT 環境以及最終業務服務的技術支柱。應用程式依賴消息收發正常運行。

在許多情況下，這些組織已開始構建新的原生雲應用程式或將應用程式遷移到 AWS。有些應用程式（例如大型機系統）的遷移成本太高。在這些情況下，本地應用程式仍必須與基於雲的元件進行交互。

借助 Amazon MQ，組織能夠在雲中的應用程式與本地應用程式之間發送消息，以實現混合環境和應用程式現代化。例如，您可以從由 Amazon MQ 代理託管的佇列和主題中調用 AWS Lambda，以將傳統系統與無伺服器架構進行集成。

該示例說明了您可以借助 ActiveMQ 的代理功能網路使用 Amazon MQ 將本地和雲環境集成在一起。該圖顯示了從本地生產者到本地代理的消息生命週期，該消息生命週期遍歷本地代理與 Amazon MQ 之間的混合連接。最後，消息遷移到 AWS 雲內使用。

有關如何使用 Amazon MQ 集成本地和雲環境的更多資訊，請閱讀此 [AWS 計算博客文章](#)。



## Amazon MQ 與 Amazon SQS 和 Amazon SNS 的對比



Amazon MQ	Amazon SQS 和 SNS
對於應用程式遷移	對於誕生於雲中的應用程式
協議：JMS、NMS、AMQP、STOMP、MQTT 和 WebSocket	協議：HTTPS
豐富功能	接近無限輸送量
按小時付費和按 GB 付費	按請求付費
可以執行發布/訂閱	無法在 Amazon SQS 中執行發布/訂閱，但是您可以在 Amazon SNS 中執行發布/訂閱

Amazon MQ 是一項託管消息代理服務，可相容許多常見消息代理。Amazon SQS 和 Amazon SNS 都是高度可擴展、易於使用且不需要您設置消息代理的佇列和主題服務。

- 如果您使用現有應用程式中的消息收發功能，並且想要將消息收發功能移至雲中，AWS 建議您使用 Amazon MQ。它支持開放標準 API 和協議。您可以從基於標準的任何消息代理切換到 Amazon MQ，無需重新編寫應用程式中的消息收發代碼。
- 如果您要在雲中構建新的應用程式，AWS 建議您使用 Amazon SQS 和 Amazon SNS。

## 第 5 節要點



- Amazon MQ 是一種適用於 Apache ActiveMQ 的託管消息代理服務，讓您能夠在雲中設置和操作消息代理
- Amazon MQ 可以管理 ActiveMQ 的預置、設置和維護，這種是常用的開源消息代理
- Amazon MQ 與多種開放標準 API 和協定相容（即 JMS、NMS、AMQP、STOMP、MQTT 和 WebSocket）
- 您可以借助 ActiveMQ 的代理功能網路使用 Amazon MQ 將本地和雲環境集成在一起

本模組中這節內容的要點包括：

- Amazon MQ 是一種適用於 Apache ActiveMQ 的託管消息代理服務，讓您能夠在雲中設置和操作消息代理
- Amazon MQ 可以管理 ActiveMQ 的預置、設置和維護，這種是常用的開源消息代理
- Amazon MQ 與多種開放標準 API 和協定相容（即 JMS、NMS、AMQP、STOMP、MQTT 和 WebSocket）
- 您可以借助 ActiveMQ 的代理功能網路使用 Amazon MQ 將本地和雲環境集成在一起

模組 12：構建解耦架構

## 模組總結



現在來回顧下本模組，並對知識測驗和對實踐認證考試問題的討論進行總結。

## 模組總結



總體來說，您在本模組中學習了如何：

- 區分緊耦合架構和松耦合架構
- 確定 Amazon SQS 的工作原理以及使用時間
- 確定 Amazon SNS 的工作原理以及使用時間
- 描述 Amazon MQ

總體來說，您在本模組中學習了如何：

- 區分緊耦合架構和松耦合架構
- 確定 Amazon SQS 的工作原理以及使用時間
- 確定 Amazon SNS 的工作原理以及使用時間
- 描述 Amazon MQ

## 完成知識測驗



現在可以完成本模組的知識測驗。

一家公司必須執行非同步處理，並已將 Amazon Simple Queue Service (Amazon SQS) 作為解耦架構的一部分實施。該公司希望確保將輪詢請求中的空回應數量保持在最低水準。

解決方案架構師應該做些什麼來確保減少空回應數量呢？

- A. 增加佇列的最大消息保留期
- B. 增加佇列的重新驅動策略的最大接收量
- C. 增加佇列的預設可見性超時
- D. 增加佇列的長輪詢等待時間

請查看答案選項，並根據之前突出顯示的關鍵字排除錯誤選項。

**正確答案是 D：**“增加佇列的輪詢等待時間。” 當隊列的 `ReceiveMessageWaitTimeSeconds` 屬性設置為大於零的值時，長輪詢將生效。長輪詢允許 Amazon SQS 等待消息可用後再向 `ReceiveMessage` 請求發送回應，從而減少空回應的數量。

## 其他資源



- [使用 Amazon SQS 和 Amazon SNS 構建松耦合且可擴展的 C# 應用程式](#)
- [Amazon SQS 資源](#)
- [Amazon SNS 資源](#)
- [Amazon MQ 資源](#)

如果您想瞭解有關本模組所涵蓋主題的更多資訊，下面這些其他資源可能會有所幫助：

- [使用 Amazon SQS 和 Amazon SNS 構建松耦合且可擴展的 C# 應用程式](#)
- [Amazon SQS 資源](#)
- [Amazon SNS 資源](#)
- [Amazon MQ 資源](#)

# 謝謝

© 2020 Amazon Web Services, Inc. 或其附屬公司。保留所有權利。未經 Amazon Web Services, Inc. 事先書面許可，不得複製或轉載本文的部分或全部內容。禁止因商業目的複製、出借或出售本文。如有對本課程的糾正或回饋意見，請發送電子郵件至：[aws-course-feedback@amazon.com](mailto:aws-course-feedback@amazon.com)。如有其他任何問題，請與我們聯繫：<https://aws.amazon.com/contact-us/aws-training/>。所有商標均為各自所有者的財產。



感謝您完成本模組的學習。