9.4

IBM MQ の計画

注記

本書および本書で紹介する製品をご使用になる前に、213 ページの『特記事項』に記載されている情報をお読みください。

本書は、 IBM® MQ バージョン 9 リリース 4、および新しい版で明記されていない限り、以降のすべてのリリースおよびモディフィケーションに適用されます。

お客様が IBM に情報を送信する場合、お客様は IBM に対し、お客様に対してなんら義務も負うことのない、自ら適切と信ずる方法で情報を使用または配布する非独占的な権利を付与します。

© Copyright International Business Machines Corporation 2007 年, 2025.

目次

計画.....5
IBM MQ リリース・タイプ: 計画上の考慮事項.....6
IBM MQ および IBM MQ Appliance の GDPR 対応に関するオンプレミス考慮事項.....9
1 つのキュー・マネージャーに基づくアーキテクチャー.....18
複数のキュー・マネージャーに基づくアーキテクチャー.....19
分散キューおよびクラスターの計画.....20
分散パブリッシュ/サブスクライブ・ネットワークの計画.....72
ストレージ要件とパフォーマンス要件の計画 (Multiplatforms).....112
ディスク・スペース要件 (Multiplatforms).....113
ファイル・システム・サポートの計画 (Multiplatforms).....116
MFT on Multiplatforms でのファイル・システム・サポートの計画.....144
循環ロギングまたはリニア・ロギングの選択 (Multiplatforms).....145
AIX 上の共有メモリー.....145
IBM MQ と UNIX System V IPC リソース.....146
IBM MQ および UNIX のプロセス優先順位.....146
Planning your IBM MQ environment on z/OS.....147
Planning for your queue manager.....147
チャネル・イニシエーターの計画.....176
Planning your queue sharing group (QSG).....181
Planning for backup and recovery.....194
Planning your z/OS UNIX environment.....203
計画中 Advanced Message Security の上 z/OS.....203
計画中 Managed File Transfer の上 z/OS.....204
Planning to use the IBM MQ Console and REST API on z/OS.....210

特記事項.....213
プログラミング・インターフェース情報.....214
商標.....214

IBM MQ アーキテクチャーの計画

IBM MQ 環境を計画する際、単一および複数キュー・マネージャーのアーキテクチャーについて、また Point-to-Point およびパブリッシュ/サブスクライブのメッセージング・スタイルについて IBM MQ が提供するサポートを考慮します。 また、リソース要件、およびロギングやバックアップの機能の使用方法を計画します。

このタスクについて

IBM MQ のアーキテクチャーを計画する前に、IBM MQ の基本的な概念をよく理解することが必要です。『IBM MQ の技術概要』を参照してください。

IBM MQ アーキテクチャーは、単一のキュー・マネージャーを使用した単純なアーキテクチャーから、より複雑な相互接続キュー・マネージャーのネットワークまで多岐にわたります。 複数のキュー・マネージャーを互いに接続するには、分散キューイング技法が使用されます。 単一のキュー・マネージャーのアーキテクチャーおよび複数のキュー・マネージャーのアーキテクチャーの計画について詳しくは、以下のトピックを参照してください。

- 18 ページの『1 つのキュー・マネージャーに基づくアーキテクチャー』
- 19 ページの『複数のキュー・マネージャーに基づくアーキテクチャー』
  - 20 ページの『分散キューおよびクラスターの計画』
  - 72 ページの『分散パブリッシュ/サブスクライブ・ネットワークの計画』

z/OS IBM MQ for z/OS® では、共有キューおよびキュー共有グループを使用することにより、ワークロード・バランシングを実装して、IBM MQ アプリケーションをスケーラブルにし、その可用性を高めることができます。 共有キューおよびキュー共有グループについて詳しくは、『共用キューとキュー共用グループ』を参照してください。

IBM MQ は、2 つの異なるリリース・モデルを提供しています。

- Long Term Support (LTS) リリースは、長期的なデプロイメントと最大限の安定性を必要とするシステムに最適です。
- Continuous Delivery (CD) リリースは、 IBM MQ の最新の機能拡張を迅速に活用する必要があるシステムを対象としています。

どちらのリリース・タイプも同じ方法でインストールできますが、理解しておく必要があるサポートと移行に関する考慮事項があります。 詳しくは、 IBM MQ のリリース・タイプとバージョン管理を参照してください。

複数のインストール済み環境、ストレージとパフォーマンスの要件、およびクライアントの使用については、他のサブトピックを参照してください。

関連概念

IBM MQ のリリース・タイプとバージョン管理

147 ページの『Planning your IBM MQ environment on z/OS』
When planning your IBM MQ environment, you must consider the resource requirements for data sets, page sets, Db2, Coupling Facilities, and the need for logging, and backup facilities. Use this topic to plan the environment where IBM MQ runs.

可用性、リカバリー、および再始動

関連タスク

要件のチェック

メッセージの消失を確実に回避する (ログ)

© Copyright IBM Corp. 2007 年, 2025

IBM MQ リリース・タイプ: 計画上の考慮事項

IBM MQ の 2 つの主なリリース・タイプは、 Long Term Support (LTS) と Continuous Delivery (CD) です。 サポートされるプラットフォームごとに、選択するリリース・タイプは、注文、インストール、保守、およびマイグレーションに影響します。

リリース・タイプについて詳しくは、 IBM MQ リリース・タイプおよびバージョン管理を参照してください。

IBM MQ for Multiplatforms の考慮事項

Multi

注文

Passport Advantage® 内には、IBM MQ 9.4 用の 2 つの別個の eAssemblies があります。 一方には IBM MQ 9.4.0 Long Term Support・リリースのインストール・イメージが含まれており、もう一方には IBM MQ 9.4.x Continuous Delivery・リリースのインストール・イメージが含まれています。 選択するリリースに応じてインストール・イメージを eAssembly からダウンロードします。

すべての IBM MQ バージョン、および IBM MQ 9.4 の LTS リリースおよび CD リリースの両方は、同じ製品 ID に属します。

IBM MQ を使用するためのライセンスは、ライセンス交付を受けたコンポーネントおよび料金設定メトリックの制約に従って、製品全体 (PID) にわたって拡張されます。 これは、 IBM MQ 9.4 の LTS リリースと CD リリースのインストール・イメージの間で自由に選択できることを意味します。

インストール

Passport Advantage からインストール・イメージをダウンロードした後、ライセンスを購入したコンポーネントのみのインストールを選択する必要があります。 各有料コンポーネントに含まれるインストール可能コンポーネントについて詳しくは、IBM MQ ライセンス情報を参照してください。

IBM MQ 9.4.0 LTS リリースと IBM MQ 9.4.x CD リリースを同じオペレーティング・システム・イメージにインストールすることができます。 これを行う場合、コンポーネントは、IBM MQ マルチバージョン・サポートによってサポートされる別個のインストールとして表示されます。 各バージョンの固有のキュー・マネージャーのセットがそのバージョンに関連付けられています。

新しい各 CD リリースがインストール・イメージとして提供されます。 新しい CD リリースは、既存のリリースと一緒にインストールすることも、インストーラーによって以前の CD リリースを新しいリリースに更新することもできます。

CD リリースには、機能拡張に加えて、障害フィックスおよびセキュリティー更新の最新セットが含まれています。 各 CD リリースは累積され、そのバージョンの IBM MQ の以前のすべてのリリースを完全に置き換えます。 そのため、企業に関連する機能が含まれていない特定の CD リリースをスキップできます。

保守

LTS リリースは、障害フィックスを提供するフィックスパック、およびセキュリティー・パッチを提供する累積セキュリティー更新 (CSU) の適用によって保守されます。 フィックスパックおよび CSU は定期的に提供され、累積されます。

CD の場合、CSU は最新の CD リリースに対してのみ作成されます。後続のバージョンの場合もあります。

IBM サポートから暫定的な修正を適用するよう指示されることがあります。 暫定修正は緊急フィックスまたはテスト・フィックスとも呼ばれ、次の保守デリバリーを待つことができない緊急更新を適用するために使用されます。

LTS リリースと CD リリースの間の移行

制約と制限はありますが、ターゲット・リリースが移行前のものより新しければ、通常は、単一キュー・マネージャーでの使用を LTS リリース・コードから CD リリース・コードに、または、CD リリース・コードから LTS リリース・コードに移行できます。

2 つの方法を使用できます。

- IBM MQ の既存のインストールが更新されるように、コードの新しいリリースを所定の位置にインストールする方法。 キュー・マネージャーがインストールに関連付けられていると、それらはすべて開始時にコードの新しいリリースを使用します。
- コードの新しいリリースを新規インストールとしてインストールし、 setmqm コマンドを使用して個別のキュー・マネージャー・インスタンスを新規インストールに移動させる方法。

キュー・マネージャーがコードの CD リリースの実行を開始すると、新しいリリース・レベルを示すようにキュー・マネージャーのコマンド・レベルが更新されます。 これは、リリースで提供される新機能が有効になり、 VRM 番号が小さいコード・リリースを使用してキュー・マネージャーを再始動できなくなることを意味します。

IBM MQ for z/OS の考慮事項

z/OS

注文

IBM MQ for z/OS 9.4 を注文する際、2 つの別個のフィーチャーが ShopZ で提供されます。 これらのフィーチャーは、LTS リリースおよび CD リリースに対応します。 どちらのフィーチャーも同じ製品 ID (PID) に適用できます。 ライセンス交付を受ける製品 ID なので、一方のフィーチャーがライセンス交付を受けている場合は、必要であれば、他方のフィーチャーも使用することができます。 発注時に、 LTS リリースまたは CD リリースのいずれかに対応するフィーチャーを選択します。

ServerPac,に含める製品を選択する場合、同じ ServerPac の順序で「LTS リリースと「CD リリースの両方を選択することはできません。

インストール

LTS リリースおよび CD リリースは、FMID の別々のセットで提供されます。 これらの FMID は、同じ SMP/E ターゲット・ゾーンにインストールできません。 LTS と CD の両方のリリースが必要な場合は、以下のようにします。

- LTS リリースと CD リリースを別々のターゲット・ゾーンにインストールします。
- 2 つのリリース用に別々のターゲット・ライブラリーと配布ライブラリーを維持します。

キュー・マネージャーがキュー共有グループ内にある場合、最新の CD バージョンにアップグレードするときに、グループ内のすべてのキュー・マネージャーをアップグレードする必要があります。

キュー・マネージャのコマンド・レベルは 3 桁の VRM レベルです。 IBM MQ プログラムは、 `MQIA_COMMAND_LEVEL` セレクタを渡して `MQINQ` を呼び出し、接続しているキュー・マネージャのコマンド・レベルを取得することができる。

これらのリリースでは異なる FMID が使用されているため、 LTS リリースまたはその逆方向の保守を使用して CD リリースを更新することはできません。 同様に、製品コードのバージョンを LTS リリースから CD リリースに切り替えたり、その逆を行ったりする方法はありません。 ただし、リリース・モデル間でキュー・マネージャーを切り替えることができます。 LTS リリースと CD リリース間のマイグレーションを参照してください。

注 :

IBM MQ 9.0.x と IBM MQ 9.1.x の CD リリースには、それぞれ異なるバージョンとリリースに依存する FMID があります。 そのため、9.0.x CD から 9.1.x CD に移行するには、少なくとも 1 つの完全な SMP/E インストールが必要です。

IBM MQ for z/OS 9.2.0 以降、 CD リリースは、 9 のバージョン番号を持つすべての IBM MQ for z/OS リリースで同じままの FMID のセットを使用します。 IBM MQ の各新規バージョンは CD と LTS の両方のリリースとして使用可能であるため、メジャー・バージョンの境界を超えても、PTF を単一の SMP/E インストールに適用することによって CD リリースをアップグレードすることができます。 例えば、PTF を適用するだけで、 IBM MQ for z/OS 9.2.0 CD から IBM MQ for z/OS 9.2.2 CD、 IBM MQ for z/OS 9.2.4 CD、 IBM MQ for z/OS 9.3.0 CD に移動することができます。

同じ VRM レベルの LTS リリースと CD リリースとを区別するには、キュー・マネージャーのジョブ・ログで CSQY000I メッセージを調べます。

保守

IBM MQ for z/OS は、保守のために PTF を使用します。

LTS PTF は、特定のリリース・レベルに対応する特定のライブラリー・セットに固有のものとなります。 UNIX System Services 機能 (つまり、JMS および WEB UI、Connector Pack、および Managed File Transfer) の場合、z/OS PTF は、Multiplatforms フィックスパックおよび累積セキュリティー更新 (CSU) と直接整合しています。 これらのフィックスは累積的であり、同等の Multiplatforms フィックスパックまたは CSU と同時に入手できます。

CD CD CSU は通常、CD リリースの間に提供されることはないが、次の。 IBM MQ for z/OS CD サポートに連絡して + + USERMOD を要求することもできます。

IBM MQ for z/OS のその他のフィックスは、特定の部分に対する別個のフィックスです。 これらのフィックスは、特定の問題を解決し、累積的な問題ではなく、作成時に使用可能になります。

LTS リリースと CD リリースの間の移行

制約と制限はありますが、ターゲット・リリースが移行前のものより新しければ、通常は、単一キュー・マネージャーでの使用を LTS リリース・コードから CD リリース・コードに、または CD リリース・コードから LTS リリース・コードに移行できます。

IBM MQ for z/OS 9.2.0 以降では、同じ VRM を持つ CD リリースと LTS リリースの間で必要な回数だけマイグレーションを行うことができ、逆方向マイグレーションの機能に影響を与えることはありません。 例えば、 IBM MQ for z/OS 9.3.0 LTS でキュー・マネージャーを開始してから、 IBM MQ for z/OS 9.3.0 CD でシャットダウンおよび開始してから、 IBM MQ for z/OS 9.3.0 LTS でシャットダウンおよび開始することができます。

IBM MQ for z/OS では従来、フォールバック機能 (逆方向マイグレーション) が提供されていました。これにより、マイグレーション後に一定期間実行した後、前のリリースにフォールバックすることができます。この機能は、 LTS リリース、および修飾子が 0 の CD リリース (9.3.0 CD など) で保持されますが、マイグレーションのソースまたはターゲットがゼロ以外の修飾子番号を持つ CD リリース (9.2.5 または 9.3.1 など) である場合は使用できません。

以下は有効な移行シナリオで、この原則がどのように当てはまるかを示しています。

ソース・リリース	宛先リリース	注
9.1.0 LTS	9.4.0 LTS または 9.4.0 CD	9.1.0 LTS が標準サポートされていないため、逆方向マイグレーションはサポートされません。
9.2.0 LTS (英語)	9.4.0 LTS または 9.4.0 CD	バックワード・マイグレーションはサポートされています。
9.3.0 LTS (LTS)	9.4.0 LTS または 9.4.0 CD	バックワード・マイグレーションはサポートされています。
9.3.5 CD	9.4.0 LTS または 9.4.0 CD	ソース・リリースが CD であり、修飾子が 0 でないため、バックワード・マイグレーションはサポートされていません。
9.4.0 LTS または 9.4.0 CD	9.4.1 CD	宛先リリースが CD であり、修飾子が 0 でないため、バックワード・マイグレーションはサポートされていません。 マイグレーションを確認するために、`Write to operator with reply` CSQY041D が発行されます。

関連タスク

z/OS z/OS での保守の適用と削除

関連情報
ダウンロード中 IBM MQ 9.4

IBM MQ および IBM MQ Appliance の GDPR 対応に関するオンプレミス考慮事項

対象 PID:

分散

- IBM MQ/IBM MQ Advanced - 5724-H72
- IBM MQ for HPE NonStop - 5724-A39

z/OS

- IBM MQ for z/OS - 5655-MQ9
- IBM MQ for z/OS Value Unit Edition - 5655-VU9
- IBM MQ Advanced for z/OS - 5655-AV9
- IBM MQ Advanced for z/OS Value Unit Edition - 5655-AV1

IBM MQ Appliance

- IBM MQ Appliance M2003 -5900-ALJ
- IBM MQ Appliance M2002 - 5737-H47

注意:

この資料は、お客様の GDPR 対応の準備を支援することを目的としています。 組織として GDPR に対応するために検討しなければならない IBM MQ の構成可能な機能と製品の使用法について説明します。 お客様が機能を選択および構成できる方法が多岐にわたっており、また製品を単体で、あるいはサード・パーティーのアプリケーションおよびシステムとともにさまざまな方法で使用できるため、この情報はすべてを網羅したリストではありません。

お客様は、欧州連合 (EU) の一般データ保護規則を含む、さまざまな法律および規制への準拠を保証する責任があります。 お客様のビジネスに影響を及ぼす可能性のある関連法令の特定およびそれらの解釈、ならびにかかる関連法令を遵守するためにお客様が講ずるべき必要措置に関する助言は、お客様の責任により適格な弁護士から得るものとします。

本書に記載の製品、サービス、および他の機能が、すべてのお客様の状況に適しているとは限らず、使用する際に制約を受ける場合があります。 IBM は、法律、会計または監査に関する助言を提供することはしませんし、IBM のサービスまたは製品が、お客様のあらゆる法令遵守の裏付けとなる表明または保証もいたしません。

目次

1. GDPR
2. GDPR のための製品構成
3. データ・ライフサイクル
4. データ収集
5. データ・ストレージ
6. データ・アクセス
7. データ処理
8. データ削除
9. データ・モニタリング
10. 個人データの使用を制限するための機能

11. ファイル処理

GDPR

一般データ保護規則 (GDPR) は、欧州連合 (EU) によって採択され、2018 年 5 月 25 日から適用されています。

GDPR が重要である理由

GDPR により、個人に関する個人データの処理に関する、より強固なデータ保護規制の枠組みが確立されます。 GDPR により以下のことがもたらされます。

- 個人の新たな権利および強化された権利
- 個人データの定義の拡大
- データを処理する人の新たな義務
- 不遵守に対して高額の制裁金の可能性
- データ漏えいの届け出の義務付け

GDPR について詳しくは、以下のサイトを参照してください。

- EU GDPR 情報ポータル
- ibm.com/GDPR の Web サイト

製品の構成 - GDPR 対応のための考慮事項

以下の各セクションでは、組織として GDPR に対応するために、IBM MQ の構成に関する注意点をまとめています。

データ・ライフサイクル

IBM MQ は、アプリケーション間でアプリケーション提供のデータを非同期に交換できるようにするための、トランザクション型のメッセージ指向ミドルウェア製品です。 IBM MQ では、アプリケーションを接続するために、さまざまなメッセージング API、プロトコル、およびブリッジをサポートしています。 そのため、IBM MQ は様々な形態のデータの交換に使用される可能性があり、その一部が GDPR の対象になる可能性があります。 IBM MQ とデータ交換を行う可能性のあるサード・パーティー製品もいくつかあります。 その一部は IBM 所有ですが、その他の多くは他のテクノロジー・サプライヤーから提供されている製品です。 Software Product Compatibility Reports Web サイトに、関連するソフトウェアのリストが記載されています。 サード・パーティー製品の GDPR 対応に関する考慮事項については、その製品の資料を調べてください。 IBM MQ 管理者は、キュー、トピック、およびサブスクリプションの定義によって、IBM MQ がそれを通過するデータと相互作用する方法を制御します。

IBM MQ を流れるデータのタイプにはどのようなものがありますか?

IBM MQ は、アプリケーション・データの非同期メッセージング・サービスを提供するため、アプリケーションのデプロイメントによってユースケースが異なり、この問いに対する 1 つの明確な答えはありません。アプリケーション・メッセージ・データは、キュー・ファイル (z/OS のページ・セットまたはカップリング・ファシリティー)、ログ、およびアーカイブに保持されています。またメッセージ自体に GDPR によって管理されるデータが含まれている場合があります。 アプリケーション提供のメッセージ・データは、エラー・ログ、トレース・ファイル、および FFST など、問題判別のために収集されたファイルにも含まれている可能性もあります。 z/OS では、アプリケーション提供のメッセージ・データは、アドレス・スペースやカップリング・ファシリティーのダンプにも含まれる可能性があります。

IBM MQ を使用して交換される可能性のある代表的な個人データの例として、以下のようなものがあります。

- お客様の雇用者の個人データ (例えば、IBM MQ を使用してお客様の給与計算システムまたは HR システムを接続する場合があります)
- お客様自身の顧客の個人データ (例えば、お客様が IBM MQ を使用して顧客に関係するデータをアプリケーション間で交換する場合があります。CRM システムで見込み客情報を取得したりデータを格納したりする場合などです)。

- お客様自身の顧客の機密性の高い個人データ (例えば、個人データの交換を必要とする業界特有の状況で IBM MQ を使用する場合があります。臨床アプリケーションを統合するときの HL7 ベースの医療記録などです)。

アプリケーション提供のメッセージ・データの他にも、IBM MQ は以下のタイプのデータを処理します。

- 認証資格情報 (ユーザー名とパスワード、API 鍵など)
- 技術的に識別可能な個人情報 (デバイス ID、使用ベースの ID、IP アドレスなど - 個人にリンクされている場合)

IBM とのオンラインによる連絡のために使用される個人データ

IBM MQ のお客様は、さまざまな方法でオンラインでコメント/フィードバック/要求を送信して、 IBM MQ 件について IBM に連絡することができます。主な方法は以下のとおりです。

- IBM Developer の IBM MQ 領域内のページにあるパブリック・コメント領域
- IBM MQ IBM Documentation の製品情報のページ上にあるパブリック・コメント領域
- IBM サポート・フォーラム内のパブリック・コメント
- IBM の統合概念におけるパブリック・コメント

通常、クライアント名と E メール・アドレスのみが使用され、コンタクトの対象となる個人の応答を使用可能にし、個人データの使用は IBM オンライン・プライバシー・ステートメントに準拠します。

データ収集

IBM MQ を使用して個人データを収集できます。 IBM MQ の使用および GDPR の要求を満たす必要性を評価する場合、ご使用の環境で IBM MQ を通過する個人データのタイプを考慮する必要があります。 次のような側面を考慮することができます。

- データはどのようにキュー・マネージャーに到着するか。 (どのプロトコルか。 データは暗号化されているか。 データは署名されているか。)
- データはどのようにキュー・マネージャーから送信されるか。 (どのプロトコルか。 データは暗号化されているか。 データは署名されているか。)
- データはキュー・マネージャーを通過するときにどのように格納されるか。 (メッセージが非持続の場合でも、メッセージング・アプリケーションはメッセージ・データをステートフル・メディアに書き込む可能性がある。 この製品を通過するアプリケーション・メッセージ・データの特定の側面が、メッセージングのフィーチャーによってどのように公開される可能性があるかを認識しているか?)
- IBM MQ がサード・パーティー・アプリケーションにアクセスするために必要なときに、資格情報をどのように収集して保管するか。

IBM MQ は、LDAP など、認証を必要とする他のシステムおよびサービスと通信する必要がある場合があります。 必要なときに、IBM MQ はそのような通信で利用するために認証データ (ユーザー ID、パスワード) を構成して保管します。 可能な限り、IBM MQ 認証に個人の資格情報を使用しないようにする必要があります。 認証データ用に使用されるストレージの保護を検討してください。 (下記の「データ・ストレージ」を参照)

データ・ストレージ

メッセージ・データがキュー・マネージャーを通過するとき、IBM MQ はそのデータ (おそらくその複数のコピー) をステートフル・メディアに直接保存します。 IBM MQ ユーザーは、メッセージ・データが保存状態である間はそれを保護するよう検討してください。

以下の項目は、IBM MQ がアプリケーション提供のデータを保持する領域を取り上げています。これらは、GDPR への準拠を確実にする場合にユーザーが考慮したいと考える項目です。

- アプリケーション・メッセージ・キュー:

  IBM MQ は、アプリケーション間での非同期データ交換を可能にするメッセージ・キューを提供します。キューに格納された非持続メッセージおよび持続メッセージは、ステートフル・メディアに書き込まれます。

- ファイル転送エージェント・キュー:

  IBM MQ Managed File Transfer はメッセージ・キューを使用してファイル・データの信頼性のある転送を調整します。個人データと転送記録を入れたファイルは、これらのキューに格納されます。
- 伝送キュー

  メッセージをキュー・マネージャー間で確実に転送するために、メッセージは一時的に伝送キューに格納されます。
- 送達不能キュー:

  メッセージは、宛先キューに書き込むことができずに送達不能キューに格納されることがあります (送達不能キューがキュー・マネージャーで構成されている場合)。
- バックアウト・キュー:

  JMS および XMS のメッセージング・インターフェースは、他の有効なメッセージを処理できるように、いくつかのバックアウトが発生すると有害メッセージをバックアウト・キューに移動できる機能を提供します。
- AMS エラー・キュー:

  IBM MQ Advanced Message Security は、セキュリティー・ポリシーに準拠しないメッセージを、送達不能キューイングと同様の方法で SYSTEM.PROTECTION.ERROR.QUEUE エラー・キューに移動します。
- 保存パブリケーション:

  IBM MQ は、サブスクライブ側のアプリケーションが前のパブリケーションを再呼び出しできるようにするために、保存パブリケーション・フィーチャーを提供します。
- 配信の遅延:

  IBM MQ は、メッセージを将来の宛先に配信できるようにする JMS 2.0 および Jakarta Messaging 3.0 の配信遅延機能をサポートします。 まだ配信されていないメッセージは、SYSTEM.DDELAY.LOCAL.QUEUE キューに保管されます。

詳しくは、以下を参照してください。

- ロギング: メッセージが失われないようにするための機能
- MFT エージェント・キュー設定
- 送達不能キューの使用
- IBM MQ classes for JMS での有害メッセージの処理
- AMS エラー処理
- 保存パブリケーション
- JMS 2.0 送達遅延

以下の項目は、IBM MQ がアプリケーション提供のデータを間接的に保持する領域を取り上げています。これらも、GDPR への準拠を保証するためにユーザーが考慮できる項目です。

- 経路トレース・メッセージング:

  IBM MQ は、アプリケーション間でメッセージが取る経路を記録する経路トレース機能を提供します。生成されるイベント・メッセージには、IP アドレスなどの技術的に識別可能な個人情報が含まれる場合があります。
- アプリケーション・アクティビティー・トレース:

  IBM MQ は、アプリケーションとチャネルのメッセージング API アクティビティーを記録するアプリケーション・アクティビティー・トレースを提供します。アプリケーション・アクティビティー・トレースでは、アプリケーション提供のメッセージ・データの内容をイベント・メッセージに記録することができます。
- サービス・トレース:

  IBM MQ は、メッセージ・データが流れる内部コード・パスを記録するサービス・トレース・フィーチャーを提供します。 これらのフィーチャーの一部として、IBM MQ では、アプリケーション提供のメッセージ・データの内容を、ディスクに保管されているトレース・ファイルに記録できます。

- キュー・マネージャー・イベント:

  IBM MQ は、権限イベント、コマンド・イベント、構成イベントなどの、個人データを含む可能性のあるイベント・メッセージを生成することがあります。

詳しくは、以下を参照してください。

- 経路トレース・メッセージング
- トレースの使用法
- イベント・モニター
- キュー・マネージャー・イベント

アプリケーション提供のメッセージ・データのコピーへのアクセスを保護するには、以下のアクションを考慮してください。

- ファイル・システム内の IBM MQ データへの特権ユーザー・アクセスを制限します。例えば、UNIX and Linux® プラットフォームでの 'mqm' グループのユーザー・メンバーシップを制限します。
- 専用キューおよびアクセス制御によって、IBM MQ データへのアプリケーションのアクセスを制限します。 必要に応じて、アプリケーション間でのキューなどのリソースの不要な共有を避け、キューおよびトピック・リソースに対してきめ細かくアクセス制御を設定します。
- 高可用性 (HA) または 災害復旧 (DR) 構成の IBM MQ データの複製コピーへのアクセスを制限し、複製に使用する接続を保護します。
- IBM MQ Advanced Message Security を使用して、メッセージ・データのエンドツーエンドの署名または暗号化 (あるいはその両方) を行います。
- ファイル・レベルまたはボリューム・レベルの暗号化を使用して、 IBM MQ データ、トレース、またはログを含む可能性のあるディレクトリーまたはファイル・システムを保護します。
- サービス・トレースを IBM にアップロードした後、個人データが入っている可能性があるコンテンツについて懸念がある場合は、サービス・トレース・ファイルおよび FFST データを削除できます。

詳しくは、以下を参照してください。

- 特権ユーザー
- ファイル・システム・サポートの計画 (Multiplatforms)
- IBM'MQ for z/OS のデータセット暗号化による静止データの機密性
- IBM MQ オペレーターのためのストレージ計画 - 暗号化
- IBM MQ Appliance 上のファイル・システム暗号化

IBM MQ 管理者は、資格情報 (ユーザー名とパスワード、API キーなど) を使用してキュー・マネージャーを構成できます。 LDAP などのサード・パーティー・サービスの場合。 このデータは、通常、ファイル・システム権限を使用して保護されたキュー・マネージャーのデータ・ディレクトリーに格納されます。

IBM MQ キュー・マネージャーが作成されるときには、IBM MQ が構成ファイルを読み取って資格情報を使用してこれらのシステムに接続できるように、グループ・ベースのアクセス制御を使用してデータ・ディレクトリーがセットアップされます。 IBM MQ 管理者は特権ユーザーと見なされ、このグループのメンバーであるため、これらのファイルへの読み取り権限があります。 一部のファイルは難読化されていますが、暗号化はされていません。 そのため、資格情報へのアクセスを完全に保護するには、以下のアクションを考慮する必要があります。

- IBM MQ データへの特権ユーザーのアクセスを制限します。例えば、UNIX and Linux プラットフォームでの「mqm」グループのメンバーシップを制限します。
- ファイル・レベルまたはボリューム・レベルの暗号化を使用して、キュー・マネージャー・データ・ディレクトリーのコンテンツを保護します。
- 実動構成ディレクトリーのバックアップを暗号化し、適切なアクセス制御を設定して保管します。
- セキュリティー・イベント、コマンド・イベント、および構成イベントでの、認証失敗、アクセス制御、および構成変更に対して監査証跡を提供することを検討してください。

詳しくは、以下を参照してください。

- IBM MQ の保護

データ・アクセス

IBM MQ キュー・マネージャー・データは、以下の製品インターフェースからアクセスできます。 リモート接続によってアクセスするように設計されているものと、ローカル接続によってアクセスするように設計されているものがあります。

- IBM MQ コンソール [リモートのみ]
- IBM MQ 管理 REST API [リモートのみ]
- IBM MQ メッセージング REST API [リモートのみ]
- MQI [ローカルとリモート]
- JMS [ローカルとリモート]
- XMS [ローカルとリモート]
- MQ Telemetry （MQTT） [リモートのみ]
- IBM MQ Light (AMQP) [リモートのみ]
- IBM MQ IMS ブリッジ [ローカルのみ]
- IBM MQ CICS ブリッジ [ローカルのみ]
- IBM MQ MFT プロトコル・ブリッジ [リモートのみ]
- IBM MQ Connect:Direct ブリッジ [リモートのみ]
- IBM MQ MQAI [ローカルおよびリモート]
- IBM MQ PCF コマンド [ローカルおよびリモート]
- IBM MQ MQSC コマンド [ローカルおよびリモート]
- IBM MQ Explorer [ローカルおよびリモート]
- IBM MQ ユーザー出口 [ローカルのみ]
- IBM MQ Internet Pass-Thru [リモートのみ]
- Red Hat® OpenShift® Monitoring (Prometheus) メトリック (メトリックは、キュー・マネージャー統計に関する数値データです)
- IBM MQ Appliance シリアル・コンソール [ローカルのみ]
- IBM MQ Appliance SSH [リモートのみ]
- IBM MQ Appliance REST API [リモートのみ]
- IBM MQ Appliance Web UI [リモートのみ]
- V 9.4.0 IBM MQ Kafka コネクター (Kafka Connect) [ローカルおよびリモート]

インターフェースは、ユーザーが IBM MQ キュー・マネージャーおよびそこに格納されたメッセージに変更を加えられるように設計されています。 管理操作およびメッセージング操作は、要求が行われたときに、関与する以下の 3 つのステージが存在するように保護されます。

- 認証
- ロール・マッピング
- 認証

認証:

メッセージ操作または管理操作がローカル接続から要求された場合、この接続のソースは、同じシステム上の実行中のプロセスです。 プロセスを実行するユーザーは、オペレーティング・システムが提供する認証ステップを通過している必要があります。 接続を行ったプロセスの所有者のユーザー名が ID として表明されます。 これは例えば、アプリケーションを開始したシェルを実行しているユーザーの名前などです。 ローカル接続に使用できる認証の形式として、次のものが挙げられます。

1. 表明されたユーザー名 (ローカル OS)

2. オプションのユーザー名とパスワード (OS、LDAP、またはカスタムのサード・パーティー・リポジトリー)
3. セキュリティー・トークン (JWT) IBM MQ のみ

管理アクションがリモート接続から要求された場合、IBM MQ との通信はネットワーク・インターフェースを介して行われます。 ネットワーク接続を介した認証では、以下の形式の ID を提示できます。

1. 表明されたユーザー名 (リモート OS 由来のもの)
2. ユーザー名とパスワード (OS、LDAP、またはカスタムのサード・パーティー・リポジトリー)
3. ソース・ネットワーク・アドレス (IP アドレスなど)
4. X.509 デジタル証明書 (相互 SSL/TLS 認証)
5. セキュリティー・トークン (LTPA2 トークンや JWT トークンなど)
6. その他のカスタム・セキュリティー (サード・パーティーの出口が提供する機能)
7. SSH 鍵

V 9.4.2 IBM MQ Appliance Web UI は、 OpenID Connect (OIDC) 認証を使用するように設定できる。

IBM MQ と IBM Cloud Pak® for Integration の統合により、 Cloud Pak を使用する IBM MQ Console: シングル・サインオンの新しい認証タイプが追加されます。 (CP4I のみ)

ロール・マッピング:

ロール・マッピング・ステージでは、認証ステージで提供された資格情報を代替ユーザー ID にマップできます。 マップされたユーザー ID が処理を許可された場合 (管理ユーザーがチャネル認証ルールによってブロックされる場合もあります)、マップされたユーザー ID は、IBM MQ リソースに対してアクティビティーを許可するときに最終ステージに持ち越されます。

authorization:

IBM MQ では、キュー、トピック、その他のキュー・マネージャー・オブジェクトなどのさまざまなメッセージング・リソースに対して、さまざまなユーザーにさまざまな権限を付与することができます。

ロギング・アクティビティー:

IBM MQ の一部のユーザーは、MQ リソースへのアクセスの監査レコードを作成する必要がある場合があります。 望ましい監査ログの例としては、変更を要求したユーザーに加えて変更に関する情報を記載した構成変更が考えられます。

この要件を実装するために以下の情報ソースを利用できます。

1. IBM MQ キュー・マネージャーは、admin コマンドが正常に実行されたときにコマンド・イベントを生成するように構成できます。
2. IBM MQ キュー・マネージャーは、キュー・マネージャー・リソースが作成、変更、または削除されたときに構成イベントを生成するように構成できます。
3. IBM MQ キュー・マネージャーは、リソースの許可検査が不合格になったときに権限イベントを生成するように構成できます。
4. 許可検査が不合格になったことを示すエラー・メッセージは、キュー・マネージャーのエラー・ログに書き込まれます。
5. IBM MQ コンソールは、認証、許可検査が不合格になったとき、またはキュー・マネージャーが作成、開始、停止、または削除されたときに、監査メッセージをログに書き込みます。
6. IBM MQ Appliance は監査メッセージをログに書き込み、ユーザー・ログインおよびシステム変更を記録します。
7. V 9.4.3 IBM MQ キューマネージャーは、リソースの認証チェックが成功したときに権限イベントを生成するように設定できます。

このようなソリューションを検討する場合、IBM MQ ユーザーは、以下の点について考慮する必要があります。

- イベント・メッセージは非持続的なので、キュー・マネージャーが再始動すると、情報は失われます。いずれのイベント・モニターも、入手可能なあらゆるメッセージを常にコンシュームしてその内容を永続メディアに転送するように構成する必要があります。
- IBM MQ 特権ユーザーは、イベントの無効化、ログのクリア、またはキュー・マネージャーの削除を行うために十分な特権を持っています。

IBM MQ データへのアクセスの保護、および監査証跡の提供について詳しくは、以下のトピックを参照してください。

- IBM MQ セキュリティー・メカニズム
- 構成イベント
- コマンド・イベント
- エラー・ログの使用

データ処理

公開鍵インフラストラクチャー (PKI) を使用した暗号化:

接続が TLS を使用するように指定することで、IBM MQ へのネットワーク接続を保護できます。TLS は、接続の開始側の相互認証も提供できます。

トランスポート・メカニズムによって提供される PKI セキュリティー機能を使用することが、IBM MQ でのデータ処理を保護するための最初のステップとなります。 しかし、追加のセキュリティー・フィーチャーを有効にしないと、コンシューム側のアプリケーションの動作は、メッセージの発信元や転送中に変更されたかどうかを検証せずに、配信されたメッセージをすべて処理するだけになってしまいます。

Advanced Message Security (AMS) 機能を使用するライセンス交付を受けた IBM MQ のユーザーは、セキュリティー・ポリシーの定義および構成を通じて、メッセージに保持されている個人データをアプリケーションが処理する方法を制御できます。 セキュリティー・ポリシーを使用すると、アプリケーション間のメッセージ・データにデジタル署名または暗号化 (あるいはその両方) を適用できます。

メッセージが本物であることを保証するために、メッセージをコンシュームするときにセキュリティー・ポリシーを使用してデジタル署名を要求および検証することができます。 AMS 暗号化は、読み取り可能な形式のメッセージ・データを、エンコード・バージョンに変換する方式を提供します。このエンコード・バージョンは、別のアプリケーションが意図されたメッセージ受信者であり、かつ正しい暗号化解除鍵にアクセスできる場合にのみ、このアプリケーションでデコードできます。

SSL および証明書を使用してネットワーク接続を保護する方法について詳しくは、IBM MQ 製品資料の以下のトピックを参照してください。

- IBM MQ の TLS セキュリティーの構成
- AMS の概要

データ削除

IBM MQ には、この製品に提供されたデータを削除するためのコマンドおよびユーザー・インターフェース・アクションが用意されています。 これによって、IBM MQ のユーザーは、特定の個人に関連するデータを削除する必要がある場合に、それらのデータを削除できます。

- GDPR クライアント・データの削除に準拠するために考慮する必要がある IBM MQ の動作の領域
  - 次のようにしてアプリケーション・キューに保管されたメッセージ・データを削除する。
    - メッセージング API またはツールを使用して、またはメッセージの有効期限を使用して、個々のメッセージを除去する。
    - 対象メッセージを、非持続メッセージ・クラスが正常の状態であるキューに保持された非持続メッセージとして指定し、キュー・マネージャーを再始動する。
    - 管理者がキューをクリアする。
    - キューを削除する。
  - 次のようにしてトピックに保管された保存パブリケーション・データを削除する。

- メッセージを非持続メッセージとして指定し、キュー・マネージャーを再始動する。
- 保存データを新規データに置き換えるか、メッセージ有効期限を使用する。
- 管理者がトピック・ストリングをクリアする。

– キュー・マネージャー全体と、高可用性または災害復旧用の複製コピーを削除することによって、キュー・マネージャーに保管されたデータを削除する。
– トレース・ディレクトリー内のファイルを削除することによって、サービス・トレース・コマンドによって保管されたデータを削除する。
– エラー・ディレクトリー内のファイルを削除することによって保管された FFST データを削除する。
– アドレス・スペースとカップリング・ファシリティー・ダンプ (z/OS 上) を削除する。
– アーカイブ、バックアップ、またはそのようなデータのその他のコピーを削除する。

• GDPR アカウント・データの削除に準拠するために考慮する必要がある IBM MQ の動作の領域
  – キュー・マネージャーとサード・パーティー・サービスに接続するために IBM MQ に保管されたアカウント・データと設定を削除するために以下を削除する (アーカイブ、バックアップ、それらの複製コピーを含む)。
    - 資格情報を格納するキュー・マネージャー認証情報オブジェクト。
    - ユーザー ID を参照するキュー・マネージャー権限レコード。
    - 特定の IP アドレス、証明書 DN、またはユーザー ID をマップまたはブロックするキュー・マネージャー・チャネル認証規則。
    - キュー・マネージャーおよびファイル・サーバーでの認証用に、IBM MQ Managed File Transfer エージェント、ロガー、および MQ Explorer MFT プラグインが使用する資格情報ファイル。
    - SSL/TLS 接続または IBM MQ Advanced Message Security (AMS) で使用する可能性がある、個人を表すかその個人についての情報を含んだ、鍵ストア由来の X.509 デジタル証明書。
    - IBM MQ Appliance の個人ユーザー・アカウント (システム・ログ・ファイル内のそれらのアカウントへの参照を含む)。
    - IBM MQ Explorer ワークスペース・メタデータおよび Eclipse 設定。
    - 「パスワード設定」で指定されている IBM MQ Explorer パスワード・ストア。
    - IBM MQ コンソールおよび mqweb サーバーの構成ファイル。
    - IBM MQ Internet Pass-Thru 構成ファイルおよび鍵ストア。

詳しくは、以下を参照してください。

- MFT と IBM MQ の接続認証
- ProtocolBridgeCredentials.xml ファイルを使用してファイル・サーバーの資格情報をマップする
- IBM MQ Console ユーザーおよび役割の構成

データのモニタリング

IBM MQ は、ユーザーがアプリケーションとキュー・マネージャーの実行状態をよりよく理解するために活用できるさまざまなモニター・フィーチャーを提供します。

さらに IBM MQ は、キュー・マネージャーのエラー・ログの管理に役立つさまざまなフィーチャーも提供します。

詳しくは、以下を参照してください。

- IBM MQ ネットワークのモニター
- 診断メッセージ・サービス
- QMErrorLog サービス
- IBM MQ オペレーター使用時のモニタリング
- IBM MQ Appliance モニターおよびレポート

個人データの使用を制限するための機能

本書に要約されている機能を使用すると、IBM MQ によって、エンド・ユーザーが自分の個人データの使用を制限できるようになります。

IBM MQ メッセージ・キューは、データベースとは異なるので、永続データ・ストアとしては使用しないでください。これは特に、GDPR の対象となるアプリケーション・データを処理する場合に当てはまります。

検索照会によってデータを検出できるデータベースとは異なり、メッセージのキュー、メッセージ、および相関 ID が分かっていないと、メッセージ・データを見つけるのが困難な場合があります。

個人データが含まれているメッセージを容易に特定して見つけることができる場合は、標準の IBM MQ メッセージング・フィーチャーを使用してメッセージ・データにアクセスしたり変更したりできます。

ファイル処理

1. IBM MQ Managed File Transfer は、転送されるファイルに対してマルウェアのスキャンを実行しません。 ファイルは現状のまま転送され、整合性検査が実行されて、転送中にファイル・データが変更されていないことが確認されます。 転送状況のパブリケーションの一部として、ソースと宛先のチェックサムがパブリッシュされます。 MFT がファイルを転送する前と、MFT がファイルをリモート・エンドポイントに配信した後に、エンド・ユーザーが自分の環境に適したマルウェアのスキャンを実装することが推奨されています。
2. IBM MQ Managed File Transfer は MIME タイプやファイル拡張子に基づくアクションを実行しません。 MFT はファイルを読み取り、入力ファイルから読み取ったとおりに正確なバイト数を転送します。

1 つのキュー・マネージャーに基づくアーキテクチャー

IBM MQ の最もシンプルなアーキテクチャーは、キュー・マネージャーを 1 つだけ構成して使用するというものです。

IBM MQ のアーキテクチャーを計画する前に、IBM MQ の基本的な概念をよく理解することが必要です。『IBM MQ の技術概要』を参照してください。

キュー・マネージャーを 1 つだけ使用したアーキテクチャーとしては、以下の各セクションで取り上げるようなアーキテクチャーが考えられます。

- 18 ページの『1 つのキュー・マネージャーで複数のローカル・アプリケーションがサービスにアクセスするアーキテクチャー』
- 18 ページの『1 つのキュー・マネージャーで複数のリモート・アプリケーションがクライアントとしてサービスにアクセスするアーキテクチャー』
- 19 ページの『1 つのキュー・マネージャーでパブリッシュ/サブスクライブを構成するアーキテクチャー』

1 つのキュー・マネージャーで複数のローカル・アプリケーションがサービスにアクセスするアーキテクチャー

1 つのキュー・マネージャーに基づく最初のアーキテクチャーは、サービスにアクセスするアプリケーションとサービスを提供するアプリケーションを同じシステムで実行するというものです。 IBM MQ キュー・マネージャーは、サービスを要求するアプリケーションとサービスを提供するアプリケーションの間の非同期相互通信を提供します。 この場合は、いずれかのアプリケーションが長期にわたってオフラインになっても、アプリケーション間の通信を継続できます。

1 つのキュー・マネージャーで複数のリモート・アプリケーションがクライアントとしてサービスにアクセスするアーキテクチャー

1 つのキュー・マネージャーに基づく 2 番目のアーキテクチャーは、サービスを提供するアプリケーションからアプリケーションをリモート実行するというものです。 つまり、サービスが存在するシステムとは異なるシステムでリモート・アプリケーションを実行します。 それらのアプリケーションは、クライアン

トとして 1 つのキュー・マネージャーに接続します。 この場合は、1 つのキュー・マネージャーで複数のシステムにサービスに対するアクセスを提供することになります。

このアーキテクチャーの場合は、アプリケーションの操作のためにネットワーク接続を有効にしておく必要がある、という制約があります。 ネットワーク接続を経由したアプリケーションとキュー・マネージャーの対話は、同期モードになります。

1 つのキュー・マネージャーでパブリッシュ/サブスクライブを構成するアーキテクチャー

1 つのキュー・マネージャーを使用するさらに別のアーキテクチャーは、パブリッシュ/サブスクライブ構成を使用するというものです。 パブリッシュ/サブスクライブ・メッセージングでは、情報のプロバイダーとコンシューマーを分離できます。 これまでに取り上げたアーキテクチャーの Point-to-Point スタイルのメッセージングとは、この点が異なります。前述のアーキテクチャーでは、アプリケーションにおいて、ターゲット・アプリケーション (メッセージの書き込み先のキュー名など) についての情報が必要になります。 IBM MQ のパブリッシュ/サブスクライブ構成を使用する場合、送信側のアプリケーションは、情報のサブジェクトに基づいて指定されたトピックにメッセージをパブリッシュします。 その後、IBM MQ がメッセージの配布を処理します。 つまり、サブスクリプションによってそのサブジェクトを興味の対象として登録しているアプリケーションにメッセージを配布します。 受信側のアプリケーションも、メッセージを受信するために、そのソースについて何かの情報を知っておく必要はありません。 詳しくは、『パブリッシュ/サブスクライブ・メッセージング』および『単一キュー・マネージャーのパブリッシュ/サブスクライブ構成の例』を参照してください。

関連概念

IBM MQ の概要

関連タスク

5 ページの『IBM MQ アーキテクチャーの計画』

IBM MQ 環境を計画する際、単一および複数キュー・マネージャーのアーキテクチャーについて、また Point-to-Point およびパブリッシュ/サブスクライブのメッセージング・スタイルについて IBM MQ が提供するサポートを考慮します。 また、リソース要件、およびロギングやバックアップの機能の使用方法を計画します。

マルチプラットフォームでのキュー・マネージャーの作成と管理

複数のキュー・マネージャーに基づくアーキテクチャー

分散メッセージ・キューイングの手法を使用して、複数のキュー・マネージャーの構成と使用を含む IBM MQ アーキテクチャーを作成できます。

IBM MQ のアーキテクチャーを計画する前に、IBM MQ の基本的な概念をよく理解することが必要です。『IBM MQ の技術概要』を参照してください。

追加のキュー・マネージャーを加えることにより、サービスを提供するアプリケーションを変更せずに IBM MQ アーキテクチャーを変更することができます。

キュー・マネージャーと同じマシン上でアプリケーションをホストしてから、別のシステム上の別のキュー・マネージャー上でホストされているサービスとの非同期通信を行うことができます。 または、サービスにアクセスしているアプリケーションをクライアントとしてキュー・マネージャーに接続してから、別のキュー・マネージャー上のサービスに非同期アクセスすることもできます。

さまざまなキュー・マネージャーとそのキューを接続する経路は、分散キューイングの手法を使用して定義します。 アーキテクチャー内のキュー・マネージャーは、チャネルを使用して接続されます。 チャネルを使用すると、キュー・マネージャーの構成に応じて、キュー・マネージャー間でメッセージが一方向に自動的に移動します。

IBM MQ ネットワークの計画の概要については、21 ページの『分散キュー・マネージャー・ネットワークの設計』を参照してください。

IBM MQ アーキテクチャー用にチャネルを計画する方法については、IBM MQ 分散キューイング技法を参照してください。

分散キュー管理を使用すると、キュー・マネージャー間の通信を作成してモニターできます。 分散キュー管理について詳しくは、分散キュー管理の概要を参照してください。

関連タスク

5 ページの『IBM MQ アーキテクチャーの計画』
IBM MQ 環境を計画する際、単一および複数キュー・マネージャーのアーキテクチャーについて、また Point-to-Point およびパブリッシュ/サブスクライブのメッセージング・スタイルについて IBM MQ が提供するサポートを考慮します。 また、リソース要件、およびロギングやバックアップの機能の使用方法を計画します。

マルチプラットフォームでのキュー・マネージャーの作成と管理

分散キューおよびクラスターの計画

分散キュー・マネージャー上でホストされるキューに手動で接続することは可能ですが、キュー・マネージャー・クラスターを作成して製品の機能によりキュー・マネージャーに自動接続することも可能です。 実際の分散メッセージング・ネットワークのために適切なトポロジーを選択するには、手動制御の要件、ネットワークのサイズ、変更の頻度、アベイラビリティー、およびスケーラビリティーについて考慮する必要があります。

始める前に

このタスクでは、担当者が分散メッセージング・ネットワークとその動作についてよく理解していることが前提とされています。 技術概要について詳しくは、『分散キューイングとクラスター』を参照してください。

このタスクについて

分散メッセージング・ネットワークを作成するには、異なる複数のキュー・マネージャー上でホストされるキューに接続するチャネルを手動で構成するか、またはキュー・マネージャー・クラスターを作成することができます。 クラスタリングを使用すると、キュー・マネージャーは、追加のチャネル定義やリモート・キュー定義をセットアップせずに相互に通信できるため、構成および管理が単純化されます。

実際の分散パブリッシュ/サブスクライブ・ネットワークのために適切なトポロジーを選択するには、以下のさまざまな要素を考慮する必要があります。

- ネットワーク内の接続に対してどの程度の手動制御を必要とするか?
- ネットワークの規模はどの程度の大きさか?
- どの程度動的に変化するか?
- アベイラビリティーとスケーラビリティーの要件は何か?

手順

- ネットワーク内の接続に対してどの程度の手動制御を必要とするかを考慮します。

  少数の接続のみ必要な場合、または個々の接続について非常に厳密な定義が必要とされる場合、ネットワークを手動で作成することが必要になるでしょう。

  論理的に関連した複数のキュー・マネージャーの間でデータやアプリケーションを共有することが必要とされる場合、それらをまとめてグループ化し、1 つのキュー・マネージャー・クラスターとすることを考慮してください。
- ネットワークの規模としてどの程度の大きさが必要かを見積もります。
  a) 何個のキュー・マネージャーが必要かを見積もります。 キューは複数のキュー・マネージャー上でホスト可能であることを念頭に置いてください。
  b) クラスターの使用を検討している場合は、フル・リポジトリーとして動作する 2 つのキュー・マネージャーを余分に追加してください。

  大規模ネットワークの場合、接続の構成と保守を手動で実行しようとすると非常に多くの時間を取られてしまうことが少なくないため、クラスターの使用を検討してください。
- ネットワーク・アクティビティーがどの程度動的に変化するかを考慮します。

使用頻度の高いキューについては、処理能力の高いキュー・マネージャー上でホストするように計画してください。

キューを頻繁に作成したり削除したりすることが予期されるなら、クラスターの使用を検討してください。

- アベイラビリティーとスケーラビリティーの要件を考慮します。
  a) キュー・マネージャーの高可用性を保証することが必要かどうかを判断します。 そのような必要がある場合は、その要件が適用されるキュー・マネージャーがどれだけあるかを見積もります。
  b) キュー・マネージャーのうちのあるものが他のものより処理能力が低いかどうかを考慮します。
  c) キュー・マネージャーのうちのいくつかへの通信リンクが他のものよりも脆弱かどうかを考慮します。
  d) 複数のキュー・マネージャー上でキューをホスティングすることについて検討します。

  手動で構成したネットワークおよびクラスターは、可用度とスケーラビリティーのどちらも高くなるよう構成することが可能です。 クラスターを使用する場合は、フル・リポジトリーとして機能する 2 つのキュー・マネージャーを余分に定義する必要があります。 フル・リポジトリーを 2 個設けるなら、フル・リポジトリーの一方が利用不可になっても、クラスターが動作し続けることが保証されます。 フル・リポジトリー・キュー・マネージャーが堅固で処理能力が高く、ネットワーク接続の点で良好であることを確認してください。 フル・リポジトリー・キュー・マネージャーを何らかの他の用途に使用するようには計画しないでください。
- これらの見積もりに基づき、以下のリンクを使用して、キュー・マネージャー間の接続を手動で構成するか、それともクラスターを使用するかを決定してください。

次のタスク

これで、分散メッセージング・ネットワークの構成作業の準備ができました。

関連タスク

分散キューイングの構成

キュー・マネージャー・クラスターの構成

分散キュー・マネージャー・ネットワークの設計

IBM MQ は、キュー・マネージャーおよびチャネルを使用して、アプリケーション間で、ネットワークを介したデータの送受信を行います。 ネットワークを介して各システムを接続するフレームワークを作成するためには、 ネットワーク計画において要件を定義する必要があります。

チャネルは、システムと、通信する必要がある他のシステムとの間に 作成できます。 直接接続していないシステムに接続するために、マルチ・ホップ・チャネルを作成することができます。 各シナリオで説明されたメッセージ・チャネル接続は、22 ページの図 1 でネットワーク・ダイアグラムとして示されています。

チャネルを別々の物理ネットワークにあるシステム間で、またはファイアウォールを介して通信するチャネル間で作成する必要がある場合、IBM MQ Internet Pass-Thru を使用することで構成が単純化される場合があります。 詳しくは、 IBM MQ Internet Pass-Thru を参照してください。

チャネルと伝送キューの名前

伝送キューには任意の名前を付ける ことができます。 ただし、混乱を避けるためには、適宜、 宛先キュー・マネージャーの名前または別名と同じ名前を 付けるようにします。 こうすると、伝送キューに、その伝送キューで使用する経路が関連付けられるため、 中間 (マルチ・ホップ) のキュー・マネージャーを介して作成された 並列経路の概要が明確になります。

チャネル名については、あまり分かりやすくはなりません。 例えば、22 ページの図 1 で示された QM2 のチャネル名は、着信チャネルと発信チャネルとで異なっていなければなりません。 この場合にも、すべてのチャネル名には伝送キューの名前を付けることができますが、これらの名前を修飾して固有なものにしなければなりません。

例えば、QM2 には、QM1 から接続されている QM3 チャネルがあり、この QM3 チャネルは QM3 に接続されています。 これらの名前を固有なものにするには、最初のチャネルには QM3_from_QM1 という名前を

付け、2 番目のチャネルには QM3_from_QM2 という名前を付けることができます。 このようにすると、チャネル名の最初の部分にその伝送キューの名前が示され、 名前の 2 番目の部分には方向および隣接キュー・マネージャーの名前が示されます。

22 ページの図 1 の場合に推奨されるチャネル名の例が、22 ページの表 1 に示してあります。

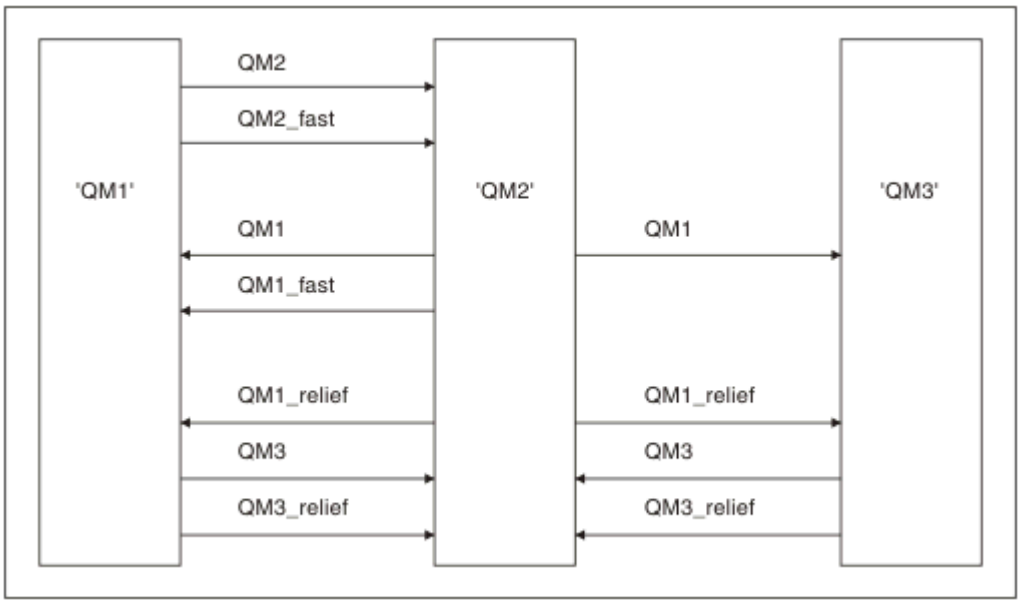


図 *1.* すべてのチャネルを表すネットワーク・ダイアグラム

表 *1.* チャネル名の例

経路名	チャネルのホストとなるキュー・マネージャー	伝送キュー名	推奨されるチャネル名
QM1	QM1 & QM2	QM1 (QM2 内)	QM1.from.QM2
QM1	QM2 & QM3	QM1 (QM3 内)	QM1.from.QM3
QM1_fast	QM1 & QM2	QM1_fast (QM2 内)	QM1_fast.from.QM2
QM1_relief	QM1 & QM2	QM1_relief (QM2 内)	QM1_relief.from.QM2
QM1_relief	QM2 & QM3	QM1_relief (QM3 内)	QM1_relief.from.QM3
QM2	QM1 & QM2	QM2 (QM1 内)	QM2.from.QM1
QM2_fast	QM1 & QM2	QM2_fast (QM1 内)	QM2_fast.from.QM1
QM3	QM1 & QM2	QM3 (QM1 内)	QM3.from.QM1
QM3	QM2 & QM3	QM3 (QM2 内)	QM3.from.QM2
QM3_relief	QM1 & QM2	QM3_relief (QM1 内)	QM3_relief.from.QM1
QM3_relief	QM2 & QM3	QM3_relief (QM2 内)	QM3_relief.from.QM2

注 :

1. z/OS IBM MQ for z/OS では、キュー・マネージャー名は 4 文字までに制限されています。
2. ネットワーク内のすべてのチャネルに固有の名前を付けてください。 22 ページの表 1 に示すように、発信元および宛先のキュー・マネージャー名をチャネル名に含める方法を推奨します。

ネットワーク計画者

ネットワークの作成にあたっては、より高レベルのネットワーク計画者 の役割が前提となります。ネットワーク計画者が立てた計画は、チームの別のメンバーによって実現されます。

広範囲に使用されるアプリケーションの場合には、 23 ページの図 2 に示すように、 メッセージ・トラフィックを集中させるローカル・アクセス・サイトを設けて、 各ローカル・アクセス・サイト間で広帯域リンクを使用すれば、コストをより低く抑えることができます。

この例では、 2 つのメイン・システムといくつかのサテライト・システムが あります。 実際の構成は、ビジネス上の考慮事項によって異なります。 2 つのキュー・マネージャー・コンセントレーターがキュー・マネージャーの間に配置されています。 各 QM コンセントレーターには、次のように、ローカル・キュー・マネージャーへのメッセージ・チャネルがあります。

- QM コンセントレーター 1 には、3 つのローカル・キュー・マネージャー QM1、QM2、QM3 のそれぞれに通じるメッセージ・チャネルがあります。 これらのキュー・マネージャーを使用するアプリケーションは、QM コンセントレーターを使用して相互に通信できます。
- QM コンセントレーター 2 には、3 つのローカル・キュー・マネージャー QM4、QM5、QM6 のそれぞれに通じるメッセージ・チャネルがあります。 これらのキュー・マネージャーを使用するアプリケーションは、QM コンセントレーターを使用して相互に通信できます。
- QM コンセントレーター間にはメッセージ・チャネルがあり、あるキュー・マネージャーのロケーションにあるアプリケーションが別のキュー・マネージャーのロケーションにある任意のアプリケーションとメッセージを交換できるようになっています。

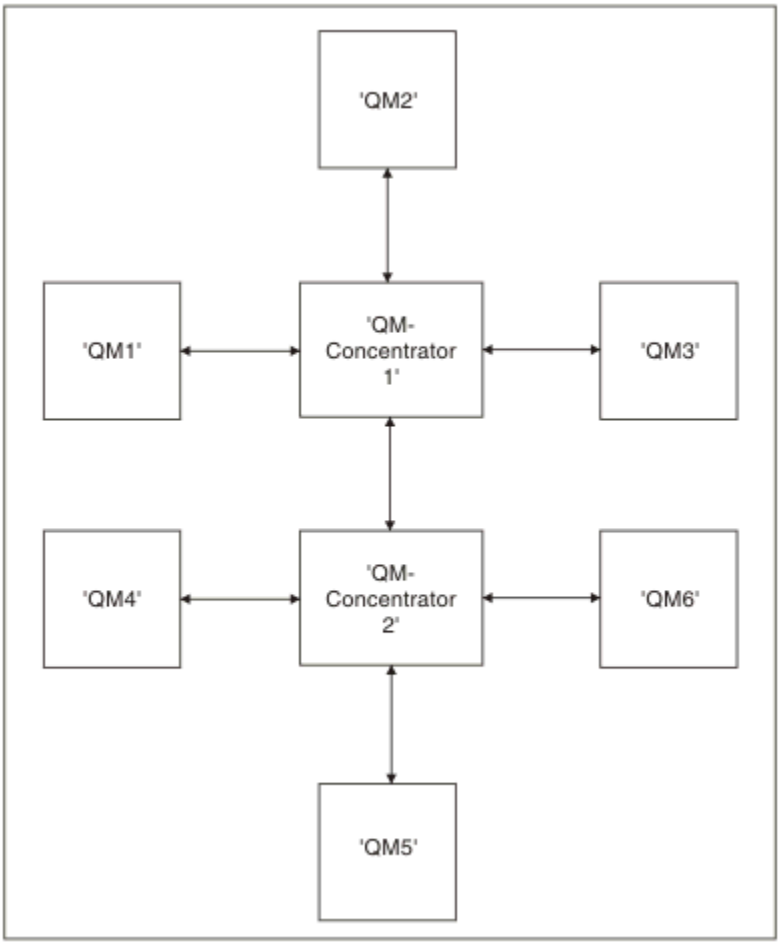


図 *2. QM* コンセントレーターを表すネットワーク・ダイアグラム

クラスターの設計

クラスターによって提供されるキュー・マネージャーを相互接続するためのメカニズムにより、初期構成と継続的な管理を簡単に行えます。 クラスターが正しく機能することと、必要なレベルの可用性と応答性をクラスターが達成することが大切ですので、クラスターは慎重に設計する必要があります。

始める前に

クラスター化の概念の概要については、以下のトピックを参照してください。

- 分散キューイングとクラスター
- 29 ページの『クラスター化と分散キューイングとの比較』
- クラスターのコンポーネント

キュー・マネージャー・クラスターを設計するときは、いくつかの決定を行う必要があります。 まず、クラスター内のどのキュー・マネージャーがクラスター情報の完全リポジトリーを保持するかを決定する必要があります。 作成するどのキュー・マネージャーもクラスター内で機能できます。 この目的のためのキュー・マネージャーはいくつでも選択できますが、理想的な数は 2 つです。 完全リポジトリーを保持するキュー・マネージャーの選択については、32 ページの『完全リポジトリーを保持するクラスター・キュー・マネージャーの選択方法』を参照してください。

クラスターの設計について詳しくは、以下のトピックを参照してください。

- 38 ページの『サンプル・クラスター』
- 33 ページの『クラスターの編成』
- 34 ページの『クラスターの命名規則』
- z/OS 35 ページの『Queue sharing groups and clusters』
- 35 ページの『クラスターのオーバーラップ』

次のタスク

クラスターの構成と処理について詳しくは、以下のトピックを参照してください。

- クラスター内での通信の確立
- キュー・マネージャー・クラスターの構成
- クラスターとの間のルーティング・メッセージ
- クラスターによるワークロードの管理

クラスターの構成に役立つ詳細情報は、36 ページの『クラスター化のヒント』を参照してください。

複数のクラスター伝送キューの使用方法の計画

伝送キューは、明示的に定義するか、システムに自動生成させることができます。 伝送キューを自分で定義する場合、キュー定義の内容を細かく制御できます。 z/OS z/OS では、メッセージが保持されるページ・セットも細かく制御できます。

伝送キューの定義

伝送キューを定義するには、次の 2 つの方法があります。

- キュー・マネージャーの属性 DEFCLXQ を以下のように使用して、自動で行う。

```
ALTER QMGR DEFCLXQ(SCTQ | CHANNEL)
```

DEFCLXQ(SCTQ) は、すべてのクラスター送信側チャネルのデフォルト伝送キューが SYSTEM.CLUSTER.TRANSMIT.QUEUE であることを示します。 これがデフォルト値です。

DEFCLXQ(CHANNEL) は、デフォルトで、 各クラスター送信側チャネルが SYSTEM.CLUSTER.TRANSMIT.*channel name* という名前の別個の伝送キューを使用することを示しま

す。 各伝送キューは、キュー・マネージャーによって自動的に定義されます。 詳しくは、26 ページの『自動的に定義されるクラスター伝送キュー』を参照してください。

- CLCHNAME 属性に指定された値によって伝送キューを定義して、手動で行う。 CLCHNAME 属性は、伝送キューを使用するクラスター送信側チャネルを示します。 z/OS 手動で送信キューを定義する場合 z/OS、 見る 27 ページの『手動で定義されるクラスター伝送キューの計画』詳細については。

必要なセキュリティー

自動または手動のいずれにおいても、切り替えを開始するには、チャネルを開始する権限が必要です。

伝送キューとして使用するキューを定義するには、キューを定義するための標準的な IBM MQ 権限が必要です。

変更を実施するのに適したタイミング

クラスター送信側チャネルで使用される伝送キューを変更する場合は、以下の点を検討して、更新を行う時間を割り振る必要があります。

- チャネルが伝送キューを切り替えるタイミングは、古い伝送キュー上にあるメッセージの総数、移動する必要があるメッセージの数、およびメッセージのサイズによって異なります。
- 変更を実行している間も、アプリケーションは引き続き伝送キューにメッセージを書き込めます。 これにより、移行時間が長くなる可能性があります。
- 伝送キューの CLCHNAME パラメーターまたは DEFCLXQ はいつでも変更できますが、可能であればワークロードの少ないときを選んでください。

  すぐには何も起こらないことに注意してください。
- チャネルを開始または再開したときにのみ、変更は行われます。 チャネルは、開始するときに現在の構成をチェックし、必要であれば新しい伝送キューに切り替えます。
- クラスター送信側チャネルと伝送キューの関連付けを変える可能性のあるいくつかの変更を次に示します。
  - 伝送キューの CLCHNAME 属性の値を変更して、CLCHNAME をより具体的ではない値またはブランクにする。
  - 伝送キューの CLCHNAME 属性の値を変更して、CLCHNAME をより具体的な値にする。
  - CLCHNAME を指定したキューを削除する。
  - キュー・マネージャー属性 DEFCLXQ を変更する。

切り替えにかかる時間

移行期間には、チャネルのすべてのメッセージが、1 つの伝送キューから別の伝送キューに移されます。 チャネルが伝送キューを切り替えるために要する時間は、 古い伝送キュー上にあるメッセージの総数、および移動する必要があるメッセージの数によって決まります。

数千個のメッセージが含まれているキューの場合、メッセージの移動にかかる時間はおそらく 1 秒未満です。 実際の時間は、メッセージの数とサイズによって決まります。 キュー・マネージャーは、メッセージを 1 秒につき何メガバイトも移動できるはずです。

変更を実行している間も、アプリケーションは引き続き伝送キューにメッセージを書き込めます。 これにより、移行時間が長くなる可能性があります。

変更を有効にするには、影響を受ける各クラスター送信側チャネルを再始動する必要があります。 そのため、伝送キュー構成は、キュー・マネージャーがビジーではなく、クラスター伝送キュー上に保管されているメッセージ数が少ないときに変更するのがベストです。

の **runswchl** 指示 z/OS または CSQUTIL の SWITCH CHANNEL(*) STATUS コマンドの上 z/OS クラスター送信チャネルのステータスと、その送信キュー構成に対して保留中の変更が何であるかを照会するために使用できます。

変更の実施方法

自動または手動で複数のクラスター伝送キューに変更を加える方法の詳細 については、複数のクラスター伝送キューを使用したシステムの実装を参照 してください。

変更の取り消し

z/OS

問題が発生した場合に変更をバックアウトする方法について詳しくは、z/OS での伝送キューの変更の取り消し を参照してください。

自動的に定義されるクラスター伝送キュー
伝送キューをシステムにより自動的に生成することができます。

始める前に

z/OS z/OS でクラスター伝送キューを手動でセットアップするには、 27 ページの『手動で定義されるクラスター伝送キューの計画』を参照してください。

このタスクについて

手動で定義されたクラスター伝送キューがチャネルに関連付けていない場合、DEFCLXQ(CHANNEL) を指定すると、チャネルの始動時にキュー・マネージャーはクラスター送信側チャネル用の永続的な動的キューを自動的に定義します。 モデル・キュー SYSTEM.CLUSTER.TRANSMIT.MODEL.queue を使用して、SYSTEM.cluster.transmit.ChannelName という名前の永続動的クラスター伝送キューが自動的に定義されます。

重要 : z/OS IBM MQ 8.0 では、キュー・マネージャーに SYSTEM.CLUSTER.TRANSMIT.MODEL.QUEUE はありません。 IBM MQ 8.0 からこのバージョンに直接マイグレーションすることはできません。 IBM MQ 8.0 からマイグレーションされたキュー・マネージャーへの SYSTEM.CLUSTER.TRANSMIT.MODEL.QUEUE の追加については、キュー・マネージャーのマイグレーションに使用した暫定バージョンの資料でこのトピックを参照してください。

手順

1. *DEFCLXQ* キュー・マネージャー属性を使用します。
   この属性について詳しくは、ALTER QMGR を参照してください。

   次の 2 つのオプションがあります。

   SCTQ
   これはデフォルトのオプションで、単一の SYSTEM.CLUSTER.TRANSMIT.QUEUE を使用することを意味します。

   CHANNEL
   複数のクラスター伝送キューを使用することを意味します。
2. 新しいアソシエーションに切り替えるには、次のようにします。
   - チャネルを停止してから再始動します。
   - チャネルは、新しい伝送キュー定義を使用します。
   - 一時的な切り替えプロセスによって古いキューから新しい伝送キューにメッセージが転送されます。

   アプリケーション・メッセージは古い定義に置かれることに注意してください。

   古いキュー上のメッセージ数がゼロになると、新しいメッセージは新しい伝送キューに直接配置されるようになります。
3. 切り替えプロセスが完了したことをモニターするには、次のようにします。

a) チャネルによって開始された伝送キューの切り替えはバックグラウンドで実行され、管理者はキュー・マネージャーのジョブ・ログをモニターすることにより、切り替えが完了したことを判別できます。

b) 切り替えの進行を示すジョブ・ログ上のメッセージをモニターします。

c) 該当するチャネルだけがこの伝送キューを使用していることを確認するには、コマンド DIS CLUSQMGR(*) を発行します (例えば、伝送キューを定義する伝送キュー・プロパティーは、APPQMGR.CLUSTER1.XMITQ です)。

d) z/OS
CSQUTIL の下で SWITCH CHANNEL (*) STATUS コマンドを使用します。
このオプションを使用すると、未処理で保留になっている変更が通知され、伝送キュー間で移動する必要のあるメッセージの数が表示されます。

タスクの結果

これで、クラスター伝送キュー (1 つまたは複数) のセットアップが完了しました。

関連タスク

27 ページの『手動で定義されるクラスター伝送キューの計画』
IBM MQ for z/OS では、伝送キューを自分で定義すると、定義、およびメッセージが保持されるページ・セットをより詳細に制御することができます。

関連資料

ALTER QMGR

DISPLAY CLUSQMGR

z/OS 手動で定義されるクラスター伝送キューの計画

IBM MQ for z/OS では、伝送キューを自分で定義すると、定義、およびメッセージが保持されるページ・セットをより詳細に制御することができます。

始める前に

クラスター伝送キューを自動的にセットアップするには、26 ページの『自動的に定義されるクラスター伝送キュー』を参照してください。

このタスクについて

管理者は、伝送キューを手動で定義し、キュー属性 CLCHNAME を使用して、このキューを伝送キューとして使用するクラスター送信側チャネルを定義します。

なお、CLCHNAME には先頭または末尾にワイルドカード文字を指定できるため、1 つのキューを複数のチャネルで使用することができます。

手順

1. 例えば、次のように入力します。

```
DEFINE QLOCAL(APPQMGR.CLUSTER1.XMITQ)
CLCHNAME(CLUSTER1.TO.APPQMGR)
USAGE(XMITQ) STGCLASS(STG1)
INDXTYPE( CORRELID ) SHARE

DEFINE STGCLASS(STG1) PSID(3)
DEFINE PSID(3) BUFFERPOOL(4)
```

ヒント: 伝送キューにどのページ・セット (およびバッファー・プール) を使用するかを計画する必要があります。 キューごとに異なるページ・セットを設定し、それらを分離することができます。そのため、1 つのページ・セットがいっぱいになっても、他のページ・セットの伝送キューには影響しません。

各チャネルが適切なキューをどのように選択するかについて詳しくは、クラスター伝送キューとクラスター送信側チャネルを参照してください。

チャネルが始動すると、チャネルの関連付けが新しい伝送キューに切り替わります。 メッセージが失われないようにするために、キュー・マネージャーは、古いクラスター伝送キューから新しい伝送キューにメッセージを順番に自動的に転送します。

2. CSQUTIL SWITCH 関数を使用して新しい関連に変更します。

   詳細については、Switch the transmission queue associated with cluster-sender channels (SWITCH) を参照してください。

   a) 伝送キューを変更するチャネル (1 つまたは複数) を STOP して、それらを STOPPED 状況にします。

   以下に例を示します。

   ```
   STOP CHANNEL(CLUSTER1.TO.APPQMGR)
   ```

   b) 伝送キューの CLCHNAME(XXXX) 属性を変更します。

   c) SWITCH 関数を使用すると、メッセージを切り替えたり、何が起きているかをモニターしたりできます。

   次のコマンドを使用すると、

   ```
   SWITCH CHANNEL(*) MOVEMSGS(YES)
   ```

   チャネルを開始せずにメッセージを移動できます。

   d) チャネル (1 つまたは複数) を開始し、チャネルで正しいキューが使用されているかどうか検査します。

   以下に例を示します。

   ```
   DIS CHS(CLUSTER1.TO.APPQMGR)
   DIS CHS(*) where(XMITQ eq APPQMGR.CLUSTER1.XMITQ)
   ```

ヒント: 以下のプロセスでは、CSQUTIL SWITCH 機能を使用します。 詳しくは、 クラスター送信側チャネルに関連付けられた伝送キューの切り替え (SWITCH)を参照してください。

必ずしもこの関数を使用する必要はありませんが、この関数を使用すると、より多くのオプションが提供されます。

- SWITCH CHANNEL (*) STATUS を使用すると、クラスター送信側チャネルの切り替え状況を簡単に識別できます。 これを使用すると、管理者は、現在切り替え中のチャネルがどれか、切り替えが保留中で次回のチャネル始動時に有効になるチャネルがどれかを確認できます。

  この機能を利用できないとすると、管理者は、複数の DISPLAY コマンドを使用し、その出力結果を処理したうえで、この情報を確認する必要があります。 また、管理者は、構成変更の結果が要求どおりであることを確認することもできます。

- CSQUTIL を使用して切り替えを開始すると、CSQUTIL によってこの操作の進行が引き続きモニターされ、切り替えが完了した時点で終了します。

  これを利用すると、これらの操作をバッチで実行するのが簡単になります。 また、CSQUTIL で複数のチャネルの切り替えを実行すると、CSQUTIL はそれらのアクションを順番に実行します。そのため、複数の切り替えを並行して実行するよりも、企業に与える影響を小さくできます。

タスクの結果

z/OS でクラスター伝送キュー (複数可) をセットアップしました。

アクセス制御と複数のクラスター伝送キュー

アプリケーションがメッセージをリモート・クラスター・キューに入れるタイミングをチェックするモードを 3 つの中から選択します。 これらのモードはそれぞれ、リモートでのクラスター・キューに対するチェック、ローカルでの SYSTEM.CLUSTER.TRANSMIT.QUEUE に対するチェック、クラスター・キューまたはクラスター・キュー・マネージャーのローカル・プロファイルに対するチェックを行います。

IBM MQ では、ユーザーにメッセージをリモート・キューに入れるためのアクセス権があることをローカルでチェックするか、またはローカルとリモートでチェックするかを選択できます。 典型的な IBM MQ アプリケーションはローカル・チェックのみを使用し、ローカル・キュー・マネージャーで行われるアクセス・

チェックを信用するリモート・キュー・マネージャーに依存します。 リモート・チェックが使用されない場合、メッセージはリモート・メッセージ・チャネル・プロセスの権限でターゲット・キューに入れられます。 リモート・チェックを使用するには、受信側チャネルの書き込み権限をコンテキスト・セキュリティーに設定する必要があります。

ローカル・チェックは、アプリケーションがオープンするキューに対して行われます。 分散キューイングでは、アプリケーションが通常、リモート・キュー定義をオープンするため、リモート・キュー定義に対してアクセス・チェックが行われます。 メッセージが完全なルーティング・ヘッダーで書き込まれている場合には、伝送キューに対するチェックが行われます。 アプリケーションがローカル・キュー・マネージャー上にはないクラスター・キューをオープンした場合、チェック対象のローカル・オブジェクトはありません。 アクセス制御チェックは、クラスター伝送キュー SYSTEM.CLUSTER.TRANSMIT.QUEUE に対して行われます。 複数のクラスター伝送キューがある場合でも、リモート・クラスター・キューのローカル・アクセス制御検査は SYSTEM.CLUSTER.TRANSMIT.QUEUE に対して行われます。

ローカル・チェックとリモート・チェックのどちらを選択するかは、2 つの極端な選択となります。 リモートでは、チェックが微細化されます。 クラスター・キューに書き込むには、すべてのユーザーがすべてのキュー・マネージャーにアクセス制御プロファイルを持っている必要があります。 ローカルでは、チェックが粗視化されます。 すべてのユーザーに必要なのは、接続先のキュー・マネージャー上のクラスター伝送キューのアクセス制御プロファイルだけです。 そのプロファイルを使用して、任意のクラスター内の任意のキュー・マネージャーにある任意のクラスター・キューにメッセージを書き込むことができます。

管理者は、別の方法でクラスター・キューのアクセス制御をセットアップできます。 **setmqaut** コマンドを使用して、クラスター内の任意のキュー・マネージャー上にクラスター・キューのセキュリティー・プロファイルを作成できます。 このプロファイルは、キュー名のみを指定してリモートクラスタキューをローカルに開いた場合に有効になります。 リモート・キュー・マネージャーのプロファイルをセットアップすることもできます。 その場合、キュー・マネージャーは、完全修飾名を指定して、クラスター・キューをオープンしたユーザーのプロファイルをチェックできます。

新しいプロファイルは、キューマネージャスタンザ **ClusterQueueAccessControl** の設定を RQMName に変更した場合にのみ動作します。 デフォルトは Xmitq です。 この設定を変更する前に、リモート・クラスタ・キューにアクセスする既存のすべてのアプリケーションに対してプロファイルを作成していることを確認してください。 プロファイルを作成せずにスタンザ設定を RQMN 名に変更すると、アプリケーションが失敗する可能性が高い。

ヒント: クラスター・キュー・アクセス検査は、リモート・キューイングには適用されません。 アクセス・チェックは、引き続きローカル定義に対して行われます。 これらの変更は、クラスター・キューおよびクラスター・トピックでのアクセス・チェックを構成する場合にも、同じ手法に従えることを意味します。

z/OS また、これらの変更は、クラスター・キューのアクセス・チェック手法をより密接に z/OS と一致させています。 z/OS でアクセス・チェックをセットアップするためのコマンドは異なりますが、どちらもオブジェクト自体ではなく、プロファイルに対してアクセス・チェックを行います。

関連概念

48 ページの『クラスター化: 複数のクラスター伝送キューの使用によるアプリケーションの分離』
クラスター内のキュー・マネージャー間のメッセージ・フローは、分離することができます。 さまざまなクラスター送信側チャネルによって転送されるメッセージを、それぞれに異なるクラスター伝送キューに配置できます。 この手法は、単一のクラスターでも、オーバーラップするクラスターでも使用できます。 このトピックでは、使用する手法を選択する際に参考となる例およびベスト・プラクティスを説明します。

関連タスク

設定 ClusterQueueAccessControl

クラスター化と分散キューイングとの比較

キュー・マネージャーに接続するために定義する必要があるコンポーネントを、分散キューイングを使用する場合とクラスター化を使用する場合で比較します。

クラスターを使用しない場合、キュー・マネージャーはそれぞれ独立したプログラムとなり、分散キューイングでデータを送受信します。 この場合、あるキュー・マネージャーから別のキュー・マネージャーにメッセージを送信するには、以下のものを定義する必要があります。

- 伝送キュー

- リモート・キュー・マネージャーへのチャネル

30 ページの図 3 に、分散キューイングの場合に必要なコンポーネントを示します。

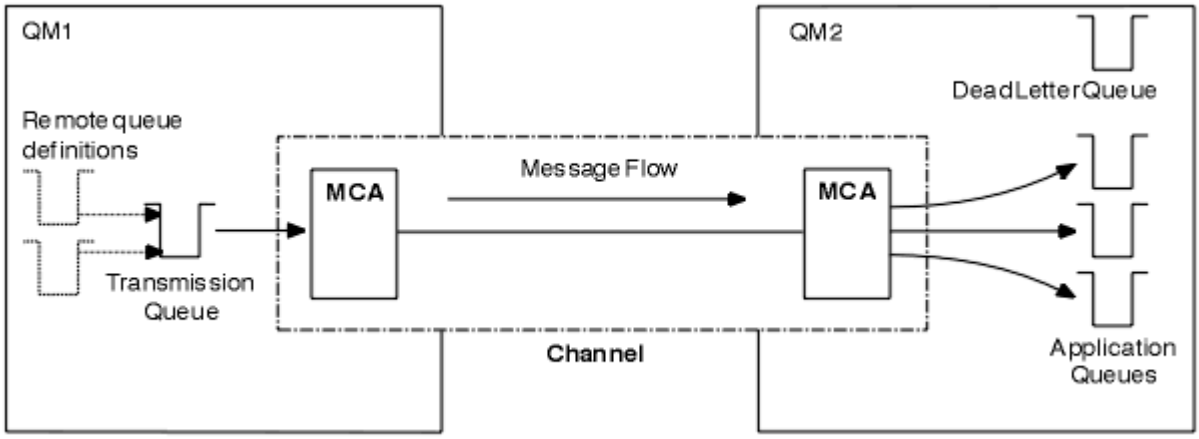


図 *3.* 分散キュー

複数のキュー・マネージャーを 1 つのクラスターにグループ化した場合、どのキュー・マネージャー上のキューも、そのクラスター内にある他のキュー・マネージャーから使用できるようになります。 どのキュー・マネージャーも、同じクラスター内の別のキュー・マネージャーへ明示的な定義なしでメッセージを送信することができます。 宛先ごとにチャネル定義、リモート・キュー定義、伝送キューを指定しません。 クラスターを構成するキュー・マネージャーには伝送キューがそれぞれ 1 つずつ設定されています。この伝送キューにより、同じクラスター内の別のキュー・マネージャーにメッセージを送信することができます。 したがって、クラスター内の各キュー・マネージャーに定義する必要があるのは次の 2 つだけです。

- クラスター受信側チャネル。メッセージを受信するために使用します。
- クラスター送信側チャネル。自分自身の情報を取り込んだり、クラスターを把握したりします。

設定時の定義項目に関するクラスターと分散キューイングとの比較

以降では、4 つのキュー・マネージャーで構成されるネットワークを設定する場合を例にして説明します。各キュー・マネージャーにはそれぞれ 2 つのキューが格納されているものとします。30 ページの図 4 は、そのようなキュー・マネージャーで構成されるネットワークの例です。 最初に、これらのキュー・マネージャーを分散キューイングで接続する場合に、いくつの項目を定義する必要があるのかについて説明します。 次に、同じネットワークをクラスターとして設定する場合には、いくつの項目を定義する必要があるのかを比較します。

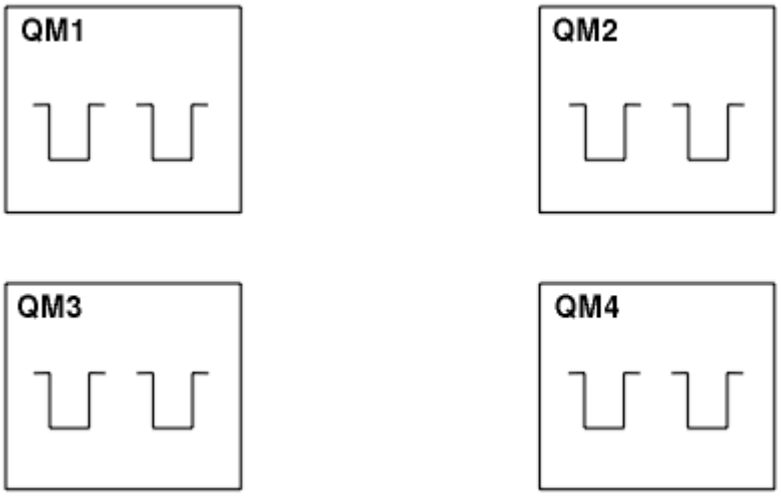


図 *4. 4* つのキュー・マネージャーで構成されるネットワークの例

分散キューイングによりネットワークを設定する場合に定義する項目

以下の表は、30 ページの図 3 に示すネットワークを分散キューイングにより設定する場合に定義する項目を示しています。

表 2. 分散キューイングの場合に定義する項目

説明	各キュー・マネージャーに定義する数	合計数
送信側チャネル定義: 他のキュー・マネージャーにメッセージを送信するときに使用するチャネルを示します。	3	12
受信側チャネル定義: 他のキュー・マネージャーからメッセージを受信するときに使用するチャネルを示します。	3	12
伝送キュー定義: 他のキュー・マネージャーにメッセージを送信するときに使用する伝送キューを示します。	3	12
ローカル・キュー定義: 個々のローカル・キューを示します。	2	8
リモート・キュー定義: メッセージの書き込み先となる個々のリモート・キューを示します。	6	24

汎用受信側チャネルを定義することにより、定義の数を減らすこともできます。 各キュー・マネージャーにつき最大で 17 の定義、つまり、ネットワークの合計で 68 の定義が必要です。

クラスター方式によりネットワークを設定する場合に定義する項目

30 ページの図 3 に示すネットワークをクラスター方式により設定する場合は、以下の表に示す項目を定義する必要があります。

表 3. クラスター方式の場合に定義する項目

説明	各キュー・マネージャーに定義する数	合計数
クラスター送信側チャネル定義: リポジトリー・キュー・マネージャーにメッセージを送信するときに使用するチャネルを示します。	1	4
クラスター受信側チャネル定義: クラスター内にある他のキュー・マネージャーからメッセージを受信するときに使用するチャネルを示します。	1	4
ローカル・キュー定義: 個々のローカル・キューを示します。	2	8

これらのキュー・マネージャーで構成されるクラスターを (2 つの完全リポジトリーと共に) 設定する場合、各キュー・マネージャーにつき 4 つの定義、つまり、合計で 16 の定義が必要です。 また、いずれか 2 つのキュー・マネージャーの定義を修正して、それらのキュー・マネージャーをそのクラスターの完全リポジトリー・キュー・マネージャーにする必要があります。

1 つの CLUSSDR チャネル定義と 1 つの CLUSRCVR チャネル定義のみ必要です。 クラスターを定義すれば、(リポジトリー・キュー・マネージャー以外の) キュー・マネージャーを追加したり削除したりしても、他のキュー・マネージャーが壊れることはありません。

クラスターを使用すると、多数のキュー・マネージャーで構成されるネットワークを設定する場合に、必要な定義の数が減ります。

定義する項目の数が少なければ、エラーが発生することも少なくなります。以下に例を示します。

- オブジェクト名が、例えば送信側と受信側が対になっているチャネル名と常に一致します。

- チャネル定義に指定された伝送キュー名は、正しい伝送キュー定義か、またはリモート・キュー定義に指定された伝送キュー名と常に一致します。
- QREMOTE 定義は、リモート・キュー・マネージャーにある正しいキューを常に指し示します。

クラスターをいったん設定したあとは、クラスター内のあるキュー・マネージャーから別のキュー・マネージャーにクラスター・キューを移動しても、その他のキュー・マネージャーでシステム管理上の作業を行う必要がありません。 したがって、チャネル、リモート・キュー、または伝送キューの定義を削除し忘れたり、変更し忘れることはありません。 新しいキュー・マネージャーをクラスターに追加する場合にも、既存のネットワーク環境を破壊することなく追加できます。

完全リポジトリーを保持するクラスター・キュー・マネージャーの選択方法

それぞれのクラスターで、完全リポジトリーを保持するためのキュー・マネージャーを少なくとも 1 つ、できれば 2 つ選択する必要があります。 きわめて例外的な状況を除き、完全リポジトリーは 2 つあれば十分です。 キュー・マネージャーは、可能であれば、永続的に接続された堅固なプラットフォームでホストされ、2 台同時に停止しないものを選択します。地理的に中心位置にあるものが理想です。 また、システムを完全リポジトリー・ホスト専用にし、他の作業に使用しないシステムにすることも検討してください。

完全リポジトリー とは、クラスターの状態に関するすべての情報を保持するキュー・マネージャーのことです。 この情報を共有するために、それぞれの完全リポジトリーは、クラスター内の他の完全リポジトリーそれぞれに CLUSSDR チャネル (およびそれらに対応する CLUSRCVR 定義) で接続されます。 これらのチャネルは手動で定義する必要があります。

図 *5.* 接続された *2* つの完全リポジトリー

クラスター内の他のキュー・マネージャーは、それぞれが現在認識している範囲で、クラスターの状態に関する情報を部分リポジトリー に保持します。 これらのキュー・マネージャーは、自分自身に関する情報をパブリッシュするときや、他のキュー・マネージャーに関する情報を要求するときには、使用可能な 2 つの完全リポジトリーを使用します。 選択された完全リポジトリーが使用不可の場合は、もう 1 つの完全リポジトリーが使用されます。 選択された完全リポジトリーが再び使用可能になると、他の完全リポジトリーから最新情報や変更情報を収集して、内容を合わせます。 すべての完全リポジトリーがサービスを休止した場合、 それ以外のキュー・マネージャーは部分リポジトリーに保存されている情報を使用します。ただし、使用できるのは自らが持っている情報に限られており、新情報や更新要求を処理することはできません。 完全リポジトリーがネットワークに再接続すると、メッセージが交換され、すべてのリポジトリー (完全リポジトリーと部分リポジトリーの両方) が最新状態になります。

完全リポジトリーの割り振りを計画する際は、以下の考慮事項を含めてください。

- 完全リポジトリーを保持するために選択するキュー・マネージャーを、信頼性があり管理されたものにする必要があるということです。 永続的に接続された堅固なプラットフォームでホストされるキュー・マネージャーを選択してください。
- 完全リポジトリーをホストするシステムの計画停止を検討して、これらのシステムが同時に停止しないようにします。
- ネットワーク・パフォーマンスを考慮します。つまり、地理的に中心に位置するキュー・マネージャーか、クラスター内の他のキュー・マネージャーと同じシステムを共有するキュー・マネージャーを選択してください。
- キュー・マネージャーが複数のクラスターのメンバーであるかどうかを検討します。 いくつかのクラスターの完全リポジトリーを同一のキュー・マネージャーを使用してホストすると、管理が容易になる場合があります。ただし、この利点をどの程度重視するかは、キュー・マネージャーの予想稼働率がどの程度になるかを踏まえて、平衡を取ることが必要です。
- 一部のシステムは完全リポジトリーだけを収容する専用システムにし、他の作業に使用しないようにすることを検討します。 これにより、これらのシステムで必要となるのはキュー・マネージャー構成の保守だけになり、他のビジネス・アプリケーションの保守のためにサービスを停止することがなくなります。

また、システム・リソースの使用に関して、リポジトリーを保守するタスクがアプリケーションと競合することがなくなります。 これは特に、クラスターの状態を保持するために完全リポジトリーのワークロードが非常に大きくなる大規模クラスター (例えば、1000 を超えるキュー・マネージャーから成るクラスター) で利点があります。

完全リポジトリーを 2 つより多くすることは可能ですが、推奨される状況はほとんどありません。 オブジェクト定義 (つまり、キュー、トピック、およびチャネル) は使用可能なすべての完全リポジトリーに流れますが、要求は部分リポジトリーから最大 2 つの完全リポジトリーにしか流れません。 つまり、完全リポジトリーを 2 つより多く定義しても、いずれか 2 つの完全リポジトリーが使用不可になった場合、部分リポジトリーによっては期待している更新を受け取らない可能性があります。 MQ Clusters: Why only two Full Repositories? を参照してください。

完全リポジトリーを 2 つより多く定義することが有用と思われる 1 つの状況は、既存の完全リポジトリーを新しいハードウェアまたは新しいキュー・マネージャーにマイグレーションする場合です。 この場合は、置換用の完全リポジトリーを導入し、そこにデータが完全に取り込まれたことを確認してから、以前の完全リポジトリーを除去する必要があります。 完全リポジトリーを追加するときは必ず、他のすべての完全リポジトリーに CLUSSDR チャネルで直接接続する必要があることに注意してください。

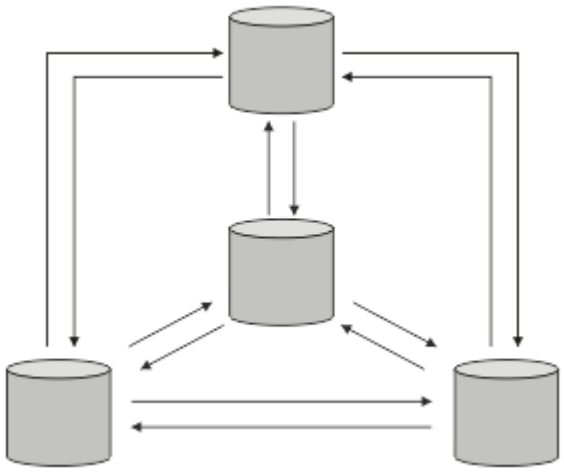


図 *6. 2* つより多く接続された完全リポジトリー

関連情報

MQ Clusters: Why only two Full Repositories?

How big can an MQ Cluster be?

クラスターの編成

どのキュー・マネージャーがどの完全リポジトリーにリンクするかを選択します。 パフォーマンスへの影響やキュー・マネージャーのバージョン、および複数の CLUSSDR チャネルが望ましいかどうかを考慮します。

完全リポジトリーを保持するためのキュー・マネージャーを選択したら、どのキュー・マネージャーをどの完全リポジトリーにリンクするかを決定する必要があります。 CLUSSDR チャネル定義は、クラスター内の他の完全リポジトリーの情報を取り出す完全リポジトリーに、キュー・マネージャーをリンクします。 それ以後、このキュー・マネージャーは任意の 2 つの完全リポジトリーにメッセージを送信します。 このとき常に、まず CLUSSDR チャネル定義がある完全リポジトリーを送信先にしようとします。 キュー・マネージャーをいずれかの完全リポジトリーにリンクすることを選択できます。 選択する場合には、構成のトポロジーを考慮する必要があります。また、キュー・マネージャーの物理的または地理的な場所も考慮する必要があります。

すべてのクラスター情報が 2 つの完全リポジトリーに送信されるため、CLUSSDR チャネル定義をもう 1 つ作成しなければならない場合があります。 多数の完全リポジトリーがあり、広範な領域に広がっている場合は、クラスターに 2 番目の CLUSSDR チャネルを定義することができます。 これで、情報の送信先となる 2 つの完全リポジトリーを制御することができます。

クラスターの命名規則

キュー・マネージャーが属するクラスターを識別する命名規則を使用して、同じクラスター内のキュー・マネージャーの命名について考えます。 チャネル名の命名に類似した命名規則を使用し、チャネルの特性を記述するために命名規則を拡張します。

MQ クラスターの命名時のベスト・プラクティス

クラスター名は最大 48 文字ですが、他のオブジェクトに命名規則を適用する場合は、比較的短いクラスター名が役立ちます。 34 ページの『クラスター・チャネル名を選択する際のベスト・プラクティス』 を参照してください。

クラスター名を選択する際には、通常、「コンテンツ」ではなく、クラスターの「目的」(存続期間が長い可能性が高い) を表すと役立ちます。 例えば、'QM1_QM2_QM3_CLUS' ではなく 'B2BPROD' または 'ACTTEST' です。

クラスター・キュー・マネージャー名を選択する際のベスト・プラクティス

新規クラスターとそのメンバーを最初から作成する場合は、クラスターの使用法を反映するキュー・マネージャーの命名規則を検討してください。 キュー・マネージャーにはそれぞれ異なる名前を付ける必要があります。 ただし、クラスター内のキュー・マネージャーに一連の類似した名前を付けることができます。これにより、論理グループ (例えば、「ACTTQM1」、「 ACTTQM2」) を識別して記憶することができます。

比較的短いキュー・マネージャー名 (例えば 8 文字未満) は、次のセクションで説明する規則、またはチャネル名に類似した規則を使用することを選択した場合に役立ちます。

クラスター・チャネル名を選択する際のベスト・プラクティス

キュー・マネージャーおよびクラスターは最大 48 文字の名前を持つことができ、チャネル名は 20 文字に制限されるため、プロジェクトの途中で命名規則を変更する必要がないように、オブジェクトに最初に命名するときは注意してください (前のセクションを参照してください)。

チャネルを定義する際には、クラスター内のいずれかのキュー・マネージャーで自動的に作成されたクラスター送信側チャネルの名前は、クラスター内の受信側キュー・マネージャーで構成された対応するクラスター受信側チャネルから取られることに注意してください。したがって、これらのチャネルは クラスター内のリモート・キュー・マネージャー上で 固有でなければなりません。

一般的な方法の 1 つは、クラスター名を前に付けたキュー・マネージャー名を使用することです。 例として、クラスター名が CLUSTER1 で、キュー・マネージャーが QM1、QM2 の場合、クラスター受信側チャネルは CLUSTER1.QM1、CLUSTER1.QM2 となります。

チャネルの優先順位が異なる場合、または異なるプロトコルを使用する場合は、この規則を拡張することができます。 以下に例を示します。

- CLUSTER1.QM1.S1
- CLUSTER1.QM1.N3
- CLUSTER1.QM1.T4

この例では、 S1 が最初の SNA チャネルになり、 N3 がネットワーク優先順位が 3 の NetBIOS チャネルになり、 T4 が IPV4 ネットワークを使用する TCP IP になります。

共用チャネル定義の命名

単一のチャネル定義を複数のクラスターで共有することができます。その場合は、ここで推奨されている命名規則を変更する必要があります。 ただし、「 チャネル定義の管理 」で説明されているように、通常はクラスターごとに個別のチャネルを定義することをお勧めします。

古いチャネルの命名規則

クラスター環境以外では、従来は「FROMQM.TO.TARGETQM」命名規則を使用することが一般的であったため、既存のクラスターで類似したもの (CLUSTER.TO.TARGET など) が使用されている場合があり

ます。 これは、新しいクラスター命名方式の一部としては推奨されません。これにより、チャネル名内の「有用な」情報を伝達するために使用可能な文字がさらに少なくなるためです。

z/OS IBM MQ for z/OS でのチャネル名

VTAM 総称リソースまたは 動的ドメイン・ネーム・サーバー (DDNS) 総称名を定義できます。 総称名を使用して、接続名を定義することができます。 しかし、クラスター受信側定義に汎用接続名を使用することはできません。

クラスター受信側定義に汎用接続名を使用する場合の問題は以下のとおりです。汎用 CONNAME を使用して CLUSRCVR を定義する場合、 CLUSSDR チャネルが意図したキュー・マネージャーを指す保証はありません。 最初の CLUSSDR は、キュー共有グループ内の、完全リポジトリーをホストする必要のない任意のキュー・マネージャーを指して終了する可能性があります。 チャネルが接続の再試行を開始すると、同じ総称名を持つ別のキュー・マネージャーに再接続して、メッセージのフローを中断する可能性があります。

z/OS *Queue sharing groups and clusters*

Shared queues can be cluster queues and queue managers in a queue sharing group can also be cluster queue managers.

On IBM MQ for z/OS you can group queue managers into queue sharing groups. A queue manager in a queue sharing group can define a local queue that is to be shared by up to 32 queue managers.

Shared queues can also be cluster queues. Furthermore, the queue managers in a queue sharing group can also be in one or more clusters.

VTAM 総称リソースまたは 動的ドメイン・ネーム・サーバー (DDNS) 総称名を定義できます。 総称名を使用して、接続名を定義することができます。 しかし、クラスター受信側定義に汎用接続名を使用することはできません。

クラスター受信側定義に汎用接続名を使用する場合の問題は以下のとおりです。汎用 CONNAME を使用して CLUSRCVR を定義する場合、 CLUSSDR チャネルが意図したキュー・マネージャーを指す保証はありません。 最初の CLUSSDR は、キュー共有グループ内の、完全リポジトリーをホストする必要のない任意のキュー・マネージャーを指して終了する可能性があります。 チャネルが接続の再試行を開始すると、同じ総称名を持つ別のキュー・マネージャーに再接続して、メッセージのフローを中断する可能性があります。

A CLUSRCVR channel that uses the group listener port can not be started because, if this were the case, it would not be possible to tell which queue manager the CLUSRCVR would connect to each time. The cluster system queues on which information is kept about the cluster are not shared. Each queue manager has its own.

Cluster channels are used not only to transfer application messages but internal system messages about the setup of the cluster. Each queue manager in the cluster must receive these internal system messages to participate properly in clustering, so needs its own unique CLUSRCVR channel on which to receive them.

A shared CLUSRCVR could start on any queue manager in the queue sharing group (QSG) and so lead to an inconsistent supply of the internal system messages to the QSG queue managers, meaning none can properly participate in the cluster. To ensure no shared CLUSRCVR channels can be used, any attempt fails with the CSQX502E message.

クラスターのオーバーラップ

クラスターのオーバーラップは、追加の管理機能を提供します。 名前リストを使用して、オーバーラップするクラスターの管理に必要なコマンドの数を減らします。

オーバーラップするクラスターを作成できます。 オーバーラップするクラスターを定義する理由には、次のようなさまざまなものがあります。

- 組織ごとに独自の管理ができるようにする。
- 独立したアプリケーションを個別に管理できるようにする。
- サービス・クラスを作成する。

36 ページの図 7 では、キュー・マネージャー STF2 は両方のクラスターのメンバーです。 1 つのキュー・マネージャーが 2 つ以上のクラスターのメンバーである場合、名前リストの利点を生かして、必要な定義の数を減らすことができます。 名前リストには、名前 (例えば、クラスター名) のリストを入れます。 クラスターを命名する名前リストを作成できます。 ALTER QMGR コマンドで STF2 に対してこの名前リストを指定して、両方のクラスターの完全リポジトリー・キュー・マネージャーにすることができます。

ネットワークに 2 つ以上のクラスターがある場合、それぞれ異なる名前を付ける必要があります。 同じ名前の 2 つのクラスターがマージされると、再び分離することは不可能です。 また、クラスターおよびチャネルに別々の名前を付けるとよいでしょう。 これにより、DISPLAY コマンドの出力を確認する際に区別しやすくなります。 正しく動作するためにはキュー・マネージャー名がクラスター内で固有でなければなりません。

サービス・クラスの定義

各職員および各学生ごとにキュー・マネージャーを持つ大学を想像してください。 職員間のメッセージは、高い優先度と高い帯域幅のチャネル上でやり取りされます。 学生間のメッセージは、安価で低速のチャネル上でやり取りされます。 このネットワークは、従来の分散キューイング技法でセットアップできます。 IBM MQ は、宛先のキュー名およびキュー・マネージャー名を探すことによって、使用するチャネルを選択します。

職員と学生をより明確に区別するために、36 ページの図 7 に示されているようにそれらのキュー・マネージャーを 2 つのクラスターにグループ化することができます。 IBM MQ は、職員クラスターに定義されているチャネルを介してのみ、メッセージをそのクラスターの会議キューに移動します。 学生クラスターのゴシップ・キューのメッセージは、そのクラスター内に定義されているチャネルに送られ、適切なサービス・クラスを受けます。

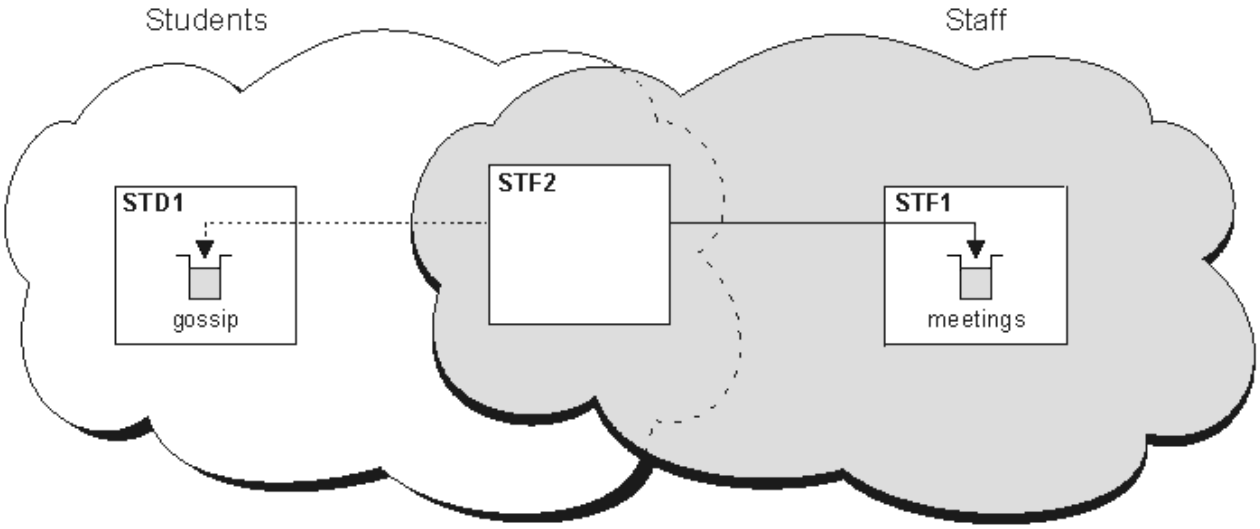


図 7. サービス・クラス

クラスター化のヒント

クラスター化を利用する場合は、その前にシステムやアプリケーションに多少の変更を加える必要が生じることがあります。 分散キューイングの動作とは、類似点も相違点もあります。

- クラスターの外側にあるキュー・マネージャーがクラスター・キューにアクセスできるように、これらのキュー・マネージャーに手動で構成定義を追加しなければなりません。
- 同じ名前の 2 つのクラスターをマージした場合、それらを再び分離することはできません。 したがって、各クラスターには重複しない名前を付けるようにしてください。
- キュー・マネージャーにメッセージが届いたときに、そのキュー・マネージャーにメッセージを受信するキューがないと、メッセージは送達不能キューに書き込まれます。 送達不能キューがない場合、そのチャネルでの受信は失敗しますが、その後に処理が再試行されます。 送達不能キューの使用法は、分散キューイングの場合と同様です。

- 持続メッセージの保全性が損なわれることはありません。 つまり、クラスターの使用によってメッセージが重複したり、消失したりすることはありません。
- クラスターを使用すると、システム管理にかかわる作業が少なくなります。 クラスターを使用すると、分散キューイングの場合よりも多数のキュー・マネージャーを大規模なネットワークに簡単に接続できるようになります。 クラスター内にあるすべてのキュー・マネージャーの間でデータを通信できるように設定すると、ネットワーク・リソースを消費しすぎることがあるので注意してください。
- IBM MQ Explorer は、キュー・マネージャーをツリー構造で表すため、大規模なクラスターの表示には適さない場合があります。
- Multi 配布先リストを使用すると、MQPUT コマンドを 1 回実行するだけで、同じメッセージを複数の宛先に送信することができます。 IBM MQ for Multiplatforms では、配布リストがサポートされています。 配布先リストとキュー・マネージャーのクラスターを併用できます。 クラスターでは MQPUT の実行時にすべてのメッセージが展開されます。 ネットワーク・トラフィックの点においてはクラスターではない環境の場合ほどのメリットはありません。 配布先リストを使用すると、多数のチャネルと伝送キューを手動で定義しなくてすむという利点があります。
- ワークロードのバランスを取るためにクラスターを使用する場合には、使用するアプリケーションを詳しく調べてください。 そのアプリケーションに、特定のキュー・マネージャーでメッセージを処理したり、メッセージを特定の順序で処理したりする必要があるかどうかを確認してください。 そのようにしてメッセージを処理する必要があるアプリケーションにはメッセージ・アフィニティーがあるといいます。 そのようなアプリケーションを複雑なクラスターで使用するには、アプリケーションを修正することが必要な場合があります。
- MQOPEN の MQOO_BIND_ON_OPEN オプションによりメッセージを特定の宛先に送信するよう選択することもできます。 宛先となるキュー・マネージャーが使用できない状態であると、そのキュー・マネージャーが再び使用可能になるまで、メッセージは送信されません。 そのような場合、メッセージを別のキュー・マネージャーに転送するとメッセージが重複してしまうことがあるため、メッセージは転送されません。
- キュー・マネージャーにクラスター・リポジトリーをホストさせる場合は、そのホスト名か IP アドレスを知る必要があります。 クラスターに加わる他のキュー・マネージャーで CLUSSDR 定義を作成する際に、CONNAME パラメーターにこの情報を指定する必要があります。 DHCP を使用する場合、システムを再始動するたびに DHCP は新しい IP アドレスを割り振ることがあるので、IP アドレスは変わる可能性があります。 したがって、この場合 CLUSSDR 定義に IP アドレスを指定することができません。 すべての CLUSSDR 定義で、IP アドレスではなくホスト名を指定したとしても、やはりそれらの定義に信頼を置くことはできません。 DHCP は、ホストの DNS ディレクトリー項目を新しいアドレスで必ず更新するとは限らないからです。 DHCP を使用するシステムでキュー・マネージャーを完全リポジトリーとして指名しなければならない場合、DNS ディレクトリーを常に最新の状態に保つことを保証するソフトウェアをインストールしてください。
- チャネルの接続名として、総称名 (例えば、VTAM 総称リソースまたはダイナミック・ドメイン・ネーム・サーバー (DDNS) 総称名) を使用しないでください。 使用すると、チャネルは予期しているものとは別のキュー・マネージャーに接続する場合があります。
- メッセージの取得はローカル・クラスター・キューからのみ行えますが、メッセージの書き込みはクラスター内のどのキューに対しても行えます。 MQGET コマンドを使用するためにキューをオープンする場合、キュー・マネージャーはローカル・キューをオープンします。
- 単純な構成の IBM MQ クラスターをセットアップする場合には、既存のアプリケーションを変更する必要はありません。 アプリケーションでは MQOPEN 呼び出しでオープンするキューの名前を指定できます。キュー・マネージャーの格納場所を知っている必要はありません。 ワークロード管理用にクラスターをセットアップする場合には、アプリケーションの内容を調べ、必要に応じてアプリケーションを修正する必要があります。
- DISPLAY CHSTATUS および DISPLAY QSTATUS **runmqsc** コマンドを使用して、チャネルまたはキューの現行のモニター・データおよび状況データを表示できます。 モニター情報を使用して、システム・パフォーマンスおよびシステム・ヘルスを測定できます。 モニターは、キュー・マネージャー、キュー、およびチャネルの属性によって制御されます。 自動定義されたクラスター送信側チャネルのモニターは、MONACLS キュー・マネージャー属性によって可能になります。

関連概念

クラスター

29 ページの『クラスター化と分散キューイングとの比較』
キュー・マネージャーに接続するために定義する必要があるコンポーネントを、分散キューイングを使用する場合とクラスター化を使用する場合で比較します。

クラスターのコンポーネント

関連タスク

キュー・マネージャー・クラスターの構成

新規クラスターのセットアップ

キュー・マネージャー・リポジトリーに情報が保管される期間

キュー・マネージャー・リポジトリーは、情報を 30 日間保管します。 自動プロセスは、使用中の情報を効率的にリフレッシュします。

キュー・マネージャーがそれ自体に関する情報を送信したときには、完全および部分リポジトリー・キュー・マネージ ャーはその情報を 30 日間保管します。 例えば、キュー・マネージャーが新しいキューの作成を通知した場合などに、情報は送信されます。 この情報の有効期限が切れるのを防ぐために、キュー・マネージャーはそ れ自体に関する情報を 27 日後に再送信します。 部分リポジトリーは、30 日の存続期間の途中で新しい情報要求を送信する 場合、有効期限は元の 30 日のままです。

情報は、有効期限が切れた場合、即時にリポジトリーから除去されるわけ ではありません。 その情報は、60 日間の猶予期間中、引き続き保持されます。 猶予期間中に更新を受け取らなかった場合、その情報は除去されます。 猶予期間は、有効期限の日にキュー・マネージャーが一時的にサービスを停止する可能性があることを考慮したものです。 キュー・マネージャーがクラスターから切断されている期間が 90 日を超えた場合は、そのキュー・マネージャーはクラスターの一部ではなくなります。 ただし、キュー・マネージャーがネットワークに再接続した場合は、再びクラスターの一部になります。 完全リポジトリーは、他のキュー・マネージャーからの新しい要求に対応するために、期限が満了した情報は使用しません。

また、キュー・マネージャーが完全リポジトリーの最新情報に対する要求を送信した場合は、その要求の存続期間は 30 日です。 IBM MQ は、27 日後に要求をチェックします。 27 日の間に参照された場合、その要求は自動的にリフレッシュされます。 この期間に参照されなかった場合、その要求は期限満了の対象としてそのまま残され、再び必要になった場合はキュー・マネージャーによりリフレッシュされます。 要求が期限満了になると、休止状態のキュー・マネージャーについての情報要求が累積 されるのを防ぐことができます。

注 : APAR PH43191 の PTF をダウンロードしてインストールする必要があります。これにより、サブスクリプションの有効期限時刻の計算におけるシステム・エラーが修正されます。 これらのエラーが原因で、サブスクリプションの有効期限が切れる (結果としてメッセージ CSQX456I が発行される) か、オブジェクトの有効期限が切れた後に有効期限が切れる (結果として MQRC 2085 (MQRC_UNKNOWN_OBJECT) エラーが発生する) 可能性があります。

大規模クラスターでは、多数のキュー・マネージャーが自身に関する全情報を同時に自動的に再送すると悪影響が及ぶ可能性があります。 大規模クラスターでのリフレッシュはクラスターのパフォーマンスと可用性に影響を与える可能性があるを参照してください。

関連概念

69 ページの『クラスター化: REFRESH CLUSTER の使用に関するベスト・プラクティス』
REFRESH CLUSTER コマンドを使用して、クラスターに関するローカルに保持されているすべての情報を破棄し、クラスターの完全リポジトリーからその情報を再作成します。 例外的な状況を除き、このコマンドを使用する必要はありません。 このコマンドを使用する必要がある場合は、使用方法に関する特別な考慮事項があります。 この情報は、テストおよびお客様からのフィードバックに基づく指針を示すものです。

サンプル・クラスター

最初に規模がもっとも小さい事例として、2 つのキュー・マネージャーから成るクラスターを示します。 2 番目と 3 番目の例では、3 つのキュー・マネージャーから成るクラスターの 2 つのバージョンを示します。

最小規模のクラスターでは、それに含まれるキュー・マネージャーは 2 つだけです。 この場合は、両方のキュー・マネージャーが完全リポジトリーを保有します。 クラスターのセットアップに必要な定義がごくわずかで済むにもかかわらず、各キュー・マネージャーに高度の自律性があります。

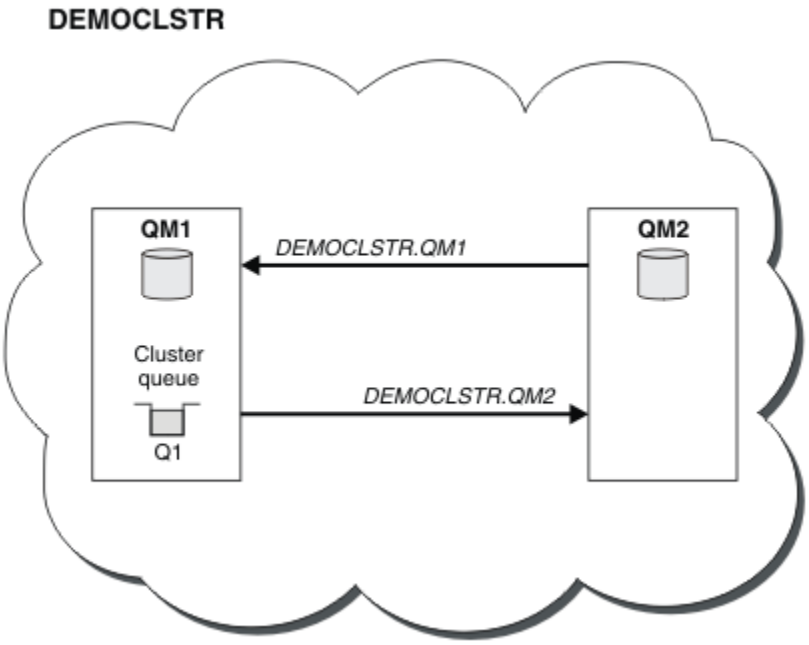


図 *8. 2* つのキュー・マネージャーで構成される小規模なクラスターの例

- キュー・マネージャーには、LONDON や NEWYORK などのロング・ネームを付けることができます。
  z/OS IBM MQ for z/OS では、キュー・マネージャー名は 4 文字までに制限されています。
- 各キュー・マネージャーは、通常別個のマシンで構成します。 ただし、同じマシンに複数のキュー・マネージャーを配置することもできます。

類似のクラスター例のセットアップ手順については、新規クラスターのセットアップ を参照してください。

40 ページの図 9 に CLSTR1 という名前のクラスターのコンポーネントを示しています。

- このクラスターには、QM1、QM2、QM3 という 3 つのキュー・マネージャーがあります。
- QM1 および QM2 では、クラスター内のすべてのキュー・マネージャーとクラスター関連オブジェクトに関する情報のリポジトリーをホストしています。 このようなキュー・マネージャーを完全リポジトリー・キュー・マネージャー といいます。 リポジトリーは、図の中で陰影の付いた円柱で示されています。
- QM2 および QM3 では、このクラスター内にあるその他のキュー・マネージャーからアクセスできるいくつかのキューをホストしています。 このクラスター内にあるその他のキュー・マネージャーからアクセスできるキューをクラスター・キュー といいます。 図の中で陰影の付いたキューの部分がクラスター・キューを表します。 クラスター・キューへはクラスター内のどこからでもアクセスできます。 IBM MQ クラスタリング・コードにより、これらのキューのリモート・キュー定義が、その定義を参照するすべてのキュー・マネージャーに必ず作成されるようになります。

  分散キューイングの場合と同様に、アプリケーションは MQPUT 呼び出しを使用してクラスター内の任意のキュー・マネージャーにあるクラスター・キューにメッセージを書き込みます。 アプリケーションは MQGET 呼び出しを使用して、キューが存在するキュー・マネージャーのみにあるクラスター・キューからメッセージを取り出します。
- 各キュー・マネージャーには、メッセージを受信できる *cluster_name. queue_manager_name* と呼ばれるチャネルの受信側に対して、手動で作成された定義があります。 受信側のキュー・マネージャーでは、*cluster_name. queue_manager_name* はクラスター受信側チャネルです。 クラスター受信側チャネルは分散キューイングで使用されている受信側チャネルに似ており、キュー・マネージャーのメッセージを受信します。 さらに、クラスターについての情報も受け取ります。

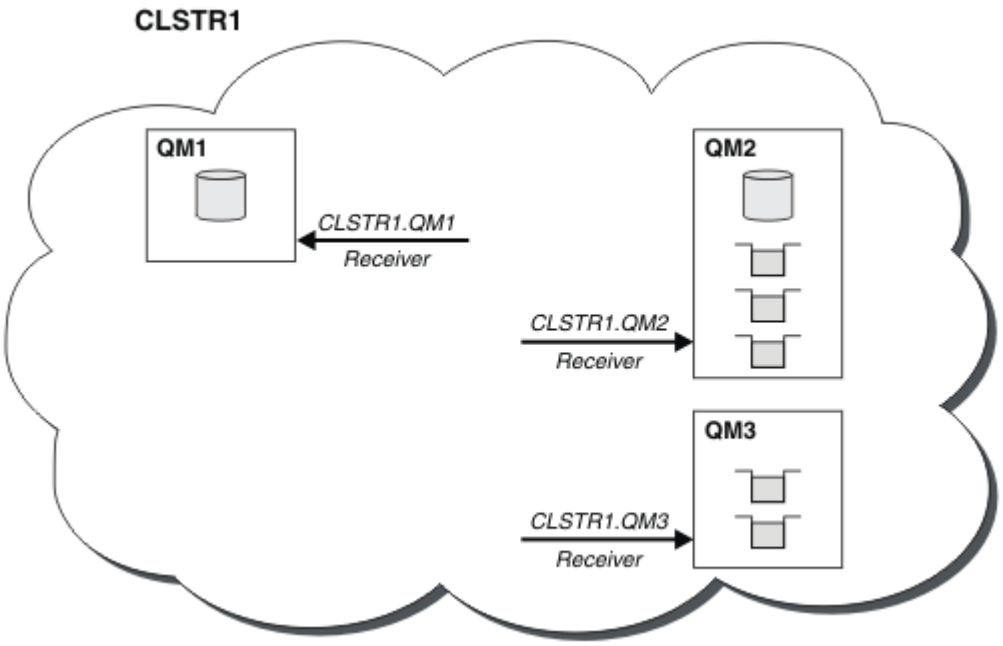
- 

図 9. 複数のキュー・マネージャーで構成されるクラスター

- 41 ページの図 10 では、各キュー・マネージャーに、チャネルの送信側であることを示す定義もあります。 送信側のチャネルは、いずれかの完全リポジトリー・キュー・マネージャーのクラスター受信側チャネルに接続しています。 送信側キュー・マネージャーでは、*cluster_name. queue_manager_name* はクラスター送信側チャネルです。 QM1 および QM3 のクラスター送信側チャネルは CLSTR1.QM2 に接続されています。点線「"2"」を参照してください。

  QM2 のクラスター送信側チャネルは CLSTR1.QM1 に接続されています。点線「"3"」を参照してください。 クラスター送信側チャネルは、分散キューイングで使用されている送信側チャネルに似ており、受信側キュー・マネージャーにメッセージを送信します。 さらに、クラスターについての情報も送信します。

  クラスター受信側チャネルとクラスター送信側チャネルの両方の定義が終わると、自動的にチャネルが始動します。

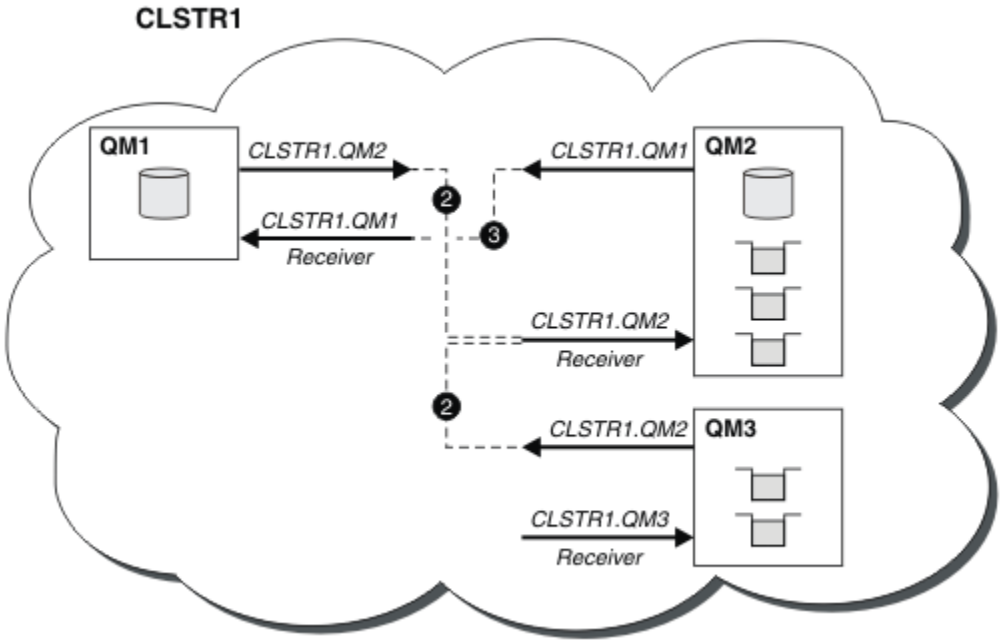


図 *10.* 送信側チャネルのある複数のキュー・マネージャーで構成されるクラスター

ローカル・キュー・マネージャーにクラスター送信側チャネルを定義すると、キュー・マネージャーがいずれかの完全リポジトリー・キュー・マネージャーで認識されます。 これにより、その完全リポジトリー・キュー・マネージャーは完全リポジトリーの情報を更新します。 そして、元のキュー・マネージャー宛てのクラスター送信側チャネルを自動的に作成して、そのキュー・マネージャーにクラスターの情報を送信します。 このように、キュー・マネージャーとクラスターは相互に認識できます。

以降に、40 ページの図 9 に示すクラスターを再び例にとって説明します。 例えば、キュー・マネージャー QM3 に接続されているアプリケーションから QM2 のキューにメッセージを送信するとします。 QM3 が初めてそれらのキューにアクセスする際には、完全リポジトリーを参照して、キューを見つけることができます。 この場合の完全リポジトリーは QM2 で、このリポジトリーへは送信側チャネル CLSTR1.QM2 を使用してアクセスします。 リポジトリーからの情報を使用して、それらのキュー用にリモート定義を自動的に作成することができます。 キューが QM1 にある場合でも、QM2 が完全リポジトリーであるため、このメカニズムは引き続き機能します。 完全リポジトリーは、クラスター内のすべてのオブジェクトの完全なレコードを保持しています。 後者の事例では、QM3 は自動的に QM1 のクラスター受信側チャネルに対応するクラスター送信側チャネルを作成することもでき、これら 2 つの間で直接通信が可能になります。

42 ページの図 11 は同じクラスターを示していますが、ここでは、自動的に作成された 2 つのクラスター送信側チャネルが追加されています。 クラスター送信側チャネルは、クラスター受信側チャネル CLSTR1.QM3 につながっている 2 本の破線で示されています。 また、この図に示されているクラスター伝送キュー SYSTEM.CLUSTER.TRANSMIT.QUEUE は、QM1 がメッセージを送信するときに使用するキューです。 クラスター伝送キューは、クラスター内のどのキュー・マネージャーにもあります。このキューにより、各キュー・マネージャーは同じクラスター内のその他のキュー・マネージャーにメッセージを送信することができます。

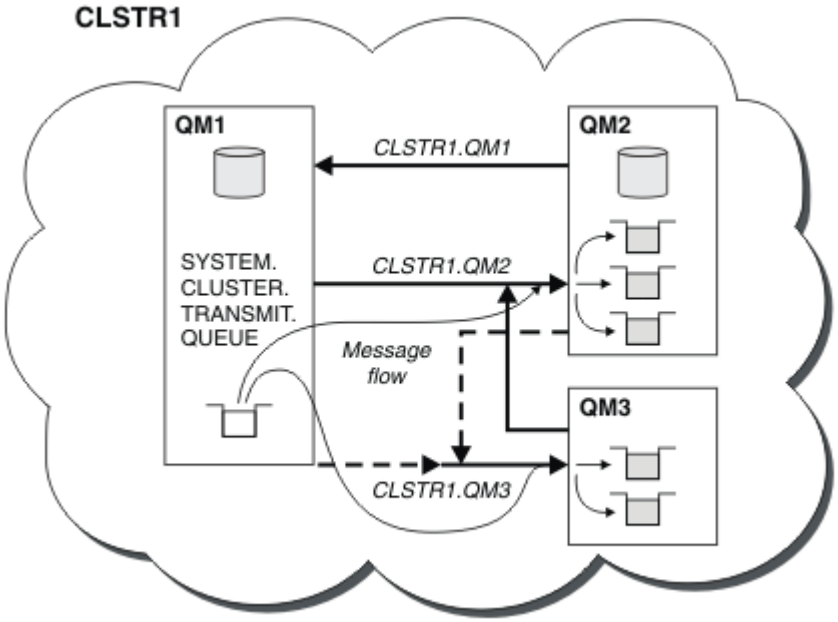


図 *11.* 複数のキュー・マネージャーで構成されるクラスター (自動定義チャネル付き)

注 : 他の図は、手動で定義を作成する、チャネルの受信側のみを示しています。 送信側は、必要な場合に自動的に定義されることが多いので、省略しています。 多くのクラスター送信側チャネルの自動定義は、クラスターの機能と効率に影響を及ぼす非常に重要なものです。

関連概念

29 ページの『クラスター化と分散キューイングとの比較』
キュー・マネージャーに接続するために定義する必要があるコンポーネントを、分散キューイングを使用する場合とクラスター化を使用する場合で比較します。

クラスターのコンポーネント

関連タスク

キュー・マネージャー・クラスターの構成

新規クラスターのセットアップ

クラスター化: ベスト・プラクティス

クラスターは、キュー・マネージャーが相互接続するためのメカニズムを提供します。 このセクションで説明しているベスト・プラクティスは、テストおよびお客様からのフィードバックに基づいています。

クラスターのセットアップを成功させるには、適切な計画を立てることと IBM MQ に関する基礎知識 (適切なアプリケーション管理やネットワーク設計など) を十分に理解することが必要です。 先に進む前に、関連トピックで提供されている情報をよく理解してください。

関連概念

分散キューイングとクラスター

クラスター

関連タスク

24 ページの『クラスターの設計』
クラスターによって提供されるキュー・マネージャーを相互接続するためのメカニズムにより、初期構成と継続的な管理を簡単に行えます。 クラスターが正しく機能することと、必要なレベルの可用性と応答性をクラスターが達成することが大切ですので、クラスターは慎重に設計する必要があります。

クラスターのモニター

クラスター化: オーバーラップするクラスターについての特殊な考慮事項
このトピックでは、IBM MQ クラスターを計画および管理するための指針について記載しています。 この情報は、テストおよびお客様からのフィードバックに基づく指針を示すものです。

クラスター所有権

以下の情報を読む前に、クラスターのオーバーラップについて十分に理解してください。 必要な情報については、35 ページの『クラスターのオーバーラップ』およびクラスター間のメッセージ・パスの構成を参照してください。

オーバーラップするクラスターで構成されたシステムを構成および管理する際には、以下に従うことが最善です。

- 前述のとおり、IBM MQ クラスターは「疎結合」されていますが、クラスターを 1 つの管理単位として考えると役立ちます。 この概念を使用する理由は、クラスターがスムーズに機能するためには、個々のキュー・マネージャーに関する定義の相互作用が重要となるためです。 例えば、ワークロード・バランシングが行われるクラスター・キューを使用する場合には、一人の管理者または 1 つのチームがメッセージに考えられるすべての宛先一式を理解していることが重要ですが、この宛先一式は、クラスター全体に分散された定義に依存します。 さらに当然のことながら、クラスター送信側/受信側チャネルのペアは、クラスター全体で互換性がなければなりません。
- 上記の概念を考えると、(別々のチームや個人によって管理されることになる) 複数のクラスターが集まる場合には、ゲートウェイ・キュー・マネージャーの管理を制御する明確なポリシーを実施することが重要です。
- オーバーラップするクラスターは単一の名前空間として扱うと便利です。チャネル名とおキュー・マネージャー名は、1 つのクラスター全体で固有でなければなりません。 トポロジー全体で固有であると、さらに管理しやすくなります。 適切な命名規則に従うことが最善です。使用可能な命名規則については、34 ページの『クラスターの命名規則』を参照してください。
- 場合によっては、管理とシステム管理の協力が不可欠です。 例えば、オーバーラップする必要がある異なるクラスターを所有する組織間の協力などです。 クラスターのオーバーラップ時にクラスタリングをスムーズに実行するには、誰が何を所有し、強制可能なルールと規則を明確に理解する必要があります。

オーバーラップするクラスター: ゲートウェイ

一般に、複数のクラスターを管理するよりも、1 つのクラスターを管理するほうが容易です。 したがって、通常は、多数の小さなクラスター (例えばアプリケーションごとのクラスター) を作成するようなことを避けるのが適切です。

ただし、サービス・クラスを提供する目的で、オーバーラップするクラスターを実装することもできます。以下に例を示します。

- 少数のクラスターがパブリッシュ/サブスクライブ用である同心円クラスターがある場合。 詳しくは、システムのサイズ変更の方法を参照してください。
- 一部のキュー・マネージャーが異なるチームによって管理される場合。 詳しくは、前のセクション 43 ページの『クラスター所有権』を参照してください。
- 組織的または地理的観点から意味がある。
- 同等のクラスターがネーム・レゾリューションで機能する場合 (例えば、既存のクラスターに TLS を実装する場合)。

クラスターのオーバーラップによるセキュリティー上の利点はありません。2 つの異なるチームによって管理されるクラスターをオーバーラップさせ、トポロジーだけでなくチームにも効果的に参加させることができます。

- そのようなクラスターで公示された名前は、他のクラスターからアクセスできます。
- 一方のクラスターでアドバタイズされた名前は、適格なメッセージをドローするためにもう一方のクラスターでアドバタイズすることができます。
- ゲートウェイに隣接するキュー・マネージャー上の公示されていないオブジェクトは、そのゲートウェイがメンバーとなっているクラスターから解決できます。

ネーム・スペースは両方のクラスターの和集合であり、1 つのネーム・スペースとして扱われる必要があります。 したがって、オーバーラップするクラスターの所有権は、両方のクラスターのすべての管理者の間で共有されます。

複数のクラスターが存在するシステムで、あるクラスターのキュー・マネージャーから別のクラスターのキュー・マネージャーにメッセージを経路指定することが要件となる場合もあります。 そのような場合には、複数のクラスターを何らかの方法で相互接続する必要があり、優れたパターンとして、クラスター間でゲートウェイ・キュー・マネージャーを使用することが挙げられます。 この配置によって、Point-to-Point チャネルが網目のように積み重なって管理しにくくなるという事態を回避し、セキュリティー・ポリシーなどの問題を一箇所で管理できるようになります。 この配置を達成するには、次に示す 2 種類の方法があります。

1. 2 つ目のクラスター受信側を定義して、1 つ (または複数) のキュー・マネージャーを両方のクラスターに配置する。 このような配置にすると管理定義は少なくなりますが、既に述べたとおり、オーバーラップするクラスターの所有権は両方のクラスターのすべての管理者の間で共有されます。
2. 従来の Point-to-Point チャネルを使用して、クラスター 1 のキュー・マネージャーとクラスター 2 のキュー・マネージャーをペアとして組み合わせる。

上記のいずれの場合でも、トラフィックを適切にルーティングするために各種のツールを使用できます。 特に、別のクラスターへのルーティングを行うには、キュー別名またはキュー・マネージャー別名を使用できます。ブランクの **RQMNAME** プロパティーを設定したキュー・マネージャー別名は、必要に応じてワークロード・バランシングを再駆動します。

関連概念

34 ページの『クラスターの命名規則』
キュー・マネージャーが属するクラスターを識別する命名規則を使用して、同じクラスター内のキュー・マネージャーの命名について考えます。 チャネル名の命名に類似した命名規則を使用し、チャネルの特性を記述するために命名規則を拡張します。

クラスター化: トポロジー設計上の考慮事項
このトピックでは、IBM MQ クラスターを計画および管理するための指針について記載しています。 この情報は、テストおよびお客様からのフィードバックに基づく指針を示すものです。

ユーザー・アプリケーションおよび内部管理プロセスをあらかじめどこに配置するのかについて考えておくことにより、多くの問題を回避できたり、あるいは後で最小化したりできます。 このトピックでは、クラスターが大きくなっていった場合にパフォーマンスを向上させ、保守タスクを簡単な操作で行えるようにするための、設計上の決定事項について説明します。

- 44 ページの『クラスター・インフラストラクチャーのパフォーマンス』
- 45 ページの『完全リポジトリー』
- 46 ページの『アプリケーションは完全リポジトリーでキューを使用しなければなりませんか』
- 47 ページの『チャネル定義の管理』
- 47 ページの『複数チャネルのワークロード・バランシング』

クラスター・インフラストラクチャーのパフォーマンス

アプリケーションがクラスター内のキュー・マネージャー上にあるキューを開こうとすると、そのキュー・マネージャーは、クラスター内における該当のキューの場所を認識できるようにするため、自らのインタレストをキューの完全リポジトリーに登録します。 キューの場所または構成に対する更新は、完全リポジトリーからインタレスト・キュー・マネージャーへ自動的に送信されます。 このようなインタレストの登録は、内部的に、サブスクリプションと呼ばれます (IBM MQ でメッセージのパブリッシュ/サブスクライブに使用される IBM MQ サブスクリプションとは異なります)。

クラスターに関するすべての情報は、あらゆる完全リポジトリーを経由して送信されます。 したがって、完全リポジトリーは、クラスター内の管理メッセージ・トラフィックのために常時使用されています。 これらのサブスクリプションの管理、伝送、および結果的に生成される構成メッセージに大量のシステム・リソースが使用されると、クラスター・インフラストラクチャーの負荷が大幅に増加する可能性があります。 この負荷を可能な限り認識し、最小限に抑えるために、いくつかの考慮事項があります。

- クラスター・キューを使用する個々のキュー・マネージャーが多いほど、システム内のサブスクリプションが増加します。したがって、変更が発生し、関連するサブスクライバーに通知する必要がある場合には、管理オーバーヘッドがさらに大きくなります。このことは、特に、完全リポジトリー・キュー・マネージャーに当てはまります。 不要なトラフィックおよび完全リポジトリーの負荷を最小限に抑える方法の 1 つは、類似するアプリケーション (すなわち、同一のキューを処理するアプリケーション) をより少ない数のキュー・マネージャーに接続することです。
- システム内のサブスクリプションの数だけでなく、クラスター・オブジェクトの構成に対する変更の頻度 (クラスター・キューの構成の頻繁な変更など) もパフォーマンスに影響を与える可能性があります。
- キュー・マネージャーが複数のクラスターのメンバーである (つまり、オーバーラップしているクラスター・システムの一部である) 場合、キューでインタレストが作成されると、メンバーとして属している各クラスターについてサブスクリプションが発生することになります。同一のキュー・マネージャーが複数のクラスターの完全リポジトリーであっても同様です。 このような配置はシステムの負荷を増加させます。したがって、単一のクラスターではなく複数のオーバーラップ・クラスターが必要かどうかを検討する理由の 1 つにもなります。
- アプリケーション・メッセージ・トラフィック (つまり、IBM MQ アプリケーションによってクラスター・キューに送信されるメッセージ) は、完全リポジトリーを経由せずに宛先キュー・マネージャーに到達します。 このメッセージ・トラフィックは、クラスターへのメッセージの入口となるキュー・マネージャーとクラスター・キューが存在するキュー・マネージャーの間で直接送信されます。 したがって、完全リポジトリー・キュー・マネージャーが偶然にもそれらの 2 つのキュー・マネージャーの 1 つである場合を除き、完全リポジトリー・キュー・マネージャーに関しては高速のアプリケーション・メッセージ・トラフィックに対応する必要はありません。 そのため、クラスター・インフラストラクチャーの負荷が高いクラスターでは、完全リポジトリー・キュー・マネージャーをアプリケーション・メッセージ・トラフィックに使用しないことをお勧めします。

完全リポジトリー

リポジトリーとは、クラスターを構成する各キュー・マネージャーについての情報の集まりを指します。クラスター内の各キュー・マネージャーについての 全情報をホストしているキュー・マネージャーには、完全リポジトリーが含まれます。 完全リポジトリーと部分リポジトリーについて詳しくは、『クラスター・リポジトリー』を参照してください。

完全リポジトリーは、信頼性が高く、可能な限り可用性の高いサーバーに保持する必要があり、単一障害点を避けなければなりません。 クラスター設計には常に 2 つの完全リポジトリーが必要です。 1 つの完全リポジトリーに障害が発生した場合でも、クラスターは動作を継続することができます。

クラスター内のキュー・マネージャーによって行われたクラスター・リソース (クラスター・キューなど) に対する更新の詳細は、そのキュー・マネージャーからクラスター内の最大 2 つの完全リポジトリー (クラスター内に完全リポジトリー・キュー・マネージャーが 1 つしかない場合は 1 つ) に送信されます。 それらの完全リポジトリーは、情報を保持し、クラスター内のインタレスト・キュー・マネージャー (つまり、完全リポジトリーにサブスクライブしているキュー・マネージャー) にその情報を伝搬します。 クラスターの各メンバーでクラスター・リソースの最新の状況が認識されるようにするため、各キュー・マネージャーが常に少なくとも 1 つの完全リポジトリー・キュー・マネージャーと通信できなければなりません。

キュー・マネージャーは、何らかの理由でいずれの完全リポジトリー・キュー・マネージャーとも通信できない場合には、既にキャッシュに入れられているレベルの情報を基にしばらくは機能し続けることができますが、新しい更新情報を取得することや、以前は使用されていなかったクラスター・リソースにアクセスすることはできません。

このため、常に 2 つの完全リポジトリーを使用可能にしておくことを目指してください。 ただし、1 つの完全リポジトリーがない場合でも短い期間であればクラスターは十分に機能するので、この配置は、極端な手段を講じなければならないということを意味するものではありません。

クラスターに 2 つの完全リポジトリー・キュー・マネージャーが必要な理由は、クラスター情報を使用可能にしておくため以外に、もう 1 つあります。その理由とは、リカバリーのため、完全リポジトリーのキャッシュに保持されるクラスター情報が 2 つの場所に保管されるようにすることです。 完全リポジトリーが 1 つだけであり、そこで保持されているクラスター情報が失われた場合、クラスターが再び機能するようにするには、クラスター内のすべてのキュー・マネージャーに対して手操作による介入を行う必要があります。 しかし、完全リポジトリーが 2 つあれば、情報は常に 2 つの完全リポジトリーにパブリッシュさ

れ、サブスクライブされるので、最小限の作業で、障害が発生した完全リポジトリーを回復することができます。

- 2 つの完全リポジトリーを採用するクラスター設計では、クラスターのユーザーに影響を及ぼすことなく、完全リポジトリー・キュー・マネージャーの保守作業を行うことができます。クラスターはリポジトリーが 1 つしかなくても機能し続けるので、可能であれば、一度に 1 つずつリポジトリーを停止して、保守作業を行い、稼働状態に戻します。 2 番目の完全リポジトリーが故障している場合でも、実行中のアプリケーションは、最低でも 3 日間は影響を受けません。
- 地理的理由から特定の場所にある完全リポジトリーを使用しているなど、3 番目のリポジトリーを使用する理由がある場合を除き、これら 2 つのリポジトリーを採用する設計を使用してください。 完全リポジトリーが 3 つあると、どの 2 つが現在使用されているのか分からず、複数のワークロード管理パラメーター間の相互作用によって管理上の問題が発生することもあります。 完全リポジトリーを 3 つ以上使用することは推奨されません。
- より優れた可用性が必要な場合は、完全リポジトリー・キュー・マネージャーを複数インスタンス・キュー・マネージャーとしてホストすることや、プラットフォーム固有の高可用性サポート機能を使用して可用性を改善することを検討してください。
- 手動で定義されたクラスター送信側チャネルを使用して、すべての完全リポジトリー・キュー・マネージャーを完全に相互接続する必要があります。 何らかの正当な理由があり、クラスターに 3 つ以上の完全リポジトリーが確保されていない場合は、特に注意しなければなりません。 そのような場合、1 つまたは複数のチャネルが見逃され、そのことがすぐに判明しない可能性が高くなります。 完全な相互接続が確立されない場合、問題の診断が困難になるということがしばしば発生します。 なぜならば、いくつかの完全リポジトリーですべてのリポジトリー・データが保持されず、その結果、クラスター内のキュー・マネージャーが認識するクラスターの全体像が、接続している完全リポジトリーによって異なることになるからです。

アプリケーションは完全リポジトリーでキューを使用しなければなりませんか

完全リポジトリーは、ほとんどの場合、他のキュー・マネージャーとまったく同じようなものです。したがって、完全リポジトリーでアプリケーション・キューをホストし、アプリケーションを他のキュー・マネージャーに直接接続することが可能です。 アプリケーションは完全リポジトリーでキューを使用しなければなりませんか

一般的に受け入れられている回答は「いいえ」です。 そのような構成を使用することはできますが、多くのお客様は完全リポジトリー・キュー・マネージャーを完全リポジトリー・クラスター・キャッシュの保守専用にしておくことを好まれます。 ここでは、どちらを選択するか決定する際の考慮事項を紹介していますが、最終的には、ご使用の環境における個々の要望に対応するにはクラスター・アーキテクチャーが適しているものと思われます。

- アップグレード: 通常、新規リリースの IBM MQ で新しいクラスター機能を使用するには、最初に、該当のクラスターの完全リポジトリー・キュー・マネージャーをアップグレードします。 クラスター内のアプリケーションで新機能を使用する必要がある場合に、共存している多数のアプリケーションをテストしなくても完全リポジトリー (および部分リポジトリーの一部のサブセット) をアップデートできるようになるので便利です。
- 保守: 同様に、完全リポジトリーに緊急保守を適用する必要がある場合は、アプリケーションには触れずに、それらのリポジトリーを再始動するか、**REFRESH** コマンドを使用してリフレッシュすることができます。
- パフォーマンス: クラスターが成長し、完全リポジトリー・クラスター・キャッシュの保守に対する要求が強まってきても、アプリケーションを切り離しておくことで、アプリケーションのパフォーマンスがシステム・リソースの競合による影響を受けるというリスクが軽減されます。
- ハードウェア要件: 通常、強力な完全リポジトリーは不要であり、例えば、可用性が十分に期待できる単純な UNIX サーバーで十分です。 あるいは、非常に大規模なクラスターまたは頻繁に変更が加えられるクラスターの場合は、完全リポジトリー・コンピューターのパフォーマンスを考慮する必要があります。
- ソフトウェア要件: 通常、完全リポジトリーでアプリケーション・キューをホストすることにする場合、これらの要件が主な理由となります。 小規模なクラスターでは、コロケーションは、全体としてキュー・マネージャーまたはサーバーの数を削減する必要があることを意味します。

チャネル定義の管理

単一クラスター内でも、2 つのキュー・マネージャー間で複数の経路を提供する複数のチャネル定義が存在することがあります。

単一クラスター内に並列チャネルが存在していることが利点である場合もありますが、この設計上の決定事項は十分に検討する必要があります。この設計では、複雑さが増すばかりでなく、チャネルの使用効率が下がり、パフォーマンスが低下する可能性があります。 そのような状況が発生するのは、通常、テストでは、一定の速度で多くのメッセージが送信されるので、並列チャネルがフルに使用される ためです。 しかし、メッセージのストリームが一定ではない現実世界の状態では、メッセージ・フローがチャネルからチャネルへと切り替えられていくにつれて、ワークロード・バランシングのアルゴリズムが原因でパフォーマンスが低下します。

キュー・マネージャーが複数のクラスターのメンバーである場合は、クラスターごとに別々の CLUSRCVR チャネルを定義する代わりに、クラスター名前リストを指定して 1 つのチャネル定義を使用するという選択肢があります。 ただし、そのようにセットアップすると、後で管理が困難になることがあります。例えば、TLS が 1 番目のクラスターに適用されるが 2 番目のクラスターには適用されない場合があります。 そのため、別々の定義を作成することが望ましく、34 ページの『クラスターの命名規則』で推奨されている命名規則はこのような場合に対応するようになっています。

複数チャネルのワークロード・バランシング

この情報は、このテーマを高度に理解してもらうことを意図したものです。 この点に関する基本的な説明 (この情報を使用する前に理解しておく必要のある知識) については、クラスターによるワークロードの管理、クラスターでのワークロード・バランシング、およびクラスター・ワークロード管理アルゴリズムを参照してください。

クラスター・ワークロード管理アルゴリズムは、多くのツールからなるセットを提供しますが、それらのツールはすべて、それらの動作方法と相互作用を十分に理解してから相互に使用すべきです。 ワークロード・バランシング・プロセスに対するチャネルの重要度がただちに明確にならない場合があります。ワークロード管理のラウンドロビン・アルゴリズムでは、クラスター・キューを所有するキュー・マネージャーへの複数のクラスター・チャネルが、そのキューの複数のインスタンスであるかのように処理されます。このプロセスについては、以下の例で詳しく説明します。

1. クラスターにはキューをホストする 2 つのキュー・マネージャー (QM1 と QM2) があります。
2. QM1 へのクラスター受信側チャネルは 5 つあります。
3. QM2 へのクラスター受信側チャネルは 1 つのみです。
4. QM3 上の **MQPUT** または **MQOPEN** が、あるインスタンスを選択すると、このアルゴリズムでは、QM1 にメッセージを送信する確率が、QM2 にメッセージを送信する確率より 5 倍高くなります。
5. ステップ 4 の状況が発生する理由は、このアルゴリズムに見えているのは、(5+1) からの選択肢としての 6 つのオプションであるため、このアルゴリズムは、QM1 への 5 つのチャネルすべてと、QM2 への 1 つのチャネルについてラウンドロビンを行うからです。

その他の優れた動作は、たまたまローカル・キュー・マネージャー上に構成されているインスタンスを 1 つもつクラスター・キューにメッセージを送信する場合でも、IBM MQ は、ローカル・クラスター受信側チャネルの状態に基づいて、メッセージをそのキューのローカル・インスタンスとリモート・インスタンスのどちらに送信するかが決まることです。 このシナリオでは、以下のようになります。

1. メッセージを送信する場合、ワークロード管理アルゴリズムでは、個々のクラスター・キューを調べるのではなく、宛先に到達できるクラスター・チャネルを調べます。
2. ローカル宛先に到達するため、(メッセージの送信には使用されませんが) ローカル受信側チャネルがこのリストに含められます。
3. ローカル受信側チャネルが停止すると、ワークロード管理アルゴリズムでは、デフォルトで代わりのインスタンスを優先的に使用します (ただし、その CLUSRCVR が停止していない場合)。 宛先のローカル CLUSRCVR インスタンスが複数存在し、少なくとも 1 つが停止状態でない場合、そのローカル・インスタンスは引き続き適格になります。

クラスター化: 複数のクラスター伝送キューの使用によるアプリケーションの分離
クラスター内のキュー・マネージャー間のメッセージ・フローは、分離することができます。 さまざまなクラスター送信側チャネルによって転送されるメッセージを、それぞれに異なるクラスター伝送キューに配置できます。 この手法は、単一のクラスターでも、オーバーラップするクラスターでも使用できます。 このトピックでは、使用する手法を選択する際に参考となる例およびベスト・プラクティスを説明します。

アプリケーションをデプロイするときには、他のアプリケーションと共有する IBM MQ リソース、および共有しないリソースを選択できます。 共有できるリソースには、さまざまなタイプがあります。そのうち主なタイプは、サーバー自体、キュー・マネージャー、チャネル、およびキューです。 少ない数の共有リソースでアプリケーションを構成し、個々のアプリケーションに個別のキュー、チャネル、キュー・マネージャー、さらにはサーバーを割り振るという方法があります。 この方法では、システム構成全体が大きくなり、複雑さも増してきます。 IBM MQ クラスターを使用すると、多数のサーバー、キュー・マネージャー・キュー、およびチャネルを管理する場合の複雑さは軽減されますが、別の共有リソースであるクラスター伝送キュー SYSTEM.CLUSTER.TRANSMIT.QUEUE が導入されます。

49 ページの図 12 は、大規模な IBM MQ デプロイメントからのスライスです。ここには、SYSTEM.CLUSTER.TRANSMIT.QUEUE を共有する重要性が示されています。 この図のアプリケーション Client App は、クラスター CL1 内のキュー・マネージャー QM2 に接続されています。 Client App からのメッセージは、アプリケーション Server App によって処理されます。 このメッセージは、Server App が CLUSTER2 内のキュー・マネージャー QM3 にあるクラスター・キュー Q1 から取得します。 クライアント・アプリケーションとサーバー・アプリケーションは同じクラスター内に配置されていないため、メッセージはゲートウェイ・キュー・マネージャー QM1 によって転送されます。

クラスター・ゲートウェイを構成する通常の方法は、ゲートウェイ・キュー・マネージャーをすべてのクラスターのメンバーにすることです。 ゲートウェイ・キュー・マネージャーには、すべてのクラスター内のクラスター・キューに対応するクラスター別名キューを定義します。 クラスター化されたキュー別名は、すべてのクラスター内で使用可能になります。 クラスター化されたキュー別名に入れられたメッセージは、ゲートウェイ・キュー・マネージャーを介して、それぞれの正しい宛先にルーティングされます。 ゲートウェイ・キュー・マネージャーは、クラスター別名キューに送信されたメッセージを QM1 上の共通 SYSTEM.CLUSTER.TRANSMIT.QUEUE に入れます。

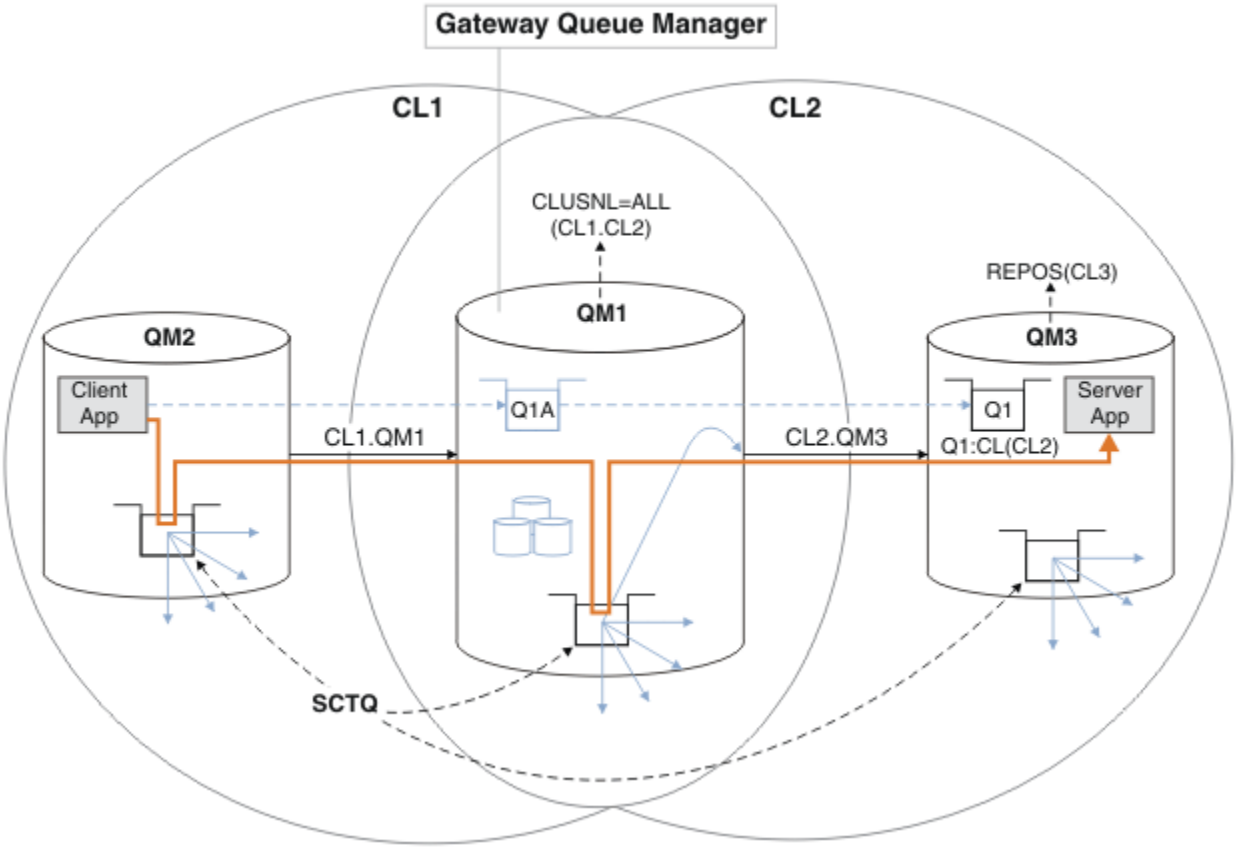
ハブ・スポーク・アーキテクチャーでは、クラスター間のすべてのメッセージが、ゲートウェイ・キュー・マネージャーを通過しなければなりません。 したがって、すべてのメッセージ・フローが、QM1 上の 1 つのクラスター伝送キュー SYSTEM.CLUSTER.TRANSMIT.QUEUE を通過することになります。

パフォーマンスの観点からすると、キューが 1 つでも問題にはなりません。 共通伝送キューは、一般にパフォーマンス・ボトルネックを意味しません。 ゲートウェイでのメッセージ・スループットは、主にそのゲートウェイに接続するチャネルのパフォーマンスによって決まります。 キューの数やチャネルが使用するキューに置かれたメッセージの数は通常、スループットに影響を与えません。

その他の観点から見ると、複数のアプリケーションに単一の伝送キューを使用することには、以下の欠点があります。

- ある宛先へのメッセージ・フローを、別の宛先へのメッセージ・フローから分離することができません。 メッセージがそれぞれに異なるクラスター内の異なるキュー・マネージャーを宛先としているとしても、メッセージのストレージを分離してからメッセージを転送することは不可能です。

  1 つのクラスター宛先が使用不可になると、その宛先のメッセージが単一の伝送キューに蓄積され、最終的には、その伝送キューがこれらのメッセージで満杯になってしまいます。 伝送キューが満杯になると、あらゆるクラスター宛先の伝送キューへのメッセージの配置が中断されます。
- さまざまなクラスター宛先へのメッセージの転送をモニターするのは容易ではありません。 すべてのメッセージは、単一の伝送キューに配置されます。 伝送キューの深さを表示しても、すべての宛先にメッセージが転送されているかどうかは、ほとんど明らかになりません。



注 : 49 ページの図 12 および以降の図には、異なるタイプの矢印が示されています。 実線の矢印は、メッセージ・フローを表します。 実線矢印のラベルは、メッセージ・チャネル名です。 グレーの実線矢印は、SYSTEM.CLUSTER.TRANSMIT.QUEUE からクラスター送信側チャネルへの潜在的メッセージ・フローです。 黒の破線は、ラベルをそのターゲットに結びます。 グレイの破線矢印は、参照を示します。例えば、Client App による MQOPEN 呼び出しから、クラスター別名キュー定義 Q1A への参照です。

図 *12.* ハブ・スポーク・アーキテクチャーにデプロイされた、*IBM MQ* クラスターを使用するクライアント/サーバー・アプリケーション

49 ページの図 12 では、Server App のクライアントはキュー Q1A をオープンします。 QM2 の SYSTEM.CLUSTER.TRANSMIT.QUEUE に入れられたメッセージは、QM1 の SYSTEM.CLUSTER.TRANSMIT.QUEUE に転送されてから、QM3 の Q1 に転送されます。このキューから、Server App アプリケーションはメッセージを受け取ります。

Client App からのメッセージは、QM2 および QM1 のシステム・クラスター伝送キューを通過します。 49 ページの図 12 で目標となっているのは、ゲートウェイ・キュー・マネージャーでメッセージ・フローをクライアント・アプリケーションから分離することなので、メッセージは SYSTEM.CLUSTER.TRANSMIT.QUEUE には保管されません。 他のすべてのクラスター・キュー・マネージャーでも、フローを分離できます。 逆の方向で、クライアントに戻るフローを分離することも可能です。 ソリューションの説明を簡潔にするため、以降の説明ではクライアント・アプリケーションからの 1 つのフローのみを検討します。

クラスター・ゲートウェイ・キュー・マネージャーでクラスター・メッセージ・トラフィックを分離するためのソリューション

問題を解決する 1 つの方法は、キュー・マネージャー別名 (リモート・キュー定義) を使用して、クラスター間にブリッジを確立することです。 クラスター化されたリモート・キュー定義、伝送キュー、およびチャネルを作成し、それぞれのメッセージ・フローをゲートウェイ・キュー・マネージャーで分離します。 リモート・キュー定義を追加して、ゲートウェイ・キュー・マネージャーから送信されたメッセージを分離するを参照してください。

IBM WebSphere® MQ 7.5 からは、クラスター・キュー・マネージャーは単一のクラスター伝送キューに制限されません。 次のいずれかを選択できます。

1. 追加のクラスター伝送キューを手動で定義し、それぞれの伝送キューからメッセージを転送するクラスター送信側チャネルを定義します。クラスター伝送キューを追加して、ゲートウェイ・キュー・マネージャーから送信されたクラスター・メッセージ・トラフィックを分離するを参照してください。
2. キュー・マネージャーが追加のクラスター伝送キューを自動的に作成して管理できるようにします。 この場合、クラスター送信側チャネルごとに異なるクラスター伝送キューが定義されます。クラスター伝送キューを区別するようにデフォルトを変更して、メッセージ・トラフィックを分離するを参照してください。

一部のクラスター送信側チャネルに手動で定義したクラスター伝送キューは、残りのクラスター送信側チャネルを管理するキュー・マネージャーに結合できます。 伝送キューの組み合わせは、クラスター伝送キューを追加して、ゲートウェイ・キュー・マネージャーから送信されたクラスター・メッセージ・トラフィックを分離するで使用されているアプローチです。 このソリューションでは、クラスター間のほとんどのメッセージが共通 SYSTEM.CLUSTER.TRANSMIT.QUEUE を使用します。 非常に重要な 1 つのアプリケーションがあり、そのすべてのメッセージ・フローは、手動で定義された 1 つのクラスター伝送キューを使用して他のメッセージ・フローから分離されます。

クラスター伝送キューを追加して、ゲートウェイ・キュー・マネージャーから送信されたクラスター・メッセージ・トラフィックを分離するでの構成には、制限があります。 あるクラスター・キューへと流れるメッセージ・トラフィックは、同じクラスター内の同じキュー・マネージャー上の別のクラスター・キューへと流れるメッセージ・トラフィックから分離されません。 メッセージ・トラフィックを個々のキューに分離するには、分散キューイングの一部であるリモート・キュー定義を使用することができます。 クラスターでは、複数のクラスター伝送キューを使用して、異なるクラスター送信側チャネルに向かうメッセージ・トラフィックを分離できます。 同じクラスター内の同じキュー・マネージャーにある複数のクラスター・キューは、クラスター送信側チャネルを共有します。 これらのキューのメッセージは、同じ伝送キューに保管された後、ゲートウェイ・キュー・マネージャーから転送されます。 クラスターおよびクラスター伝送キューを追加して、ゲートウェイ・キュー・マネージャーから送信されたクラスター・メッセージ・トラフィックを分離する場合の構成では、この制限を回避するために、もう 1 つのクラスターを追加し、そのキュー・マネージャーとクラスター・キューをその新しいクラスターのメンバーにします。 新しいキュー・マネージャーが、そのクラスター内の唯一のキュー・マネージャーである場合もあります。 クラスターにさらにキュー・マネージャーを追加し、同じクラスターを使用して、これらのキュー・マネージャーでもクラスター・キューを分離するという方法もあります。

関連概念

28 ページの『アクセス制御と複数のクラスター伝送キュー』
アプリケーションがメッセージをリモート・クラスター・キューに入れるタイミングをチェックするモードを 3 つの中から選択します。 これらのモードはそれぞれ、リモートでのクラスター・キューに対するチェック、ローカルでの SYSTEM.CLUSTER.TRANSMIT.QUEUE に対するチェック、クラスター・キューまたはクラスター・キュー・マネージャーのローカル・プロファイルに対するチェックを行います。

クラスター伝送キューとクラスター送信側チャネルの操作

35 ページの『クラスターのオーバーラップ』
クラスターのオーバーラップは、追加の管理機能を提供します。 名前リストを使用して、オーバーラップするクラスターの管理に必要なコマンドの数を減らします。

関連タスク

リモート・クラスター・キューへのメッセージ書き込み権限の付与

リモート・キュー定義を追加して、ゲートウェイ・キュー・マネージャーから送信されたメッセージを分離する

クラスター伝送キューを追加して、ゲートウェイ・キュー・マネージャーから送信されたクラスター・メッセージ・トラフィックを分離する

クラスターおよびクラスター伝送キューを追加して、ゲートウェイ・キュー・マネージャーから送信されたクラスター・メッセージ・トラフィックを分離する

クラスター伝送キューを区別するようにデフォルトを変更して、メッセージ・トラフィックを分離する

ゲートウェイ・キュー・マネージャーを使用した 2 つのオーバーラップするクラスターの作成

クラスター間のメッセージ・パスの構成

セキュリティー

関連資料

setmqaut

クラスター化: クラスター伝送キューの構成方法の計画

クラスター伝送キューの選択について手順を追って説明します。 1 つの共通デフォルト・キュー、個々のデフォルト・キュー、または手動で定義したキューを構成できます。

始める前に

53 ページの『使用するクラスター伝送キュー・タイプの選択方法』を参照してください。

このタスクについて

キュー・マネージャーを構成する方法を計画する際には、クラスター伝送キューに関して以下の選択肢があります。

1. クラスター・メッセージを転送するデフォルト・クラスター伝送キューをどれにするか。
   a. 共通クラスター伝送キュー SYSTEM.CLUSTER.TRANSMIT.QUEUE。
   b. 個々のクラスター伝送キュー。 キュー・マネージャーが、個々のクラスター伝送キューを管理します。 これらのキューは、モデル・キュー SYSTEM.CLUSTER.TRANSMIT.MODEL.QUEUE から永続動的キューとして作成します。 使用するクラスター送信側チャネルごとに 1 つのクラスター伝送キューを作成します。
2. 手動で作成することにしたクラスター伝送キューについては、さらに 2 つの選択肢があります。
   a. 手動で構成することにしたクラスター送信側チャネルごとに個別の伝送キューを定義する。 この場合、伝送キューの **CLCHNAME** キュー属性をクラスター送信側チャネルの名前に設定します。 この伝送キューからメッセージを転送するクラスター送信側チャネルを選択します。
   b. クラスター送信側チャネルのグループのメッセージ・トラフィックを同じクラスター伝送キューに結合する (52 ページの図 13 を参照)。 この場合、それぞれの共通伝送キューの **CLCHNAME** キュー属性を総称クラスター送信側チャネル名に設定します。 総称クラスター送信側チャネル名は、クラスター送信側チャネル名をグループ化するためのフィルターです。 例えば、SALES.* は、名前が SALES. で始まるすべてのクラスター送信側チャネルをグループ化します。 フィルター・ストリングには、任意の場所に複数のワイルドカード文字を配置できます。 ワイルドカード文字は、アスタリスク「"*"」です。 これは、ゼロから任意の数の文字を表します。

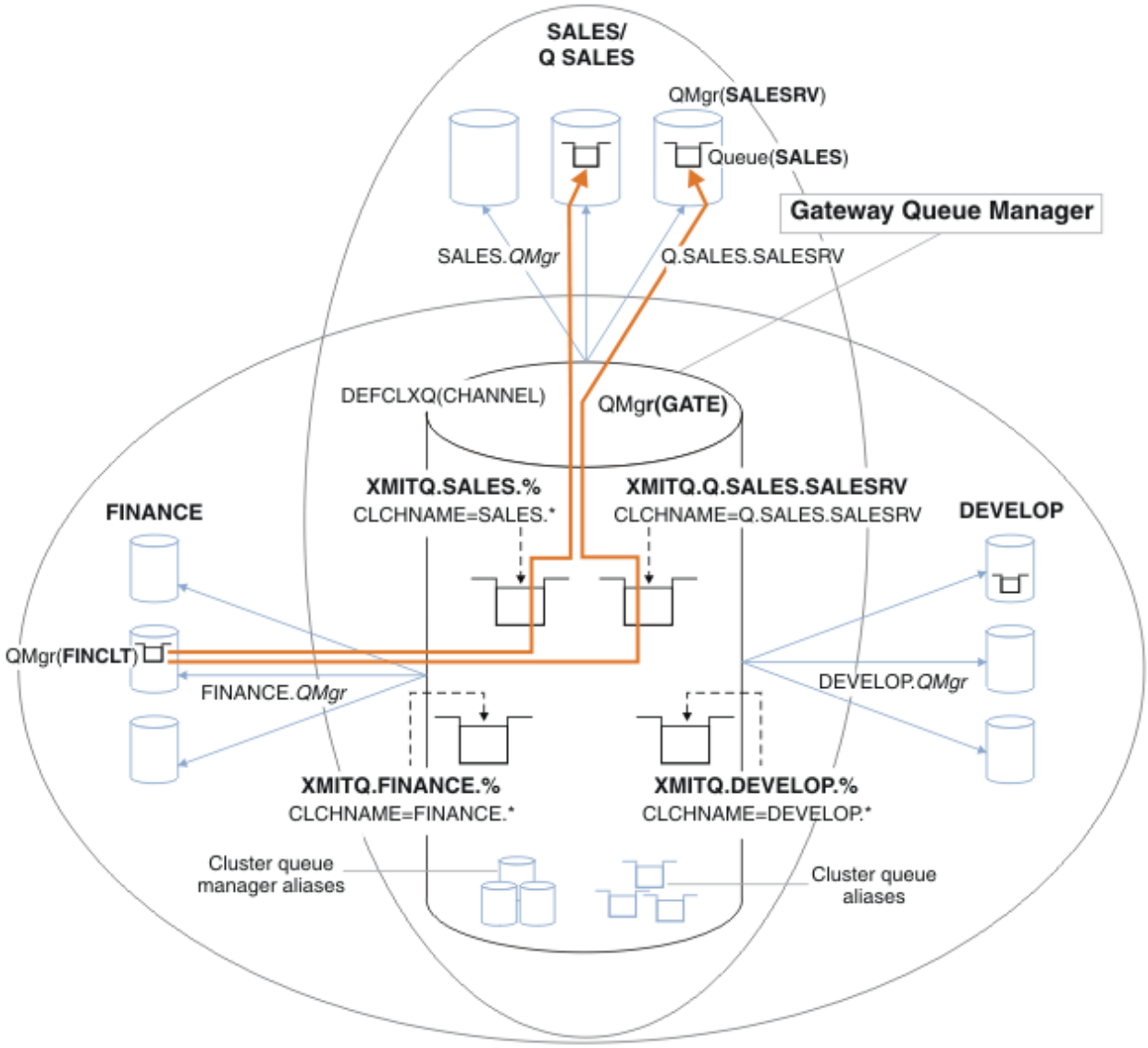


図 *13.* さまざまな部門別 *IBM MQ* クラスターに固有の伝送キューの例

手順

1. 使用するデフォルト・クラスター伝送キューのタイプを選択します。
   - 単一のクラスター伝送キューを選択するか、またはクラスター接続ごとに個別のキューを選択します。

   デフォルト設定のままにするか、または以下の **MQSC** コマンドを実行します。

   ```
   ALTER QMGR DEFCLXQ(CHANNEL)
   ```

2. 他のフローとクラスター伝送キューを共有させないメッセージ・フローを分離します。
   - 55 ページの『クラスター化: 複数のクラスター伝送キューの構成例』を参照。 この例で、分離しなければならない SALES キューは SALES クラスターのメンバーであり、SALESRV 上にあります。 SALES キューを分離するには、新しいクラスター Q.SALES を作成し、SALESRV キュー・マネージャーをそのメンバーにして、SALES が Q.SALES に属するように変更します。
   - メッセージを SALES に送信するキュー・マネージャーも、新しいクラスターのメンバーにする必要があります。 この例のようにクラスター・キュー別名とゲートウェイ・キュー・マネージャーを使用すると、多くの場合、ゲートウェイ・キュー・マネージャーを新しいクラスターのメンバーにするための変更を制限できます。

- 一方、ゲートウェイから宛先へのフローを分離しても、ゲートウェイからソース・キュー・マネージャーへのフローを分離することにはなりません。 ただし、ゲートウェイへのフローを分離しなくても、ゲートウェイからのフローを分離するだけで十分である場合もあります。 それで十分でない場合には、ソース・キュー・マネージャーを新しいクラスターに追加してください。 メッセージがゲートウェイを通過するようにするには、クラスター別名を新しいクラスターに移動し、メッセージをターゲット・キュー・マネージャーに直接送信するのではなく、引き続きゲートウェイ上のクラスター別名に送信します。

以下の手順に従って、メッセージ・フローを分離します。

a) フローの宛先を構成して、それぞれのフローのターゲット・キューが特定のクラスター内の当該キュー・マネージャー上の唯一のキューとなるようにします。

b) 体系的な命名規則に従って、作成した新規クラスターのクラスター送信側チャネルとクラスター受信側チャネルを作成します。

- 43 ページの『クラスター化: オーバーラップするクラスターについての特殊な考慮事項』を参照。

c) メッセージをターゲット・キューに送信するすべてのキュー・マネージャーに、分離された各宛先のクラスター伝送キューを定義します。

- クラスター伝送キューの命名規則は、接頭部 `XMITQ.` が付いた、クラスター・チャネル名属性 `CLCHNAME` の値を使用することです。

3. ガバナンスまたはモニター要件を満たすクラスター伝送キューを作成します。

- 典型的なガバナンスおよびモニター要件では、クラスターごとに 1 つの伝送キュー、またはキュー・マネージャーごとに 1 つの伝送キューを作成することになります。 クラスター・チャネルの命名規則 *`ClusterName. QueueManagerName`* に従うと、キュー・マネージャーのクラスター、またはキュー・マネージャーがメンバーとなっているすべてのクラスターを選択する汎用チャネル名を容易に作成できます (『55 ページの『クラスター化: 複数のクラスター伝送キューの構成例』』を参照)。
- 汎用チャネル名に対応するように、アスタリスク記号を % 記号に置き換えてクラスター伝送キューの命名規則を拡張します。 例:

```
DEFINE QLOCAL(XMITQ.SALES.%) USAGE(XMITQ) CLCHNAME(SALES.*)
```

関連概念

クラスター伝送キューとクラスター送信側チャネルの操作

28 ページの『アクセス制御と複数のクラスター伝送キュー』
アプリケーションがメッセージをリモート・クラスター・キューに入れるタイミングをチェックするモードを 3 つの中から選択します。 これらのモードはそれぞれ、リモートでのクラスター・キューに対するチェック、ローカルでの SYSTEM.CLUSTER.TRANSMIT.QUEUE に対するチェック、クラスター・キューまたはクラスター・キュー・マネージャーのローカル・プロファイルに対するチェックを行います。

35 ページの『クラスターのオーバーラップ』
クラスターのオーバーラップは、追加の管理機能を提供します。 名前リストを使用して、オーバーラップするクラスターの管理に必要なコマンドの数を減らします。

関連タスク

リモート・キュー定義を追加して、ゲートウェイ・キュー・マネージャーから送信されたメッセージを分離する

クラスター伝送キューを追加して、ゲートウェイ・キュー・マネージャーから送信されたクラスター・メッセージ・トラフィックを分離する

クラスターおよびクラスター伝送キューを追加して、ゲートウェイ・キュー・マネージャーから送信されたクラスター・メッセージ・トラフィックを分離する

クラスター伝送キューを区別するようにデフォルトを変更して、メッセージ・トラフィックを分離する

ゲートウェイ・キュー・マネージャーを使用した 2 つのオーバーラップするクラスターの作成

クラスター間のメッセージ・パスの構成

使用するクラスター伝送キュー・タイプの選択方法
各種のクラスター伝送キュー構成オプションの中から選択する方法。

クラスター送信側チャネルに関連付けられるクラスター伝送キューを選択できます。

1. 単一のデフォルト・クラスター送信側キュー SYSTEM.CLUSTER.TRANSMIT.QUEUE にすべてのクラスター送信側チャネルを関連付けることができます。このオプションがデフォルトです。
2. すべてのクラスター送信側チャネルが別個のクラスター伝送キューと自動的に関連付けられるように設定することができます。 これらのキューは、キュー・マネージャーによってモデル・キュー SYSTEM.CLUSTER.TRANSMIT.MODEL.QUEUE から作成され、SYSTEM.CLUSTER.TRANSMIT.*ChannelName* という名前が付けられます。 キュー・マネージャーの属性 **DEFCLXQ** が CHANNEL に設定されている場合、チャネルは一意的に名前が付けられたクラスター伝送キューを使用します。
3. 単一のクラスター伝送キューで処理する特定のクラスター送信側チャネルを設定することができます。 このオプションを選択するには、伝送キューを作成し、その **CLCHNAME** 属性をクラスター送信側チャネルの名前に設定します。
4. 単一のクラスター伝送キューで処理するクラスター送信側チャネルのグループを選択することができます。 このオプションを選択するには、伝送キューを作成し、その **CLCHNAME** 属性を総称チャネル名 (*ClusterName*.* など) に設定します。 43 ページの『クラスター化: オーバーラップするクラスターについての特殊な考慮事項』 で説明されている命名規則に従ってクラスター・チャネルの名前を設定する場合、この名前によって、クラスター *ClusterName* 内のキュー・マネージャーに接続されたすべてのクラスター・チャネルが選択されます。

一部のクラスター送信側チャネルのデフォルト伝送キュー・オプションのいずれかを、任意の数の特定および汎用クラスター伝送キュー構成と組み合わせることができます。

ベスト・プラクティス

ほとんどの場合、既存の IBM MQ インストール済み環境ではデフォルトの構成が最適な選択となります。 クラスター・キュー・マネージャーは、クラスター・メッセージを単一のクラスター伝送キュー SYSTEM.CLUSTER.TRANSMIT.QUEUE に保管します。 このデフォルトを変更して、異なるキュー・マネージャーと異なるクラスターのメッセージを個々の伝送キューに保管することも、独自の伝送キューを定義することもできます。

ほとんどの場合、新しい IBM MQ インストール済み環境でもデフォルトの構成が最適な選択となります。 デフォルト構成から、クラスター送信側チャネルごとに 1 つの伝送キューを使用する代替デフォルトへの切り替えプロセスは、自動的に行われます。 元の状態への切り替えも自動です。 どちらを選択するかは重要ではありません。選択は元に戻すことができます。

別の構成を選択する理由は、機能やパフォーマンスよりも、ガバナンスおよび管理に関係します。 いくつかの例外を除き、複数のクラスター伝送キューを構成することによって、キュー・マネージャーの動作に利益がもたらされることはありません。 結果的にキューの数が増えて、すでにセットアップした、単一の伝送キューを参照するモニターおよび管理プロシージャーを変更しなければならなくなります。 そのため、結局は、別の選択を行う強力なガバナンス上または管理上の理由がない限り、デフォルト構成のままにすることが最善の選択です。

これらの例外は、どちらも、SYSTEM.CLUSTER.TRANSMIT.QUEUE に保管されるメッセージの数が増加した場合に起こる事態に関係しています。 ある宛先へのメッセージを他の宛先へのメッセージから分離するためにあらゆる措置を講じると、ある宛先でのチャネルおよび配信の問題が別の宛先への配信に影響しなくなるはずです。 ただし、SYSTEM.CLUSTER.TRANSMIT.QUEUE に保管されるメッセージの数は、ある 1 つの宛先へのメッセージの配信速度が遅いために増加することもあります。 1 つの宛先に対する SYSTEM.CLUSTER.TRANSMIT.QUEUE 上のメッセージ数が他の宛先へのメッセージの配信に影響することがあります。

1 つの伝送キューがいっぱいになったために問題が発生するということがないようにするため、構成で十分なキャパシティーを確保するようにしてください。 そうすれば、宛先で障害が発生し、メッセージ・バックログが溜まり始めたとしても、問題を修正する時間をとることができます。

メッセージが、クラスター・ゲートウェイなどのハブ・キュー・マネージャー経由でルーティングされる場合、これらのメッセージは共通伝送キュー SYSTEM.CLUSTER.TRANSMIT.QUEUE を共有します。 ゲートウェイ・キュー・マネージャー上の SYSTEM.CLUSTER.TRANSMIT.QUEUE に保管されているメッセージ数が最大件数に達すると、キュー・マネージャーは、件数が減るまで、この伝送キューへの新しいメッセージを拒否し始めます。 この輻輳 (ふくそう) は、ゲートウェイ経由でルーティングされるすべての宛先へ

のメッセージに影響を与えます。 メッセージは、メッセージをゲートウェイに送信する他のキュー・マネージャー上の伝送キューにバックアップされます。 この問題は、メッセージがキュー・マネージャーのエラー・ログに書き込まれ、メッセージのスループットが低下し、メッセージが送信されてからその宛先に到着するまでの経過時間が長くなるという形で現れます。

単一の伝送キューでの輻輳の影響は、伝送キューがフルになる前から明らかになる可能性があります。 メッセージ・トラフィックに大規模な非持続メッセージと小さなメッセージが混在する場合、小さなメッセージの配信時間は伝送キューがいっぱいになるにつれ長くなります。 この遅延は、通常はディスクに書き込まれない大規模な非持続メッセージがディスクに書き込まれるために発生します。 速度が重視される重大なメッセージ・フローと他の混在メッセージ・フローがクラスター伝送キューを共有している場合には、その重大なメッセージ・フローを他のメッセージ・フローと切り分けるための特別メッセージ・パスを構成することを検討する価値があります。 クラスターおよびクラスター伝送キューを追加して、ゲートウェイ・キュー・マネージャーから送信されたクラスター・メッセージ・トラフィックを分離するを参照してください。

個々のクラスター伝送キューを構成するもう 1 つの理由は、ガバナンス要件を満たすため、あるいは、さまざまなクラスター宛先に送信されるメッセージのモニターを簡素化するためです。 例えば、ある宛先へのメッセージが、他の宛先へのメッセージと伝送キューを決して共有しないことを実証しなければならない場合があります。

すべてのクラスター送信側チャネルに異なるクラスター伝送キューを作成するには、デフォルト・クラスター伝送キューを制御するキュー・マネージャー属性 **DEFCLXQ** を変更します。 複数の宛先が 1 つのクラスター送信側チャネルを共有することは可能であるため、この目標を完全に満たすラスターを計画する必要があります。 クラスターおよびクラスター伝送キューを追加して、ゲートウェイ・キュー・マネージャーから送信されたクラスター・メッセージ・トラフィックを分離するの方式をすべてのクラスター・キューに体系的に適用します。 目標とする結果は、別のクラスター宛先とクラスター送信側チャネルを共有するクラスター宛先が 1 つもなくなることです。 その結果、クラスター宛先へのメッセージはいずれも、別の宛先へのメッセージとクラスター伝送キューを共有することがなくなります。

特定のメッセージ・フローに別個のクラスター伝送キューを作成すると、その宛先へのメッセージのフローを簡単にモニターできるようになります。 新しいクラスター伝送キューを使用するには、キューを定義し、そのキューをクラスター送信側チャネルに関連付けて、チャネルを停止して開始します。 これは永続的な変更である必要はありません。 しばらくの間、伝送キューをモニターするためにメッセージ・フローを切り分けて、後でデフォルトの伝送キューを再び使用するように戻すこともできます。

関連タスク

クラスター化: 複数のクラスター伝送キューの構成例
このタスクでは、複数のクラスター伝送キューの計画手順を、3 つのオーバーラップするクラスターに適用します。 ここで要件となるのは、1 つのクラスター・キューへのメッセージ・フローを、その他すべてのメッセージ・フローから分離すること、そして各クラスターへのメッセージをそれぞれ異なるクラスター伝送キューに保管することです。

クラスター化: クラスター伝送キューの切り替え
既存の実働キュー・マネージャーのクラスター伝送キューに加えた変更を有効にしていく方法を計画します。

クラスター化: 複数のクラスター伝送キューの構成例
このタスクでは、複数のクラスター伝送キューの計画手順を、3 つのオーバーラップするクラスターに適用します。 ここで要件となるのは、1 つのクラスター・キューへのメッセージ・フローを、その他すべてのメッセージ・フローから分離すること、そして各クラスターへのメッセージをそれぞれ異なるクラスター伝送キューに保管することです。

このタスクについて

このタスクのステップでは、51 ページの『クラスター化: クラスター伝送キューの構成方法の計画』で説明した手順を適用して、56 ページの図 14 に示す構成を達成する方法を説明します。 これは、個々のクラスター伝送キューを設定して構成されたゲートウェイ・キュー・マネージャーを使用した、3 つのオーバーラップするクラスターの一例です。 クラスターを定義するための MQSC コマンドについては、58 ページの『サンプル・クラスターの作成』で説明しています。

この例には、2 つの要件があります。 1 つは、ゲートウェイ・キュー・マネージャーから売り上げをログに記録する販売アプリケーションへのメッセージ・フローを分離することです。 2 つ目の要件は、さまざまな部門領域への送信を待機しているメッセージの数を任意の時点で照会することです。 SALES、FINANCE、および DEVELOP の各クラスターがすでに定義されています。 クラスター・メッセージは現在、SYSTEM.CLUSTER.TRANSMIT.QUEUE から転送されています。

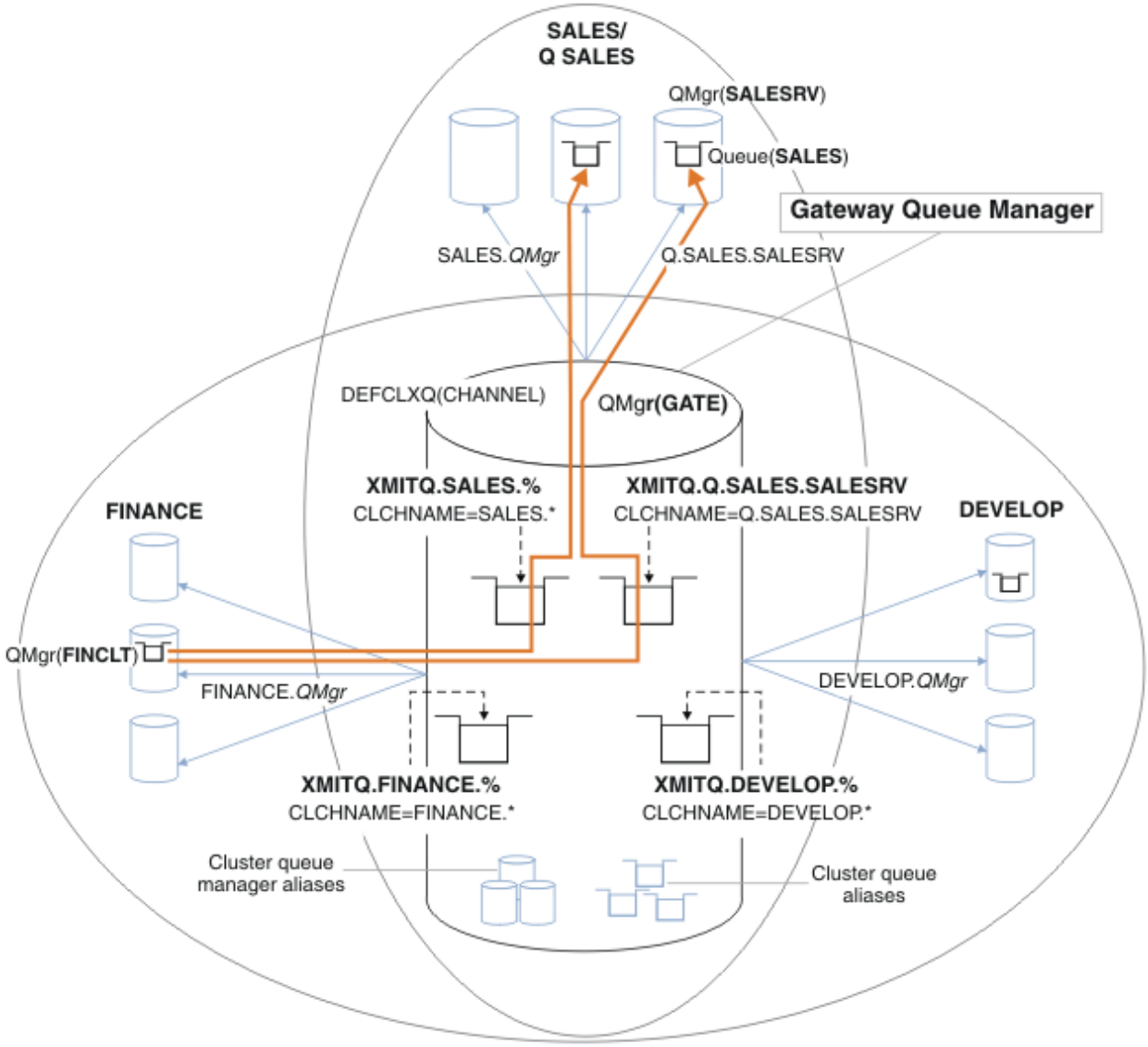


図 *14.* さまざまな部門別 *IBM MQ* クラスターに固有の伝送キューの例

クラスターを変更する手順は以下のとおりです。 定義については、 新しいクラスターで販売キューを分離し、ゲートウェイ・クラスター伝送キューを分離するための変更を参照してください。

手順

1. 最初の構成ステップは、『" 使用するデフォルト・クラスター伝送キューのタイプを選択します "』に対するものです。

   決定内容は、個々のデフォルト・クラスター伝送キューを作成することです。それには、GATE キュー・マネージャーで以下の **MQSC** コマンドを実行します。

   ```
   ALTER QMGR DEFCLXQ(CHANNEL)
   ```

手動でクラスター伝送キューを定義することが目的なので、このデフォルトを選択する強力な理由はありません。 この選択を診断する価値は低いものです。 手動による定義が誤って行われ、メッセージがデフォルト・クラスター伝送キューに流れる場合、それは、永続動的クラスター伝送キューの作成という形で現れます。

2. 2 番目の構成ステップは、『" 他のフローとクラスター伝送キューを共有させないメッセージ・フローを分離します "』です。

   この例では、SALESRV 上のキュー SALES からメッセージを受信する販売アプリケーションを分離する必要があります。 ゲートウェイ・キュー・マネージャーからのメッセージの分離のみが必要です。 3 つのサブステップは以下のとおりです。

   a) " フローの宛先を構成して、それぞれのフローのターゲット・キューが特定のクラスター内の当該キュー・マネージャー上の唯一のキューとなるようにします ".

      この例では、キュー・マネージャー SALESRV を販売部門内の新しいクラスターに追加する必要があります。 分離を必要とするキューがほとんどない場合、SALES キューに固有のクラスターを作成することもできます。 クラスター名に使用できる命名規則は、クラスターに `Q.QueueName` のような名前 (例えば、Q.SALES) を付けることです。 多数のキューを分離しなければならない場合には、必要な場所と時機に応じて、分離されたキューのクラスターを作成するほうが、より実用的な代替手段となります。 クラスター名は、`QUEUES.n` のようになります。

      この例では、新しいクラスターを Q.SALES という名前にしています。 新規クラスターを追加するには、 新規クラスター内の販売キューを分離し、ゲートウェイ・クラスター伝送キューを分離するための変更の定義を参照してください。 定義の変更内容を以下に要約します。

      i) リポジトリー・キュー・マネージャーのクラスターの名前リストに、Q.SALES を追加します。 この名前リストは、キュー・マネージャーの **REPOSNL** パラメーターで参照されます。

      ii) ゲートウェイ・キュー・マネージャーのクラスターの名前リストに、Q.SALES を追加します。 この名前リストは、ゲートウェイ・キュー・マネージャーのすべてのクラスター・キュー別名定義およびクラスター・キュー・マネージャー別名定義で参照されます。

      iii) キュー・マネージャー SALESRV に、このキュー・マネージャーがメンバーとなる両方のクラスターの名前リストを作成し、SALES キューのクラスター・メンバーシップを変更します。

      ```
      DEFINE NAMELIST(CLUSTERS) NAMES(SALES, Q.SALES) REPLACE
      ALTER QLOCAL(SALES) CLUSTER(' ') CLUSNL(SALESRV.CLUSTERS)
      ```

      SALES キューは、遷移の目的でのみ、両方のクラスターのメンバーになります。 新しい構成が実行中になった後、SALES クラスターから SALES キューを削除します (61 ページの図 15 を参照)。

   b) " 体系的な命名規則に従って、作成した新規クラスターのクラスター送信側チャネルとクラスター受信側チャネルを作成します ".

      i) 各リポジトリー・キュー・マネージャーに、クラスター受信側チャネル `Q.SALES.RepositoryQMgr` を追加します。

      ii) 各リポジトリー・キュー・マネージャーに、クラスター送信側チャネル `Q.SALES.OtherRepositoryQMgr` を追加し、他方のリポジトリー・マネージャーに接続します。 これらのチャネルを開始します。

      iii) いずれか実行中のリポジトリー・キュー・マネージャーに、クラスター受信側チャネル Q.SALES.SALESRV および Q.SALES.GATE を追加します。

      iv) SALESRV および GATE キュー・マネージャーに、クラスター送信側チャネル Q.SALES.SALESRV および Q.SALES.GATE を追加します。 クラスター送信側チャネルを、クラスター受信側チャネルを作成したリポジトリー・キュー・マネージャーに接続します。

   c) " メッセージをターゲット・キューに送信するすべてのキュー・マネージャーに、分離された各宛先のクラスター伝送キューを定義します ".

ゲートウェイ・キュー・マネージャーに、Q.SALES.SALESRV クラスター送信側チャネルのクラスター伝送キュー XMITQ.Q.SALES.SALESRV を定義します。

```
DEFINE QLOCAL(XMITQ.Q.SALES.SALESRV) USAGE(XMITQ) CLCHNAME(Q.SALES.SALESRV) REPLACE
```

3. 3 番目の構成ステップは、『" ガバナンスまたはモニター要件を満たすクラスター伝送キューを作成します "』です。

   ゲートウェイ・キュー・マネージャーに、クラスター伝送キューを定義します。

```
DEFINE QLOCAL(XMITQ.SALES)  USAGE(XMITQ) CLCHNAME(SALES.*)  REPLACE
DEFINE QLOCAL(XMITQ.DEVELOP) USAGE(XMITQ) CLCHNAME(DEVELOP.*) REPLACE
DEFINE QLOCAL(XMITQ.FINANCE) USAGE(XMITQ) CLCHNAME(SALES.*)  REPLACE
```

次のタスク

ゲートウェイ・キュー・マネージャーで新しい構成に切り替えます。

切り替えをトリガーするには、新規チャネルを開始し、別の伝送キューに関連付けられるようになったチャネルを再始動します。 あるいは、ゲートウェイ・キュー・マネージャーを停止してから再始動するという方法もあります。

1. ゲートウェイ・キュー・マネージャーで以下のチャネルを停止します。

```
SALES. Qmgr
DEVELOP. Qmgr
FINANCE. Qmgr
```

2. ゲートウェイ・キュー・マネージャーで以下のチャネルを開始します。

```
SALES. Qmgr
DEVELOP. Qmgr
FINANCE. Qmgr
Q.SALES.SAVESRV
```

切り替えが完了したら、SALES クラスターから SALES キューを削除します (61 ページの図 15 を参照)。

関連概念

使用するクラスター伝送キュー・タイプの選択方法
各種のクラスター伝送キュー構成オプションの中から選択する方法。

関連タスク

クラスター化: クラスター伝送キューの切り替え
既存の実働キュー・マネージャーのクラスター伝送キューに加えた変更を有効にしていく方法を計画します。

サンプル・クラスターの作成
サンプル・クラスターを作成し、それを変更して SALES キューを分離し、ゲートウェイ・キュー・マネージャーでメッセージを分離するための定義と手順を説明します。

このタスクについて

FINANCE クラスター、 SALES クラスター、および Q.SALES クラスターを作成するための完全な **MQSC** コマンドは、 基本クラスターの定義、 新規クラスターで販売キューを分離し、ゲートウェイ・クラスター伝送キューを分離するための変更、および キュー・マネージャー SALESRV の販売キューを販売クラスターから削除するに記載されています。 定義を短縮するため、DEVELOP クラスターは定義から省略されています。

手順

1. SALES および FINANCE の各クラスター、およびゲートウェイ・キュー・マネージャーを作成します。

   a) キュー・マネージャーを作成します。

59 ページの表 4 に示されている各キュー・マネージャー名に対して、コマンド `crtmqm -sax -u SYSTEM.DEAD.LETTER.QUEUE` *QmgrName* を実行します。

表 4. キュー・マネージャー名およびポート番号

説明	キュー・マネージャー名	ポート番号
財務リポジトリー	FINR1	1414
財務リポジトリー	FINR2	1415
財務クライアント	FINCLT	1418
販売リポジトリー	SALER1	1416
販売リポジトリー	SALER2	1417
販売サーバー	SALESRV	1419
ゲートウェイ	GATE	1420

b) すべてのキュー・マネージャーを開始します。

59 ページの表 4 に示されている各キュー・マネージャー名に対して、コマンド `strmqm` *QmgrName* を実行します。

c) 各キュー・マネージャーの定義を作成します。

コマンド `runmqsc` *QmgrName* < *filename* を実行します。これらのファイルは 基本クラスターの定義にリストされており、ファイル名はキュー・マネージャー名と一致します。

基本クラスターの定義
finr1.txt

```
DEFINE LISTENER(1414) TRPTYPE(TCP) IPADDR(localhost) CONTROL(QMGR) PORT(1414) REPLACE
START LISTENER(1414)
ALTER QMGR REPOS(FINANCE)
DEFINE CHANNEL(FINANCE.FINR2) CHLTYPE(CLUSSDR) CONNAME('localhost(1415)')
CLUSTER(FINANCE) REPLACE
DEFINE CHANNEL(FINANCE.FINR1) CHLTYPE(CLUSRCVR) CONNAME('localhost(1414)')
CLUSTER(FINANCE) REPLACE
```

finr2.txt

```
DEFINE LISTENER(1415) TRPTYPE(TCP) IPADDR(localhost) CONTROL(QMGR) PORT(1415) REPLACE
START LISTENER(1415)
ALTER QMGR REPOS(FINANCE)
DEFINE CHANNEL(FINANCE.FINR1) CHLTYPE(CLUSSDR) CONNAME('localhost(1414)')
CLUSTER(FINANCE) REPLACE
DEFINE CHANNEL(FINANCE.FINR2) CHLTYPE(CLUSRCVR) CONNAME('localhost(1415)')
CLUSTER(FINANCE) REPLACE
```

finclt.txt

```
DEFINE LISTENER(1418) TRPTYPE(TCP) IPADDR(localhost) CONTROL(QMGR) PORT(1418) REPLACE
START LISTENER(1418)
DEFINE CHANNEL(FINANCE.FINR1) CHLTYPE(CLUSSDR)  CONNAME('localhost(1414)')
CLUSTER(FINANCE) REPLACE
DEFINE CHANNEL(FINANCE.FINCLT) CHLTYPE(CLUSRCVR) CONNAME('localhost(1418)')
CLUSTER(FINANCE) REPLACE
DEFINE QMODEL(SYSTEM.SAMPLE.REPLY) REPLACE
```

saler1.txt

```
DEFINE LISTENER(1416) TRPTYPE(TCP) IPADDR(localhost) CONTROL(QMGR) PORT(1416) REPLACE
START LISTENER(1416)
ALTER QMGR REPOS(SALES)
DEFINE CHANNEL(SALES.SALER2) CHLTYPE(CLUSSDR) CONNAME('localhost(1417)')
CLUSTER(SALES) REPLACE
```

```
DEFINE CHANNEL(SALES.SALER1) CHLTYPE(CLUSRCVR) CONNAME('localhost(1416)')
CLUSTER(SALES) REPLACE
```

saler2.txt

```
DEFINE LISTENER(1417) TRPTYPE(TCP) IPADDR(localhost) CONTROL(QMGR) PORT(1417) REPLACE
START LISTENER(1417)
ALTER QMGR REPOS(SALES)
DEFINE CHANNEL(SALES.SALER1) CHLTYPE(CLUSSDR) CONNAME('localhost(1416)')
CLUSTER(SALES) REPLACE
DEFINE CHANNEL(SALES.SALER2) CHLTYPE(CLUSRCVR) CONNAME('localhost(1417)')
CLUSTER(SALES) REPLACE
```

salesrv.txt

```
DEFINE LISTENER(1419) TRPTYPE(TCP) IPADDR(localhost) CONTROL(QMGR) PORT(1419) REPLACE
START LISTENER(1419)
DEFINE CHANNEL(SALES.SALER1) CHLTYPE(CLUSSDR) CONNAME('localhost(1416)')
CLUSTER(SALES) REPLACE
DEFINE CHANNEL(SALES.SALESRV) CHLTYPE(CLUSRCVR) CONNAME('localhost(1419)')
CLUSTER(SALES) REPLACE
DEFINE QLOCAL(SALES) CLUSTER(SALES) TRIGGER INITQ(SYSTEM.DEFAULT.INITIATION.QUEUE)
PROCESS(ECHO) REPLACE
DEFINE PROCESS(ECHO) APPLICID(AMQSECH) REPLACE
```

gate.txt

```
DEFINE LISTENER(1420) TRPTYPE(TCP) IPADDR(LOCALHOST) CONTROL(QMGR) PORT(1420) REPLACE
START LISTENER(1420)
DEFINE NAMELIST(ALL) NAMES(SALES, FINANCE)
DEFINE CHANNEL(FINANCE.FINR1) CHLTYPE(CLUSSDR) CONNAME('LOCALHOST(1414)')
CLUSTER(FINANCE) REPLACE
DEFINE CHANNEL(FINANCE.GATE) CHLTYPE(CLUSRCVR) CONNAME('LOCALHOST(1420)')
CLUSTER(FINANCE) REPLACE
DEFINE CHANNEL(SALES.SALER1) CHLTYPE(CLUSSDR) CONNAME('LOCALHOST(1416)')
CLUSTER(SALES) REPLACE
DEFINE CHANNEL(SALES.GATE)  CHLTYPE(CLUSRCVR) CONNAME('LOCALHOST(1420)')
CLUSTER(SALES) REPLACE
DEFINE QALIAS(A.SALES) CLUSNL(ALL) TARGET(SALES) TARGTYPE(QUEUE) DEFBIND(NOTFIXED)
REPLACE
DEFINE QREMOTE(FINCLT) RNAME(' ') RQMNAME(FINCLT) CLUSNL(ALL) REPLACE
DEFINE QREMOTE(SALESRV) RNAME(' ') RQMNAME(SALESRV) CLUSNL(ALL) REPLACE
```

2. サンプル要求プログラムを実行して、構成をテストします。
   a) SALESRV キュー・マネージャーでトリガー・モニター・プログラムを開始します。

      Windows で、コマンド・ウィンドウを開き、コマンド `runmqtrm -m SALESRV` を実行します。

   b) サンプル要求プログラムを実行し、要求を送信します。

      Windows で、コマンド・ウィンドウを開き、コマンド `amqsreq A.SALES FINCLT` を実行します。

      要求メッセージがエコー出力され、15 秒後にサンプル・プログラムが終了します。
3. ゲートウェイ・キュー・マネージャーで、Q.SALES クラスター内の SALES キューを分離して、SALES および FINANCE クラスターへのクラスター・メッセージを分離するための定義を作成します。

   コマンド `runmqsc` *QmgrName* < *filename* を実行します。これらのファイルは以下のリストにリストされており、ファイル名はキュー・マネージャー名とほぼ一致しています。

   新規クラスター内で販売キューを分離し、ゲートウェイ・クラスター伝送キューを分離するための変更
   chgsaler1.txt

```
DEFINE NAMELIST(CLUSTERS) NAMES(SALES, Q.SALES)
ALTER QMGR REPOS(' ') REPOSNL(CLUSTERS)
DEFINE CHANNEL(Q.SALES.SALER2) CHLTYPE(CLUSSDR) CONNAME('localhost(1417)')
CLUSTER(Q.SALES) REPLACE
DEFINE CHANNEL(Q.SALES.SALER1) CHLTYPE(CLUSRCVR) CONNAME('localhost(1416)')
CLUSTER(Q.SALES) REPLACE
```

   chgsaler2.txt

```
DEFINE NAMELIST(CLUSTERS) NAMES(SALES, Q.SALES)
ALTER QMGR REPOS(' ') REPOSNL(CLUSTERS)
```

```
DEFINE CHANNEL(Q.SALES.SALER1) CHLTYPE(CLUSSDR) CONNAME('localhost(1416)')
CLUSTER(Q.SALES) REPLACE
DEFINE CHANNEL(Q.SALES.SALER2) CHLTYPE(CLUSRCVR) CONNAME('localhost(1417)')
CLUSTER(Q.SALES) REPLACE
```

chgsalesrv.txt

```
DEFINE NAMELIST (CLUSTERS) NAMES(SALES, Q.SALES)
DEFINE CHANNEL(Q.SALES.SALER1) CHLTYPE(CLUSSDR) CONNAME('localhost(1416)')
CLUSTER(Q.SALES) REPLACE
DEFINE CHANNEL(Q.SALES.SAVESRV) CHLTYPE(CLUSRCVR) CONNAME('localhost(1419)')
CLUSTER(Q.SALES) REPLACE
ALTER QLOCAL (SALES) CLUSTER(' ') CLUSNL(CLUSTERS)
```

chggate.txt

```
ALTER NAMELIST(ALL) NAMES(SALES, FINANCE, Q.SALES)
ALTER QMGR DEFCLXQ(CHANNEL)
DEFINE CHANNEL(Q.SALES.SALER1) CHLTYPE(CLUSSDR)  CONNAME('localhost(1416)')
CLUSTER(Q.SALES) REPLACE
DEFINE CHANNEL(Q.SALES.GATE)  CHLTYPE(CLUSRCVR) CONNAME('localhost(1420)')
CLUSTER(Q.SALES) REPLACE
DEFINE QLOCAL (XMITQ.Q.SALES.SALESRV) USAGE(XMITQ) CLCHNAME(Q.SALES.SALESRV) REPLACE
DEFINE QLOCAL (XMITQ.SALES)     USAGE(XMITQ) CLCHNAME(SALES.*)  REPLACE
DEFINE QLOCAL (XMITQ.FINANCE)    USAGE(XMITQ) CLCHNAME(FINANCE.*) REPLACE
```

4. SALES キューを SALES クラスターから削除します。

   61 ページの図 15 に記載されている **MQSC** コマンドを実行します。

   ```
   ALTER QLOCAL(SALES) CLUSTER('Q.SALES') CLUSNL(' ')
   ```

   図 *15.* 販売クラスターからのキュー・マネージャー *SALESRV* 上の販売キューの削除

5. チャネルを新しい伝送キューに切り替えます。

   要件は、GATE キュー・マネージャーが使用しているすべてのチャネルを停止してから開始することです。 これを最も少ない数のコマンドで行うには、キュー・マネージャーを停止して、始動します。

   ```
   endmqm -i GATE
   strmqm GATE
   ```

次のタスク

1. サンプル要求プログラムを再実行して、新しい構成が機能することを検証します。ステップ 60 ページの『2』 を参照してください。
2. GATE キュー・マネージャー上のすべてのクラスター伝送キューを介したメッセージ・フローをモニターします。

   a. クラスター伝送キューごとの定義を変更し、キュー・モニターを有効にします。

   ```
   ALTER QLOCAL(SYSTEM.CLUSTER.TRANSMIT.
   name) STATQ(ON)
   ```

   b. キュー・マネージャー統計モニターが OFF になっていることを確認し (出力を最小限にするため)、モニター間隔を低い値に設定します (複数のテストを効率的に行うため)。

   ```
   ALTER QMGR STATINT(60) STATCHL(OFF) STATQ(OFF) STATMQI(OFF) STATACLS(OFF)
   ```

   c. GATE キュー・マネージャーを再始動します。

   d. サンプル要求プログラムを何回か実行し、SYSTEM.CLUSTER.TRANSMIT.Q.SALES.SALESRV および SYSTEM.CLUSTER.TRANSMIT.QUEUE を通過するメッセージが同じ数であることを確認します。 要求は SYSTEM.CLUSTER.TRANSMIT.Q.SALES.SALESRV を経由し、応答は SYSTEM.CLUSTER.TRANSMIT.QUEUE を経由します。

   ```
   amqsmon -m GATE -t statistics
   ```

e. いくつかの間隔の結果は以下のとおりです。

```
C:\Documents and Settings\Admin>amqsmon -m GATE -t statistics
MonitoringType: QueueStatistics
QueueManager: 'GATE'
IntervalStartDate: '2012-02-27'
IntervalStartTime: '14.59.20'
IntervalEndDate: '2012-02-27'
IntervalEndTime: '15.00.20'
CommandLevel: 700
ObjectCount: 2
QueueStatistics: 0
QueueName: 'SYSTEM.CLUSTER.TRANSMIT.QUEUE'
CreateDate: '2012-02-24'
CreateTime: '15.58.15'
...
Put1Count: [0, 0]
Put1FailCount: 0
PutBytes: [435, 0]
GetCount: [1, 0]
GetBytes: [435, 0]
...
QueueStatistics: 1
QueueName: 'SYSTEM.CLUSTER.TRANSMIT.Q.SALES.SAVESRV'
CreateDate: '2012-02-24'
CreateTime: '16.37.43'
...
PutCount: [1, 0]
PutFailCount: 0
Put1Count: [0, 0]
Put1FailCount: 0
PutBytes: [435, 0]
GetCount: [1, 0]
GetBytes: [435, 0]
...
MonitoringType: QueueStatistics
QueueManager: 'GATE'
IntervalStartDate: '2012-02-27'
IntervalStartTime: '15.00.20'
IntervalEndDate: '2012-02-27'
IntervalEndTime: '15.01.20'
CommandLevel: 700
ObjectCount: 2
QueueStatistics: 0
QueueName: 'SYSTEM.CLUSTER.TRANSMIT.QUEUE'
CreateDate: '2012-02-24'
CreateTime: '15.58.15'
...
PutCount: [2, 0]
PutFailCount: 0
Put1Count: [0, 0]
Put1FailCount: 0
PutBytes: [863, 0]
GetCount: [2, 0]
GetBytes: [863, 0]
...
```

```
QueueStatistics: 1
QueueName: 'SYSTEM.CLUSTER.TRANSMIT.Q.SALES.SAVESRV'
CreateDate: '2012-02-24'
CreateTime: '16.37.43'
...
PutCount: [2, 0]
PutFailCount: 0
Put1Count: [0, 0]
Put1FailCount: 0
PutBytes: [863, 0]
GetCount: [2, 0]
GetBytes: [863, 0]
...
2 Records Processed.
```

最初の間隔では 1 つの要求および応答メッセージが送信され、2 回目の間隔では 2 つ送信されました。 要求メッセージは SYSTEM.CLUSTER.TRANSMIT.Q.SALES.SAVESRV に配置され、応答メッセージは SYSTEM.CLUSTER.TRANSMIT.QUEUE に配置されたと推測できます。

クラスター化: クラスター伝送キューの切り替え
既存の実働キュー・マネージャーのクラスター伝送キューに加えた変更を有効にしていく方法を計画します。

始める前に

切り替えプロセスによって新規伝送キューに転送する必要のあるメッセージの数を減らすと、切り替えはより迅速に完了します。 さらに続行する前に、伝送キューを空にすることを試みるべき理由については、クラスター送信側チャネルを異なる伝送キューに切り替えるプロセスの仕組みを参照してください。

このタスクについて

クラスター伝送キューに加える変更を有効にする方法には、次の 2 つの選択肢があります。

1. キュー・マネージャーが自動で変更するようにする。 これはデフォルトです。 次回クラスター送信側チャネルが開始するときに、キュー・マネージャーはクラスター送信側チャネルを、保留中の伝送キューの変更内容に切り替えます。
2. 手動で変更する。 クラスター送信側チャネルが停止しているときに、それに対する変更を行うことができます。 クラスター伝送キューを別のものに切り替えてから、クラスター送信側チャネルを開始します。

これら 2 つのオプションのどちらを選ぶか、また切り替えをどのように行うかを決定するに当たって、どのような要素を考慮すればよいのでしょうか。

手順

- オプション 1: キュー・マネージャーが自動で変更するようにする。64 ページの『アクティブなクラスター送信側チャネルを別のクラスター送信側キューのセットに切り替える』を参照。

  キュー・マネージャー側で自動に切り替わるようにするには、このオプションを選択します。

  言い換えれば、このオプションの場合、ユーザーが強制的にチャネルを停止しなくても、キュー・マネージャーはクラスター送信側チャネルを切り替えます。 早く切り替わるように、チャネルを強制的に停止してから開始することもできます。 このスイッチは、チャネルの開始時に開始され、チャネルの実行中に実行されます。これはオプション 2 とは異なります。 オプション 2 では、チャネルが停止しているときにスイッチが実行されます。

  このオプションを選択して、切り替えが自動的に行われるようにすると、クラスター送信側チャネルが開始したときに、切り替えプロセスが開始します。 チャネルが停止していない場合は、処理するメッセージがあれば、チャネルが活動状態でなくなってから、切り替えプロセスが開始します。 チャネルが停止している場合は、START CHANNEL コマンドでチャネルを開始します。

切り替えプロセスは、チャネルがサービスを提供する伝送キューにクラスター送信側チャネルのメッセージがなくなるとすぐに完了します。 この状態になると、それ以後、クラスター送信側チャネルの新着メッセージは新しい伝送キューに直接格納されます。 それまでは、メッセージは古い伝送キューに格納され、切り替えプロセスがメッセージを古い伝送キューから新しい伝送キューに転送します。 クラスター送信側チャネルは、切り替えプロセスの間中、メッセージを新しいクラスター伝送キューから転送します。

切り替えプロセスがいつ完了するかは、システムの状態によって左右されます。 保守時間枠で変更を行う場合は、切り替えプロセスが時間内で完了するかどうかを前もって査定しておいてください。 時間内に完了するかどうかは、古い伝送キューからの転送を待機しているメッセージの数がゼロになるかどうかによって決まります。

1 番目の方法の利点は、自動であることです。 欠点としては、構成変更を行えるのが保守時間枠に限られている場合、保守時間枠内で切り替えプロセスを完了するようにシステムを制御できるという確信がなければならないということです。 その確信がない場合は、オプション 2 をお勧めします。

- オプション 2: 手動で変更する。65 ページの『停止したクラスター送信側チャネルから別のクラスター伝送キューへの切り替え』を参照。

  切り替えプロセス全体を手動で制御する場合や、停止したまたは活動状態にないチャネルを切り替える場合は、このオプションを選択します。 いくつかのクラスター送信側チャネルを切り替える場合、その切り替えを保守時間枠に行うのであれば、このオプションが適しています。

  言い換えれば、このオプションの場合、クラスター送信側チャネルが停止している間に、自分でクラスター送信側チャネルを切り替えます。

  このオプションを選択すると、いつ切り替えるかをすべて自分で決めることができます。

  保守時間枠内の一定の時間内に切り替えプロセスが完了することを確信できます。 切り替えにかかる時間は、一方の伝送キューから他方の伝送キューに転送しなければならないメッセージの数によって左右されます。 継続的にメッセージが着信する場合は、プロセスがすべてのメッセージを転送するのに時間がかかる可能性があります。

  古い伝送キューからメッセージを転送せずにチャネルを切り替えるというオプションもあります。 スイッチは"一瞬"にして終わります。

  クラスター送信側チャネルを再始動すると、新たに割り当てられた伝送キュー上のメッセージの処理が開始します。

  2 番目の方法の利点は、切り替えプロセス全体をコントロールできることです。 欠点は、切り替えるクラスター送信側チャネルを識別し、必要なコマンドを実行し、クラスター送信側チャネルを停止できなくさせかねない未確定チャネルがあればそれを解決しなければならないことです。

関連概念

使用するクラスター伝送キュー・タイプの選択方法
各種のクラスター伝送キュー構成オプションの中から選択する方法。

クラスター送信側チャネルを異なる伝送キューに切り替えるプロセスの仕組み

関連タスク

クラスター化: 複数のクラスター伝送キューの構成例
このタスクでは、複数のクラスター伝送キューの計画手順を、3 つのオーバーラップするクラスターに適用します。 ここで要件となるのは、1 つのクラスター・キューへのメッセージ・フローを、その他すべてのメッセージ・フローから分離すること、そして各クラスターへのメッセージをそれぞれ異なるクラスター伝送キューに保管することです。

アクティブなクラスター送信側チャネルを別のクラスター送信側キューのセットに切り替える
この作業には、アクティブなクラスター送信側チャネルを切り替えるための 3 つのオプションがあります。 1 つのオプションは、キュー・マネージャーが自動で切り替えるようにする方法で、アプリケーションの実行への影響はありません。 その他のオプションは、チャネルを手動で停止および開始する、またはキュー・マネージャーを再始動するという方法です。

始める前に

クラスター伝送キューの構成を変更します。 **DEFCLXQ** キュー・マネージャー属性を変更するか、または伝送キューの **CLCHNAME** 属性を追加または変更することができます。

切り替えプロセスによって新規伝送キューに転送する必要のあるメッセージの数を減らすと、切り替えはより迅速に完了します。 さらに続行する前に、伝送キューを空にすることを試みるべき理由については、クラスター送信側チャネルを異なる伝送キューに切り替えるプロセスの仕組みを参照してください。

このタスクについて

この作業のステップを見本として使用し、クラスター伝送キュー構成変更を実行する独自の計画を立ててください。

手順

1. オプション: 現在のチャネル状況を記録します。

   クラスター伝送キューにサービスを提供している現在のチャネルおよび保存済みのチャネルの状況を記録します。 次のコマンドを実行すると、システム・クラスター伝送キューに関連する状況が表示されます。 自分で定義したクラスター伝送キューに関連する状況を表示するために、独自のコマンドを追加できます。 `XMITQ. ChannelName` などの規則を使用して、定義したクラスター伝送キューに名前を付け、それらの伝送キューのチャネル状況を簡単に表示できるようにします。

   ```
   DISPLAY CHSTATUS(*) WHERE(XMITQ LK 'SYSTEM.CLUSTER.TRANSMIT.*')
   DISPLAY CHSTATUS(*) SAVED WHERE(XMITQ LK 'SYSTEM.CLUSTER.TRANSMIT.*')
   ```

2. 伝送キューを切り替えます。

   - 何もしない。 クラスター送信側チャネルが停止または非アクティブ状態から再始動すると、キュー・マネージャーはクラスター送信側チャネルを切り替えます。

     キュー・マネージャー構成の変更に関して規則や考慮事項がない場合は、このオプションを選択します。 実行中のアプリケーションは、この変更の影響を受けません。
   - キュー・マネージャーを再始動する。 要求に応じて、すべてのクラスター送信側チャネルは自動的に停止して再始動します。

     すべての変更の適用を即時に開始するには、このオプションを選択します。 実行中のアプリケーションは、キュー・マネージャーがシャットダウンして再始動する際に、そのキュー・マネージャーによって中断されます。
   - 個々のクラスター送信側チャネルを停止して再始動します。

     いくつかのチャネルを即時に変更するには、このオプションを選択します。 メッセージ・チャネルを一度停止して再び開始するまでの間に、実行中のアプリケーションにメッセージ転送の短い遅延が生じます。 クラスター送信側チャネルは、停止させた時間を除いて、実行を継続します。 切り替えプロセス中、メッセージは古い伝送キューに送達され、切り替えプロセスによって新しい伝送キューに転送され、クラスター送信側チャネルによって新しい伝送キューから転送されます。

3. オプション: 切り替え時のチャネルをモニターします。

   切り替え中のチャネル状況と伝送キュー・サイズを表示します。 次の例では、システム・クラスター伝送キューの状況が表示されます。

   ```
   DISPLAY CHSTATUS(*) WHERE(XMITQ LK 'SYSTEM.CLUSTER.TRANSMIT.*')
   DISPLAY CHSTATUS(*) SAVED WHERE(XMITQ LK 'SYSTEM.CLUSTER.TRANSMIT.*')
   DISPLAY QUEUE('SYSTEM.CLUSTER.TRANSMIT.*') CURDEPTH
   ```

4. オプション: キュー・マネージャーのエラー・ログに書き込まれるメッセージ「AMQ7341 チャネル *ChannelName* の伝送キューがキュー *QueueName* から *QueueName* に切り替えられました」をモニターします。

停止したクラスター送信側チャネルから別のクラスター伝送キューへの切り替え
手動での変更を選択する場合、クラスター送信側チャネルが停止しているときに、このチャネルに対する変更を行います。また、クラスター伝送キューを別のものに切り替えてから、クラスター送信側チャネルを開始します。

始める前に

何らかの構成変更を行った後に、影響を受けるクラスター送信側チャネルを始動せずに、変更を有効にすることが必要な場合があります。 あるいは、作業のステップの 1 つとして、必要な構成変更を行うこともできます。

切り替えプロセスによって新規伝送キューに転送する必要のあるメッセージの数を減らすと、切り替えはより迅速に完了します。 さらに続行する前に、伝送キューを空にすることを試みるべき理由については、クラスター送信側チャネルを異なる伝送キューに切り替えるプロセスの仕組みを参照してください。

このタスクについて

この作業では、停止または非アクティブのクラスター送信側チャネルによって扱われる伝送キューを切り替えます。 クラスター送信側チャネルが停止しており、その伝送キューを即時に切り替えることが必要な場合に、この作業を行います。 例えば、何らかの理由でクラスター送信側チャネルが開始されていない場合や、構成に関連した他の問題がある場合が該当します。 問題を解決するには、クラスター送信側チャネルを作成し、古いクラスター送信側チャネルの伝送キューを、新しく定義したクラスター送信側チャネルと関連付けるようにします。

さらによくある状況は、クラスター伝送キューの再構成を実行するタイミングを制御することが必要となるケースです。 再構成を完全に制御するには、チャネルを停止し、構成を変更してから、伝送キューを切り替えます。

手順

1. 切り替えるチャネルを停止します。
   a) 切り替える対象で実行中のチャネルや非アクティブなチャネルがあれば、すべて停止します。 非アクティブ・クラスター送信側チャネルを停止すると、構成の変更中にこのチャネルが開始しなくなります。

   ```
   STOP CHANNEL(ChannelName) MODE(QUIESCE) STATUS(STOPPED)
   ```

2. オプション: 構成を変更します。

   例えば、55 ページの『クラスター化: 複数のクラスター伝送キューの構成例』 を参照してください。
3. クラスター送信側チャネルを新しいクラスター伝送キューに切り替えます。

   Multi マルチプラットフォーム で以下のコマンドを出します。

   ```
   runswchl -m QmgrName -c ChannelName
   ```

   z/OS z/OS では、CSQUTIL コマンドの SWITCH 機能を使用して、メッセージの切り替えや発生している状態のモニターを行います。 以下のコマンドを使用します。

   ```
   SWITCH CHANNEL(channel_name) MOVEMSGS(YES)
   ```

   詳しくは、SWITCH 関数を参照してください。

   runswchl または CSQUTIL SWITCH コマンドを実行すると、古い伝送キュー上のすべてのメッセージが新しい伝送キューに転送されます。 このチャネルの古い伝送キュー上にあるメッセージの数がゼロになると、切り替えが完了します。 コマンドは同期的に実行されます。 コマンドは、切り替えプロセス中、進行状況のメッセージをウィンドウに表示します。

   転送フェーズ中は、クラスター送信側チャネルに送信された既存および新規メッセージが順番に新しい伝送キューに転送されます。

   クラスター送信側チャネルは停止しているため、メッセージは新しい伝送キューに蓄積されます。 停止しているクラスター送信側チャネルを、64 ページの『アクティブなクラスター送信側チャネルを別のクラスター送信側キューのセットに切り替える』 のステップ 65 ページの『2』 と対比してください。 そのステップでは、クラスター送信側チャネルが実行中であるため、必ずしもメッセージが新しい伝送キューに蓄積されるとは限りません。

4. オプション: 切り替え時のチャネルをモニターします。

   別のコマンド・ウィンドウに、切り替え中の伝送キュー・サイズを表示します。 次の例では、システム・クラスター伝送キューの状況が表示されます。

   ```
   DISPLAY QUEUE('SYSTEM.CLUSTER.TRANSMIT.*') CURDEPTH
   ```

5. オプション: キュー・マネージャーのエラー・ログに書き込まれるメッセージ「AMQ7341 チャネル *ChannelName* の伝送キューがキュー *QueueName* から *QueueName* に切り替えられました」をモニターします。
6. 停止したクラスター送信側チャネルを再始動します。

   チャネルを停止し、STOPPED 状況にしたため、それらは自動では始動しません。

   ```
   START CHANNEL(ChannelName)
   ```

関連資料

runswchl
RESOLVE CHANNEL
STOP CHANNEL

クラスター化: マイグレーションと変更に関するベスト・プラクティス
このトピックでは、IBM MQ クラスターを計画および管理するための指針について記載しています。 この情報は、テストおよびお客様からのフィードバックに基づく指針を示すものです。

1. 67 ページの『クラスター内でのオブジェクトの移動』 (IBM MQ のフィックスパックや新バージョンはインストールせずに、クラスター内でオブジェクトを移動する場合のベスト・プラクティスです)。
2. 68 ページの『インストールのアップグレードと保守』 (作業中のクラスター・アーキテクチャーを稼働させたまま、保守またはアップグレードの適用および新規アーキテクチャーのテストを行う場合のベスト・プラクティスです)。

クラスター内でのオブジェクトの移動

アプリケーションとそのキュー

あるキュー・マネージャーでホストされているキュー・インスタンスを別のキュー・マネージャーに移動して、そこでホストされるように必要がある場合は、ワークロード・バランシングのパラメーターを操作して、スムーズな移行を行うことができます。

キューのインスタンスを、それが新たにホストされる場所に作成します。ただし、アプリケーションの切り替え準備が完了するまで、クラスター・ワークロード・バランシング設定を使用して、引き続き、元のインスタンスにメッセージが送信されるようにしてください。 そのためには、以下の手順を実行します。

1. 既存のキューの **CLWLRANK** プロパティーを高い値 (例えば、5) に設定します。
2. キューの新規インスタンスを作成し、その **CLWLRANK** プロパティーを 0 に設定します。
3. 新規システムのためのその他の構成作業 (キューの新規インスタンスに対するコンシューム側アプリケーションのデプロイおよび始動など) を完了させます。
4. 新規キュー・インスタンスの **CLWLRANK** プロパティーを元のインスタンスよりも高い値 (例えば、9) に設定します。
5. システムでキューに入っているすべてのメッセージを元のキュー・インスタンスが処理できるようにしてから、そのキューを削除します。

キュー・マネージャー全体の移動

キュー・マネージャーが同じホストに留まっていても、その IP アドレスを変更する場合、プロセスは以下のようになります。

- DNS は、正しく使用されれば、このプロセスを簡単にするのに役立ちます。 接続名 (CONNAME) チャネル属性を設定して DNS を使用する方法について詳しくは、『ALTER CHANNEL』を参照してください。
- 完全リポジトリーを移動する場合は、変更を加える前に、順調に (例えば、チャネル状況に関する問題なく) 稼働している完全リポジトリーが少なくとも 1 つは他にあることを確認します。
- トラフィックが蓄積していくのを避けるために、SUSPEND QMGR コマンドを使用してキュー・マネージャーを中断します。
- コンピューターの IP アドレスを変更します。 CLUSRCVR チャネル定義の CONNAME フィールドで IP アドレスを使用する場合、この IP アドレスの項目を変更します。 更新内容がどこでも有効になるようにするため、DNS キャッシュをフラッシュすることが必要な場合があります。
- キュー・マネージャーが完全リポジトリーに再接続すると、チャネルの自動定義が自動的にそれ自体を解決します。
- キュー・マネージャーで完全リポジトリーをホストしており、IP アドレスを変更する場合は、新しい場所に対して手動で定義されているあらゆる CLUSSDR チャネルをポイントするように、部分リポジトリーが可能な限り速やかに切り替えられるようにすることが重要です。 この切り替えが実施されるまで、それらのキュー・マネージャーは残りの (変更されていない) 完全リポジトリーとは通信できるかもしれませんが、チャネル定義が不正であるという警告メッセージが表示されることがあります。
- RESUME QMGR コマンドを使用してキュー・マネージャーを再開します。

キュー・マネージャーを新しいホストに移動する必要がある場合は、キュー・マネージャー・データをコピーして、バックアップから復元することが可能です。 ただし、他に選択肢がない場合を除き、そのようなプロセスは推奨されません。前のセクションで説明しているように、新しいマシンでキュー・マネージャーを作成し、キューおよびアプリケーションを複製することをお勧めします。 そのようにすることで、円滑なロールオーバーおよびロールバック・メカニズムが実現します。

バックアップを使用してキュー・マネージャー全体を移動する場合は、以下のベスト・プラクティスに従ってください。

- バックアップからキュー・マネージャーを復元するものとしてプロセス全体を処理し、オペレーティング・システム環境に応じてシステム・リカバリーで通常使用するあらゆるプロセスを適用してください。
- マイグレーション後に **REFRESH CLUSTER** コマンドを使用して、ローカルに保持されているクラスター情報 (未確定の自動定義チャネルを含む) をすべて破棄し、その再作成を強制します。

  注 : 大規模クラスターでは、処理中のクラスターに **REFRESH CLUSTER** コマンドを使用すると、破壊的な影響を及ぼす恐れがあります。その後、クラスター・オブジェクトが 27 日間隔で対象のキュー・マネージャーすべてに状況の更新を自動的に送信する際にも同様のことが起こり得ます。 大規模クラスターでのリフレッシュはクラスターのパフォーマンスと可用性に影響を与える可能性があるを参照してください。

キュー・マネージャーを作成し、クラスター内の既存のキュー・マネージャーからセットアップを複製するときには (このトピックの前述の説明を参照)、決して 2 つの異なるキュー・マネージャーを実際には同じものとして扱わないでください。 特に、新しいキュー・マネージャーに同じキュー・マネージャー名と IP アドレスを指定しないようにしてください。 代替キュー・マネージャーに「ドロップイン」しようとすると、IBM MQ クラスターで問題が発生する原因となることがよくあります。 キャッシュは、**QMID** 属性を含む更新情報を受信することを予期しているため、破損状態になる可能性があります。

誤って 2 つの異なるキュー・マネージャーを同じ名前で作成した場合には、RESET CLUSTER **QMID** コマンドを使用して、誤ったエントリーをクラスターから排除することをお勧めします。

インストールのアップグレードと保守

いわゆる「ビッグ・バン・シナリオ」(例えば、クラスターとキュー・マネージャー・アクティビティーをすべて停止し、すべてのアップグレードと保守をすべてのキュー・マネージャーに適用してから、すべてを同時に開始すること) は避けてください。 クラスターは複数のバージョンのキュー・マネージャーが共存

している場合でも動作するよう設計されているので、十分な計画を立てたうえで段階的な保守を行うことをお勧めします。

以下に示しているバックアップ・プランを立ててください。

- バックアップを取りましたか
- すぐに新しいクラスター機能を使用するのは避けてください。すべてのキュー・マネージャーが新しいレベルにアップグレードされたことを確認し、それらをいずれもロールバックしないことを確認するまでは、使用を待ってください。 一部のキュー・マネージャーがまだ初期レベルにあるクラスターで新しいクラスター機能を使用すると、動作が未定義になる可能性があります。

  リポジトリーでは、受信したレコードをそれ自体のバージョンで保管します。 そのリポジトリーが受信したレコードがより新しいバージョンの場合、レコードを保管する際、より新しいバージョンの属性は廃棄されます。 IBM MQ 9.4 キュー・マネージャーに関する情報を受け取る IBM MQ 9.3 キュー・マネージャーは、 IBM MQ 9.3 情報のみを保管します。 IBM MQ 9.3 レコードを受け取る IBM MQ 9.4 リポジトリーには、新しいバージョンで導入された属性のデフォルト値が保管されます。 デフォルト値は、受信するレコードに含まれていない属性の値を定義するものです。

最初に完全リポジトリーをマイグレーションしてください。 完全リポジトリーは、自らが認識できない情報を転送できますが、そのような状態を維持することはできません。したがって、絶対必要な場合を除き、そのようなことはお勧めできません。 詳しくは、キュー・マネージャー・クラスターの移行を参照してください。

クラスター化*: REFRESH CLUSTER* の使用に関するベスト・プラクティス

REFRESH CLUSTER コマンドを使用して、クラスターに関するローカルに保持されているすべての情報を破棄し、クラスターの完全リポジトリーからその情報を再作成します。 例外的な状況を除き、このコマンドを使用する必要はありません。 このコマンドを使用する必要がある場合は、使用方法に関する特別な考慮事項があります。 この情報は、テストおよびお客様からのフィードバックに基づく指針を示すものです。

REFRESH CLUSTER は本当に実行する必要がある場合にのみ実行する

IBM MQ クラスター・テクノロジーを使用することにより、クラスター構成に対するあらゆる変更 (クラスター・キューへの変更など) が、その情報を認識する必要があるクラスターのすべてのメンバーに自動的に認識されるようになります。 このような情報伝達を実現するためにさらに管理手順を実行する必要はありません。

そのような情報を必要とするクラスター内のキュー・マネージャーに情報が届かない場合 (あるクラスター・キューをアプリケーションが初めて開こうとしたときにそのキューがクラスター内の他のキュー・マネージャーに認識されない場合など)、クラスター・インフラストラクチャーに問題があることを意味します。 例えば、キュー・マネージャーと完全リポジトリー・キュー・マネージャーの間のチャネルが開始されないということがあり得ます。 そのため、不整合が認められる場合は調査する必要があります。 可能であれば、**REFRESH CLUSTER** コマンドを使用しないでこの状況を解決してください。

この製品資料の他の場所に記載されているまれな状況で、または IBM サポートから要請された場合に、**REFRESH CLUSTER** コマンドを使用して、クラスターに関するローカルに保持されているすべての情報を破棄し、クラスターの完全リポジトリーからその情報を再作成することができます。

大規模クラスターでのリフレッシュはクラスターのパフォーマンスと可用性に影響を与える可能性がある

REFRESH CLUSTER コマンドを使用すると、処理中のクラスターに悪影響が及ぶ可能性があります。例えば、完全リポジトリーで、キュー・マネージャー・クラスター・リソースを再伝搬している最中に、作業負荷が急激に増加するというようなことが起こります。 大規模なクラスター (数百のキュー・マネージャーが存在するクラスター) をリフレッシュする場合、このコマンドは可能な限り日常の作業では使用せず、別の方法で個々の不整合を修正してください。 例えば、クラスター・キューがクラスター全体で正しく伝搬されていない場合、クラスター・キュー定義の更新 (定義の記述の変更など) を調査するために最初に適用する手法によって、キュー構成がクラスター全体に再伝搬されます。 このプロセスは、問題を特定するのに役立ちます。また、一時的な不整合を解消するのに役立つこともあります。

代替方法を使用できず、**REFRESH CLUSTER** を大規模なクラスターで実行しなければならない場合は、ユーザーのワークロードに影響を与えないようにするため、オフピーク時または保守時間枠の間に行ってください。 また、単一のバッチで大規模なクラスターをリフレッシュするのではなく、70 ページの『クラスター・オブジェクトが自動更新を送信するときにパフォーマンスおよび可用性の問題を回避する』で説明されているように、時間をずらしてアクティビティーを実行してください。

クラスター・オブジェクトが自動更新を送信するときにパフォーマンスおよび可用性の問題を回避する

新規クラスター・オブジェクトがキュー・マネージャーで定義されると、定義された時刻から 27 日ごとにこのオブジェクトの更新が生成され、クラスター内のすべての完全リポジトリーおよびその他関係するキュー・マネージャーに送信されます。 **REFRESH CLUSTER** コマンドをキュー・マネージャーに発行すると、指定したクラスターのローカルで定義されているすべてのオブジェクトでこの自動更新のクロックが再設定されます。

単一のバッチで大規模なクラスター (数百のキュー・マネージャーが存在するクラスター) をリフレッシュする場合、または構成バックアップからシステムを再作成するなどその他の状況では、それらすべてのキュー・マネージャーは、27 日後にすべてのオブジェクト定義を完全リポジトリーに同時に再通知します。 この動作によってもシステムの実行速度が著しく低下する場合があり、すべての更新が完了するまで利用不可になる可能性さえあります。 そのため、大規模なクラスターで複数のキュー・マネージャーをリフレッシュするまたは再作成する必要がある場合は、数時間または数日を空けてこのアクティビティーを実行し、後続の自動更新がシステム・パフォーマンスに定期的に影響を与えることがないようにしてください。

システム・クラスター履歴キュー

REFRESH CLUSTER が実行されると、キュー・マネージャーは、クラスターの状態をリフレッシュして SYSTEM.CLUSTER.HISTORY.QUEUE (SCHQ) に保存する前に、そのスナップショットをとります (キュー・マネージャーで定義されている場合)。 このスナップショットは、後でシステムの問題が発生した場合に備えて、IBM サービスに使用することのみを目的として作成されます。

デフォルトでは SCHQ は、始動時に分散キュー・マネージャーに対して定義されます。 z/OS のマイグレーションの場合は、SCHQ を手動で定義する必要があります。

SCHQ 上のメッセージの有効期限は 3 カ月です。

関連概念

107 ページの『パブリッシュ/サブスクライブ・クラスターの REFRESH CLUSTER についての考慮事項』
REFRESH CLUSTER コマンドを発行すると、キュー・マネージャーで、ローカルに保持されているクラスター情報 (クラスター・トピックおよびそれらが関連付けられているプロキシー・サブスクリプションを含む) が、当面の間、破棄されることになります。

関連資料

REFRESH CLUSTER の実行中に発生するアプリケーションの問題

MQSC コマンドのリファレンス: REFRESH CLUSTER

クラスター化: 可用性、複数インスタンス、および災害復旧

このトピックでは、IBM MQ クラスターを計画および管理するための指針について記載しています。 この情報は、テストおよびお客様からのフィードバックに基づく指針を示すものです。

IBM MQ クラスター化自体は高可用性ソリューションではありませんが、状況によっては、IBM MQ を使用してサービスの可用性を向上させることができます。例えば、異なるキュー・マネージャーに複数のキュー・インスタンスを作成するというような方法をとります。 このセクションでは、IBM MQ インフラストラクチャーをそのようなアーキテクチャーで使用できるように、インフラストラクチャーで最大限の可用性が実現されるようにするための指針を提供します。

注 : IBM MQ では、その他の高可用性ソリューションおよび災害復旧ソリューションを使用できます。 高可用性、リカバリー、および再始動の構成を参照してください。

クラスター・リソースの可用性

2 つの完全リポジトリーを保持することが推奨される一般的な理由は、1 つが失われてもクラスターの円滑な稼働に重大な影響が出ないことです。 両方が使用不可になった場合でも、部分リポジトリーに

既存のナレッジが保持されるため、60 日の猶予期間が与えられます。ただし、このイベントでは、新規リソースまたは以前にアクセスされたリソース (キューなど) は使用できません。

クラスターによるアプリケーションの可用性の向上

クラスターは、キューとアプリケーションの複数のインスタンスを使用することにより、高い可用性を備えたアプリケーション (要求/応答タイプのサーバー・アプリケーションなど) の設計に役立ちます。必要な場合は、例えばキュー・マネージャーまたはチャネルが使用不可になっているというようなことがない限り、優先順位属性により、稼働中のアプリケーションが優先されるようにすることができます。 これは、問題が発生した場合に速やかに切り替えが行われ、引き続き新しいメッセージが処理されるようにするために極めて有効です。

ただし、クラスター内の特定のキュー・マネージャーに配信されたメッセージは、当該キュー・インスタンスでのみ保持されるため、そのキュー・マネージャーが回復するまで処理できません。 このため、本当の意味で高いデータ可用性を実現するには、複数インスタンス・キュー・マネージャーなど、他のテクノロジーの利用を検討することもできます。

複数インスタンス・キュー・マネージャー

ソフトウェアの高可用性 (複数インスタンス) は、既存のメッセージを常に使用可能にしておくための組み込みオファリングです。 詳しくは、『高可用性構成と共に IBM MQ を使用する』、『複数インスタンス・キュー・マネージャーの作成』、および次のセクションを参照してください。 この手法を利用してクラスター内の任意のキュー・マネージャーで高い可用性を実現することができます。ただし、クラスター内のすべてのキュー・マネージャーが、IBM WebSphere MQ 7.0.1 以降を使用して実行されていることが前提になります。 クラスター内のいずれかのキュー・マネージャーが以前のレベルである場合、そのキュー・マネージャーは、セカンダリー IP へのフェイルオーバーを行うと、複数インスタンス・キュー・マネージャーへの接続を失う可能性があります。

このトピックで既に説明したように、2 つの完全リポジトリーを構成している限り、ほとんどの場合は、本来、可用性が高くなります。 必要に応じて、IBM MQ ソフトウェア高可用性/複数インスタンス・キュー・マネージャーを完全リポジトリーに使用することができます。 これらの方法を使用することを強く勧める根拠はありません。むしろ、一時的な障害の場合、これらの方法を使用すると、フェイルオーバーの際に余分にパフォーマンス・コストが発生することもあります。 2 つの完全リポジトリーを稼働させる代わりにソフトウェア高可用性を使用することはお勧めしません。なぜならば、例えば、チャネルが 1 つだけ停止した場合に、フェイルオーバーは必要ないかもしれませんが、部分リポジトリーがクラスター・リソースを照会できないままになる可能性があります。

災害時リカバリー

災害復旧 (キュー・マネージャーのデータが保管されているディスクが破損した場合の復旧など) にきちんと対処するのは難しいものです。IBM MQ は役立ちはしますが、自動的に災害復旧できるわけではありません。 IBM MQ (オペレーティング・システムおよび他の基本的複製テクノロジーは除く) における「真の」災害復旧オプションは、バックアップからの回復だけです。 そのような状態におけるクラスター固有の考慮事項として、以下のようなものがあります。

- 災害復旧シナリオのテスト時には注意してください。 例えば、バックアップ・キュー・マネージャーの動作をテストする場合に、同じネットワーク内でそれらのキュー・マネージャーをオンラインにするときには注意が必要です。稼働中のクラスター・キュー・マネージャーのキューと同じ名前が付けられているキューがホストされることにより、誤ってバックアップ・キュー・マネージャーが稼働中のクラスターに加わり、メッセージを「盗み」始める可能性があります。
- 災害復旧のテストは、稼働中のクラスターに干渉してはなりません。 干渉を避けるための手法を以下に示します。
  - 完全なネットワーク分離、またはファイアウォール・レベルでのネットワーク分離。
  - z/OS チャネルの初期設定、または z/OS の **chinit** アドレス・スペースを開始しない。
  - 実際の災害復旧シナリオが実施されるまで、または実施されない限り、災害復旧システムには稼働中の TLS 証明書を発行しない。
- クラスター内のキュー・マネージャーのバックアップをリストアする場合、バックアップがそのクラスターの残りの部分と同期していない可能性もあります。 **REFRESH CLUSTER** コマンドを使用すると、更新を解決してクラスターと同期することができます。ただし、 **REFRESH CLUSTER** コマンドは最後の手段として使用してください。 69 ページの『クラスター化: REFRESH CLUSTER の使用に関するベスト・プラクティス』を参照してください。 組織内の手続き関連の資料および IBM MQ の資料

を読み返して、最後の手段としてこのコマンドを使用する前に、簡単な手段を見過ごしていないかどうか確認してください。

- どのようなリカバリーにおいても、アプリケーションはデータの再生と損失に対処する必要があります。 キューをクリアして既知の状態にするかどうか、あるいは再生を管理するのに十分な情報が他の場所にあるかどうかを判断する必要があります。

分散パブリッシュ/サブスクライブ・ネットワークの計画

1 つのキュー・マネージャー上で作成されるサブスクリプションが、ネットワーク内の別のキュー・マネージャーに接続されたアプリケーションによって公開されるメッセージのうち一致するものを受け取るような、キュー・マネージャーのネットワークを作成することができます。 適切なトポロジーを選択するには、手動制御の要件、ネットワークのサイズ、変更の頻度、アベイラビリティー、およびスケーラビリティーについて考慮する必要があります。

始める前に

このタスクでは、担当者が分散パブリッシュ/サブスクライブ・ネットワークとその動作についてよく理解していることが前提とされています。 技術概要について詳しくは、『分散パブリッシュ/サブスクライブのネットワーク』を参照してください。

このタスクについて

パブリッシュ/サブスクライブ・ネットワークの基本的なトポロジーとしては、以下の 3 つがあります。

- 直接ルーティング型クラスター
- トピック・ホスト・ルーティング型クラスター
- 階層

最初の 2 つのトポロジーの場合、開始点は IBM MQ クラスター構成です。 3 番目のトポロジーは、クラスターを使用する場合と使用しない場合のいずれも作成できます。 背景となるキュー・マネージャー・ネットワークの計画については、20 ページの『分散キューおよびクラスターの計画』を参照してください。

直接ルーティング型クラスター は、クラスターが既に存在する場合に構成するトポロジーとして最もシンプルなものです。 どのキュー・マネージャー上で定義するトピックも、クラスター内のあらゆるキュー・マネージャー上で自動的に使用可能となります。 また、パブリケーションは、パブリッシュ側アプリケーションの接続されている任意のキュー・マネージャーから、一致するサブスクリプションの存在するキュー・マネージャーのそれぞれに直接ルーティングされます。 構成がこのようにシンプルなのは、IBM MQ が、クラスター内のキュー・マネージャー間で情報と接続の共有を高水準で維持していることによります。 小規模でシンプルなネットワーク (キュー・マネージャーの数が少なく、またパブリッシャーとサブスクライバーの集合が固定している) では、これで十分です。 しかし、より大規模な、または動的に変化する環境で使用した場合は、オーバーヘッドの点で問題が発生する可能性があります。 『78 ページの『パブリッシュ/サブスクライブ・クラスターでの直接ルーティング』』を参照してください。

トピック・ホスト・ルーティング型クラスター には直接ルーティング型クラスターの場合と同じメリットがあり、クラスター内のどのキュー・マネージャーで定義するトピックも、クラスター内のあらゆるキュー・マネージャー上で自動的に使用可能になります。 しかし、トピック・ホスト・ルーティング型クラスターの場合、各トピックのホストとなるキュー・マネージャーを注意深く選択する必要があります。 そのトピックのための情報とパブリケーションのすべてが、それらのトピック・ホスト・キュー・マネージャーを通過することになるからです。 つまり、システムは、キュー・マネージャーすべての間でのチャネルと情報フローを保守する必要がありません。 しかし、そのことはまた、パブリケーションがサブスクライバーに直接送信されなくなり、トピック・ホスト・キュー・マネージャー経由でルーティングされることになる可能性があることも意味しています。 そのような理由で、特に、トピックのホスティング担当のキュー・マネージャーでは、システムに追加の負荷がかかる可能性があります。 トポロジーは、十分に注意深く計画する必要があります。 このトポロジーは、含まれるキュー・マネージャーの数の多いネットワークにおいて、あるいはパブリッシャーとサブスクライバーの動的セットをホストする (つまり、パブリッシャーやサブスクライバーの追加と削除が頻繁になされる) ネットワークにおいて、特に有効です。 経路のアベイラビリティーを向上させ、パブリケーションのワークロードを水平方向にスケーリングするため、追加のトピック・ホストを定義することができます。 『82 ページの『パブリッシュ/サブスクライブ・クラスターでのトピック・ホスト・ルーティング』』を参照してください。

階層 は、構成の手動セットアップを必要とする度合いが最も高く、トポロジーの変更が最も困難です。 階層内の各キュー・マネージャーの間の関連と、その直接の関係を手動で構成する必要があります。 関係を構成した後、(前の 2 つのトポロジーの場合) パブリケーションは、階層内の他のキュー・マネージャー上のサブスクリプションにルーティングされます。 パブリケーションは、階層関係を使用してルーティングされます。 これにより、特定のトポロジーをさまざまな要件に合わせて構成することができますが、これにより、サブスクリプションに到達するために、中間キュー・マネージャーを介して多くの"ホップ"を必要とするパブリケーションも生成されます。 階層を経由するパブリケーションの経路は常に 1 つのみであるため、それぞれのキュー・マネージャーのアベイラビリティーが重要になります。 多くの場合、階層を使用することが望ましいのは、単一クラスターを構成することが不可能な場合のみです。 例えば、複数の組織にまたがる場合などです。 『108 ページの『パブリッシュ/サブスクライブ階層でのルーティング』』を参照してください。

必要な場合には、上記の 3 種類のトポロジーを組み合わせることにより、特定の地理的要件に対処することができます。 その例については、複数のクラスターのトピック・スペースの結合を参照してください。

実際の分散パブリッシュ/サブスクライブ・ネットワークのために適切なトポロジーを選択するには、以下のさまざまな要素を考慮する必要があります。

- ネットワークの規模はどの程度の大きさか?
- 構成に対してどの程度の手動制御を必要とするか?
- トピックとサブスクリプションの点で、またキュー・マネージャーに関して、システムはどの程度動的に変化するか?
- アベイラビリティーとスケーラビリティーの要件は何か?
- すべてのキュー・マネージャーを相互に直接接続することは可能か?

手順

- ネットワークの規模としてどの程度の大きさが必要かを見積もります。
  a) 必要なトピックの数を見積もります。
  b) 予期されるパブリッシャーとサブスクライバーの数を見積もります。
  c) パブリッシュ/サブスクライブ・アクティビティーに関与するキュー・マネージャーの数を見積もります。

  92 ページの『パブリッシュ/サブスクライブのクラスター化: ベスト・プラクティス』、特に以下のセクションも参照してください。

  – システムのサイズを決める方法
  – パブリッシュ/サブスクライブ・アクティビティーに参加するクラスター・キュー・マネージャーの数を制限するべき理由
  – どのトピックをクラスター化するかを決める方法

  ネットワーク内のキュー・マネージャーの数が多く、処理するパブリッシャーとサブスクライバーの数も多い場合は、トピック・ホスト・ルーティング型クラスターまたは階層を使用することが必要になることでしょう。 直接ルーティング型クラスターは、手動構成がほとんど不要であり、小規模なネットワーク、または変化のない固定したネットワークに適したソリューションとなり得ます。
- トピック、パブリッシャー、またはサブスクライバーのそれぞれをどのキュー・マネージャーがホストするのかについて、手動制御がどの程度必要かを考慮します。
  a) キュー・マネージャーのうちのあるものが他のものより処理能力が低いかどうかを考慮します。
  b) キュー・マネージャーのうちのいくつかへの通信リンクが他のものよりも脆弱かどうかを考慮します。
  c) トピックのパブリケーションの数が多く、サブスクライバーの数が少ないのがどんな場合かを特定します。
  d) トピックのサブスクライバーの数が多く、パブリケーションの数が少ないのはどんな場合かを特定します。

すべてのトポロジーにおいて、パブリケーションは、他のキュー・マネージャー上のサブスクリプションに配信されます。 直接ルーティング型クラスターの場合、それらのパブリケーションは、サブスクリプションに至る最短パスを経由します。 トピック・ホスト・ルーティング型クラスターまたは階層の場合、パブリケーションの経路を自分で制御することになります。 キュー・マネージャーごとに処理能力が異なる場合、またはアベイラビリティーとコネクティビティーの点でレベルが異なる場合、特定のワークロードを特定のキュー・マネージャーに割り当てるようにするとよいでしょう。 それは、トピック・ホスト・ルーティング型クラスターまたは階層を使用することにより実現できます。

すべてのトポロジーにおいて、パブリッシュ側アプリケーションを可能な限りサブスクリプションと同じキュー・マネージャー上に配置するなら、オーバーヘッドを最小限に抑え、パフォーマンスを最大にすることができます。 トピック・ホスト・ルーティング型クラスターの場合は、トピックをホストするキュー・マネージャー上にパブリッシャーまたはサブスクライバーを置くことを考慮してください。 これにより、パブリケーションをサブスクライバーに渡す際のキュー・マネージャー間で余分な"ホップ"が除去されます。 このアプローチは、トピックのパブリッシャーの数が多くてサブスクライバーの数が少ない場合、またはサブスクライバーの数が多くてパブリッシャーの数が少ない場合に特に有効です。 例えば、集中型パブリッシャーまたはサブスクライバーを使用したトピック・ホスト・ルーティングを参照してください。

92 ページの『パブリッシュ/サブスクライブのクラスター化: ベスト・プラクティス』、特に以下のセクションも参照してください。

- どのトピックをクラスター化するかを決める方法
- パブリッシャーおよびサブスクリプションの場所

- ネットワーク・アクティビティーがどの程度動的に変化するかを考慮します。

  a) さまざまな異なるトピックについて、サブスクライバーがどれほど頻繁に追加されたり削除されたりするかを見積もります。

  キュー・マネージャーのサブスクリプションが追加されたり削除されたりする場合、それがその特定のトピック・ストリングの最初または最後のサブスクリプションであるなら、その情報はトポロジー内の他のキュー・マネージャーに伝達されます。 直接ルーティング型クラスターおよび階層の場合、そのサブスクリプション情報は、トピックのパブリッシャーが属するものも属していないものも含め、トポロジー内のあらゆるキュー・マネージャーに伝搬されます。 トポロジーが多数のキュー・マネージャーで構成されている場合、これによりパフォーマンス上の大きなオーバーヘッドが発生する可能性があります。 トピック・ホスト・ルーティング型クラスターの場合、この情報は、サブスクリプションのトピック・ストリングにマップされるクラスター化トピックをホストするキュー・マネージャーにのみ伝搬されます。

  92 ページの『パブリッシュ/サブスクライブのクラスター化: ベスト・プラクティス』の中のサブスクリプションの変更と動的トピック・ストリングのセクションも参照してください。

  注 : 非常に動的なシステムでは、多数の固有のトピック・ストリングのセットが常に急速に変化しているため、モデルを "「どこでもパブリッシュ」" モードに切り替えるのが最善の方法である可能性があります。 パブリッシュ/サブスクライブ・ネットワークでのサブスクリプションのパフォーマンスを参照してください。

  b) トポロジー内のキュー・マネージャーがどの程度動的に変化するかを考慮します。

  階層では、トポロジー内のキュー・マネージャーに変更があるたびに、手動で階層に挿入したり階層から削除したりすることが必要になるため、階層内の上位でキュー・マネージャーに変更がある場合には、十分な注意が必要です。 階層内のキュー・マネージャーでは、手動構成によるチャネル接続も使用されていることが少なくありません。 階層でキュー・マネージャーを追加/削除するたびに、チャネルを追加/削除することによってこれらの接続を保守する必要があります。

  パブリッシュ/サブスクライブ・クラスターの場合、キュー・マネージャーは、クラスターに参加することが初めて必要になった時点で他のキュー・マネージャーに自動的に接続され、トピックとサブスクリプションを自動的に認識するようになります。

- 経路のアベイラビリティーとパブリケーション・トラフィックのスケーラビリティーに関する要件を考慮します。

  a) あるキュー・マネージャーが使用不可の場合であっても、パブリッシュ側キュー・マネージャーからサブスクライブ側キュー・マネージャーに至る利用可能な経路が常に存在することが必要かどうかを判断します。

b) ネットワークがどの程度スケーラブルであることが必要かを考慮します。 パブリケーション・トラフィックのレベルが単一キュー・マネージャーまたはチャネル経由でルーティングするには高すぎるかどうかを判別します。 また、そのレベルのパブリケーション・トラフィックを単一トピック・ブランチで処理しなければならないのか、それとも複数のトピック・ブランチに分散させることが可能かどうかを判断します。

c) メッセージの順序を維持することが必要かどうかを考慮します。

直接ルーティング型クラスターでは、パブリッシュ側キュー・マネージャーからのメッセージをサブスクライブ側キュー・マネージャーに直接送信するため、経路沿いにある中間キュー・マネージャーのアベイラビリティーについて心配する必要はありません。 同じように、中間キュー・マネージャーへのスケーリングも考慮する必要はありません。 しかし、前述のように、特に規模の大きい、または動的に変化する環境の場合、クラスター内のすべてのキュー・マネージャーの間でのチャネルと情報フローの自動保守のためのオーバーヘッドは、パフォーマンスに大きく影響する可能性があります。

トピック・ホスト・ルーティング型クラスターは、個々のトピックに合わせて調整することが可能です。 パブリケーション・ワークロードがかなり大きいトピック・ツリーの各ブランチがそれぞれ異なるキュー・マネージャー上に定義されていること、また、各キュー・マネージャーに、トピック・ツリーのそのブランチで予期されるワークロードに対して十分の処理能力と可用性があることを確認してください。 また、各トピックを複数のキュー・マネージャーで定義することにより、アベイラビリティーと水平スケーリングをさらに向上させることができます。 これにより、使用不可状態のトピック・ホスト・キュー・マネージャーを回避したルーティングと、その間でのパブリケーション・トラフィックのワークロード・バランシングが、システムにおいて可能になります。 しかし、特定のトピックを複数のキュー・マネージャーで定義する場合、以下の制約も発生することになります。

- パブリケーション間のメッセージ順序付けは失われます。
- 保存パブリケーションは使用できません。 『106 ページの『パブリッシュ/サブスクライブ・クラスターでの保存パブリケーションに関する設計上の考慮事項』』を参照してください。

多重経路による階層においてルーティングの高可用性とスケーラビリティーはいずれも構成できません。

92 ページの『パブリッシュ/サブスクライブのクラスター化: ベスト・プラクティス』のパブリケーション・トラフィックも参照してください。

- これらの見積もりに基づき、以下のリンクを使用して、トピック・ホスト・ルーティング型クラスター、直接ルーティング型クラスター、階層のうちどれを使用するか、それともそれらのトポロジーの混合を使用するのかを決定してください。

次のタスク

これで、分散パブリッシュ/サブスクライブ・ネットワークの構成作業の準備ができました。

関連タスク

キュー・マネージャー・クラスターの構成

分散キューイングの構成

パブリッシュ/サブスクライブ・クラスターの構成

パブリッシュ/サブスクライブ階層へのキュー・マネージャーの接続

パブリッシュ/サブスクライブ・クラスターの設計

パブリッシュ/サブスクライブの基本的なクラスター・トポロジーには、直接ルーティング とトピック・ホスト・ルーティング の 2 つがあります。 それぞれ、異なる利点を持っています。 パブリッシュ/サブスクライブ・クラスターを設計するときには、想定されるネットワーク要件に最適なトポロジーを選択してください。

2 つのパブリッシュ/サブスクライブ・クラスター・トポロジーの概要について詳しくは、『パブリッシュ/サブスクライブ・クラスター』を参照してください。 ネットワーク要件を評価するための参考として、72 ページの『分散パブリッシュ/サブスクライブ・ネットワークの計画』および 92 ページの『パブリッシュ/サブスクライブのクラスター化: ベスト・プラクティス』を参照してください。

一般に、この 2 つのクラスター・トポロジーはどちらも、以下のような利点を提供します。

- Point-to-Point クラスター・トポロジーを基礎としたシンプルな構成。

- クラスターへのキュー・マネージャーの参加と退去を自動的に処理。
- サブスクリプションとパブリッシャーを追加する場合のスケーリングが容易。キュー・マネージャーを追加して、追加のサブスクリプションとパブリッシャーをそれらに分散させることができます。

しかし、要件がさらに具体的になると、2 つのトポロジーに異なる利点があることがわかります。

直接ルーティング型パブリッシュ/サブスクライブ・クラスター

直接ルーティングの場合、クラスター内のすべてのキュー・マネージャーは、サブスクリプションが一致するクラスター内の他のすべてのキュー・マネージャーに対して、接続されたアプリケーションからパブリケーションを直接送信します。

直接ルーティング型パブリッシュ/サブスクライブ・クラスターには、以下の利点があります。

- 同一クラスター内の特定のキュー・マネージャー上のサブスクリプション宛のメッセージは、そのキュー・マネージャーに直接移送されるため、中間キュー・マネージャーを経由して移動する必要がありません。 これにより、トピック・ホスト・ルーティング型トポロジーや階層型トポロジーと比べて、パフォーマンスが向上することがあります。
- すべてのキュー・マネージャーが相互に直接接続しているため、このトポロジーのルーティング・インフラストラクチャーには単一障害点が存在しません。 あるキュー・マネージャーが使用不可になっても、クラスター内の他のキュー・マネージャー上のサブスクリプションは、使用可能なキュー・マネージャー上のパブリッシャーからメッセージを引き続き受け取ることができます。
- 特に既存のクラスター上に構築する場合、ごく容易に構成できます。

直接ルーティング型パブリッシュ/サブスクライブ・クラスターを使用する場合の考慮事項として、以下の点が挙げられます。

- クラスター内のすべてのキュー・マネージャーが、クラスター内の他のすべてのキュー・マネージャーを認識するようになります。
- クラスター・トピックに対する 1 つ以上のサブスクリプションをホストするクラスター内のキュー・マネージャーは、どのクラスター・トピックに対してもメッセージをパブリッシュしないキュー・マネージャーを含め、クラスター内の他のすべてのキュー・マネージャーに対してクラスター送信側チャネルを自動的に作成します。
- クラスター・トピック下のトピック・ストリングに対するキュー・マネージャー上の最初のサブスクリプションでは、クラスター内の他のすべてのキュー・マネージャーにメッセージが送信されます。 同様に、削除対象のトピック・ストリングに対する最後のサブスクリプションでも、メッセージが送信されます。クラスター・トピック下で使用される個々のトピック・ストリングが増加し、サブスクリプションの変更率が高くなると、キュー・マネージャー間の通信も増加します。
- クラスター内のすべてのキュー・マネージャーは、サブスクライブされたトピック・ストリング情報が通知されると、そのキュー・マネージャーがそれらのトピックのパブリッシュにもサブスクライブにも関与していない場合でも、通知された情報を保持します。

上記の理由から、クラスター内で直接ルーティング型のトピックを定義されたすべてのキュー・マネージャーには、付加的なオーバーヘッドが掛かります。 クラスター内のキュー・マネージャーが多くなるほど、そのオーバーヘッドは大きくなります。 同様に、サブスクライブされるトピック・ストリングが多くなり、その変更率が高くなるほど、オーバーヘッドが大きくなります。 その結果、大規模な、または動的な直接ルーティング型のパブリッシュ/サブスクライブ・クラスター内の小規模なシステムで実行されているキュー・マネージャーに掛かる負荷が大きくなりすぎることがあります。 詳細については、直接ルーティング型パブリッシュ/サブスクライブ・クラスターのパフォーマンスを参照してください。

直接ルーティング型クラスター・パブリッシュ/サブスクライブのオーバーヘッドにクラスターが対応できないことが分かっている場合は、代わりにトピック・ホスト・ルーティング型パブリッシュ/サブスクライブを使用できます。 あるいは、極端な状況では、クラスター内のすべてのキュー・マネージャーでキュー・マネージャー属性 **PSCLUS** を DISABLED に設定することにより、クラスター・パブリッシュ/サブスクライブ機能を完全に無効にすることもできます。 103 ページの『クラスター化されたパブリッシュ/サブスクライブの禁止』を参照してください。 こうすると、クラスター・トピックが作成されなくなるため、クラスター・パブリッシュ/サブスクライブに関連したオーバーヘッドがネットワークに掛からなくなります。

トピック・ホスト・ルーティング型パブリッシュ/サブスクライブ・クラスター

トピック・ホスト・ルーティングでは、クラスター・トピックが管理的に定義されたキュー・マネージャーが、パブリケーションのルーターになります。 クラスター内の非ホスティング・キュー・マネージャーからのパブリケーションは、ホスティング・キュー・マネージャーを経由して、クラスター内でサブスクリプションが一致するキュー・マネージャーにルーティングされます。

トピック・ホスト・ルーティング型パブリッシュ/サブスクライブ・クラスターには、直接ルーティング型パブリッシュ/サブスクライブ・クラスターに比べて、以下の利点があります。

- トピック・ホスト・ルーティング型のトピックが定義されたキュー・マネージャーのみが、クラスター内の他のすべてのキュー・マネージャーを認識します。
- トピック・ホスト・キュー・マネージャーだけがクラスター内の他のすべてのキュー・マネージャーに接続できればよく、通常はサブスクリプションが存在するキュー・マネージャーにのみ接続することになります。 したがって、キュー・マネージャー間で実行されるチャネルは大幅に少なくなります。
- クラスター・トピックに対する 1 つ以上のサブスクリプションをホストするクラスター・キュー・マネージャーは、サブスクリプションのトピック・ストリングにマップするクラスター・トピックをホストするキュー・マネージャーに対してのみクラスター送信側チャネルを自動的に作成します。
- クラスター・トピック下のトピック・ストリングに対してキュー・マネージャーに最初のサブスクリプションが行われると、クラスター内のそのクラスター・トピックをホストしているキュー・マネージャーにメッセージが送信されます。 同様に、削除対象のトピック・ストリングに対する最後のサブスクリプションでも、メッセージが送信されます。 クラスター・トピック下で使用される個々のトピック・ストリングが増加し、サブスクリプションの変更率が高くなると、キュー・マネージャー間の通信も増加しますが、それはサブスクリプション・ホストとトピック・ホストの間のみです。
- 物理構成を制御できる程度が大きくなります。 直接ルーティングでは、すべてのキュー・マネージャーがパブリッシュ/サブスクライブ・クラスターに参加する必要があるため、オーバーヘッドが大きくなります。 トピック・ホスト・ルーティングでは、トピック・ホスト・キュー・マネージャーだけが他のキュー・マネージャーとそのサブスクリプションを認識します。 トピック・ホスト・キュー・マネージャーを明示的に選択して、それらのキュー・マネージャーが適切な性能の機器で実行され、他のキュー・マネージャーには性能の低いシステムを使用することができます。

トピック・ホスト・ルーティング型パブリッシュ/サブスクライブ・クラスターを使用する場合の考慮事項として、以下の点が挙げられます。

- パブリッシャーまたはサブスクライバーがトピック・ホスティング・キュー・マネージャー上にない場合は、パブリッシュ側のキュー・マネージャーとサブスクライブ側のキュー・マネージャーの間に余分な"ホップ"が発生します。 余分な"ホップ"によって生じる待ち時間のために、そのトピック・ホスト・ルーティングが直接ルーティングよりも効率が低くなることがあります。
- 大規模なクラスターでは、トピック・ホスト・ルーティングを使用することにより、直接ルーティングで発生する可能性のある重大なパフォーマンスおよびスケーリングの問題が緩和されます。
- すべてのトピックを単一のキュー・マネージャーに定義するか、ごく少数のキュー・マネージャーに定義するかを選択できます。 この選択を行うときには、接続性のよい強力なシステムでトピック・ホスト・キュー・マネージャーがホストされるようにしてください。
- 複数のキュー・マネージャーに同じトピックを定義することもできます。 これによってトピックの可用性が向上するほか、スケーラビリティーも向上します。IBM MQ は、トピックのパブリケーションのワークロード・バランシングをそのトピックのすべてのホスト間で行うからです。 ただし、複数のキュー・マネージャーに同じトピックを定義すると、そのトピックのメッセージの順序が失われることに注意してください。
- トピックごとに異なるキュー・マネージャーでホストすることにより、メッセージの順序を失わずにスケーラビリティーを向上させることができます。

関連タスク

シナリオ: パブリッシュ/サブスクライブ・クラスターの作成

パブリッシュ/サブスクライブ・クラスターの構成

分散パブリッシュ/サブスクライブ・ネットワークのチューニング

分散パブリッシュ/サブスクライブの問題のトラブルシューティング

パブリッシュ/サブスクライブ・クラスターでの直接ルーティング

パブリッシュ側キュー・マネージャーからのパブリケーションは、クラスター内の他のキュー・マネージャーのうち一致するサブスクリプションのあるものに、直接ルーティングされます。

パブリッシュ/サブスクライブ階層およびクラスター内のキュー・マネージャー間でメッセージがルーティングされる方法については、『分散パブリッシュ/サブスクライブのネットワーク』を参照してください。

直接ルーティング型パブリッシュ/サブスクライブ・クラスターの動作は、以下のとおりです。

- すべてのキュー・マネージャーは、他のすべてのキュー・マネージャーを自動認識します。
- クラスター・トピックに対するサブスクリプションがあるすべてのキュー・マネージャーは、クラスター内の他のすべてのキュー・マネージャーに対してチャネルを作成し、それぞれのサブスクリプションについて通知します。
- アプリケーションのパブリッシュするメッセージは、それが接続されているキュー・マネージャーから、一致するサブスクリプションの存在する各キュー・マネージャーに直接ルーティングされます。

次の図は、パブリッシュ/サブスクライブ・アクティビティーまたは Point-to-Point アクティビティーに現在使用されていないキュー・マネージャー・クラスターを示しています。 クラスター内のどのキュー・マネージャーも、フル・リポジトリー・キュー・マネージャーとのみ接続していることに注意してください。

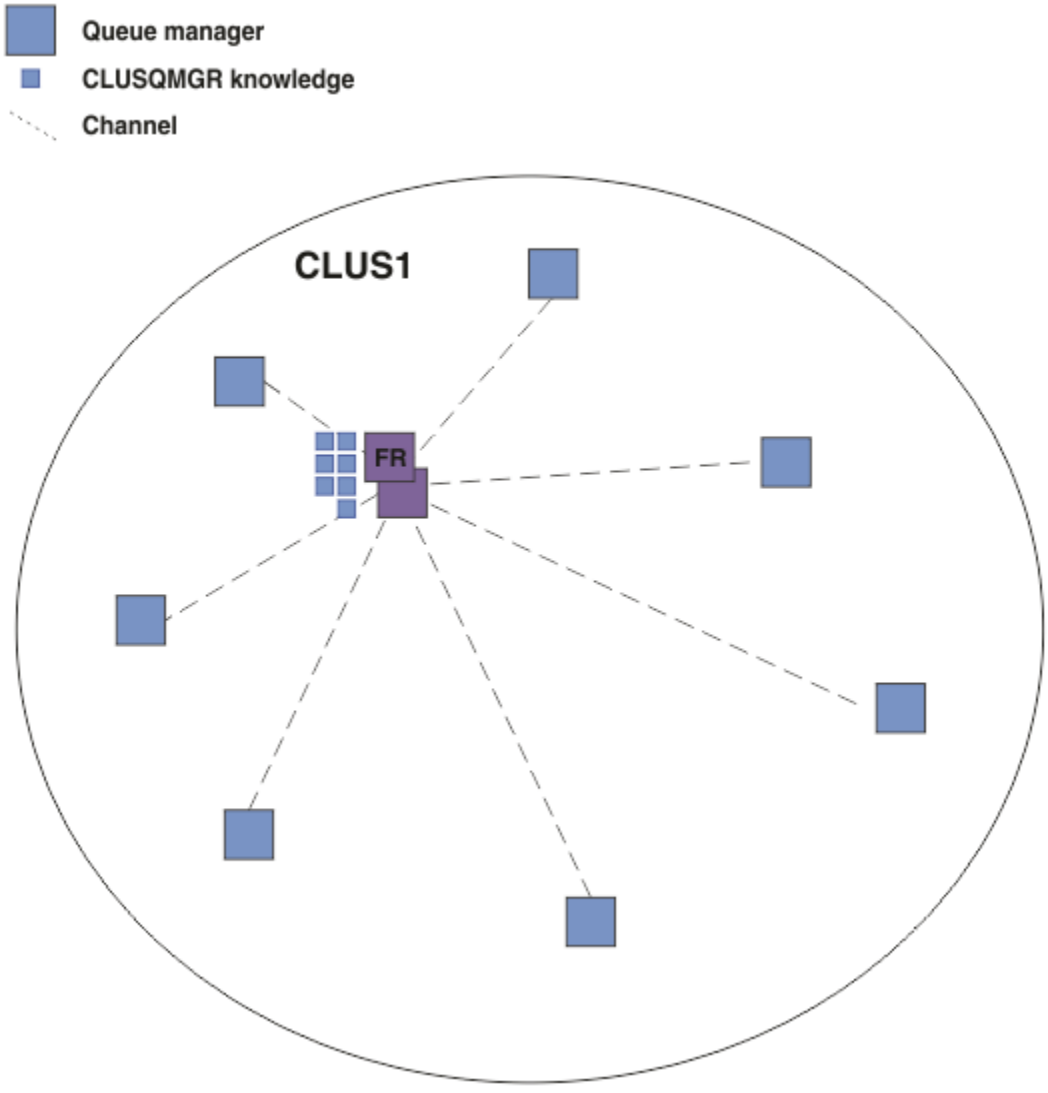


図 *16.* キュー・マネージャー・クラスター

直接ルーティング型クラスター内のキュー・マネージャーの間をパブリケーションが流れるためには、パブリッシュ/サブスクライブ・クラスターの構成で説明されているようにしてトピック・ツリーのブランチをクラスター化し、直接ルーティング を指定します (デフォルト)。

直接ルーティング型パブリッシュ/サブスクライブ・クラスターでは、クラスター内のどのキュー・マネージャーにおいても、トピック・オブジェクトを定義します。 それをした場合、オブジェクトの情報、およびクラスター中のその他のすべてのキュー・マネージャーの情報が、フル・リポジトリー・キュー・マネージャーによって自動的にクラスター内のすべてのキュー・マネージャーに送られます。 これは、キュー・マネージャーがトピックを参照する前になされます。

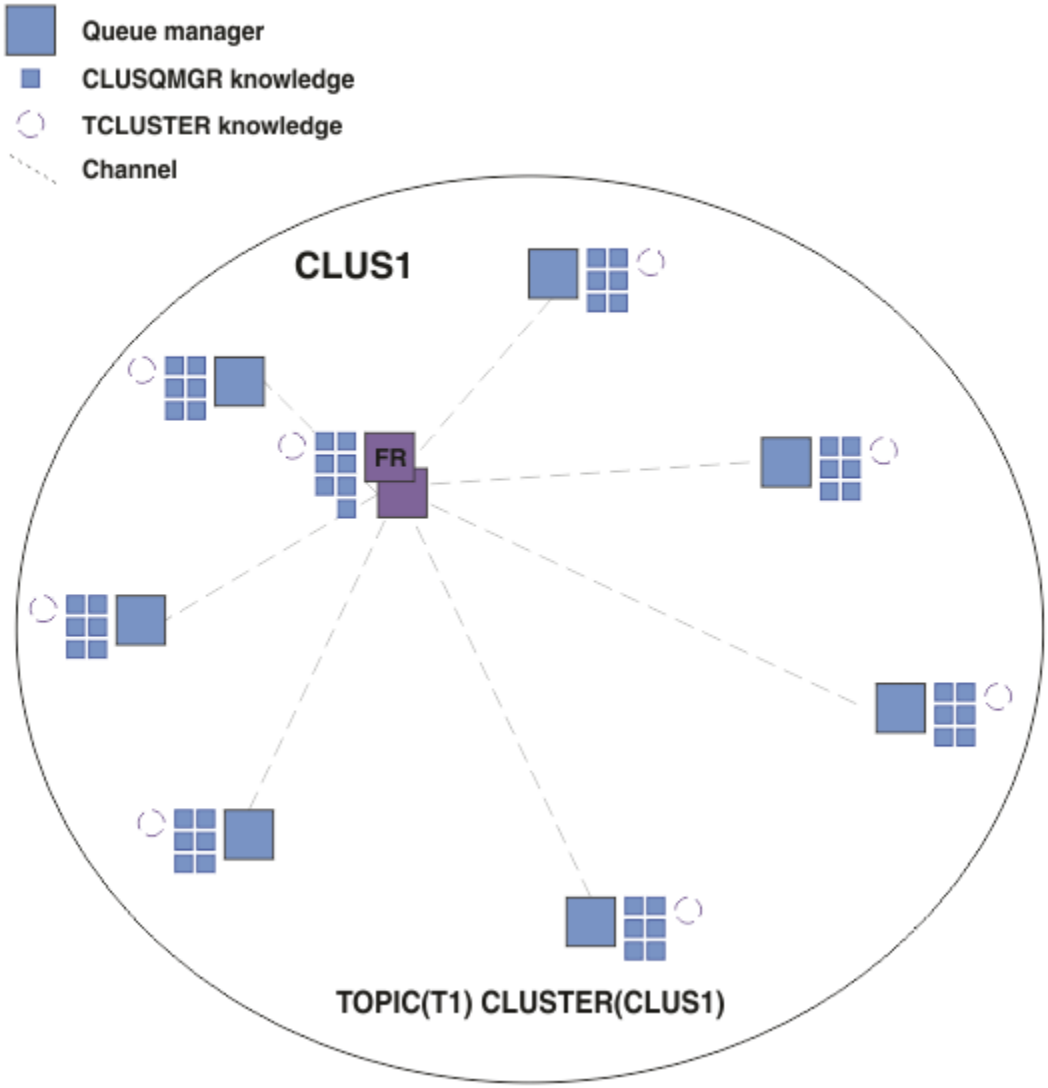


図 17. 直接ルーティング型パブリッシュ/サブスクライブ・クラスター

サブスクリプションが作成されると、そのサブスクリプションをホストするキュー・マネージャーは、クラスター内のあらゆるキュー・マネージャーへのチャネルを確立し、そのサブスクリプションに関する詳細を送信します。 この分散サブスクリプションのナレッジは、各キュー・マネージャー上のプロキシー・サブスクリプションによって表されます。 そのプロキシー・サブスクリプションのトピック・ストリングと一致するクラスター内のいずれかのキュー・マネージャーでパブリケーションが作成されると、パブリッシャー・キュー・マネージャーからサブスクリプションをホストする各キュー・マネージャーへのクラスター・チャネルが確立され、それぞれにメッセージが送信されます。

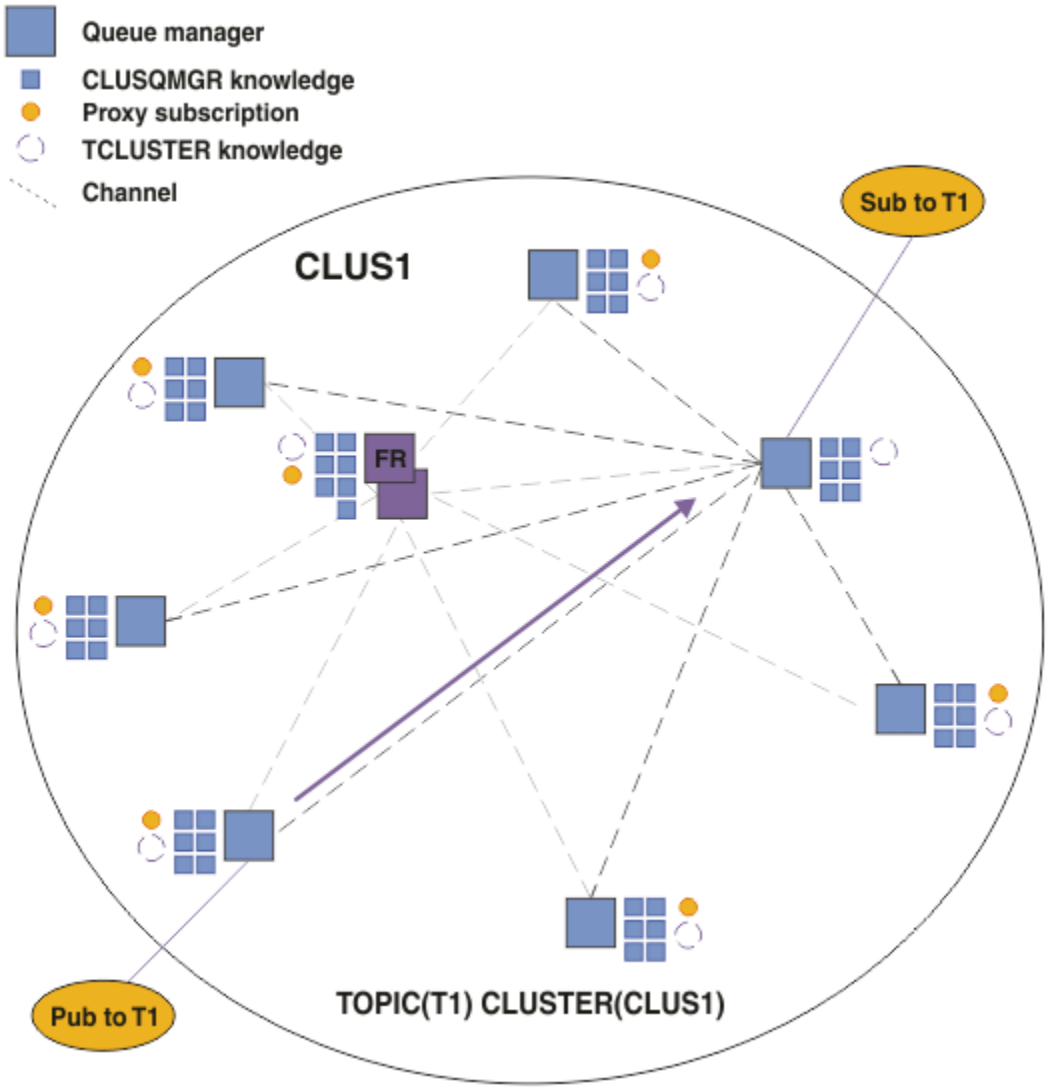


図 *18.* クラスター化トピックに対する *1* つのパブリッシャーと *1* つのサブスクライバーによる直接ルーティング型パブリッシュ/サブスクライブ・クラスター

パブリケーションを、サブスクリプションのホスティング・キュー・マネージャーに直接ルーティングすることにより、構成が簡素化され、パブリケーションをサブスクリプションに配信する際の遅延が最小限に抑えられます。

しかし、サブスクリプションとパブリッシャーの位置によっては、クラスターがすぐに完全に相互接続された状態になり、どのキュー・マネージャーも他のあらゆるキュー・マネージャーに直接接続された状態になります。 実際の環境において、それでいい場合もあれば、望ましくない場合もあります。 同じように、サブスクライブ対象のトピック・ストリングのセットが頻繁に変化する場合、すべてのキュー・マネージャー間にその情報を伝搬するためのオーバーヘッドが、無視できないほどになる可能性もあります。 直接ルーティング型パブリッシュ/サブスクライブ・クラスター内のすべてのキュー・マネージャーは、それらのオーバーヘッドに対処できるようになっていなければなりません。

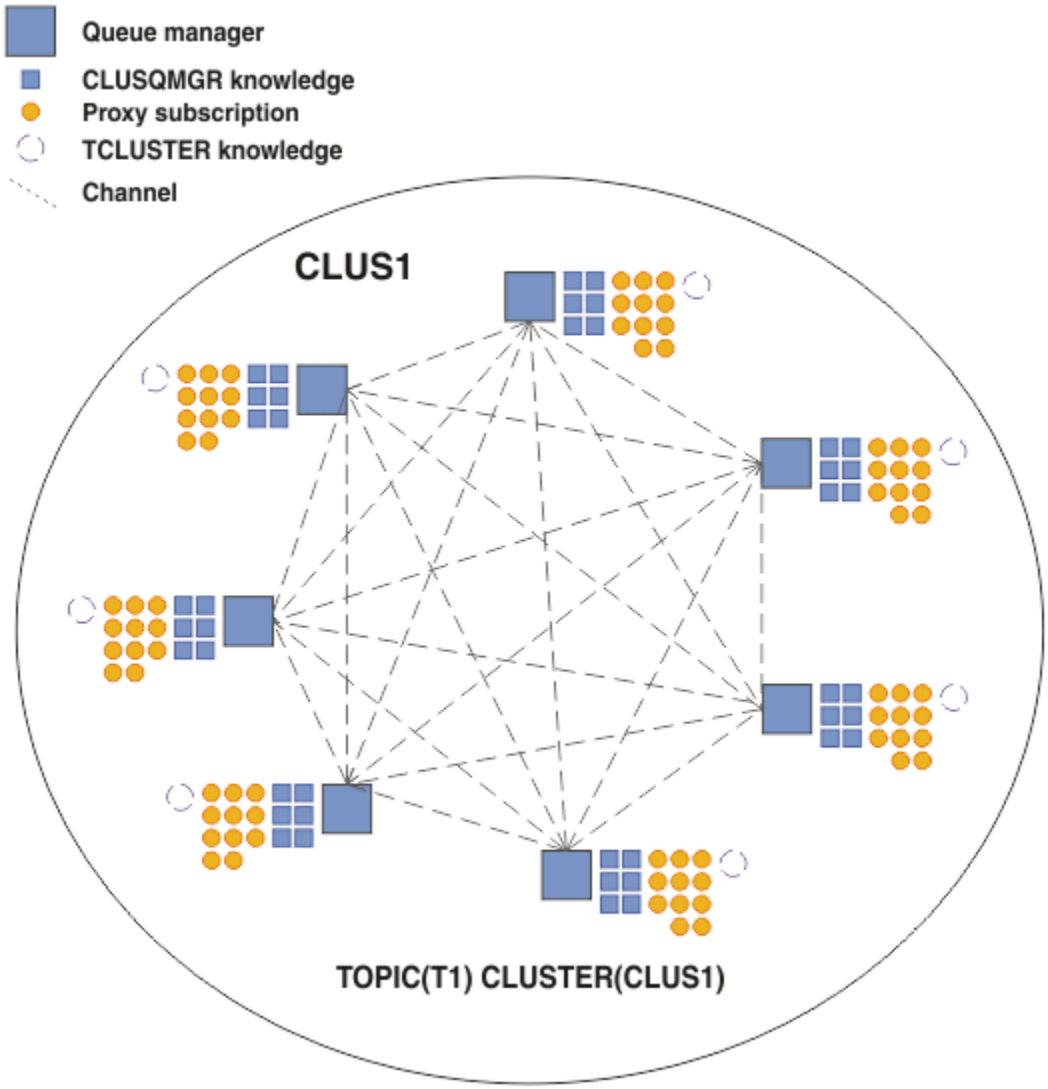


図 19. 完全に相互接続された直接ルーティング型パブリッシュ/サブスクライブ・クラスター

まとめと付加的な考慮事項

直接ルーティング型パブリッシュ/サブスクライブ・クラスターでは、作成または管理のための手操作による介入はほとんど必要なく、パブリッシャーとサブスクライバーの間の直接ルーティングを提供します。特定の構成においては、これが最適のトポロジーです。 特に、キュー・マネージャーの数の少ないクラスターの場合、あるいは、キュー・マネージャーの高い接続率が受け入れ可能でありサブスクリプションがあまりに変化しない場合には、これが最適です。 しかし、この場合、システムにおいて特定の制約も発生します。

- 各キュー・マネージャーの負荷は、クラスター内のキュー・マネージャーの総数に比例します。 したがって、クラスターが大きくなるにつれて、個々のキュー・マネージャーとシステム全体でパフォーマンスの問題が発生する可能性があります。
- デフォルトでは、サブスクライブ対象のすべてのクラスター化トピック・ストリングがクラスターを通じて伝搬します。 また、パブリケーションは、関連するトピックのサブスクリプションのあるリモート・キュー・マネージャーにのみ伝搬します。 したがって、サブスクリプションのセットが急速に変化すると、それは制限要因となり得ます。 デフォルトのこの動作は変更可能であり、すべてのパブリケーションがすべてのキュー・マネージャーに伝搬するようにすれば、プロキシー・サブスクリプションの必要がなくなります。 こうすると、サブスクリプションに関する情報のトラフィックは削減されますが、パブリケーションのトラフィック、および各キュー・マネージャーが確立するチャネルの数は増加する可能性

があります。『パブリッシュ/サブスクライブ・ネットワークでのサブスクリプションのパフォーマンス』を参照してください。

注: これと同じような制限が階層にも当てはまります。

- パブリッシュ/サブスクライブ・キュー・マネージャーの相互接続された性質のため、プロキシー・サブスクリプションがネットワーク内のすべてのノードを伝搬するのに時間がかかります。 リモート・パブリケーションでは、必ずしもサブスクライブがすぐに開始されるわけではないため、早い時期のパブリケーションが新しいトピック・ストリングのサブスクリプション後になっても送信されないという可能性があります。 すべてのパブリケーションがすべてのキュー・マネージャーに伝搬するようにして、プロキシー・サブスクリプションが不要であるようにすれば、サブスクリプションの遅れによって発生する問題はなくなります。『パブリッシュ/サブスクライブ・ネットワークでのサブスクリプションのパフォーマンス』を参照してください。

  注: この制限は、階層にも当てはまります。

直接ルーティングを使用する前に、82 ページの『パブリッシュ/サブスクライブ・クラスターでのトピック・ホスト・ルーティング』および 108 ページの『パブリッシュ/サブスクライブ階層でのルーティング』で詳しく説明されている別のアプローチを検討してください。

パブリッシュ/サブスクライブ・クラスターでのトピック・ホスト・ルーティング

クラスター内の非ホスティング・キュー・マネージャーからのパブリケーションは、ホスティング・キュー・マネージャーを経由して、クラスター内でサブスクリプションが一致するキュー・マネージャーにルーティングされます。

パブリッシュ/サブスクライブ階層およびクラスター内のキュー・マネージャー間でメッセージがルーティングされる方法については、『分散パブリッシュ/サブスクライブのネットワーク』を参照してください。

トピック・ホスト・ルーティングの動作方法とメリットについて理解するには、まず 78 ページの『パブリッシュ/サブスクライブ・クラスターでの直接ルーティング』について理解するのが最善です。

トピック・ホスト・ルーティングされるパブリッシュ/サブスクライブ・クラスターは、次のように動作します。

- クラスター化管理トピック・オブジェクトは、クラスター内の個々のキュー・マネージャーにおいて手動で定義されます。 それらはトピック・ホスト・キュー・マネージャー と呼ばれます。
- クラスター・キュー・マネージャーでサブスクリプションがなされた場合、サブスクリプション・ホスト・キュー・マネージャーからトピック・ホスト・キュー・マネージャーへのチャネルが作成され、プロキシー・サブスクリプションはトピックをホストするキュー・マネージャーでのみ作成されます。
- アプリケーションがトピックに情報をパブリッシュすると、接続されているキュー・マネージャーは、常にそのトピックをホストする 1 つのキュー・マネージャーにそのパブリケーションを転送します。 そしてそこからクラスター内のキュー・マネージャーのうち、そのトピックに対する一致するサブスクリプションのあるものすべてに渡されます。

以下、このプロセスについて例により詳しく説明します。

単一のトピック・ホストを使用したトピック・ホスト・ルーティング

トピック・ホスト・ルーティング型クラスター内のキュー・マネージャー間をパブリケーションが流れるためには、パブリッシュ/サブスクライブ・クラスターの構成で説明されているようにしてトピック・ツリーのブランチをクラスター化し、トピック・ホスト・ルーティング を指定します。

トピック・ホスト・ルーティング型トピック・オブジェクトを、クラスター内の複数のキュー・マネージャー上で定義することには、いくつかの理由があります。 しかし、説明を簡単にするため、単一のトピック・ホストから始めることにします。

次の図は、パブリッシュ/サブスクライブ・アクティビティーまたは Point-to-Point アクティビティーに現在使用されていないキュー・マネージャー・クラスターを示しています。 クラスター内のどのキュー・マネージャーも、フル・リポジトリー・キュー・マネージャーとのみ接続していることに注意してください。

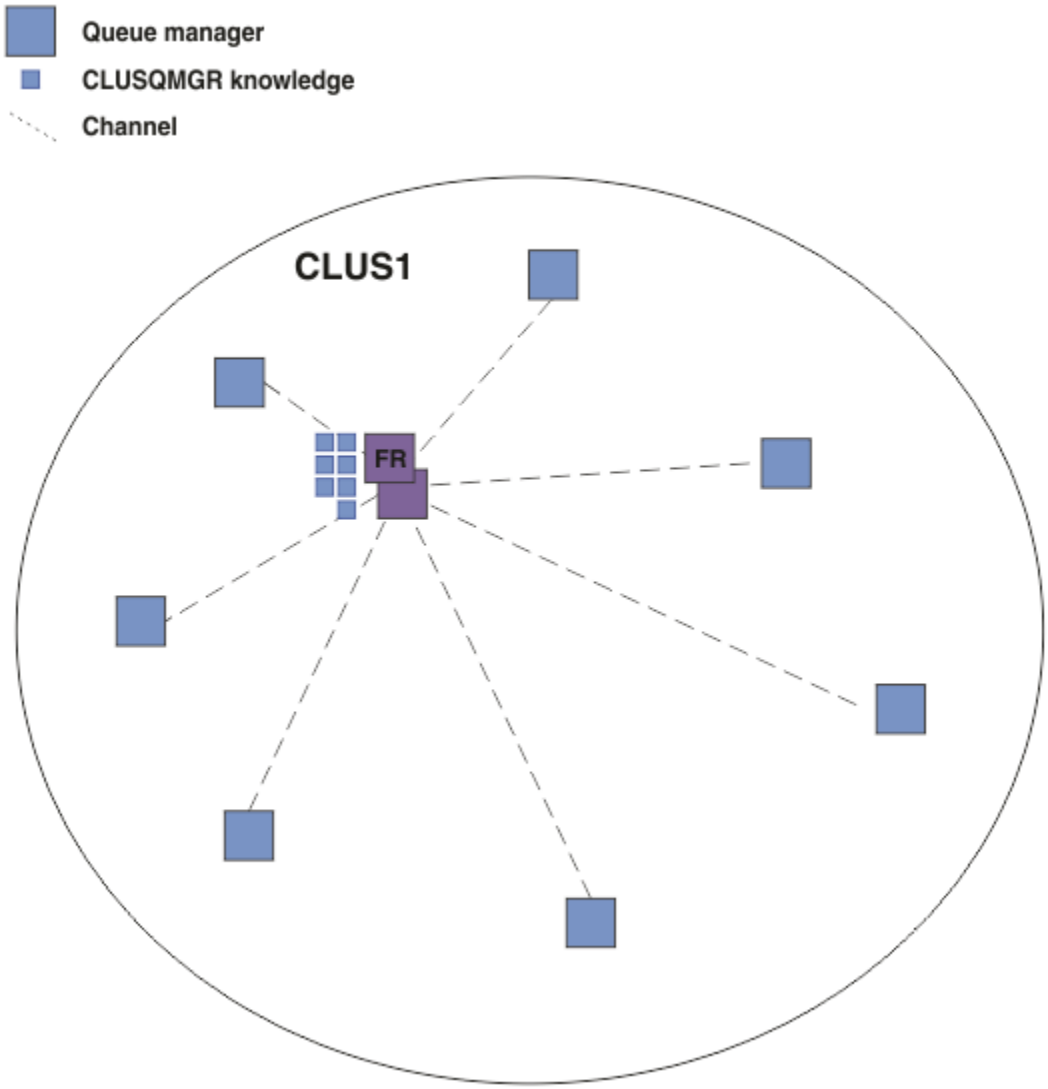


図 *20.* キュー・マネージャー・クラスター

トピック・ホスト・ルーティング型パブリッシュ/サブスクライブ・クラスターにおいては、クラスター内の特定のキュー・マネージャーにおいてトピック・オブジェクトを定義します。 パブリッシュ/サブスクライブ・トラフィックはそのキュー・マネージャーを通過するため、そのキュー・マネージャーはクラスター内の重要なキュー・マネージャーとなり、そのワークロードが増加します。 そのため、フル・リポジトリー・キュー・マネージャーを使用することはお勧めできません。 クラスター内の別のキュー・マネージャーを使用するようにしてください。 ホスト・キュー・マネージャー上でトピック・オブジェクトを定義すると、そのオブジェクトとそのホストの情報が、フル・リポジトリー・キュー・マネージャーによってクラスター内の他のすべてのキュー・マネージャーに自動的に送られます。 直接ルーティング とは異なり、各キュー・マネージャーに、クラスター内の他のあらゆるキュー・マネージャーに関する情報が伝えられるわけではないことに注意してください。

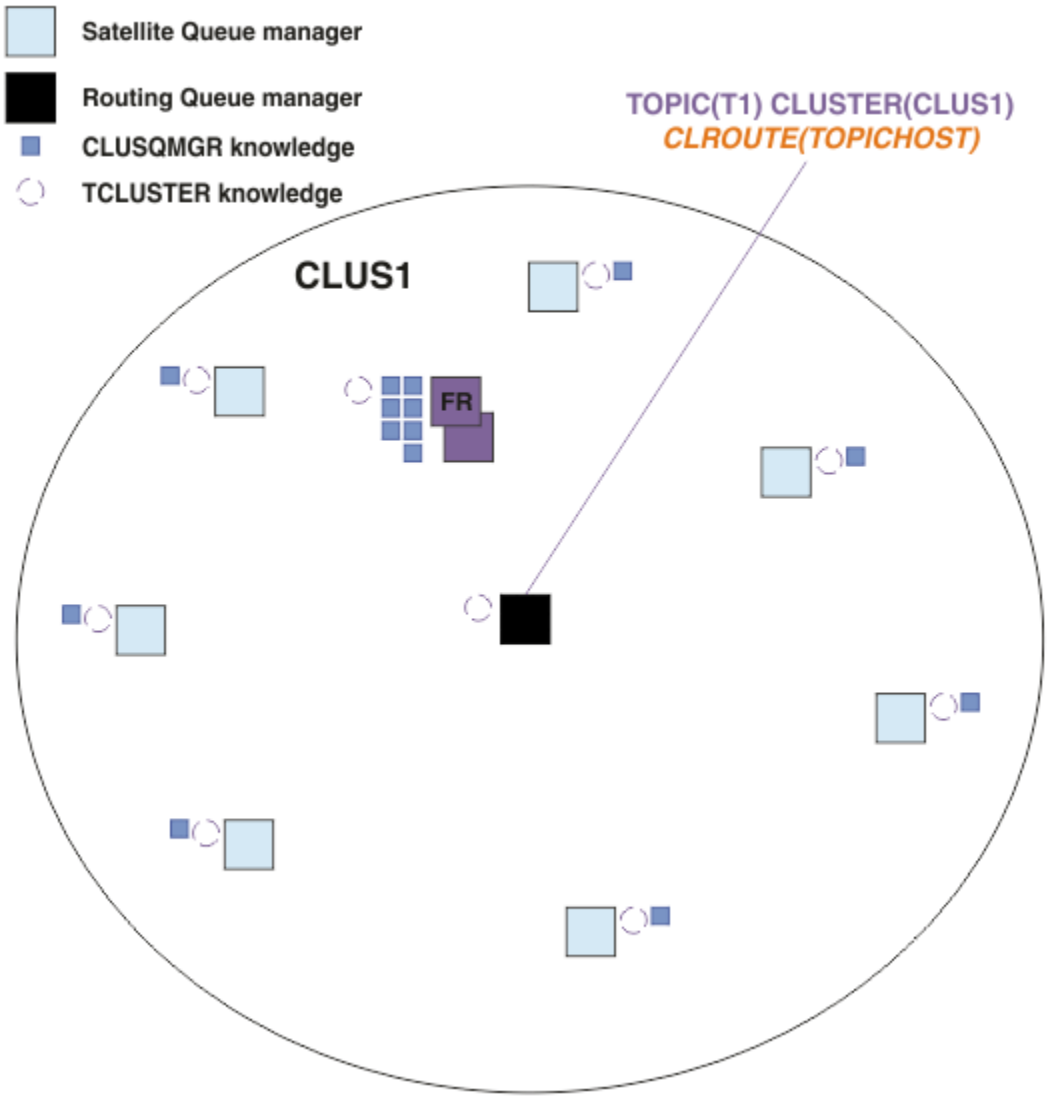


図 *21. 1* つのトピック・ホスト上で *1* つのトピックが定義されているトピック・ホスト・ルーティング型パブリッシュ/サブスクライブ・クラスター

あるキュー・マネージャー上でサブスクリプションが作成されると、サブスクライブ側キュー・マネージャーとトピック・ホスト・キュー・マネージャーとの間にチャネルが作成されます。 サブスクライブ側キュー・マネージャーはトピック・ホスト・キュー・マネージャーにのみ接続し、サブスクリプションの詳細を (プロキシー・サブスクリプション の形で) 送信します。 トピック・ホスト・キュー・マネージャーは、サブスクリプションについてのこの情報を、クラスター内の他のキュー・マネージャーには転送しません。

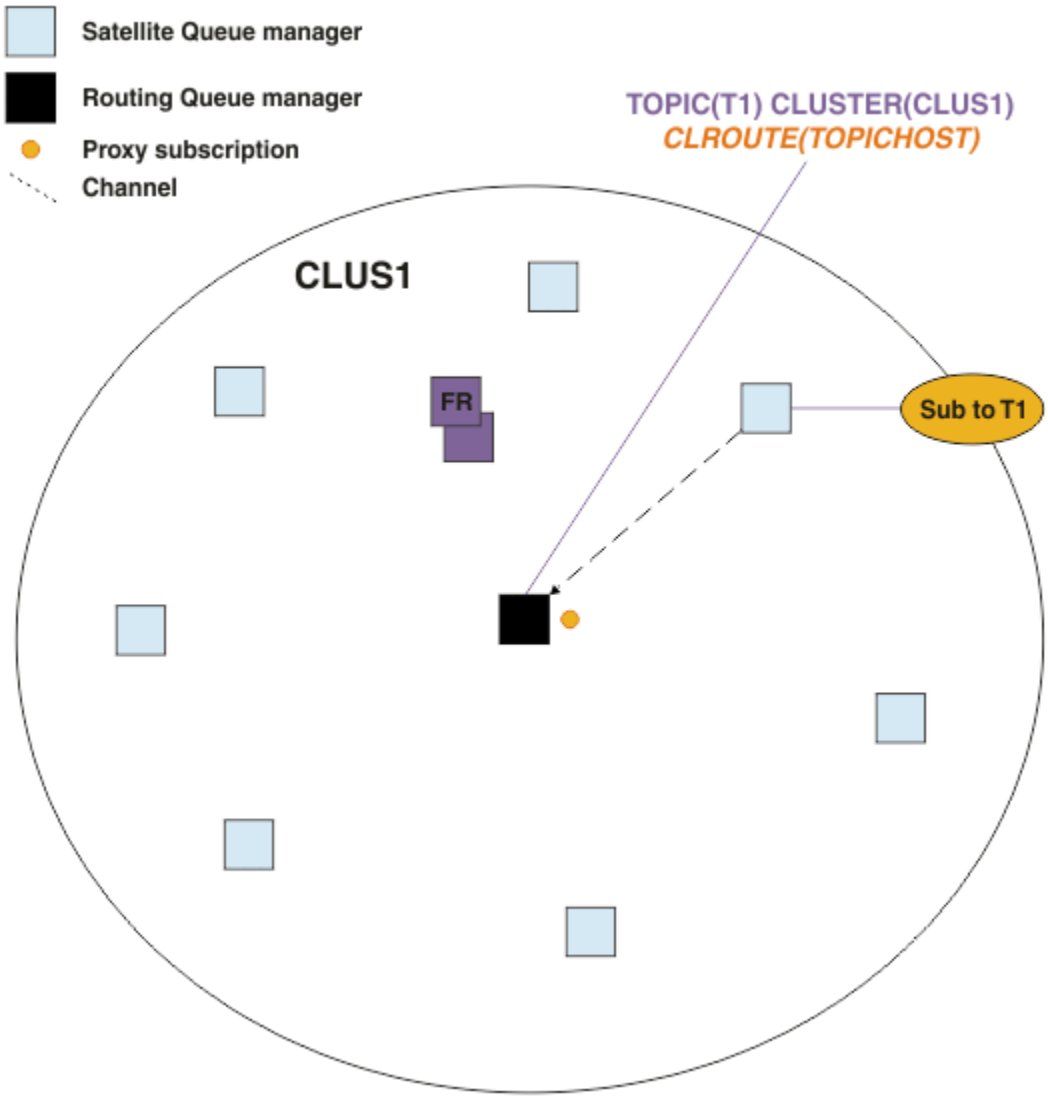


図 *22. 1* つのトピック・ホスト上で *1* つのトピックが定義され、*1* つのサブスクライバーのあるトピック・ホスト・ルーティング型パブリッシュ/サブスクライブ・クラスター

パブリッシュ側アプリケーションが別のキュー・マネージャーに接続してメッセージがパブリッシュされると、パブリッシュ側キュー・マネージャーとトピック・ホスト・キュー・マネージャーとの間にチャネルが作成され、メッセージがそのキュー・マネージャーに転送されます。 パブリッシュ側キュー・マネージャーには、クラスター内の他のキュー・マネージャー上のサブスクリプションに関するナレッジがないため、クラスター内にそのトピックのサブスクライバーがない場合でも、メッセージがトピック・ホスト・キュー・マネージャーに転送されます。 パブリッシュ側キュー・マネージャーは、トピック・ホスト・キュー・マネージャーにのみ接続します。 パブリケーションはトピック・ホストを経由してサブスクライブ側キュー・マネージャーにルーティングされます (サブスクライブ側キュー・マネージャーが存在する場合)。

パブリッシャーと同じキュー・マネージャー上のサブスクリプションは、まずトピック・ホスト・キュー・マネージャーにメッセージを送信するのではなく、直接に対応されます。

各トピック・ホスト・キュー・マネージャーの果たす役割が非常に重要であるため、負荷、アベイラビリティー、および接続に関するトピック・ホスティングの要件を処理することの可能なキュー・マネージャーを選択する必要があります。

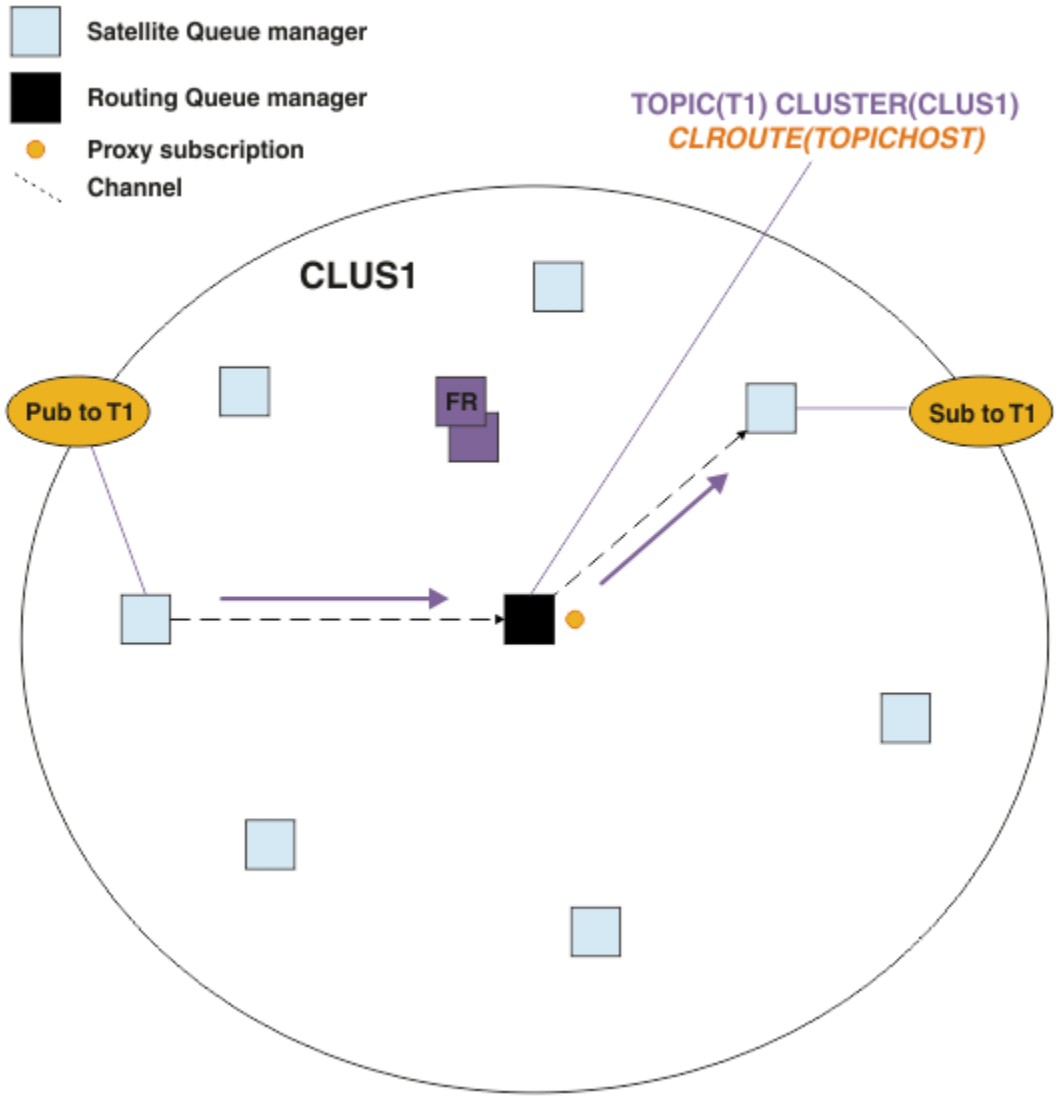


図 *23. 1* つのトピック、*1* つのサブスクライバー、および *1* つのパブリッシャーのあるトピック・ホスト・ルーティング型パブリッシュ/サブスクライブ・クラスター

複数のキュー・マネージャーの間でのトピック・ツリーの分割

ルーティング型トピック・ホスティング・キュー・マネージャーは、その管理対象トピック・オブジェクトが構成されているトピック・ツリーのブランチに関連するサブスクリプションの情報とパブリケーション・メッセージのみ担当します。 クラスター内の複数の異なるパブリッシュ/サブスクライブ・アプリケーションにより複数の異なるトピックが使用される場合、トピック・ツリーの複数の異なるクラスター化ブランチをホストするために複数の異なるキュー・マネージャーを構成することができます。 そのようにするなら、クラスター内の各トピック・ホスト・キュー・マネージャー上のパブリケーション・トラフィック、サブスクリプションの情報、およびチャネルが少なくなり、スケーリングが可能になります。 トピック・ツリーの別個のブランチが大量にある場合、この方法を使用してください。

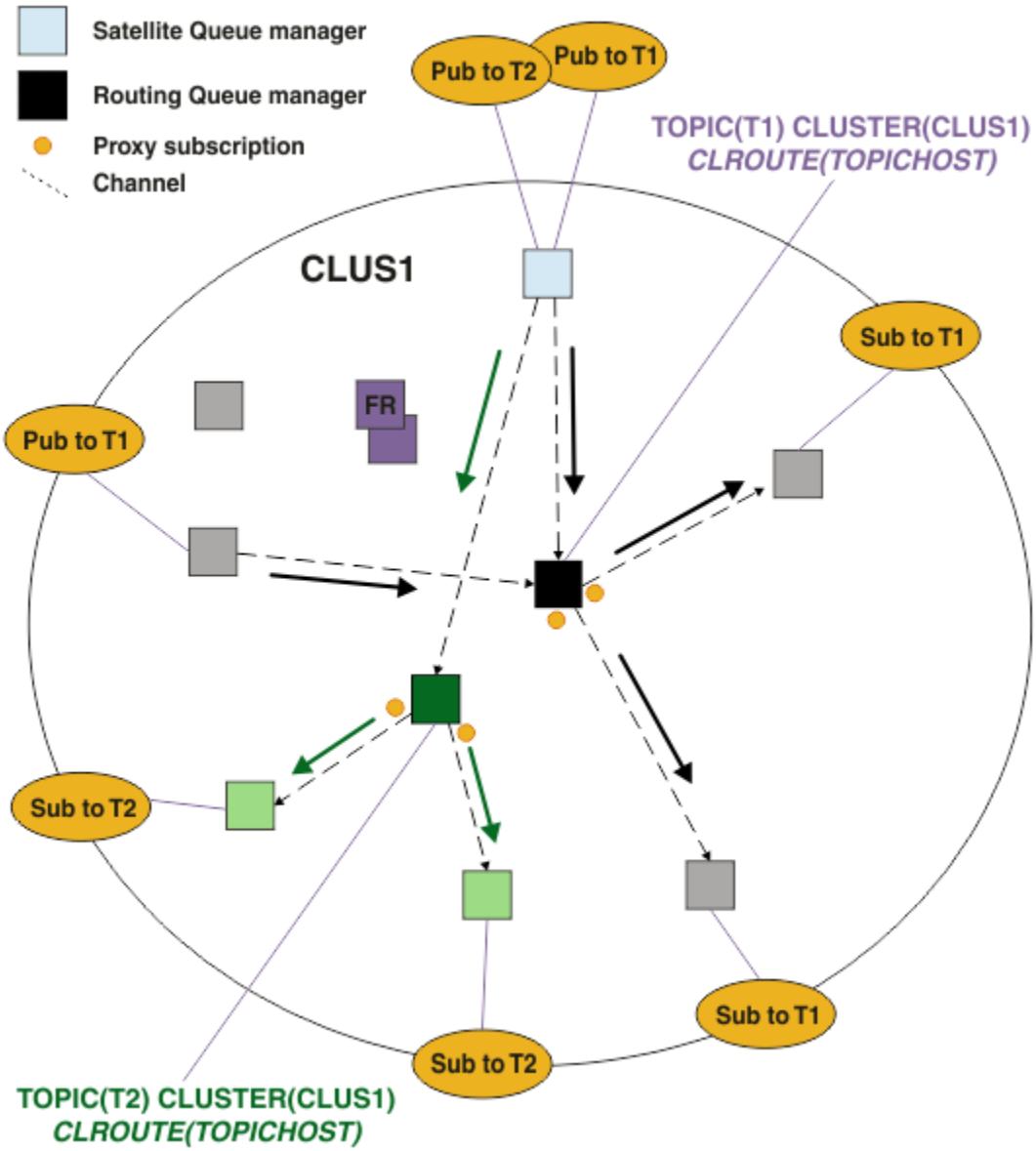


図 *24. 2* つのトピックのそれぞれが *1* つのトピック・ホスト上で定義されているトピック・ホスト・ルーティング型パブリッシュ/サブスクライブ・クラスター

例えば、トピック・ツリーで説明されているトピックを使用する場合、トピック T1 がトピック・ストリング /USA/Alabama で構成され、トピック T2 がトピック・ストリング /USA/Alaska で構成されていたとすると、/USA/Alabama/Mobile にパブリッシュされるメッセージは、T1 をホストするキュー・マネージャーを経由してルーティングされ、/USA/Alaska/Juneau にパブリッシュされるメッセージは T2 をホストするキュー・マネージャーを経由してルーティングされることになります。

注 : トピック・ツリーのうち、クラスター化されている点より上にワイルドカードを使用することにより、単一のサブスクリプションで、トピック・ツリーの複数のクラスター化ブランチをカバーすることはできません。 ワイルドカード・サブスクリプションを参照してください。

単一のトピックに複数のトピック・ホストを使用するトピック・ホスト・ルーティング

単一のキュー・マネージャーがトピックのルーティングを担当している場合に、そのキュー・マネージャーが使用不可になるか、またはワークロードの処理ができない状態になると、パブリケーションがサブスクリプションに速やかに流れなくなります。

回復力、スケーラビリティー、およびワークロード・バランシングを、1 つのキュー・マネージャーのみでトピックを定義する場合よりもさらに強化する必要があるなら、トピックを複数のキュー・マネージャー上で定義することができます。 パブリッシュされる個々のメッセージのそれぞれが、単一トピック・ホスト経由でルーティングされます。 一致するトピック・ホスト定義が複数存在する場合は、それらのトピック・ホストの中から 1 つが選択されます。 その選択は、クラスター・キューの場合と同じ方法でなされます。 これにより、使用不可状態のトピック・ホストを回避して使用可能なトピック・ホストにメッセージがルーティングされ、複数のトピック・ホスト・キュー・マネージャーおよびチャネルの間でメッセージ負荷のワークロード・バランシングが可能になります。 しかし、クラスター内で同じトピックに複数のトピック・ホストを使用する場合に、複数のメッセージの順序付けは維持されません。

次の図に、2 つのキュー・マネージャー上で同じトピックが定義されているトピック・ホスト・ルーティング型クラスターを示します。 この例では、サブスクライブ側キュー・マネージャーが、サブスクライブ対象のトピックに関する情報を、プロキシー・サブスクリプションの形で、2 つのトピック・ホスト・キュー・マネージャーの両方に送信します。

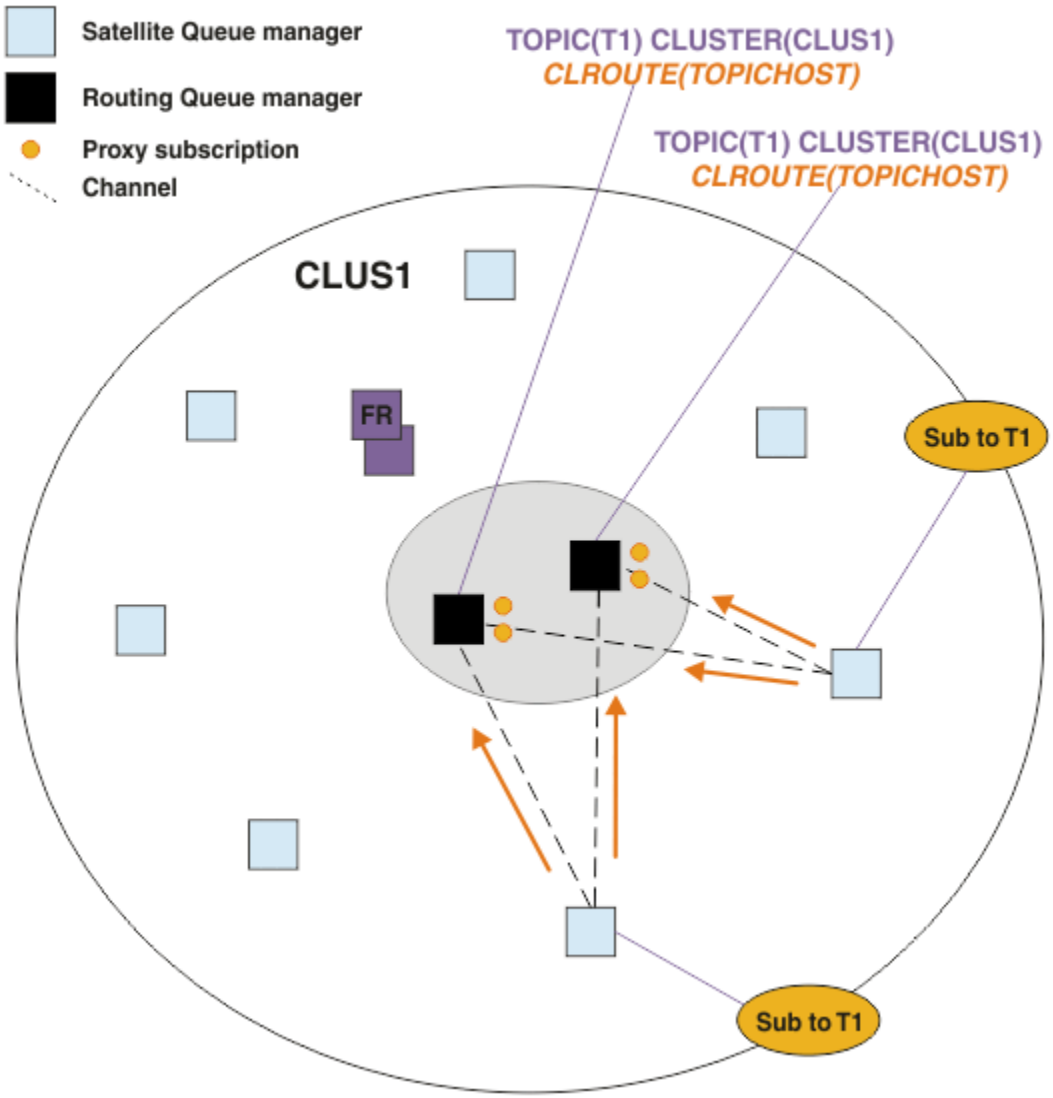


図 25. 複数のトピック・ホスト・パブリッシュ/サブスクライブ・クラスターでのプロキシー・サブスクリプションの作成

非ホスティング・キュー・マネージャーからパブリケーションがなされると、キュー・マネージャーはパブリケーションのコピーを、そのトピックのトピック・ホスト・キュー・マネージャーの *1* つ に送信します。 クラスター・ワークロード管理アルゴリズムのデフォルトの動作に基づいて、システムによりホストが選択されます。 典型的なシステムでは、各トピック・ホスト・キュー・マネージャーを通じたラウンド

ロビン分散に近いものとなります。 同じパブリッシュ側アプリケーションからのメッセージの間に親和性はありません。 これは、NOTFIXED のクラスター・バインド・タイプを使用する場合と同等です。

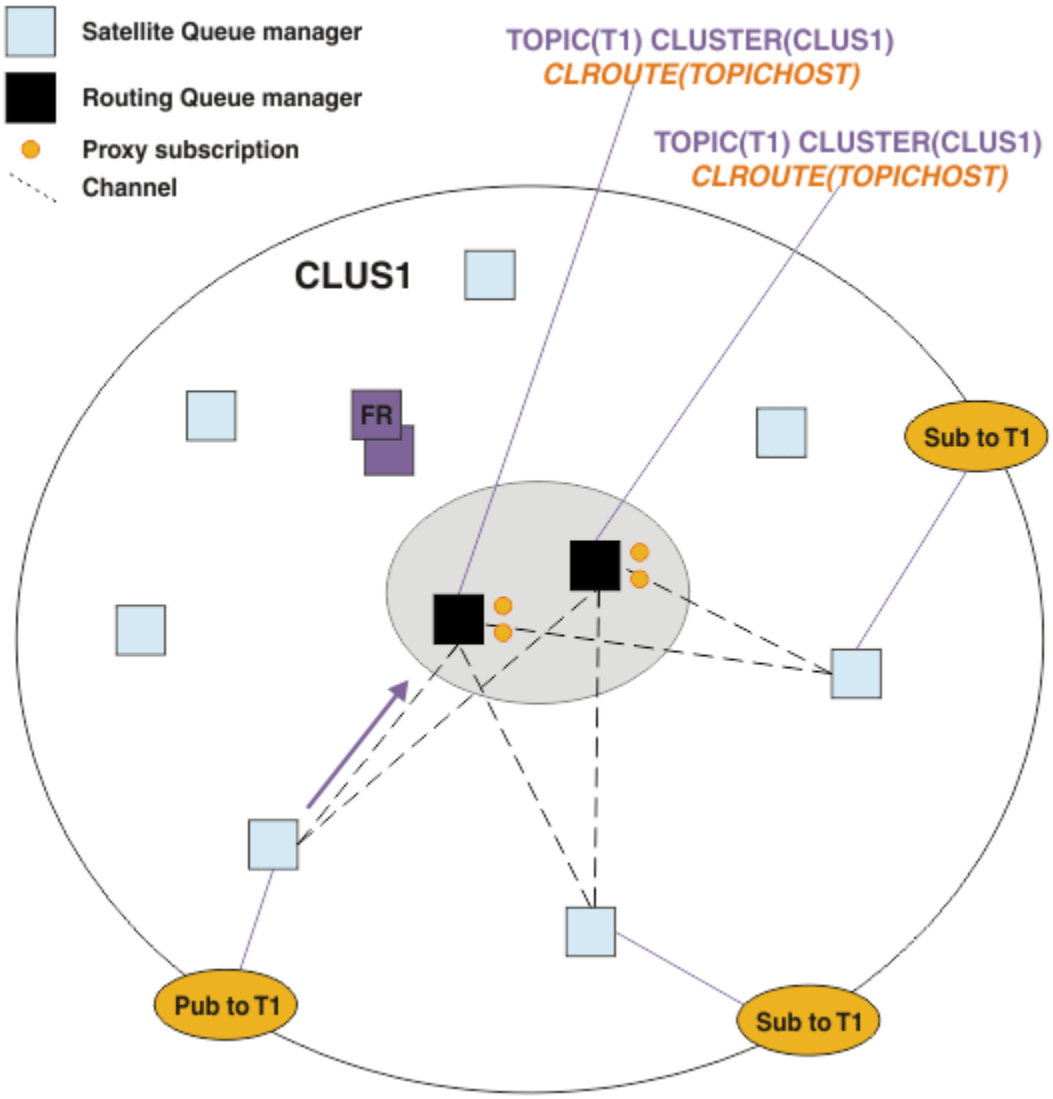


図 26. 複数のトピック・ホスト・パブリッシュ/サブスクライブ・クラスターでのパブリケーションの受信

選択されたトピック・ホスト・キュー・マネージャーへのインバウンド・パブリケーションは、一致するプロキシー・サブスクリプションが登録されているすべてのキュー・マネージャーに転送されます。

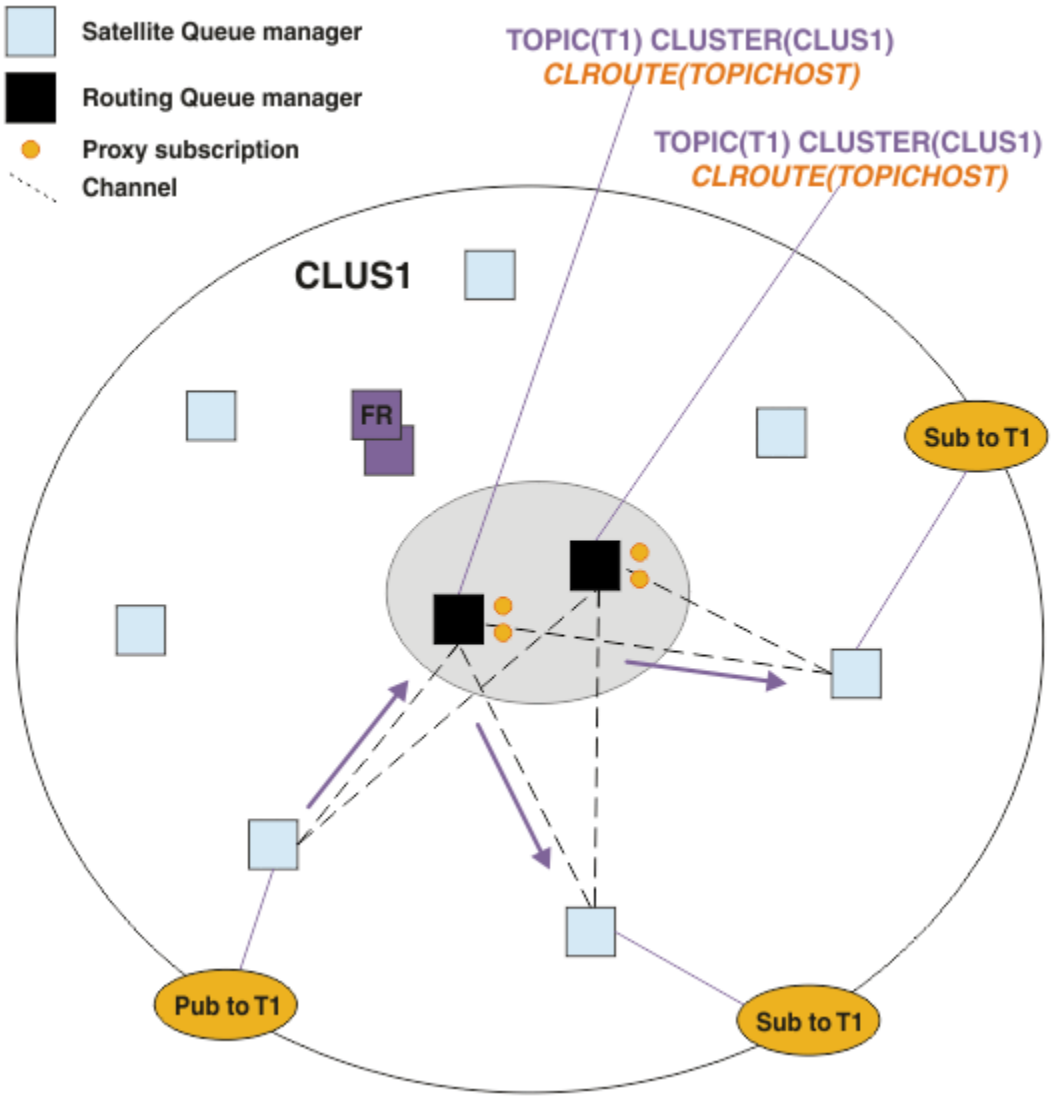


図 27. 複数のトピック・ホスト・パブリッシュ/サブスクライブ・クラスターのサブスクライバーへのパブリケーションのルーティング

サブスクリプションおよびパブリッシャーをトピック・ホスト・キュー・マネージャーにとってローカルにする

前述の例は、管理対象ルーティング型トピック・オブジェクトをホストしていないキュー・マネージャー上のパブリッシャーとサブスクライバーの間のルーティングを示しています。 それらのトポロジーにおいて、メッセージがサブスクリプションに到達するには、複数のホップ が必要です。

追加のホップが望ましくない場合は、キー・パブリッシャーをトピック・ホスティング・キュー・マネージャーに接続するとよいでしょう。 しかし、1 つのトピックについて複数のトピック・ホストがある一方でパブリッシャーは 1 つのみの場合、すべてのパブリケーション・トラフィックは、そのパブリッシャーの接続先のトピック・ホスト・キュー・マネージャー経由でルーティングされることになります。

同じように、キー・サブスクリプションがある場合、それらをトピック・ホスト・キュー・マネージャー上に配置することも可能です。 しかし、ルーティングされるトピックに複数のホストがある場合、追加のホップが回避されるのはパブリケーションの一部のみであり、残りは他のトピック・ホスト・キュー・マネージャーをまず経由してルーティングされます。

このようなトポロジーについては、集中型パブリッシャーまたはサブスクライバーを使用したトピック・ホスト・ルーティングで説明されています。

注 : パブリッシャーまたはサブスクリプションをルーティング型トピック・ホストと共存させている場合、ルーティング型トピック構成を変更するには、特別な計画が必要です。 例えば、トピック・ホスト・ルーティング型クラスターへのさらなるトピック・ホストの追加を参照してください。

まとめと付加的な考慮事項

トピック・ホスト・ルーティング型パブリッシュ/サブスクライブ・クラスターを使用すると、各トピックをどのキュー・マネージャーがホストするかを細かく制御することができます。 それらのキュー・マネージャーは、トピック・ツリーのそのブランチのルーティング ・キュー・マネージャーとなります。 さらに、サブスクリプションもパブリッシャーもないキュー・マネージャーはトピック・ホスト・キュー・マネージャーに接続する必要がなく、サブスクリプションのあるキュー・マネージャーがトピックをホストしていないキュー・マネージャーに接続する必要もありません。 この構成では、クラスター内のキュー・マネージャー間の接続の数、およびキュー・マネージャー間で渡される情報の量がかなり少なくなる可能性があります。 特に、大規模クラスターで、パブリッシュ/サブスクライブの作業を実行するのがキュー・マネージャーのうちのあるサブセットのみという場合、これが当てはまります。 さらにこの構成では、クラスター内の個々のキュー・マネージャーの負荷をある程度制御できるため、例えば、アクティブ度の高いトピックを、処理能力と回復力の高いシステムでホストするように選択することが可能です。 特定の構成、特に大規模クラスターでは、通常、こちらのほうが直接ルーティング よりも適したトポロジーです。

しかし、トピック・ホスト・ルーティングでは、システムにいくつかの点で制約が課せられることにもなります。

- 直接ルーティングの場合よりも、システム構成および保守を十分に計画する必要があります。 トピック・ツリーでクラスター化を実行するポイントや、クラスター内のトピック定義の場所を決定する必要があります。
- 直接ルーティング型トピックとちょうど同じように、新しいトピック・ホスト・ルーティング型トピックが定義されると、情報がフル・リポジトリー・キュー・マネージャーにプッシュされ、そこからクラスター内のすべてのメンバーに送信されます。 フル・リポジトリーからクラスターの各メンバーへのチャネルがまだ開始されていないのであれば、このイベントによりそれが開始されることになります。
- クラスター内にサブスクリプションがない場合でも、非ホスト・キュー・マネージャーからホスト・キュー・マネージャーにパブリケーションが常に送信されます。 そのため、通常はサブスクリプションが存在することが予期される場合、あるいはグローバルな接続とナレッジによるオーバーヘッドが余分のパブリケーション・トラフィックのリスクより大きい場合は、ルーティング型トピックを使用してください。

  注 : 前述のように、パブリッシャーをトピック・ホストにとってローカルにすると、このリスクが軽減される場合があります。
- 非ホスト・キュー・マネージャー上でパブリッシュされるメッセージは、サブスクリプションをホストするキュー・マネージャーに直接には到達しません。 それらは、常にトピック・ホスト・キュー・マネージャーを経由してルーティングされます。 このアプローチにより、クラスターの総オーバーヘッドが増加し、メッセージの遅延が大きくなり、パフォーマンスが低下することがあります。

  注 : 前述のように、サブスクリプションとパブリッシャーをトピック・ホストにとってローカルにすると、このリスクが軽減される場合があります。
- 単一のトピック・ホスト・キュー・マネージャーを使用すると、トピックに対してパブリッシュされるすべてのメッセージに対して単一障害点が存在することになります。 その単一障害点は、複数のトピック・ホストを定義すると除去できます。 しかし、ホストが複数になると、サブスクリプションの受け取るパブリッシュ・メッセージの順序に影響します。
- トピック・ホスト・キュー・マネージャーを使用すると、複数のキュー・マネージャーからのパブリケーション・トラフィックを処理する必要があるため、追加のメッセージ負荷が発生します。 この負荷を軽減するには、単一トピックに対して複数のトピック・ホストを使用するか (この場合、メッセージの順序は維持されない)、またはトピック・ツリーの複数の異なるブランチに対して、ルーティングされるトピックをホストするために複数の異なるキュー・マネージャーを使用することができます。

トピック・ホスト・ルーティングを使用する前に、78 ページの『パブリッシュ/サブスクライブ・クラスターでの直接ルーティング』、および 108 ページの『パブリッシュ/サブスクライブ階層でのルーティング』で詳しく説明されている別のアプローチについて検討してください。

パブリッシュ/サブスクライブのクラスター化: ベスト・プラクティス

クラスター・トピックを使用すると、キュー・マネージャー間のパブリッシュ/サブスクライブ・ドメインを簡単に拡張できるようになりますが、その機構と影響について十分に理解しておかないと、問題につながる恐れがあります。 情報の共有とパブリケーションのルーティングには、2 つのモデルがあります。 個々のビジネス・ニーズに最も適合し、選択したクラスター上で最高のパフォーマンスを発揮できるモデルを実装してください。

以下のセクションで説明するベスト・プラクティス情報は、何にでも当てはまるソリューションではなく、一般的な問題を解決するための共通の方法を示すものです。 ここでは、IBM MQ クラスター、およびパブリッシュ/サブスクライブ・メッセージングの基本を理解していること、および『分散パブリッシュ/ サブスクライブのネットワーク』 および 75 ページの『パブリッシュ/サブスクライブ・クラスターの設計』の情報についての知識があることを前提としています。

Point-to-Point メッセージングのためにクラスターを使用する場合、クラスター内の各キュー・マネージャーは「知る必要性」に基づいて機能します。 つまり、他のクラスター・リソース (クラスター内の他のキュー・マネージャーや、クラスター・キューなど) に関する情報は、それらに接続するアプリケーションが使用することを要求した場合にのみ検索されます。 クラスターにパブリッシュ/サブスクライブ・メッセージングを追加すると、クラスター・キュー・マネージャー間に、情報および接続のさまざまなレベルの共有が導入されます。 パブリッシュ/サブスクライブ・クラスターのベスト・プラクティスに従うには、動作におけるこの変化の影響について十分に理解しておく必要があります。

明確なニーズに基づいて最善のアーキテクチャーを構築できるようにするため、パブリッシュ/サブスクライブ・クラスター内での情報の共有およびパブリケーションのルーティングに関して、直接ルーティングとトピック・ホスト・ルーティング という 2 つのモデルが用意されています。 適切な選択を行うためには、両方のモデルについて、また各モデルが満たしているべき異なる要件について理解している必要があります。 これらの要件については、以下のセクション、および72 ページの『分散パブリッシュ/サブスクライブ・ネットワークの計画』で説明しています。

- 92 ページの『パブリッシュ/サブスクライブ・アクティビティーに参加するクラスター・キュー・マネージャーの数を制限するべき理由』
- 93 ページの『どのトピックをクラスター化するかを決める方法』
- 93 ページの『システムのサイズを決める方法』
- 95 ページの『パブリッシャーおよびサブスクリプションの位置』
- 95 ページの『パブリケーション・トラフィック』
- 96 ページの『サブスクリプションの変更と動的トピック・ストリング』

パブリッシュ/サブスクライブ・アクティビティーに参加するクラスター・キュー・マネージャーの数を制限するべき理由

クラスターでパブリッシュ/サブスクライブ・メッセージングを使用する際には、容量とパフォーマンスに関する考慮事項があります。 したがって、ベスト・プラクティスとなるのは、複数のキュー・マネージャーにまたがるパブリッシュ/サブスクライブ・アクティビティーの必要性について慎重に検討し、パブリッシュ/サブスクライブ・アクティビティーを必要な数のキュー・マネージャーに限定することです。 トピックに対してパブリッシュおよびサブスクライブが必要なキュー・マネージャーの最低限のセットを特定した後、それらのキュー・マネージャーのみがメンバーとなり、他のキュー・マネージャーは含まないクラスターを作成できます。

この方法が特に役立つのは、Point-to-Point メッセージングで適切に機能するクラスターが既に確立されている場合です。 既存の大規模なクラスターをパブリッシュ/サブスクライブ・クラスターに転換する場合は、最初にパブリッシュ/サブスクライブ作業用に別個のクラスターを作成し、現行のクラスターを使用するのではなく、作業用のクラスターでアプリケーションを試すという方法をお勧めします。 既に 1 つ以上の Point-to-Point クラスターに含まれている既存のキュー・マネージャーのサブセットを使用し、そのサブセットを新しいパブリッシュ/サブスクライブ・クラスターのメンバーにすることができます。 ただし、この新しいクラスターのフル・リポジトリー・キュー・マネージャーを、他のどのクラスターのメンバーにも含めてはなりません。既存のクラスターのフル・リポジトリーに追加の負荷が掛からないようにするためです。

新しいクラスターを作成することができず、既存の大規模なクラスターをパブリッシュ/サブスクライブ・クラスターに転換する必要がある場合は、直接ルーティング・モデルは使用しないでください。 トピック・ホスト・ルーティング・モデルは、通常、大規模なクラスターでのパフォーマンスが良好です。パブリッシュ/サブスクライブ情報の共有と接続性が、概してパブリッシュ/サブスクライブ作業を活発に実行するキュー・マネージャーのセットに限定され、トピックをホストしているキュー・マネージャーに集中するからです。 ただし、トピック定義をホストしているキュー・マネージャー上でサブスクリプション情報の手動リフレッシュが呼び出された場合は例外で、トピック・ホスト・キュー・マネージャーはその時点でクラスター内のすべてのキュー・マネージャーと接続します。 プロキシー・サブスクリプションの再同期を参照してください。

サイズまたは現行の負荷を理由に、ある特定のクラスターをパブリッシュ/サブスクライブに使用しないことに決めた場合は、そのクラスターが予期せずにパブリッシュ/サブスクライブ・クラスターに変換されるのを防止することをお勧めします。 **PSCLUS** キュー・マネージャー・プロパティーを使用すると、クラスター内のキュー・マネージャーに対して誰もクラスター・トピックを追加できなくなります。 103 ページの『クラスター化されたパブリッシュ/サブスクライブの禁止』を参照してください。

どのトピックをクラスター化するかを決める方法

クラスターにどのトピックを追加するかは、慎重に選ぶことが重要です。トピック・ツリーでの位置が高いほど、そのトピックが使用される範囲が広くなります。 その結果、サブスクリプション情報とパブリケーションが必要以上に波及することがあります。 トピック・ツリーに複数の独立したブランチがあり、そのうちの一部はクラスター化が必要で、それ以外はクラスター化が必要ないという場合には、クラスター化が必要な各ブランチのルートに管理用のトピック・オブジェクトを作成してクラスターに追加します。 例えば、ブランチ /A、/B、および /C にクラスター化が必要な場合は、各ブランチに対して別個のクラスター・トピック・オブジェクトを定義します。

注 : システムは、トピック・ツリー内でネストしたクラスター・トピック定義が発生しないようにします。各サブブランチについて、トピック・ツリー内の 1 カ所にのみクラスター・トピックを設定できます。 例えば、クラスター・トピック・オブジェクトを /A および /A/B に対して定義することはできません。 クラスター・トピックがネストしていると、特にサブスクリプションでワイルドカードが使用されている場合に、どのクラスター・オブジェクトがどのサブスクリプションに適用されるかについて混乱が生じる恐れがあります。 トピック・ホスト・ルーティングを使用する場合は、ルーティングの決定がトピック・ホストの割り振りによって正確に定義されるため、このことの重要性が増します。

クラスター・トピックをトピック・ツリーの高い位置に追加する必要がある場合に、ツリーでクラスター・ポイントより低い位置にある一部のブランチについてクラスター動作が必要ないときは、サブスクリプションおよびパブリケーションのスコープ属性を使用して、サブスクリプションおよびパブリケーションの共有レベルがそのトピックにまで波及しないように制限できます。

予想できる動作を考慮に入れずにトピックのルート・ノードをクラスターに追加することは避けてください。 グローバル・トピックは、可能な限り明白にしてください。例えば、トピック・ストリングで /global や /cluster といった高位修飾子を使用します。

ルート・トピック・ノードをクラスター化するのが望ましくないことには、別の理由もあります。 それは、すべてのキュー・マネージャーがルート・ノードのローカル定義として SYSTEM.BASE.TOPIC トピック・オブジェクトを持っていることです。 このオブジェクトをクラスター内の 1 つのキュー・マネージャー上でクラスター化すると、その他のすべてのキュー・マネージャーはそれを認識します。 しかし、同じオブジェクトのローカル定義が存在すると、そのプロパティーによってクラスター・オブジェクトがオーバーライドされます。 その結果、それらのキュー・マネージャーは、トピックがクラスター化されていないかのように動作します。 これを解決するには、SYSTEM.BASE.TOPIC のすべての定義をクラスター化する必要があります。 これを行うとすべてのキュー・マネージャーがトピック・ホストになるため、直接ルーティング定義ではこれを行うことができますが、トピック・ホスト・ルーティングの定義ではこれを行うことができません。

システムのサイズを決める方法

パブリッシュ/サブスクライブ・クラスターを構築すると、通常、クラスター内の Point-to-Point メッセージングとは異なるパターンのクラスター・チャネルが作成されます。 Point-to-Point モデルは「オプトイン」モデルですが、パブリッシュ/サブスクライブ・クラスターは、特に直接ルーティング・トピックを使用する場合に、サブスクリプションが多分岐するという判別性の低い性質を持っています。 したがって、

パブリッシュ/サブスクライブ・クラスター内のどのキュー・マネージャーがクラスター・チャネルを使用して他のキュー・マネージャーに接続し、それをどのような環境で行うかについて識別することが重要です。

次の表に、通常の稼働時にパブリッシュ/サブスクライブ・クラスター内の各キュー・マネージャーについて想定されるクラスター送信側チャネルおよび受信側チャネルの標準的なセットを、パブリッシュ/サブスクライブ・クラスター内でのキュー・マネージャーの役割に応じて記載します。

表 5. ルーティング方式ごとのクラスター送信側チャネルおよび受信側チャネル

キュー・マネージャーの役割	直接クラスター受信側	直接クラスター送信側	トピック・クラスター受信側	トピック・クラスター送信側
フル・リポジトリー	AllQmgrs	AllQmgrs	AllQmgrs	AllQMgrs
トピック定義のホスト	n/a	n/a	AllSubs+AllPubs (1)	AllSubs (1)
サブスクリプションが作成された	AllPubs (1)	AllQMgrs	AllHosts	AllHosts
パブリッシャーが接続された	AllSubs (1)	AllSubs (1)	AllHosts	AllHosts
パブリッシャーまたはサブスクライバーなし	AllSubs (1)	None (1)	None (2)	None (2)

キー:

AllQmgrs
クラスター内のすべてのキュー・マネージャーとの間のチャネル。

AllSubs
サブスクリプションが作成されたすべてのキュー・マネージャーとの間のチャネル。

AllPubs
パブリッシュ側アプリケーションが接続されたすべてのキュー・マネージャーとの間のチャネル。

AllHosts
クラスター・トピック・オブジェクトの定義が構成されたすべてのキュー・マネージャーとの間のチャネル。

なし
パブリッシュ/サブスクライブ・メッセージングの目的でのみ作成される、クラスター内の他のキュー・マネージャーとの間のチャネルなし。

注 :

1. キュー・マネージャーによるプロキシー・サブスクリプションのリフレッシュがこのキュー・マネージャーから行われた場合、クラスター内の他のすべてのキュー・マネージャーとの間のチャネルが自動的に作成される場合があります。
2. キュー・マネージャーによるプロキシー・サブスクリプションのリフレッシュがこのキュー・マネージャーから行われた場合、クラスター・トピックの定義をホストするクラスター内の他のキュー・マネージャーとの間のチャネルが自動的に作成される場合があります。

前の表に示されているとおり、一般にトピック・ホスト・ルーティングで使用される送信側および受信側チャネルの数は、直接クラスター・ルーティングより大幅に少なくなります。 そのため、容量のため、または特定のチャネルを確立できるかどうかが不確か (例えば、ファイアウォールを通過する場合) なため、クラスター内の特定のキュー・マネージャーについてチャネルの接続に懸念がある場合は、トピック・ホスト・ルーティングが望ましいソリューションになります。

パブリッシャーおよびサブスクリプションの位置

クラスター化されたパブリッシュ/サブスクライブでは、1 つのキュー・マネージャーでパブリッシュされたメッセージを、クラスター内の必要な他のすべてのキュー・マネージャー上のサブスクリプションに配信できます。 Point-to-Point メッセージングと同様に、キュー・マネージャー間でメッセージを伝送するコストがパフォーマンスに悪影響を与えることがあります。 そのため、トピックに対してサブスクリプションを作成する処理を、メッセージがパブリッシュされるのと同じキュー・マネージャーで行うことを検討する必要があります。

クラスター内でトピック・ホスト・ルーティングを使用する場合は、トピック・ホスティング・キュー・マネージャーから見たサブスクリプションおよびパブリッシャーの位置を検討することも重要です。 クラスター・トピックのホストであるキュー・マネージャーにパブリッシャーが接続されていないと、パブリッシュされるメッセージは、常にトピック・ホスティング・キュー・マネージャーに送信されます。 同様に、クラスター・トピックのトピック・ホストではないキュー・マネージャーでサブスクリプションが作成される場合、クラスター内の他のキュー・マネージャーからパブリッシュされたメッセージは、常に最初にトピック・ホスティング・キュー・マネージャーに送信されます。 さらに具体的に言うと、トピックをホストする 1 つのキュー・マネージャー上にサブスクリプションがあり、同じトピックをホストするキュー・マネージャーが他に 1 つ以上存在する場合、他のキュー・マネージャーからの特定の比率のパブリケーションがそれらの他のトピック・ホスティング・キュー・マネージャー経由でルーティングされることになります。 パブリッシャーとサブスクリプションの距離を最短化するようにトピック・ホスト・ルーティング型のパブリッシュ/サブスクライブ・クラスターを設計する方法について詳しくは、集中型パブリッシャーまたはサブスクライバーを使用したトピック・ホスト・ルーティングを参照してください。

パブリケーション・トラフィック

クラスター内の 1 つのキュー・マネージャーに接続されたアプリケーションによってパブリッシュされるメッセージは、クラスター送信側チャネルを使用して他のキュー・マネージャーのサブスクリプションに送信されます。

直接ルーティングを使用している場合、パブリッシュされたメッセージはキュー・マネージャー間の最短パスを通ります。 つまり、パブリッシュ側キュー・マネージャーから、サブスクリプションを持つ各キュー・マネージャーにメッセージが直接送信されます。 メッセージは、そのトピックに対するサブスクリプションを持たないキュー・マネージャーには送信されません。 『パブリッシュ/サブスクライブ・ネットワークでのプロキシー・サブスクリプション』を参照してください。

クラスター内のどれか 1 つのキュー・マネージャーともう 1 つのキュー・マネージャーとの間でパブリケーション・メッセージの比率が高い場合、これら 2 地点間のクラスター・チャネル・インフラストラクチャーは、その比率に対応できる必要があります。 これには、使用するチャネルおよび伝送キューのチューニングが含まれることもあります。

トピック・ホスト・ルーティングを使用している場合、トピック・ホストではないキュー・マネージャーでパブリッシュされた各メッセージは、トピック・ホスト・キュー・マネージャーに送信されます。 これは、クラスター内のどこかに 1 つ以上のサブスクリプションが存在するかどうかとは無関係に行われます。このことから、計画の際に考慮に入れるべき別の要因が発生します。

- 各パブリケーションをトピック・ホスト・キュー・マネージャーに最初に送信する際の付加的な待ち時間は、受け入れ可能な範囲か?
- 各トピック・ホスト・キュー・マネージャーは、インバウンドおよびアウトバウンドのパブリケーション・レートに対応可能か? 多くの異なるキュー・マネージャーにパブリッシャーがあるシステムについて考えてみます。 それらすべてのキュー・マネージャーからごく少数のトピック・ホスティング・キュー・マネージャーのセットに対してメッセージが送信されると、これらのメッセージの処理とサブスクライブ側キュー・マネージャーへのルーティングにおいて、これらのトピック・ホストがボトルネックになる可能性があります。
- パブリッシュされるメッセージのうち相当な比率のメッセージが、一致するサブスクライバーを持たないことが想定されるか? その場合に、そのようなメッセージのパブリッシュ率が高いときは、そのパブリッシャーのキュー・マネージャーをトピック・ホストにするのが最善であることがあります。 そのようにした状況では、クラスター内にサブスクリプションが存在しないメッセージがパブリッシュされた場合、そのメッセージは他のキュー・マネージャーに送信されません。

これらの問題も、複数のトピック・ホストを導入して、パブリケーションの負荷を全体に拡散することによって緩和できます。

- 複数の個別のトピックがあり、それぞれがかなりの比率のパブリケーション・トラフィックを占めている場合は、それらを異なるキュー・マネージャーでホストすることを検討します。
- トピックを異なるトピック・ホストに分離することができない場合は、同じトピック・オブジェクトを複数のキュー・マネージャーに定義することを検討します。 そうすれば、パブリケーションをルーティングする作業負荷がそれぞれのキュー・マネージャーに均等に割り振られます。 ただし、この処置が適切なのは、パブリケーション・メッセージの順序付けが不要な場合のみです。

サブスクリプションの変更と動的トピック・ストリング

別の考慮事項は、プロキシー・サブスクリプションを伝搬する処理がシステムのパフォーマンスに与える影響です。 通常、キュー・マネージャーは、特定のクラスター・トピック・ストリング (構成されたトピック・オブジェクトだけではない) についてそのキュー・マネージャー上で初めてサブスクリプションが作成された時点で、プロキシー・サブスクリプション・メッセージをクラスター内の他の特定のキュー・マネージャーに送信します。 同様に、特定のクラスター・トピック・ストリングに対する最後のサブスクリプションが削除された時点で、プロキシー・サブスクリプション削除メッセージが送信されます。

直接ルーティングの場合、サブスクリプションを持つ各キュー・マネージャーは、これらのプロキシー・サブスクリプションをクラスター内の他のすべてのキュー・マネージャーに送信します。 トピック・ホスト・ルーティングの場合、サブスクリプションを持つ各キュー・マネージャーは、これらのプロキシー・サブスクリプションを、そのクラスター・トピックの定義をホストしている各キュー・マネージャーだけに送信します。 したがって、直接ルーティングでは、クラスター内のキュー・マネージャーが増えると、それらのキュー・マネージャー間でプロキシー・サブスクリプションを維持するためのオーバーヘッドが大きくなります。 それに対して、トピック・ホスト・ルーティングでは、クラスター内のキュー・マネージャーの数は問題になりません。

どちらのルーティング・モデルでも、パブリッシュ/サブスクライブ・ソリューションに含まれる固有のトピック・ストリングが多い場合や、クラスター内のキュー・マネージャー上のトピックのサブスクライブとアンサブスクライブが頻繁に繰り返される場合は、プロキシー・サブスクリプションを配布および削除するメッセージが定常的に生成されるため、そのキュー・マネージャーに対する顕著なオーバーヘッドが表れます。 直接ルーティングでは、これらのメッセージをクラスター内のすべてのキュー・マネージャーに送信する必要があるため、問題がさらに悪化します。

サブスクリプションの変更率が高すぎて、トピック・ホスト・ルーティング型のシステムでも処理できない場合は、プロキシー・サブスクリプションのオーバーヘッドを削減する方法について、パブリッシュ/サブスクライブ・ネットワークでのサブスクリプションのパフォーマンスを参照してください。

クラスター・トピックの定義

クラスタ・トピックは、**cluster** 属性が定義された管理トピックです。 クラスター・トピックに関する情報は、クラスターのすべてのメンバーにプッシュされ、ローカル・トピックと結合されて、複数のキュー・マネージャーにわたるトピック・スペースの一部を形成します。 これにより、あるトピックに対して 1 つのキュー・マネージャーでパブリッシュされたメッセージが、クラスター内の他のキュー・マネージャーのサブスクリプションに配信されます。

キュー・マネージャーにクラスター・トピックを定義すると、そのクラスター・トピック定義が完全リポジトリー・キュー・マネージャーに送信されます。 完全リポジトリーは、そのクラスター・トピック定義をクラスター内のすべてのキュー・マネージャーに伝搬し、クラスター内のあらゆるキュー・マネージャーで、同じクラスター・トピックがパブリッシャーおよびサブスクライバーに使用可能になるようにします。 クラスター・トピックを作成するキュー・マネージャーは、クラスター・トピック・ホストと呼ばれます。 クラスター・トピックはクラスター内の任意のキュー・マネージャーで使用できますが、クラスター・トピックに対する変更は、そのトピックが定義されているキュー・マネージャー (クラスター・トピック・ホスト) で行う必要があります。変更を行った時点で、その変更は完全リポジトリーを介してクラスターのすべてのメンバーに伝搬されます。

直接ルーティングを使用する場合は、クラスター内のすべてのキュー・マネージャーがトピック定義を同じように使用するため、クラスター化されたトピック定義の場所がシステムの動作に直接影響を与えることはありません。 そのため、トピックの定義は、トピックが必要な間ずっとクラスターのメンバーであり、

フル・リポジトリー・キュー・マネージャーと定期的に情報交換していると信頼してよいシステムにあるものであればどのキュー・マネージャー上で行うこともできます。

トピック・ホスト・ルーティングを使用する場合は、クラスター内の他のキュー・マネージャーが、クラスター・トピック定義を行うキュー・マネージャーに対してチャネルを作成して、サブスクリプション情報とパブリケーションを送信するため、クラスター・トピックをどこで定義するかが非常に重要になります。 トピック定義をホストするための最適なキュー・マネージャーを選択するには、トピック・ホスト・ルーティングについて理解しておく必要があります。 82 ページの『パブリッシュ/サブスクライブ・クラスターでのトピック・ホスト・ルーティング』を参照。

クラスター化されたトピックとローカル・トピック・オブジェクトがある場合は、ローカル・トピックが優先されます。 99 ページの『同じ名前の複数のクラスター・トピック定義』を参照。

クラスター・トピックを表示するために使用するコマンドについては、関連情報を参照してください。

クラスター・トピックの継承

通常、クラスター化されたパブリッシュ/サブスクライブ・トポロジー内のパブリッシュ側およびサブスクライブ側のアプリケーションは、クラスター内のどのキュー・マネージャーから接続された場合にも同じ動作をすることが想定されています。 クラスター化された管理トピック・オブジェクトをクラスター内のすべてのキュー・マネージャーに伝搬するのは、そこに理由があります。

管理トピック・オブジェクトは、トピック・ツリー内の上位にある他の管理トピック・オブジェクトから動作を継承します。 この継承は、トピック・パラメーターに明示的な値が設定されなかった場合に発生します。

クラスター化されたパブリッシュ/サブスクライブの場合、継承が原因で、どのキュー・マネージャーに接続するかに応じてパブリッシャーおよびサブスクライバーの動作が異なる可能性が生じるため、継承についてよく検討することが重要です。 クラスター・トピック・オブジェクトに上位のトピック・オブジェクトから任意のパラメーターを継承させると、クラスター内の異なるキュー・マネージャーでトピックの動作が異なる可能性があります。 同様に、トピック・ツリー内でクラスター・トピック・オブジェクトより下位にあるトピック・オブジェクト定義をローカルで定義すると、これらの下位トピックは引き続きクラスター化されていますが、ローカル・オブジェクトの動作がクラスター内の他のキュー・マネージャーと異なるようになる可能性があります。

ワイルドカード・サブスクリプション

プロキシー・サブスクリプションが作成されるのは、クラスター・トピック・オブジェクトまたはその下位で解決するトピック・ストリングに対してローカル・サブスクリプションが作成されるときです。 ワイルドカード・サブスクリプションがいずれかのクラスター・トピックよりも高いトピック階層で作成された場合、一致するクラスター・トピックのプロキシー・サブスクリプションがクラスター全体に送信されないため、キュー・マネージャーはクラスターの他のメンバーからのパブリケーションを受け取りません。ただし、ローカル・キュー・マネージャーからのパブリケーションは受け取ります。

一方、別のアプリケーションがクラスター・トピックまたはその下位のトピックに解決されるトピック・ストリングにサブスクライブすると、プロキシー・サブスクリプションが生成され、パブリケーションがこのキュー・マネージャーに伝搬されます。 オリジナルが到着すると、上位のワイルドカード・サブスクリプションがそれらのパブリケーションの正当な宛先であると見なされ、コピーを受け取ります。 この動作が不要な場合は、クラスター・トピックに対して **WILDCARD(BLOCK)** を設定してください。 こうすると、オリジナルのワイルドカード・サブスクリプションは正当なサブスクリプションとは見なされなくなるため、クラスター・トピックまたはそのサブトピックに関するパブリケーションを (ローカル、またはクラスター内の別の場所から) 受け取らなくなります。

関連概念
管理トピックの操作
サブスクリプションの操作
関連資料
表示トピック
DISPLAYTPSTATUS
表示サブ

クラスター・トピックの属性

トピック・オブジェクトがクラスター名属性セットを持つ場合、トピック定義はクラスター内のすべてのキュー・マネージャーに伝搬されます。 各キュー・マネージャーは、伝搬されたトピック属性を使用してパブリッシュ/サブスクライブ・アプリケーションの動作を制御します。

トピック・オブジェクトには、パブリッシュ/サブスクライブ・クラスターに適用されるいくつかの属性があります。 パブリッシュ側およびサブスクライブ側のアプリケーションの全体的な動作を制御する属性や、クラスター全体でトピックがどのように使用されるかを制御する属性があります。

クラスター・トピック・オブジェクト定義は、クラスター内のすべてのキュー・マネージャーが正しく使用できるような方法で構成する必要があります。

例えば、管理サブスクリプションに使用するモデル・キュー (MDURMDL および MNDURMDL) にデフォルト以外のキュー名を設定する場合は、その名前のモデル・キューを、管理サブスクリプションが作成されるすべてのキュー・マネージャーに定義する必要があります。

同様に、いずれかの属性が ASPARENT に設定されている場合、トピックの動作はクラスター内の個々のキュー・マネージャー上のトピック・ツリー内の上位ノードに応じて決まります (『管理トピック・オブジェクト』を参照)。 このため、異なるキュー・マネージャーからパブリッシュまたはサブスクライブした場合の動作がそれぞれ異なることがあります。

クラスター全体にわたるパブリッシュ/サブスクライブの動作に直接関連する属性には、主として以下のものがあります。

CLROUTE

このパラメーターは、パブリッシャーが接続されているキュー・マネージャーと、一致するサブスクリプションが存在するキュー・マネージャーとの間のメッセージのルーティングを制御します。

- ルートは、それらのキュー・マネージャー間を直接接続する構成、またはクラスター・トピックの定義をホストするキュー・マネージャーを経由して接続する構成のどちらかにすることができます。 詳しくは、『パブリッシュ/サブスクライブ・クラスター』 を参照してください。
- **CLUSTER** パラメーターが設定されている間は、**CLROUTE** を変更できません。 **CLROUTE** を変更するには、まず **CLUSTER** プロパティーをブランクに設定します。 これにより、トピックを使用するアプリケーションのクラスター方式の動作が停止されます。 そうなると、今度は、サブスクリプションに対して配信されているパブリケーションが途絶えるため、変更を実行している間はパブリッシュ/サブスクライブ・メッセージングを静止する必要もあります。

PROXYSUB

このパラメーターは、プロキシー・サブスクリプションがいつ行われるかを制御します。

- FIRSTUSE がデフォルト値で、分散パブリッシュ/サブスクライブ・トポロジーにおいて、プロキシー・サブスクリプションが、キュー・マネージャーに対するローカル・サブスクリプションに応答して送信され、必要なくなった時点でキャンセルされます。 この属性をデフォルト値の FIRSTUSE から変更する必要のあるケースとその理由について詳しくは、個別プロキシー・サブスクリプション転送と全対象パブリッシュ を参照してください。
- 全対象パブリッシュ を有効にするには、上位レベルのトピック・オブジェクトに対して **PROXYSUB** パラメーターを FORCE に設定します。 この結果、トピック・ツリー内のこのトピック・オブジェクトの下のすべてのトピックに一致する単一ワイルドカードのプロキシー・サブスクリプションが得られます。

注 : PROXYSUB(FORCE) 属性を、大規模な、またはトラフィック量の多いパブリッシュ/サブスクライブ・クラスターで設定すると、システム・リソースに過大な負荷が掛かる可能性があります。 **PROXYSUB(FORCE)** 属性は、トピックが定義されたキュー・マネージャーだけでなく、すべてのキュー・マネージャーに伝搬されます。 そのため、クラスター内のすべてのキュー・マネージャーがワイルドカード・プロキシー・サブスクリプションを作成します。

クラスター内のいずれかのキュー・マネージャーからパブリッシュされたこのトピックに対するメッセージのコピーは、**CLROUTE** の設定に応じて、直接、またはトピック・ホスト・キュー・マネージャー経由で、クラスター内のすべてのキュー・マネージャーに送信されます。

トピックが直接ルーティングされる場合は、すべてのキュー・マネージャーが他のすべてのキュー・マネージャーに対してクラスター送信側チャネルを作成します。 トピックがトピック・ホスト・ルーテ

ィングされる場合は、クラスター内のすべてのキュー・マネージャーから各トピック・ホスト・キュー・マネージャーに対するチャネルが作成されます。

クラスター内で **PROXYSUB** パラメーターを使用するケースについて詳しくは、直接ルーティング型パブリッシュ/サブスクライブのパフォーマンスを参照してください。

PUBSCOBE および SUBSCOPE

これらのパラメーターにより、このキュー・マネージャーがパブリケーションを、トポロジー (パブリッシュ/サブスクライブ・クラスターまたは階層) 内のキュー・マネージャーに伝搬するか、あるいはその有効範囲をそのローカル・キュー・マネージャーのみに制限するかが決まります。
MQPMO_SCOPE_QMGR および MQSO_SCOPE_QMGR を使用すると、これに相当するジョブをプログラムで実行できます。

PUBSCOPE

PUBSCOPE(QMGR) を指定してクラスター・トピック・オブジェクトを定義した場合、定義がクラスターと共有されますが、そのトピックに基づくパブリケーションの有効範囲はローカルのみであり、それらはクラスター内の他のキュー・マネージャーへ送信されません。

SUBSCOPE

SUBSCOPE(QMGR) を指定してクラスター・トピック・オブジェクトを定義した場合、その定義はクラスターと共有されますが、そのトピックに基づくサブスクリプションの有効範囲はローカルのみであり、プロキシー・サブスクリプションはクラスター内の他のキュー・マネージャーに送信されません。

これらの 2 つの属性は普通、特定のトピックに関してクラスターの他のメンバーと対話しないように、キュー・マネージャーを分離させるために一緒に使用されます。 そのキュー・マネージャーは、それらのトピックに関するパブリケーションを、クラスターの他のメンバーへパブリッシュせず、またそれらのメンバーから受け取ることもありません。 トピック・オブジェクトがサブトピックに定義されている場合、この状態ではパブリケーションもサブスクリプションも妨げられることはありません。

トピックのローカル定義で **SUBSCOPE** を QMGR に設定しても、クラスター内の他のキュー・マネージャーは、**SUBSCOPE(ALL)** が設定されているクラスター・バージョンのトピックを使用している場合には、プロキシー・サブスクリプションをキュー・マネージャーに伝搬できなくなることはありません。 ただし、ローカル定義でも **PUBSCOPE** が QMGR に設定されていると、キュー・マネージャーからパブリケーションがそれらのプロキシー・サブスクリプションに送信されなくなります。

関連概念

パブリケーション有効範囲
サブスクリプション有効範囲

同じ名前の複数のクラスター・トピック定義

同じ名前が付いたクラスター・トピック・オブジェクトをクラスター内の複数のキュー・マネージャーで定義することができます。 シナリオによっては、これによって可能になる特定の動作があります。 同じ名前のクラスター・トピック定義が複数存在する場合、プロパティーの大多数が一致するはずです。 一致しない場合、不一致の重要度に応じてエラーまたは警告が報告されます。

一般に、複数のクラスター・トピック定義のプロパティーに不一致が存在する場合には警告が出され、トピック・オブジェクト定義の 1 つがクラスター内の各キュー・マネージャーによって使用されます。 各キュー・マネージャーでどの定義が使用されるかは決定的ではなく、クラスター内の複数のキュー・マネージャー間で一貫していません。 そのような不一致は可能な限り早く解決する必要があります。

クラスターのセットアップまたは保守の際、同一ではない複数のクラスター・トピック定義の作成が必要になる場合があります。 ただし、これは一時的な手段でしかなく、潜在的なエラー状態として扱われます。

不一致が検出されると、以下の警告メッセージが各キュー・マネージャーのエラー・ログに書き込まれます。

- Multi マルチプラットフォーム の場合: AMQ9465 および AMQ9466。
- z/OS z/OS の場合: CSQX465I および CSQX466I。

各キュー・マネージャー上の任意のトピック・ストリングに対して選択されたプロパティーは、トピック・オブジェクト定義ではなく、トピック状況を表示することによって判別できます。例えば、 **DISPLAY TPSTATUS** を使用します。

状況によっては、構成プロパティーの競合が重大であるために、トピック・オブジェクトの作成が停止したり、一致しないオブジェクトが無効としてマークされてクラスター全体に伝搬されないことがあります (DISPLAY TOPIC の「 **CLSTATE** 」を参照してください)。 このような状況は、トピック定義のクラスター・ルーティング・プロパティー (**CLROUTE**) に競合が存在する場合に発生します。 また、トピック・ホスト・ルーティング定義間の整合性が重要であるため、不整合がさらに生じると拒否されます。 これについては、この記事の後続のセクションで詳しく説明します。

オブジェクト定義の際に競合が検出されると、構成変更は拒否されます。 フル・リポジトリー・キュー・マネージャーによって後で検出される場合、以下の警告メッセージがキュー・マネージャーのエラー・ログに書き込まれます。

- Multi マルチプラットフォーム の場合: AMQ9879
- z/OS z/OS の場合: CSQX879E

同じトピック・オブジェクトの複数の定義がクラスターで定義される場合、ローカルの定義がリモートの定義よりも優先されます。 そのため、定義に違いがあると、複数の定義をホストするキュー・マネージャーは互いに異なる動作をします。

別のキュー・マネージャーのクラスター・トピックと同じ名前で非クラスター・トピックを定義する効果

クラスター内のキュー・マネージャーでクラスター化されていない管理対象トピック・オブジェクトを定義しつつ、同時にクラスター・トピック定義と同じ名前のトピック・オブジェクトを異なるキュー・マネージャーで定義することができます。 その場合、ローカル定義のトピック・オブジェクトが、同じ名前のリモート定義すべてより優先されます。

これには、このキュー・マネージャーから使用されるときにトピックのクラスター化の動作を防止する効果があります。 つまり、サブスクリプションはリモート・パブリッシャーからパブリケーションを受け取らない可能性があり、パブリッシャーからのメッセージがクラスター内のリモート・サブスクリプションに伝搬されない可能性があります。

そのようなシステムを構成する場合は、事前に注意深く検討する必要があります。 これによって動作に混乱が生じる可能性があるからです。

注 : 個々のキュー・マネージャーでパブリケーションおよびサブスクリプションがクラスター全体に伝搬されないようにする必要がある場合、トピックがどこか他の場所でクラスター化されている場合でも、代わりの方法としてパブリケーションとサブスクリプションの有効範囲をローカル・キュー・マネージャーのみに設定することができます。 98 ページの『クラスター・トピックの属性』を参照。

直接ルーティングされるクラスター内の複数のクラスター・トピック定義

直接ルーティングの場合、通常は複数のクラスター・キュー・マネージャーに同じクラスター・トピックを定義することはありません。 これは、トピックがどのキュー・マネージャーで定義されたかにかかわらず、直接ルーティングによってクラスター内のすべてのキュー・マネージャーでトピックが使用可能になるためです。 さらに、複数のクラスター・トピック定義を追加すると、システム・アクティビティーと管理の複雑さが大幅に増加し、複雑さが増すと人的なエラーが発生する確率が高くなります。

- それぞれの定義により、追加のクラスター・トピック・オブジェクトがクラスター内の他のキュー・マネージャー (他のクラスター・トピック・ホスト・キュー・マネージャーを含む) にプッシュされます。
- クラスター内では特定のトピックに関するすべての定義が同じでなければなりません。そうでない場合、どのトピック定義がキュー・マネージャーで使用されるかを確定することが困難になります。

また、トピックがクラスター全体で正しく機能するために、単一のホスト・キュー・マネージャーが常時使用可能である必要はありません。 それは、クラスター・トピック定義が、フル・リポジトリー・キュー・マネージャー、および部分クラスター・リポジトリーの他のすべてのキュー・マネージャーによってキャ

ッシュされるためです。 詳しくは、直接ルーティングを使用するトピック・ホスト・キュー・マネージャーの可用性を参照してください。

クラスター・トピックを 2 番目のキュー・マネージャーに一時的に定義することが必要になる可能性がある場合は (例えば、トピックの既存のホストがクラスターから除去されることになっているなど)、別のキュー・マネージャーへのクラスター・トピック定義の移動を参照してください。

クラスター・トピック定義を変更する必要がある場合、定義したのと同じキュー・マネージャーで慎重に変更を行ってください。 別のキュー・マネージャーから変更を試行すると、競合するトピック属性を持つトピックの 2 番目の定義が作成されてしまう可能性があります。

トピック・ホスト・ルーティングされるクラスター内の複数のクラスター・トピック定義

クラスター・トピックがトピック・ホスト のクラスター経路で定義される場合、直接 ルーティングされるトピックの場合と同じように、トピックはクラスター内のすべてのキュー・マネージャー間に伝搬されます。 また、そのトピックのパブリッシュ/サブスクライブ・メッセージングはすべて、トピックが定義されているキュー・マネージャー経由でルーティングされます。 したがって、クラスター内のトピックの定義の場所と数が重要になります (82 ページの『パブリッシュ/サブスクライブ・クラスターでのトピック・ホスト・ルーティング』 を参照)。

十分な可用性と拡張容易性を確保するために、可能であれば、複数のトピック定義を用意するのが適切です。 トピック・ホスト・ルーティングを使用するトピック・ホスト・キュー・マネージャーの可用性を参照してください。

クラスター内でトピック・ホスト・ルーティングされるトピックの追加定義を追加または削除する際、構成変更時のメッセージのフローを検討する必要があります。 変更時にクラスター内のメッセージがトピックにパブリッシュされる場合、トピック定義の追加または除去には段階的なプロセスが必要です。 別のキュー・マネージャーへのクラスター・トピック定義の移動およびトピック・ホスト・ルーティングされるクラスターへのトピック・ホストの追加を参照してください。

前述したとおり、複数定義のプロパティーは一致している必要があります (ただし、**PUB** パラメーターの場合は例外となる可能性があり、これについては次のセクションで説明します)。 パブリケーションをトピック・ホスト・キュー・マネージャー経由でルーティングされる場合には、複数定義が整合していることがなお一層重要になります。 したがって、1 つ以上のトピック定義がトピック・ホスト・クラスター・ルーティング用に構成されている場合、トピック・ストリングかクラスター名のいずれかで不整合が検出されると、拒否されます。

注 : 既存のクラスター・トピック定義がトピック・ホスト・ルーティング用に構成されているトピック・ツリーにおいて、クラスター・トピック定義を別のトピックの上または下で構成しようとする場合にも、それらの定義は拒否されます。 これにより、ワイルドカード付きサブスクリプションに関してパブリケーションのルーティングがあいまいになることが防止されます。

PUB パラメーターの特別な処理

PUB パラメーターは、アプリケーションによるトピックへのパブリッシュのタイミングを制御するために使用します。 クラスター内のトピック・ホスト・ルーティングの場合、パブリケーションのルーティングに使用するトピック・ホスト・キュー・マネージャーの制御も行えます。 そのため、PUB パラメーターの設定が異なる、同じトピック・オブジェクトの複数定義をクラスター内に置くことが許可されています。

このパラメーターの設定がトピックの複数のリモート・クラスター定義で異なる場合、以下の条件が満たされていれば、トピックによってパブリケーションがサブスクリプションに送信および送達できるようになります。

- パブリッシャーが接続されており、PUB(DISABLED) に設定されているキュー・マネージャーで、一致するトピック・オブジェクトが定義されていない。
- クラスター内の複数トピック定義の 1 つ以上が PUB(ENABLED) に設定されているか、または複数トピック定義の 1 つ以上が PUB(ASPARENT) に設定されており、パブリッシャーが接続されてサブスクリプションが定義されているローカル・キュー・マネージャーがトピック・ツリー内のより高いポイントで PUB(ENABLED) に設定されている。

トピック・ホスト・ルーティングでは、トピック・ホストではないキュー・マネージャーに接続されているアプリケーションによってメッセージがパブリッシュされると、**PUB** パラメーターが明示的に DISABLED に設定されていないキュー・マネージャーをホストするトピックにのみ、メッセージがルーティングされます。 そのため、PUB(DISABLED) 設定を使用することで、特定のトピック・ホストを経由するメッセージ・トラフィックを静止することができます。 キュー・マネージャーの保守や除去の準備のため、またはトピック・ホスト・ルーティングされるクラスターへのトピック・ホストの追加で説明されている理由でこれを行うこともできます。

クラスター・トピック・ホスト・キュー・マネージャーの可用性
トピック・ホスト・キュー・マネージャーが使用不可になった場合にトピックのトラフィックをクラスターで処理できなくなるリスクを最小限に抑えるために、パブリッシュ/サブスクライブ・クラスターを設計します。 トピック・ホスト・キュー・マネージャーが使用不可になった場合の影響は、トピック・ホスト・ルーティングと直接ルーティングのどちらをクラスターで使用しているかに応じて異なります。

直接ルーティングを使用するトピック・ホスト・キュー・マネージャーの可用性

直接ルーティングの場合、通常は複数のクラスター・キュー・マネージャーに同じクラスター・トピックを定義することはありません。 これは、トピックがどのキュー・マネージャーで定義されたかにかかわらず、直接ルーティングによってクラスター内のすべてのキュー・マネージャーでトピックが使用可能になるためです。直接ルーティングされるクラスター内の複数のクラスター・トピック定義を参照してください。

クラスター内では、クラスター化オブジェクト (クラスター化されたキュー、クラスター化されたトピックなど) のホストが長時間にわたって使用不可になった場合、クラスター内のその他のメンバーはこれらのオブジェクトに関する情報をやがて失います。 クラスター化されたトピックの場合、クラスター・トピック・ホスト・キュー・マネージャーが使用不可になると、その他のキュー・マネージャーはそのトピックに関するパブリッシュ/サブスクライブ要求を直接クラスター化という方法で処理します (つまりリモート・キュー・マネージャーでサブスクリプションにパブリケーションを送信します)。トピック・ホスト・キュー・マネージャーがフル・リポジトリー・キュー・マネージャーと最後に通信した時点から少なくとも 60 日間にわたって、このような処理を継続します。 クラスター・トピック・オブジェクトを定義したキュー・マネージャーがそれ以降も使用可能にならなかった場合、その他のキュー・マネージャー上のキャッシュに入ったトピック・オブジェクトはやがて削除され、トピックはローカル・トピックに戻ります。その場合、サブスクリプションは、リモート・キュー・マネージャーに接続されたアプリケーションからパブリケーションを受け取らなくなります。

クラスター・トピック・オブジェクトを定義したキュー・マネージャーのリカバリー期間は 60 日あるため、クラスター・トピック・ホストの可用性を保障するために特別な措置を講じる必要はほとんどありません (ただし、使用不可になったクラスター・トピック・ホストで定義されたサブスクリプションはもはや使用可能でないことに注意してください)。 60 日間という期間は、技術上の問題に対処するのに十分な期間です。その期間を超過する理由として考えられるのは、管理上の誤りのみです。 そのような誤りの可能性を軽減するために、クラスター・トピック・ホストが使用不可になった場合は、クラスターのすべてのメンバーが、キャッシュされたクラスター・トピック・オブジェクトがリフレッシュされなかったことを報告するエラー・ログ・メッセージを 1 時間ごとに書き出すようにしてください。 これらのメッセージに対する対応として、クラスター・トピック・オブジェクトが定義されているキュー・マネージャーが稼働中であることを確認します。 クラスター・トピック・ホスト・キュー・マネージャーを再び使用可能にすることができない場合は、正確に同じ属性を使用して同じクラスター化トピック定義をクラスター内の別のキュー・マネージャー上で定義してください。

トピック・ホスト・ルーティングを使用するトピック・ホスト・キュー・マネージャーの可用性

トピック・ホスト・ルーティングの場合、トピックに関するパブリッシュ/サブスクライブ・メッセージングはすべて、そのトピックが定義されているキュー・マネージャー経由でルーティングされます。 このため、クラスター内のこのようなキュー・マネージャーの継続的な可用性を考慮することが非常に重要です。あるトピック・ホストが使用不可になり、しかもそのトピックのホストが他に存在しない場合には、パブリッシャーからクラスター内のさまざまなキュー・マネージャー上のサブスクライバーに向かうトラフィックは、そのトピックに関して直ちに停止します。 追加のトピック・ホストが使用可能である場合、クラ

スター・キュー・マネージャーはこれらのトピック・ホストを介して新しいパブリケーション・トラフィックをルーティングします。これにより、メッセージ経路が継続的に確保されます。

直接トピックの場合、60 日が経過した後も最初のトピック・ホストがまだ使用不可であれば、そのトピック・ホストのトピックに関する情報がクラスターから除去されます。 これがクラスター内でこのトピックに関する残存する最後の定義である場合、他のすべてのキュー・マネージャーは、ルーティングのためにトピック・ホストにパブリケーションを転送する操作を停止します。

したがって、十分な可用性と拡張容易性を確保するために、可能であれば、各トピックを少なくとも 2 つのクラスター・キュー・マネージャーで定義するのが適切です。 これにより、いずれかのトピック・ホスト・キュー・マネージャーが使用不可になっても保護されます。 トピック・ホスト・ルーティングされるクラスター内の複数のクラスター・トピック定義も参照してください。

複数のトピック・ホストを構成できない (メッセージ順序を保持する必要がある場合など) 状況で、構成済みトピック・ホストが 1 つだけでは困る場合 (1 つのキュー・マネージャーの可用性がクラスター内のすべてのキュー・マネージャーにおけるサブスクリプションへのパブリケーション・フローに影響があってはならない場合)、トピックを直接ルーティング・トピックとして構成することを考慮してください。 これにより、クラスター全体で 1 つのキュー・マネージャーに依存することを防止できますが、ローカル・ホストされるサブスクリプションおよびパブリッシャーを処理するために個々のキュー・マネージャーは引き続き可用性を保つ必要があります。

クラスター化されたパブリッシュ/サブスクライブの禁止

クラスター内に最初の直接ルーティング型クラスター・トピックを導入すると、クラスター内のすべてのキュー・マネージャーが他のキュー・マネージャーを認識するようになるとともに、相互にチャネルを作成する可能性もあります。 これが望ましくない場合は、代わりにトピック・ホスト・ルーティング型のパブリッシュ/サブスクライブを構成してください。 各キュー・マネージャーのスケーリング上の懸念のため、直接ルーティング・クラスター・トピックが存在するとクラスターの安定性が損なわれる恐れがある場合は、クラスター化されたパブリッシュ/サブスクライブの機能を完全に無効にすることができます。それには、クラスター内のすべてのキュー・マネージャーで **PSCLUS** を DISABLED に設定します。

78 ページの『パブリッシュ/サブスクライブ・クラスターでの直接ルーティング』に説明があるとおり、クラスターに直接ルーティング・クラスター・トピックを導入すると、すべての部分リポジトリーに対してクラスター内の他のすべてのメンバーが自動的に通知されます。 また、クラスター・トピックによって他のすべてのノードにサブスクリプションが作成され (例えば **PROXYSUB(FORCE)** が指定されている場合)、ローカル・サブスクリプションが存在しないときでもキュー・マネージャーから多数のチャネルが開始されてしまう場合があります。 この場合、クラスター内の各キュー・マネージャーにすぐに追加の負荷が加わります。 キュー・マネージャーが多数あるクラスターの場合、これによってパフォーマンスが大幅に低下する場合があります。 そのため、クラスターへの直接ルーティング型パブリッシュ/サブスクライブの導入は、注意深く計画する必要があります。

直接ルーティング型パブリッシュ/サブスクライブのオーバーヘッドにクラスターが対応できないことが分かっている場合は、代わりにトピック・ホスト・ルーティング型パブリッシュ/サブスクライブを使用できます。 違いの概要については、75 ページの『パブリッシュ/サブスクライブ・クラスターの設計』を参照してください。

クラスターのパブリッシュ/サブスクライブ機能を完全に無効にしたほうがよい場合は、クラスター内のすべてのキュー・マネージャーでキュー・マネージャー属性 **PSCLUS** を DISABLED に設定することができます。 この設定にすると、キュー・マネージャー機能に関する次の 3 つの側面が変更されて、クラスター内の直接ルーティング型とトピック・ホスト・ルーティング型のパブリッシュ/サブスクライブはどちらも無効になります。

- このキュー・マネージャーの管理者は、 Topic オブジェクトをクラスターとして定義できなくなりました。
- 他のキュー・マネージャーから着信するトピック定義やプロキシー・サブスクリプションは拒否され、構成が正しくないことを管理者に通知する警告メッセージがログに記録されます。
- 完全リポジトリーは、トピック定義を受け取るときに、すべてのキュー・マネージャーに関する情報を他のすべての部分リポジトリーと自動共有することがなくなりました。

PSCLUS はクラスター内の個々のキュー・マネージャーのパラメーターではあっても、クラスター内のキュー・マネージャーのサブセットでパブリッシュ/サブスクライブを選択的に無効にすることは意図されてい

ません。 そのようにして選択的に無効にすると、頻繁にエラー・メッセージが表示されるようになります。これは、プロキシー・サブスクリプションとトピック定義が絶えず参照され、**PSCLUS** が有効になっているキュー・マネージャーでトピックがクラスター化されている場合に拒否されるからです。

したがって、クラスター内のすべてのキュー・マネージャーで **PSCLUS** を DISABLED に設定するようにしてください。 ただし、実際には、この状態を達成し、維持するのは困難なことがあります。例えば、キュー・マネージャーがいつでもクラスターに参加したり、クラスターから退出したりできます。 最低限、すべてのフル・リポジトリー・キュー・マネージャーでは確実に **PSCLUS** を DISABLED に設定する必要があります。 そうしておけば、その後、クラスター内で ENABLED になっているキュー・マネージャーでクラスター・トピックが定義されたとしても、フル・リポジトリーがすべてのキュー・マネージャーに対して他のすべてのキュー・マネージャーを通知することはないため、すべてのキュー・マネージャー全体についてクラスターでスケーリングの問題が起きないように保護されます。 このシナリオでは、クラスター・トピックの発信元が、フル・リポジトリー・キュー・マネージャーのエラー・ログに報告されます。

ある 1 つのキュー・マネージャーが 1 つ以上のパブリッシュ/サブスクライブ・クラスターに参加しており、同時に 1 つ以上の Point-to-Point クラスターにも参加している場合は、そのキュー・マネージャーで **PSCLUS** を ENABLED に設定する必要があります。 この理由のため、Point-to-Point クラスターとパブリッシュ/サブスクライブ・クラスターがオーバーラップする場合には、各クラスター内でフル・リポジトリーの別々のセットを使用するようにしてください。 この方法により、トピック定義とすべてのキュー・マネージャーに関する情報がパブリッシュ/サブスクライブ・クラスター内でのみ流れるようにできます。

構成が不整合にならないようにするために、**PSCLUS** を ENABLED から DISABLED に変更する際は、そのキュー・マネージャーがメンバーであるどのクラスターにもクラスター・トピック・オブジェクトは存在できません。 このようなトピックは、リモート定義されたものであっても、**PSCLUS** を DISABLED に変更する前に削除しておく必要があります。

PSCLUS の詳細については、ALTER QMGR (PSCLUS) を参照してください。

関連概念

直接ルーティング型パブリッシュ/サブスクライブ・クラスターのパフォーマンス

パブリッシュ/サブスクライブと複数クラスター

1 つのキュー・マネージャーは、複数のクラスターのメンバーになることができます。 このような配置のことを、オーバーラップするクラスター と言います。 このようなオーバーラップにより、クラスター・キューに複数のクラスターからアクセスできるようになるため、Point-to-Point メッセージ・トラフィックを 1 つのクラスターのキュー・マネージャーから別のクラスターのキュー・マネージャーにルーティングできるようになります。 パブリッシュ/サブスクライブ・クラスターのクラスター・トピックは、これと同じ機能を提供してはいません。 そのため、複数のクラスターを使用する場合は、クラスターの動作を明確に理解する必要があります。

キューの場合とは異なり、1 つのトピック定義を複数のクラスターに関連付けることはできません。 クラスター・トピックの有効範囲は、トピックが定義されているのと同じクラスター内のキュー・マネージャーに限定されます。 このため、パブリケーションは、同じクラスター内のキュー・マネージャーのサブスクリプションにのみ伝搬されます。

キュー・マネージャーのトピック・ツリー

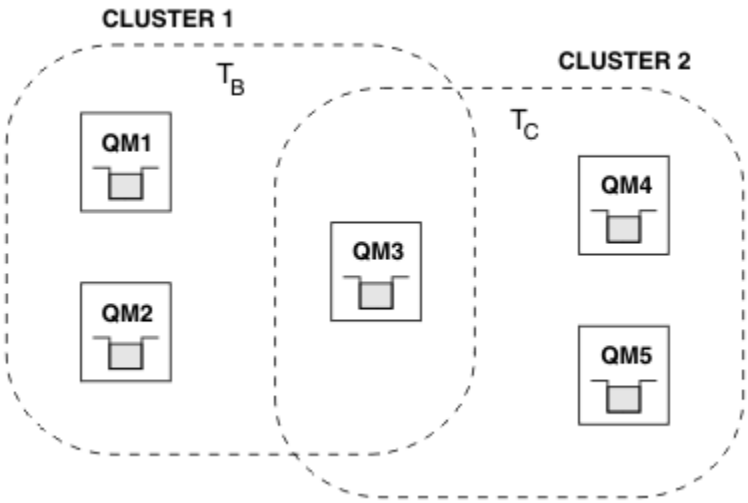


図 *28.* オーバーラップするクラスター*:* それぞれ異なるトピックにサブスクライブしている *2* つのクラスター

あるキュー・マネージャーが複数のクラスターのメンバーである場合、そのキュー・マネージャーは各クラスター内で定義されたすべてのクラスター・トピックを認識します。 例えば、前の図 QM3 では、T_B と T_C の両方の管理対象クラスター・トピック・オブジェクトが認識されていますが、QM1 は T_B のみを認識しています。 QM3 では、トピック定義の両方をそのローカル・トピックに適用するので、特定のトピックについては、QM1 に対して異なる動作をします。 この理由で、異なるクラスターからのクラスター・トピックが相互に干渉しないようにすることが重要です。 干渉が発生するのは、1 つのクラスター・トピックが異なるクラスター内の別のクラスター・トピックの上位または下位で定義された場合 (例えば、トピック・ストリング /Sport および /Sport/Football が定義された場合) や、さらには両方で同じトピック・ストリングが定義された場合です。 別の形式の干渉は、異なるクラスター内に同じオブジェクト名で管理用のクラスター・トピック・オブジェクトが定義され、それぞれが異なるトピック・ストリングを持つ場合に発生します。

このような構成が行われた場合、一致するサブスクリプションへのパブリケーションの配信は、クラスターから見たパブリッシャーおよびサブスクライバーの相対的な位置に大きく依存することになります。 この理由から、この構成を信頼することはできないため、構成を変更し、干渉しているトピックを除去する必要があります。

パブリッシュ/サブスクライブ・メッセージングでオーバーラップするクラスターのトポロジーを計画している場合、トピック・ツリーおよびクラスター・トピック・オブジェクト名がトポロジー内でオーバーラップするすべてのクラスターにまたがるものと想定して取り扱うことにより、干渉を避けることができます。

複数のパブリッシュ/サブスクライブ・クラスターの統合

パブリッシュ/サブスクライブ・メッセージングを異なるクラスター内のキュー・マネージャーにまたがるように構成する必要がある場合は、次の 2 つのオプションがあります。

- パブリッシュ/サブスクライブ階層構成を使用することにより、クラスターを一緒に接続します。 複数のクラスターのトピック・スペースの結合を参照してください。
- 既存のクラスターと重なり合い、特定のトピックに対するパブリッシュまたはサブスクライブが必要なすべてのキュー・マネージャーが含まれる追加のクラスターを作成します。

後者のオプションを使用する場合は、クラスターのサイズおよび効率の良いクラスター・ルーティング機構を注意深く検討する必要があります。 『75 ページの『パブリッシュ/サブスクライブ・クラスターの設計』』を参照してください。

パブリッシュ/サブスクライブ・クラスターでの保存パブリケーションに関する設計上の考慮事項

パブリッシュ/サブスクライブ・クラスターでの保存パブリケーションの処理を設計する場合、考慮すべき制約事項がいくつかあります。

考慮事項

考慮事項 *1:* 次のクラスター・キュー・マネージャーには、常に最新バージョンの保存パブリケーションが格納されます。

- パブリッシャーのキュー・マネージャー
- トピック・ホスト経路指定クラスター内のトピック・ホスト (この記事の次のセクションで説明されているように、トピックに対してトピック・ホストが 1 つのみの場合)
- 保存パブリケーションのトピック・ストリングに一致するサブスクリプションのあるすべてのキュー・マネージャー

考慮事項 *2:* キュー・マネージャーは、サブスクリプションがない場合は更新された保存パブリケーションを受信しません。 したがって、既にトピックをサブスクライブしていないキュー・マネージャーに格納されている保存パブリケーションは不整合になります。

考慮事項 *3:* サブスクリプションの作成時に、トピック・ストリングに関して保存パブリケーションのローカル・コピーがある場合、ローカル・コピーがサブスクリプションに対して送信されます。 あるトピック・ストリングの最初のサブスクライバーになった場合、一致する保存パブリケーションが次のクラスター・メンバーのいずれかからも送信されます。

- 直接経路指定クラスター内のパブリッシャーのキュー・マネージャー
- トピック・ホスト経路指定クラスター内の該当トピックのトピック・ホスト

トピック・ホスト、またはサブスクライブするキュー・マネージャーにパブリッシュするキュー・マネージャーからの保存パブリケーションの送達は、MQSUB 呼び出しとは非同期です。 したがって、MQSUBRQ 呼び出しを使用する場合、後続の MQSUBRQ が呼び出されるまで、保存パブリケーションは最新ではない可能性があります。

影響

パブリッシュ/サブスクライブ・クラスターでは、最初のサブスクリプションが実行されるときに、不整合な保存パブリケーションがローカル・キュー・マネージャーに格納されている可能性があり、これが新規サブスクリプションに対して送信されます。 ローカル・キュー・マネージャーにサブスクリプションが存在する場合、保存パブリケーションが次回更新されるときにこの問題が解決されます。

トピック・ホスト経路指定パブリッシュ/サブスクライブ・クラスターの場合、あるトピックについて複数のトピック・ホストを構成すると、新規サブスクライバーは、トピック・ホストから最新の保存パブリケーションを受け取ることもあれば、最新バージョンが失われているために別のトピック・ホストから不整合な保存パブリケーションを受け取ることもあります。 トピック・ホスト・ルーティングでは、通常はあるトピックについて複数のトピック・ホストを構成します。 ただし、アプリケーションで保存パブリケーションを使用する場合は、トピックごとに 1 つのトピック・ホストのみを構成する必要があります。

トピック・ストリングについては、単一のパブリッシャーを使用する必要があり、パブリッシャーが常に同じキュー・マネージャーを使用することを確認してください。 これをしない場合、同じトピックに関して異なる複数のキュー・マネージャーでそれぞれ異なる保存パブリケーションがアクティブになり、予期しない動作が発生する可能性があります。 複数のプロキシー・サブスクリプションが配布され、複数の保存パブリケーションを受け取ることになる場合があります。

サブスクライバーが不整合なパブリケーションを使用することが心配な場合は、各保存パブリケーションの作成時に、メッセージの有効期限を設定することを考慮してください。

CLEAR TOPICSTR コマンドを使用すると、パブリッシュ/サブスクライブ・クラスターから保存パブリケーションを削除できます。 特定の環境下では、パブリッシュ/サブスクライブ・クラスターの複数のメンバーにコマンドを発行する必要がある場合があります (**CLEAR TOPICSTR** を参照してください)。

ワイルドカード・サブスクリプションと保存パブリケーション

ワイルドカード・サブスクリプションを使用する場合、パブリッシュ/サブスクライブ・クラスターの他のメンバーに送信される対応プロキシー・サブスクリプションは、最初のワイルドカード文字の直前のトピック分離文字からワイルドカードにします。 『ワイルドカードとクラスター・トピック』を参照してください。

したがって、使用されるワイルドカードによって、サブスクライブ・アプリケーションで一致するよりも多くのトピック・ストリングと保存パブリケーションに一致することがあります。

保存パブリケーションに必要なストレージの量が増えるので、ホスティング・キュー・マネージャーに十分なストレージ容量があることを確認する必要があります。

関連概念

保存パブリケーション

個々のプロキシー・サブスクリプション転送と全対象パブリッシュ

パブリッシュ/サブスクライブ・クラスターの *REFRESH CLUSTER* についての考慮事項

REFRESH CLUSTER コマンドを発行すると、キュー・マネージャーで、ローカルに保持されているクラスター情報 (クラスター・トピックおよびそれらが関連付けられているプロキシー・サブスクリプションを含む) が、当面の間、破棄されることになります。

REFRESH CLUSTER コマンドが発行されてから、クラスター・パブリッシュ/サブスクライブに関する必要な全情報をキュー・マネージャーが再び取得する時点までに要する時間は、クラスターの規模、可用性、および完全リポジトリー・キュー・マネージャーの応答性によって異なります。

リフレッシュ処理の間、パブリッシュ/サブスクライブ・クラスター内のパブリッシュ/サブスクライブ・トラフィックで障害が発生します。 大規模クラスターでは、**REFRESH CLUSTER** コマンドを使用すると、処理中のクラスターが中断される可能性があります。その後、クラスター・オブジェクトが 27 日間隔で対象のキュー・マネージャーすべてに状況の更新を自動的に送信する際にも同様のことが起こり得ます。 大規模クラスターでのリフレッシュはクラスターのパフォーマンスと可用性に影響を与える可能性があるを参照してください。 これらの理由から、'**REFRESH CLUSTER** コマンドは、'IBM サポートの指導の下でのみパブリッシュ／サブスクライブ・クラスターで使用されなければならない。

クラスターへの悪影響は以下の症状として表面化することがあります。

- あるキュー・マネージャーのクラスター・トピックに対するサブスクリプションで、クラスター内の他のキュー・マネージャーに接続されているパブリッシャーからのパブリケーションを受け取っていません。
- あるキュー・マネージャーのクラスター・トピックにパブリッシュされたメッセージが、他のキュー・マネージャーのサブスクリプションに伝搬されません。
- この期間に作成されたキュー・マネージャーのクラスター・トピックに対するサブスクリプションで、クラスターの他のメンバーにプロキシー・サブスクリプションが一貫して送信されません。
- この期間に削除されたキュー・マネージャーのクラスター・トピックに対するサブスクリプションで、クラスターの他のメンバーからプロキシー・サブスクリプションが一貫して削除されません。
- メッセージ送信が 10 秒以上一時停止します。
- **MQPUT** の失敗。例えば、MQRC_PUBLICATION_FAILURE です。
- パブリケーションが MQRC_UNKNOWN_REMOTE_Q_MGR の理由で送達不能キューに置かれます。

このような理由から、**REFRESH CLUSTER** コマンドを発行する前にパブリッシュ/サブスクライブ・アプリケーションを静止させる必要があります。

パブリッシュ/サブスクライブ・クラスター内のキュー・マネージャーに対して **REFRESH CLUSTER** コマンドを発行した後、すべてのクラスター・キュー・マネージャーとクラスター・トピックが正常にリフレッシュされるまで待ち、プロキシー・サブスクリプションの再同期の説明に従ってプロキシー・サブスクリプションを再同期してください。 すべてのプロキシー・サブスクリプションが正しく再同期されてから、パブリッシュ/サブスクライブ・アプリケーションを再始動してください。

REFRESH CLUSTER コマンドが完了するのに長い時間がかかっている場合は、SYSTEM.CLUSTER.COMMAND.QUEUE の CURDEPTH を調べてコマンドをモニターしてください。

関連概念

69 ページの『クラスター化: REFRESH CLUSTER の使用に関するベスト・プラクティス』

REFRESH CLUSTER コマンドを使用して、クラスターに関するローカルに保持されているすべての情報を破棄し、クラスターの完全リポジトリーからその情報を再作成します。 例外的な状況を除き、このコマンドを使用する必要はありません。 このコマンドを使用する必要がある場合は、使用方法に関する特別な考慮事項があります。 この情報は、テストおよびお客様からのフィードバックに基づく指針を示すものです。

関連資料

REFRESH CLUSTER の実行中に発生するアプリケーションの問題

MQSC コマンドのリファレンス: REFRESH CLUSTER

パブリッシュ/サブスクライブ階層でのルーティング

分散キュー・マネージャー・トポロジーとしてパブリッシュ/サブスクライブ階層を使用している場合、あるキュー・マネージャーでサブスクリプションがなされると、デフォルトでは、階層内のすべてのキュー・マネージャーのそれぞれにおいて、プロキシー・サブスクリプションが作成されます。 いずれかのキュー・マネージャーで受信するパブリケーションは、階層内をルーティングされて、一致するサブスクリプションをホストする各キュー・マネージャーに届けられます。

パブリッシュ/サブスクライブ階層およびクラスター内のキュー・マネージャー間でメッセージがルーティングされる方法については、『分散パブリッシュ/サブスクライブのネットワーク』を参照してください。

分散パブリッシュ/サブスクライブ階層内のキュー・マネージャーでトピックのサブスクリプションがなされると、キュー・マネージャーは、接続されているキュー・マネージャーへサブスクリプションを伝搬するプロセスを管理します。 プロキシー・サブスクリプション はネットワーク内のすべてのキュー・マネージャーに流れます。 プロキシー・サブスクリプションは、キュー・マネージャーに対して、そのトピックのサブスクリプションをホストするキュー・マネージャーにパブリケーションを転送するために必要な情報を提供します。 パブリッシュ/サブスクライブ階層内の各キュー・マネージャーで認識しているのは、その直接的関係のみです。 1 つのキュー・マネージャーに入れられたパブリケーションは、その直接的関係により、サブスクリプション側キュー・マネージャーに送信されます。 これを次の図に示します。 サブスクライバー *1* は、*Asia* キュー・マネージャー上で特定のトピックのサブスクリプションを登録します (1)。 *Asia* キュー・マネージャー上でのこのサブスクリプションのプロキシー・サブスクリプションは、ネットワーク内の他のすべてのキュー・マネージャーに転送されます (2,3,4)。

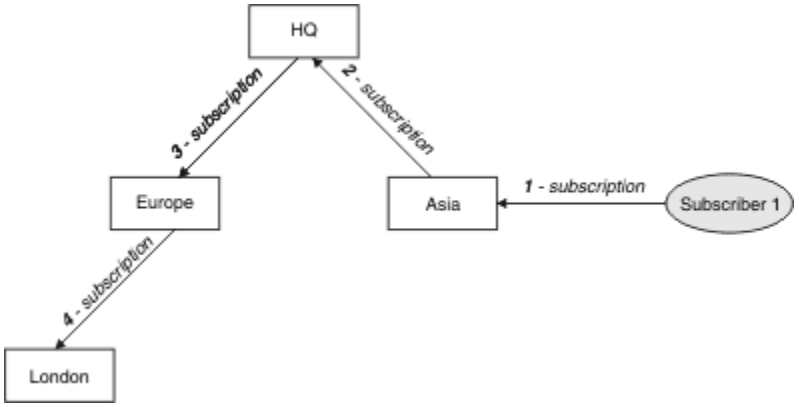


図 *29.* キュー・マネージャー・ネットワークを介したサブスクリプションの伝搬

キュー・マネージャーは、そこで作成されるすべてのサブスクリプションを、それがローカル・アプリケーションからかリモート・キュー・マネージャーからかに関係なく統合します。 プロキシー・サブスクリプションが既に存在しているのでない限り、サブスクリプションのトピックに対するプロキシー・サブスクリプションを、近隣との間で作成します。 これを次の図に示します。 サブスクライバー *2* は、108 ページの図 29 の場合と同じトピックのサブスクリプションを、*HQ* キュー・マネージャー上で登録します (5)。 このトピックに対するサブスクリプションは *Asia* キュー・マネージャーに転送されるので、ネットワーク上の他の場所にもそれらのサブスクリプションが存在することが分かります (6)。 サブスクリプションは *Europe* キュー・マネージャーには転送されません。これは、このトピックに対するサブスクリプションが既に登録されているためです。108 ページの図 29 のステップ 3 を参照してください。

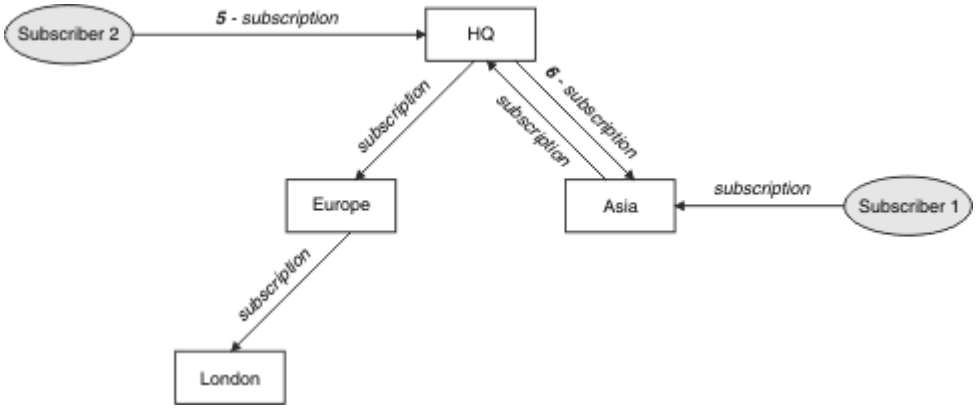


図 *30.* 複数のサブスクリプション

アプリケーションがトピックに対して情報をパブリッシュすると、デフォルトでは、それは受信側キュー・マネージャーにより、そのトピックに対する有効なサブスクリプションのあるすべてのキュー・マネージャーに転送されます。 その転送では、1 つ以上の中間キュー・マネージャーを経由する可能性があります。これを次の図に示します。 パブリッシャーは、109 ページの図 30 の場合と同じトピックについて、パブリケーションを *Europe* キュー・マネージャーに送信します (7)。 このトピックに対するサブスクリプションが *HQ* から *Europe* に対して存在するため、パブリケーションが *HQ* キュー・マネージャーへ転送されます (8)。 しかし、*London* から *Europe* に対してサブスクリプションが存在しない (*Europe* から *London* に対してのみ) ため、*London* キュー・マネージャーにはパブリケーションは転送されません。 *HQ* キュー・マネージャーはパブリケーションをサブスクライバー *2* および *Asia* キュー・マネージャーへ直接送信します (9)。 パブリケーションが *Asia* からサブスクライバー *1* へ転送されます (10)。

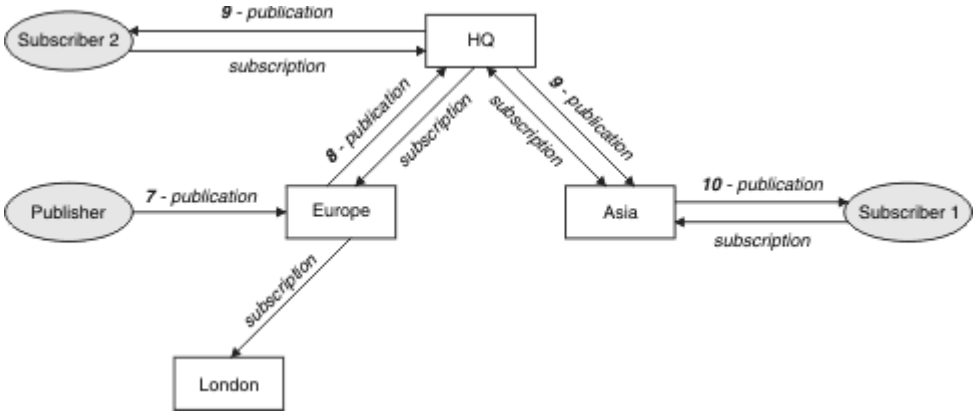


図 *31.* キュー・マネージャー・ネットワークを介したパブリケーションの伝搬

キュー・マネージャーは、パブリケーションまたはサブスクリプションを別のキュー・マネージャーへ送信する際に、メッセージにそれ自体のユーザー ID を設定します。 パブリッシュ/サブスクライブ階層を使用しており、メッセージ内のユーザー ID の権限を使用してメッセージを書き込むように着信チャネルがセットアップされている場合、送信側のキュー・マネージャーのユーザー ID に権限を与えなければなりません。 キュー・マネージャー階層でのデフォルトのユーザー ID の使用を参照してください。

注 : 一方、パブリッシュ/サブスクライブ・クラスターを使用する場合、許可はクラスターによって処理されます。

まとめと付加的な考慮事項

パブリッシュ/サブスクライブ階層では、キュー・マネージャー間の関係について細かく制御できます。 作成後に管理作業を実行するには、手操作による介入が若干必要です。 しかし、この場合、システムにおいて特定の制約も発生します。

- 階層内の上位ノード (特にルート・ノード) は、堅固な、可用性の高い、そして処理能力の高い機器でホストされていなければなりません。 それは、より多くのパブリケーション・トラフィックがそれらのノードを通過することになるからです。
- 階層内のリーフでないどのキュー・マネージャーについても、それぞれのアベイラビリティーが、パブリッシャーから他のキュー・マネージャー上のサブスクライバーへのメッセージ・フローに関して、ネットワークの可用性に影響することになります。
- デフォルトでは、サブスクライブ対象のすべてのトピック・ストリングが階層を通じて伝搬します。 また、パブリケーションは、関連するトピックのサブスクリプションのあるリモート・キュー・マネージャーにのみ伝搬します。 したがって、サブスクリプションのセットが急速に変化すると、それは制限要因となり得ます。 デフォルトのこの動作は変更可能であり、すべてのパブリケーションがすべてのキュー・マネージャーに伝搬するようにすれば、プロキシー・サブスクリプションの必要がなくなります。 『パブリッシュ/サブスクライブ・ネットワークでのサブスクリプションのパフォーマンス』を参照してください。

  注: これと同じような制限が直接ルーティング型クラスターにも適用されます。
- パブリッシュ/サブスクライブ・キュー・マネージャーの相互接続された性質のため、プロキシー・サブスクリプションがネットワーク内のすべてのノードを伝搬するのに時間がかかります。 リモート・パブリケーションでは、必ずしもサブスクライブがすぐに開始されるわけではないため、早い時期のパブリケーションが新しいトピック・ストリングのサブスクリプション後になっても送信されないという可能性があります。 すべてのパブリケーションがすべてのキュー・マネージャーに伝搬するようにして、プロキシー・サブスクリプションが不要であるようにすれば、サブスクリプションの遅れによって発生する問題はなくなります。 『パブリッシュ/サブスクライブ・ネットワークでのサブスクリプションのパフォーマンス』を参照してください。

  注: この制限は、直接ルーティング型クラスターにも当てはまります。
- パブリッシュ/サブスクライブ階層の場合、キュー・マネージャーを追加したり除去したりするには、階層を手動で構成しなければならず、それらのキュー・マネージャーの位置や他のキュー・マネージャーへの依存関係について注意深く考慮することが必要になります。 階層の最も下部にあって、それより下にブランチがないキュー・マネージャーを追加または削除するのでない限り、階層内の他のキュー・マネージャーについても構成作業が必要になります。

パブリッシュ/サブスクライブ階層をルーティング・メカニズムとして使用する前に、78 ページの『パブリッシュ/サブスクライブ・クラスターでの直接ルーティング』と 82 ページの『パブリッシュ/サブスクライブ・クラスターでのトピック・ホスト・ルーティング』で詳しく説明されている別のアプローチについて検討してください。

分散パブリッシュ/サブスクライブのシステム・キュー

キュー・マネージャーは、パブリッシュ/サブスクライブ・メッセージング用に 4 つのシステム・キューを使用します。 それらのキューの存在について意識する必要があるのは、問題判別とキャパシティー・プランニングを行う場合のみです。

これらのキューをモニターするための手引きは、パブリッシュ/サブスクライブ・ネットワークにおけるプロデューサーとコンシューマーのバランシングを参照してください。

表 6. システム・キューのパブリッシュ/サブスクライブ *(Multiplatforms)*

システム・キュー	目的
SYSTEM.INTER.QMGR.CONTROL	IBM MQ 分散パブリッシュ/サブスクライブの制御キュー
SYSTEM.INTER.QMGR.FANREQ	IBM MQ 分散パブリッシュ/サブスクライブの内部プロキシー・サブスクリプション多分岐プロセスの入力キュー
SYSTEM.INTER.QMGR.PUBS	IBM MQ 分散パブリッシュ/サブスクライブのパブリケーション
SYSTEM.HIERARCHY.STATE	IBM MQ 分散パブリッシュ/サブスクライブ階層関係の状態

z/OS z/OS では、キュー・マネージャーを作成する際に、CSQINP2 初期設定入力データ・セットの CSQ4INSX、CSQ4INSR および CSQ4INSG のサンプルを含めることによって、必要なシステム・オブジェクトをセットアップします。 詳しくは、作業 13: 初期設定入力データ・セットをカスタマイズするを参照してください。

パブリッシュ/サブスクライブ・システム・キューの属性を 111 ページの表 7 に示します。

表 7. パブリッシュ/サブスクライブ・システム・キューの属性

属性	デフォルト値
DEFPSIST	はい
DEFSOPT	SHARED
MAXMSGL	Multi マルチプラットフォーム の場合: ALTER QMGR コマンドの MAXMSGL パラメーターの値 z/OS z/OS の場合: 4194304 (つまり、4 MB)
MAXDEPTH	999999999
SHARE	N/A
z/OS z/OS STGCLASS	この属性は、z/OS プラットフォームのみで使用されます。

注: アプリケーションによって書き込まれるメッセージを含む唯一のキューは SYSTEM.INTER.QMGR.PUBS です。 **MAXDEPTH** にはこのキューの最大値を設定し、障害の発生時や過負荷の際にパブリッシュされたメッセージを一時的に蓄積できるようにします。 システム上で稼働しているキュー・マネージャーのキューの深さでは入りきらない場合は、この値を調整する必要があります。

関連タスク

分散パブリッシュ/サブスクライブの問題のトラブルシューティング

分散パブリッシュ/サブスクライブ・システム・キューのエラー

分散パブリッシュ/サブスクライブ・キュー・マネージャーのキューが使用不可の場合、エラーが発生することがあります。 これは、サブスクリプション情報をパブリッシュ/サブスクライブ・ネットワーク全体に伝搬する処理と、リモート・キュー・マネージャー上のサブスクリプションに対するパブリケーションに影響を与えます。

多分岐要求キュー SYSTEM.INTER.QMGR.FANREQ を使用できない場合、サブスクリプションの作成でエラーが発生することがあり、直接接続されているキュー・マネージャーにプロキシー・サブスクリプションを配信する必要が生じた時点で、キュー・マネージャーのエラー・ログにエラー・メッセージが書き込まれます。

階層関係状態キュー SYSTEM.HIERARCHY.STATE が使用不可になっていると、エラー・メッセージがキュー・マネージャーのエラー・ログに書き込まれ、パブリッシュ/サブスクライブ・エンジンが COMPAT モードになります。 パブリッシュ/サブスクライブのモードを表示するには、DISPLAY QMGR PSMODE コマンドを使用します。

その他の SYSTEM.INTER.QMGR キューが使用できない場合、キュー・マネージャーのエラー・ログにエラー・メッセージが書き込まれます。機能が使用不可になっていない場合でも、このキュー・マネージャーまたはリモート・キュー・マネージャーのキューにパブリッシュ/サブスクライブ・メッセージが蓄積されていく可能性があります。

親キュー・マネージャー、子キュー・マネージャー、またはパブリッシュ/サブスクライブ・クラスター・キュー・マネージャーに対するパブリッシュ/サブスクライブ・システム・キュー、または必要な伝送キューが使用不可であると、以下のような結果になります。

- パブリケーションが配信されません。パブリッシュ側アプリケーションがエラーを受け取る可能性があります。 パブリッシュ側アプリケーションがどのような時にエラーを受け取るかについて詳しくは、**DEFINE TOPIC** コマンドのパラメーター **PMSGDLV**、**NPMSGDLV**、および **USEDLQ** を参照してください。
- 受け取ったキュー・マネージャー間パブリケーションは入力キューにバックアウトされ、後で再試行されます。 バックアウトしきい値に達すると、未配布のパブリケーションは送達不能キューに書き込まれます。 キュー・マネージャーのエラー・ログに問題の詳細が記載されます。
- 未配布のプロキシー・サブスクリプションは、多分岐要求キューにバックアウトされ、後で再試行されます。 バックアウトしきい値に達すると、未配布のプロキシー・サブスクリプションは、接続されているどのキュー・マネージャーに対しても配信されず、送達不能キューに書き込まれます。 キュー・マネージャーのエラー・ログに、問題の詳細が、必要な管理上の修正処置の 詳細を含めて記載されます。
- 階層関係プロトコル・メッセージは失敗し、接続状況に ERROR フラグが立てられます。 接続状況を表示するには、コマンド **DISPLAY PUBSUB** を使用します。

関連タスク

分散パブリッシュ/サブスクライブの問題のトラブルシューティング

Multi ストレージ要件とパフォーマンス要件の計画 (Multiplatforms)

IBM MQ システムに対して現実的で達成可能なストレージおよびパフォーマンスの目標を設定する必要があります。 以下のリンク先で、ご使用のプラットフォームでストレージおよびパフォーマンスに影響を与える要因についての情報を参照してください。

要件は、IBM MQ を使用するシステム、および使用するコンポーネントによって異なります。

サポートされるハードウェアおよびソフトウェア環境に関する最新情報については、IBM MQ システム要件を参照してください。

IBM MQ は、キュー・マネージャー・データをファイル・システムに保管します。 以下のリンク先で、IBM MQ で使用するためのディレクトリー構造の計画および構成についての情報を参照してください。

- 116 ページの『ファイル・システム・サポートの計画 (Multiplatforms)』
- 117 ページの『共有ファイル・システムの要件 (Multiplatforms)』
- 126 ページの『IBM MQ ファイルの共有 (Multiplatforms)』
- Linux AIX 128 ページの『AIX and Linux システムでのディレクトリー構造』
- Windows 138 ページの『Windows システムでのディレクトリー構造』
- IBM i 141 ページの『IBM i でのディレクトリー構造』

AIX and Linux でのシステム・リソース、共有メモリー、およびプロセスの優先順位について詳しくは、以下のリンク先を参照してください。

- Linux AIX 146 ページの『IBM MQ と UNIX System V IPC リソース』
- AIX 145 ページの『AIX 上の共有メモリー』
- Linux AIX 146 ページの『IBM MQ および UNIX のプロセス優先順位』

ログ・ファイルについては、以下のリンクの情報を参照してください。

- 145 ページの『循環ロギングまたはリニア・ロギングの選択 (Multiplatforms)』
- ログのサイズの計算

関連概念

147 ページの『Planning your IBM MQ environment on z/OS』
When planning your IBM MQ environment, you must consider the resource requirements for data sets, page sets, Db2, Coupling Facilities, and the need for logging, and backup facilities. Use this topic to plan the environment where IBM MQ runs.

関連タスク

5 ページの『IBM MQ アーキテクチャーの計画』

IBM MQ 環境を計画する際、単一および複数キュー・マネージャーのアーキテクチャーについて、また Point-to-Point およびパブリッシュ/サブスクライブのメッセージング・スタイルについて IBM MQ が提供するサポートを考慮します。 また、リソース要件、およびロギングやバックアップの機能の使用方法を計画します。

関連資料

AIX and Linux でのハードウェア要件とソフトウェア要件

Windows でのハードウェア要件とソフトウェア要件

Multi ディスク・スペース要件 (Multiplatforms)

IBM MQ のストレージ要件は、インストールするコンポーネント、および必要なワークスペース量によって異なります。

ディスク・ストレージはインストールするオプション・コンポーネント用に必要となりますが、それにはオプション・コンポーネントが必要とする前提条件のコンポーネントも含まれます。 使用するキューの数、キューに入れるメッセージの数とサイズ、およびメッセージが持続メッセージかどうかによって、ストレージ要件の合計は異なります。 そのほかに、ディスクやテープなどのメディアにアーカイブとして保存するための容量も必要であり、所有アプリケーション・プログラムのためのスペースももちろん必要です。

以下の表に、さまざまなプラットフォームにいろいろな製品を組み合わせてインストールしたときに必要となる、おおよそのディスク・スペースを示します。 (値は 5 MB 単位になるように切り上げています。1 MB は 1,048,576 バイトです。)

- LTS 113 ページの『ディスク・スペース要件 (Long Term Support)』
- CD 114 ページの『ディスク・スペース要件 (Continuous Delivery)』

ディスク・スペース要件 (Long Term Support)

LTS V 9.4.0

表 8. *IBM MQ のディスク・スペース要件 (Long Term Support の Multiplatforms)*

プラットフォーム	クライアント・インストール 114 ページの『1』	サーバー・インストール 114 ページの『2』	フルインストール 114 ページの『3』
AIX AIX	335 MB	375 MB	1810 MB
IBM i IBM i (IBM i に関するその他の注意事項を参照)	485 MB	845 MB	1965 MB
Linux Linux for x86-64	270 MB	295 MB	2010 MB
Linux Linux on Power® Systems - Little Endian	170 MB	190 MB	1400 MB (英語)
Linux Linux on IBM Z®	255 MB	290 MB	1485 MB
Windows Windows (64 ビット・インストール) 114 ページの『4』	295 MB	425 MB	2310 MB

注 :

1. クライアント・インストールには、以下のコンポーネントが含まれます。
   - のランタイム
   - クライアント
2. サーバー・インストールには、以下のコンポーネントが含まれます。
   - のランタイム
   - サーバー
3. フルインストールには、選択可能なすべてのコンポーネントが含まれます。
4. Windows ここにリストされているすべてのコンポーネントが、Windows システムにインストール可能なフィーチャーであるわけではありません。 それらの機能が別のフィーチャーに組み込まれている場合があります。 Windows システムの IBM MQ 機能を参照してください。

IBM i に関するその他の注意事項 : IBM i

1. IBM i では、サーバーからネイティブ・クライアントを切り離せません。 表に示しているこのサーバーの数値は、Java なし、英語ロード (2924) の 5724H72*BASE の場合の数値です。 言語ロードは 22 種類あります。
2. 表に示している数値は、ネイティブ・クライアント 5725A49 *BASE (Java なし) の場合の数値です。
3. Java クラスおよび JMS クラスは、サーバー・バインディングとクライアント・バインディングの両方に追加できます。 これらの機能を組み込む場合は、110 MB を追加してください。
4. クライアントまたはサーバーのいずれかにサンプル・ソースを追加する場合は、さらに 10 MB を追加します。
5. Java および JMS クラスにサンプルを追加すると、さらに 5 MB 追加されます。

ディスク・スペース要件 (Continuous Delivery)

V 9.4.0 CD

表 *9. IBM MQ* のディスク・スペース要件 *(Continuous Delivery* の *Multiplatforms)*

プラットフォーム/CD リリース	クライアント・インストール 115 ページの『1』	サーバー・インストール 116 ページの『2』	フルインストール 116 ページの『3』
AIX **AIX**			
V 9.4.0 IBM MQ 9.4.0	355 MB	390 MB	1440 MB
V 9.4.1 IBM MQ 9.4.1	355 MB	390 MB	1440 MB
V 9.4.2 IBM MQ 9.4.2	390 MB	450 MB	1815 MB
V 9.4.3 IBM MQ 9.4.3	390 MB	450 MB	1815 MB
Linux **Linux for x86-64 (64 ビット)**			
V 9.4.0 IBM MQ 9.4.0	280 MB	295 MB	1195 MB
V 9.4.1 IBM MQ 9.4.1	280 MB	295 MB	1195 MB

表 9. *IBM MQ* のディスク・スペース要件 *(Continuous Delivery* の *Multiplatforms) (*続き*)*			
プラットフォーム/CD リリース	**クライアント・インストール** 115 ページの『1』	**サーバー・インストール** 116 ページの『2』	**フルインストール** 116 ページの『3』
V 9.4.2 IBM MQ 9.4.2	300 MB	305 MB	2010 MB
V 9.4.3 IBM MQ 9.4.3	300 MB	305 MB	2010 MB
Linux **Linux on Power Systems - Little Endian**			
V 9.4.0 IBM MQ 9.4.0	170 MB	195 MB	1075 MB
V 9.4.1 IBM MQ 9.4.1	170 MB	195 MB	1075 MB
V 9.4.2 IBM MQ 9.4.2	185 MB	210 メガバイト	1400 MB (英語)
V 9.4.3 IBM MQ 9.4.3	185 MB	210 メガバイト	1400 MB (英語)
Linux **Linux for IBM Z**			
V 9.4.0 IBM MQ 9.4.0	260 MB	290 MB	1160 MB
V 9.4.1 IBM MQ 9.4.1	260 MB	290 MB	1160 MB
V 9.4.2 IBM MQ 9.4.2	270 MB	290 MB	1485 MB
V 9.4.3 IBM MQ 9.4.3	270 MB	290 MB	1485 MB
Windows **Windows (64 ビット・インストール)** 116 ページの『4』			
V 9.4.0 IBM MQ 9.4.0	300 MB	425 MB	1785 MB
V 9.4.1 IBM MQ 9.4.1	300 MB	425 MB	1785 MB
V 9.4.2 IBM MQ 9.4.2	315 メガバイト	425 MB	2310 MB
V 9.4.3 IBM MQ 9.4.3	315 メガバイト	425 MB	2310 MB

注:

1. クライアント・インストールには、以下のコンポーネントが含まれます。
   - のランタイム
   - クライアント

2. サーバー・インストールには、以下のコンポーネントが含まれます。
   - のランタイム
   - サーバー
3. フルインストールには、選択可能なすべてのコンポーネントが含まれます。
4. Windows ここにリストされているすべてのコンポーネントが、Windows システムにインストール可能なフィーチャーであるわけではありません。 それらの機能が別のフィーチャーに組み込まれている場合があります。 Windows システムの IBM MQ 機能を参照してください。

関連概念
IBM MQ のコンポーネントと機能

Multi ファイル・システム・サポートの計画 (Multiplatforms)

キュー・マネージャー・データはファイル・システムに格納されます。 キュー・マネージャーはファイル・システム・ロックを使用して、複数インスタンスキュー・マネージャーの複数インスタンスが同時にアクティブにならないようにします。

ファイル共有システム

ファイル共有システムでは、複数のシステムが同じ物理ストレージ・デバイスに同時にアクセスできるようにします。 複数のシステムが、ロックと並行性制御を適用する方法をとらずに、同じ物理ストレージ・デバイスに直接アクセスした場合に破損が起こることがあります。 オペレーティング・システムには、ローカル・プロセスのロックと並行性制御が用意されたローカル・ファイル・システムがあります。 ネットワーク・ファイル・システムには、分散システム用にロックと並行性制御が用意されています。

これまでのネットワーク・ファイル・システムはそれほど高速で稼働していませんでした。つまり、メッセージのログ記録の要件を満たす十分なロックと並行性制御を提供していませんでした。 現在のネットワーク・ファイル・システムでは、良好なパフォーマンスを提供し、 *RFC 3530* のネットワーク・ファイル・システム *(NFS)* バージョン *4* プロトコルなどの 信頼性の高いネットワーク・ファイル・システム・プロトコルの実装を可能にして、 確実にメッセージのログ記録の要件を満たすことができます。

ファイル共有システムおよび IBM MQ

複数インスタンスのキュー・マネージャーのキュー・マネージャー・データは、共有ネットワーク・ファイル・システムに保管されます。 AIX, Linux, and Windows システムでは、キュー・マネージャーのデータ・ファイルおよびログ・ファイルは、共有ネットワーク・ファイル・システムに置く必要があります。

IBM i IBM i では、ログ・ファイルではなくジャーナルが使用され、ジャーナルは共有できません。 IBM i の複数インスタンス・キュー・マネージャーはジャーナルの複製、つまり切り替え可能ジャーナルを使用して、ジャーナルを異なるキュー・マネージャー・インスタンス間で使用できるようにします。

IBM MQ はロックを使用して、同じ複数インスタンス・キュー・マネージャーの複数インスタンスが同時にアクティブにならないようにします。 またこの同じロックは、2 つの異なるキュー・マネージャーが、同じセットのキュー・マネージャー・データ・ファイルを不注意に使用できないようにします。 ロックを使用できるキュー・マネージャーのインスタンスは一度に 1 つだけです。 そうすることで、IBM MQ は、ファイル共有システムとしてアクセスされるネットワーク・ストレージに保管されたキュー・マネージャー・データをサポートします。

ネットワーク・ファイル・システムのロック・プロトコルがすべて堅固であるわけではなく、またファイル・システムはデータ保全性というよりもパフォーマンスを優先して構成される場合があるため、**amqmfsck** コマンドを実行して、ネットワーク・ファイル・システムがキュー・マネージャー・データおよびログへのアクセスを正しく制御するかどうかをテストする必要があります。 このコマンドは、UNIX、Linux、および IBM i の各システムにのみ適用されます。 Windows では、サポートされるネットワーク・ファイル・システムは 1 つしかないため、**amqmfsck** コマンドは必要ありません。

関連タスク
119 ページの『共有ファイル・システムの動作の検証 (Multiplatforms)』

amqmfsck を実行して、AIX、Linux、または IBM i 上の共有ファイル・システムが、複数インスタンス・キュー・マネージャーのキュー・マネージャー・データを保管するための要件を満たすかどうか確認します。 (Windows 構成の唯一の要件は、共有ストレージのプロビジョンに SMB 3 を使用することです。)

Multi 共有ファイル・システムの要件 (Multiplatforms)

ファイル共有システムは、IBM MQ での作業の信頼性を高めるために、データ書き込み整合性、ファイルに対する排他的アクセスの保証、および障害時のロック解除を実現する必要があります。

ファイル共有システムが満たす必要がある要件

ファイル共有システムが IBM MQ で確実に機能するためには、次の 3 つの基本要件を満たしている必要があります。

1. データ書き込み整合性

   データ書き込み整合性は、「フラッシュ時のディスクへの書き込み」と呼ばれることがあります。 キュー・マネージャーは、 物理デバイスに正常にコミットされているデータと同期できなければなりません。 トランザクション・システムでは、他の処理を続行する前に、いくつかの書き込みが安全にコミットされたことを確認する必要があります。

   より具体的には、IBM MQ for AIX or Linux プラットフォームは *O_SYNC* オープン・オプションと `fsync()` システム呼び出しを使用して、 リカバリー可能なメディアへの書き込みを明示的に強制します。書き込み操作は、これらのオプションが正しく作動することに依存しています。

   重要 : Linux async オプションを使用して、ファイル・システムをマウントする必要があります。 このオプションを使用すると、引き続き同期書き込みのオプションがサポートされ、sync オプションよりもパフォーマンスが向上します。

   しかし、ファイル・システムが Linux からエクスポートされた場合は、引き続き sync オプションを使用してファイル・システムをエクスポートしなければならないことに注意してください。

2. ファイルに対する排他的アクセスの保証

   複数のキュー・マネージャーを同期するために、 キュー・マネージャーにはファイルへの排他ロックを獲得する手段が必要です。

3. 障害時のロック解除

   キュー・マネージャーに障害が発生したり、ファイル・システムとの通信障害がある場合は、キュー・マネージャーがファイル・システムに再接続するのを待たずに、キュー・マネージャーによってロックされたファイルをアンロックして、他のプロセスで使用できるようにしなければなりません。

ファイル共有システムでは、IBM MQ を確実に作動させるためにこれらの要件を満たす必要があります。 そうしないと、ファイル共有システムを複数インスタンス・キュー・マネージャー構成で使用するときに、キュー・マネージャー・データおよびログが破損します。

Microsoft Windows 上の複数インスタンス・キュー・マネージャーの場合、ネットワーク・ストレージには、 Microsoft Windows ネットワークで使用される Server Message Block (SMB) プロトコルによってアクセスする必要があります。 Server Message Block (SMB) クライアントは、 Microsoft Windows 以外のプラットフォームでセマンティクスをロックするための IBM MQ 要件を満たしていないため、 Microsoft Windows 以外のプラットフォームで実行される複数インスタンス・キュー・マネージャーは、共有ファイル・システムとして Server Message Block (SMB) を使用してはなりません。

その他のサポートされているプラットフォームの複数インスタンスのキュー・マネージャーでは、Posix に準拠し、リース・ベースのロックをサポートしているネットワーク・ファイル・システム・プロトコルによってストレージにアクセスする必要があります。 ネットワーク・ファイル・システム 4 は、この要件を満たしています。 以前のファイル・システム (ネットワーク・ファイル・システム (NFS) バージョン 3 など) は、障害後にロックを解除するための信頼性のある仕組みを持たないため、複数インスタンス・キュー・マネージャーで使用してはなりません。

ファイル共有システムが要件を満たすかどうかの確認

使用する予定のファイル共有システムが、これらの要件を満たすかどうかを確認する必要があります。 また、ファイル・システムが信頼性を得られるように 正しく構成されているかどうかも確認する必要があります。 ファイル共有システムは、 信頼性を犠牲にしてパフォーマンスを向上させるような構成オプションを提供することがあります。

詳細については、Testing statement for IBM MQ multi-instance queue manager file systems を参照してください。

通常の環境では、IBM MQ は属性キャッシュを使用して正しく作動するため、例えば NFS マウントで NOAC を設定するなどしてキャッシュを使用不可にする必要はありません。 複数のファイル・システム・クライアントがファイル・システム・サーバー上にある同じファイルへの書き込みアクセスを得ようと競合する場合、 属性キャッシュは問題を引き起こすことがあります。 これは、各クライアントが使用するキャッシュ内属性がサーバー上の属性と同じではない場合があるためです。 このようにアクセスされるファイルの例は、マルチインスタンス・キュー・マネージャー用のキュー・マネージャー・エラー・ログです。 キュー・マネージャー・エラー・ログはアクティブ・キュー・マネージャー・インスタンスとスタンバイ・キュー・マネージャー・インスタンスの 両方によって書き込まれることがあり、 キャッシュに入っているファイル属性が存在すると、ファイルのロールオーバーが発生する前にエラー・ログが予想以上に大きくなることもあります。

ファイル・システムを検査するには、『共有ファイル・システムの動作の検証』のタスクが役立ちます。 このタスクは、共有ファイル・システムが要件 2 と 3 を満たすかどうかを検査します。 要件 1 については、ファイル共有システムの資料で、またはディスクへのデータのログ記録を検証することによって、確認する必要があります。

ディスク障害はディスクへの書き込み時にエラーを引き起こすことがあり、これは IBM MQ では初期障害データ・キャプチャー機能エラーとして報告されます。 ファイル共有システムにディスク障害がないかどうかを検査するには、 オペレーティング・システム用のファイル・システム検査プログラムを実行します。 以下に例を示します。

- Linux AIX AIX and Linux のファイル・システム検査プログラムは fsck と呼ばれます。
- Windows Windows プラットフォームのファイル・システム検査プログラムは CHKDSK または SCANDISK と呼ばれます。

NFS サーバー・セキュリティー

注 :

- IBM MQ インストール・ディレクトリーを保持するために使用されるマウント・ポイントに対して **nosuid** オプションまたは **noexec** オプションを使用することはできません。 これは、 IBM MQ には setuid/setgid 実行可能プログラムが含まれており、これらが適切に実行されないようにしてはならないためです。
- キュー・マネージャー・データをネットワーク・ファイル・システム (NFS) サーバーにのみ書き込む場合は、mount コマンドで以下の 3 つのオプションを使用して、キュー・マネージャーの実行に悪影響を与えずにシステムを保護することができます。

  noexec
  このオプションを使用すると、バイナリー・ファイルを NFS 上で実行できなくなります。 こうすることで、リモート・ユーザーがシステム上で望ましくないコードを実行できないようにします。

  nosuid
  このオプションを使用すると、セット・ユーザー ID ビットとセット・グループ ID ビットを使用できなくなります。 こうすることで、リモート・ユーザーが上位の特権を取得できないようにします。

  nodev
  このオプションを使用すると、文字およびブロック特殊装置が使用または定義できなくなります。 こうすることで、リモート・ユーザーが chroot ジェイルから出られないようにします。

IBM i Linux AIX 共有ファイル・システムの動作の検証 *(Multiplatforms)*

amqmfsck を実行して、AIX、Linux、または IBM i 上の共有ファイル・システムが、複数インスタンス・キュー・マネージャーのキュー・マネージャー・データを保管するための要件を満たすかどうか確認します。 (Windows 構成の唯一の要件は、共有ストレージのプロビジョンに SMB 3 を使用することです。)

始める前に

ネットワーク・ストレージを備えた 1 台のサーバーと、それに接続しており、IBM MQ がインストールされているさらに 2 台のサーバーが必要です。 ファイル・システムを構成するには管理者 (root) 権限が必要であり、 **amqmfsck** を実行するには IBM MQ 管理者でなければなりません。

このタスクについて

117 ページの『共有ファイル・システムの要件 (Multiplatforms)』 では、複数インスタンス・キュー・マネージャーで共有ファイル・システムを使用するための、ファイル・システム要件について説明します。 IBM MQ の技術情報、Testing statement for IBM MQ multi-instance queue manager file systems に、IBM でテスト済みの共有ファイル・システムがリストされています。 この作業の手順は、リストに含まれていないファイル・システムでデータの整合性が保持されるかどうかを評価するために、ファイル・システムをテストする方法について説明しています。

複数インスタンス・キュー・マネージャーのフェイルオーバーは、ハードウェアまたはソフトウェアの障害によりトリガーできます。そうした障害には、キュー・マネージャーからデータやログ・ファイルに書き込みができなくなるネットワークの問題も含まれます。 ファイル・サーバーに障害を発生させることがテストの主眼となります。 しかし、ロックが正常に解放されることをテストするため、IBM MQ サーバーにも障害を発生させる必要があります。 共有ファイル・システムの信頼性を確認するため、以下の障害、および環境に固有のその他の障害についてすべてテストしてください。

1. ディスクを同期させて、ファイル・サーバーのオペレーティング・システムをシャットダウンする。
2. ディスクを同期させないで、ファイル・サーバーのオペレーティング・システムを停止する。
3. 各サーバーのリセット・ボタンを押す。
4. 各サーバーのネットワーク・ケーブルを抜く。
5. 各サーバーの電源ケーブルを抜く。
6. 各サーバーの電源をオフにする。

キュー・マネージャーのデータおよびログを共有するために使用するディレクトリーを、ネットワーク・ストレージ上に作成します。 ディレクトリーの所有者は、IBM MQ 管理者であるか、または AIX and Linux 上の `mqm` グループのメンバーである必要があります。 テストを実行するユーザーは、IBM MQ 管理者権限を持っている必要があります。

ファイル・システムのエクスポートとマウントの例については、Linux での複数インスタンス・キュー・マネージャーの作成 または IBM i でのジャーナル・ミラーリングと NetServer を使用した複数インスタンス・キュー・マネージャーの作成 を参照してください。 それぞれのファイル・システムでは異なる構成ステップが必要です。 ファイル・システムの資料を参照してください。

注 : IBM MQ MQI client のサンプル・プログラム **amqsfhac** を **amqmfsck** と並行して実行し、障害時にキュー・マネージャーがメッセージの整合性を保持することを実証します。

手順

各検査では、ファイル・システム検査プログラムの実行中に、前のリストの障害をすべて発生させてください。 **amqmfsck** と同時に **amqsfhac** を実行する場合は、このタスクと並行して、124 ページの『amqsfhac を実行してメッセージの整合性を検査する』 のタスクを実行してください。

1. エクスポートされたディレクトリーを 2 つの IBM MQ サーバーにマウントします。

   ファイル・システム・サーバー上に共有ディレクトリー `shared` と、複数インスタンス・キュー・マネージャーのデータを保存するためのサブディレクトリー `qmdata` を作成します。 Linux での複数インスタンス・キュー・マネージャーの共有ディレクトリーのセットアップ例については、 Linux での複数インスタンス・キュー・マネージャーの作成 を参照してください。

2. ファイル・システムの基本的な動作を検査します。

   1 台の IBM MQ サーバー上で、パラメーターを付けずにファイル・システム検査プログラムを実行します。

   IBM MQ サーバー 1:

   ```
   amqmfsck /shared/qmdata
   ```

3. 両方の IBM MQ サーバーから同一のディレクトリーに同時に書き込む検査をします。

   両方の IBM MQ サーバー上で、-c オプションを指定して、ファイル・システム検査プログラムを同時に実行します。

   IBM MQ サーバー 1:

   ```
   amqmfsck -c /shared/qmdata
   ```

   IBM MQ サーバー 2:

   ```
   amqmfsck -c /shared/qmdata
   ```

4. 両方の IBM MQ サーバー上で、ロックの待機と解放を検査します。

   両方の IBM MQ サーバー上で、-w オプションを指定して、ファイル・システム検査プログラムを同時に実行します。

   IBM MQ サーバー 1:

   ```
   amqmfsck -w /shared/qmdata
   ```

   IBM MQ サーバー 2:

   ```
   amqmfsck -w /shared/qmdata
   ```

5. データの整合性を検査します。

   a) テスト・ファイルをフォーマット設定します。

   検査対象のディレクトリー内に大きなファイルを作成します。 このファイルは、後続のフェーズを正常に完了できるよう、フォーマット設定されます。 フェイルオーバーをシミュレートするために第 2 のフェーズを中断するための時間が十分とれるよう、このファイルは十分な大きさであることが必要です。 デフォルト値の 262144 ページ (1 GB) を試してください。 低速のファイル・システムでは、フォーマット設定が約 60 秒以内で完了するよう、このデフォルト値がプログラムによって自動的に削減されます。

   IBM MQ サーバー 1:

   ```
   amqmfsck -f /shared/qmdata
   ```

   サーバーは、次のメッセージで応答します。

   ```
   Formatting test file for data integrity test.

   Test file formatted with 262144 pages of data.
   ```

   b) 障害を発生させながら、ファイル・システム検査プログラムを使用してテスト・ファイルにデータを書き込みます。

   2 台のサーバー上で、検査プログラムを同時に実行します。 障害を発生させるサーバー上で検査プログラムを開始してから、障害を回避させるサーバー上で検査プログラムを開始します。 調査対象の障害を発生させます。

   最初の検査プログラムが、エラー・メッセージを表示して停止します。 2 番目の検査プログラムがテスト・ファイルに対するロックを取得し、最初の検査プログラムが書き込みを終了した箇所から、テスト・ファイルにデータを書き込みます。 2 番目の検査プログラムの完了を待ちます。

表 *10. 2* 台のサーバー上でデータの整合性検査を同時に実行する

IBM MQ サーバー 1	**IBM MQ サーバー 2**
`amqmfsck -a /shared/qmdata`	

`Please start this program on a second machine`
`with the same parameters.`
 	
`File lock acquired.`
 	
`Start a second copy of this program with the same parameters on another server.`
 	
`Writing data into test file.`
 	
`To increase the effectiveness of the test, interrupt the writing by ending the process,`
`temporarily breaking the network connection`
`to the networked storage,`
`rebooting the server or turning off the power.`
	`amqmfsck -a /shared/qmdata`
	
	`Waiting for lock...`
	
	`Waiting for lock...`
	
	`Waiting for lock...`
	
	`Waiting for lock...`
	
	`Waiting for lock...`
	
	`Waiting for lock...`

Turn the power off here.	

	`File lock acquired.`
	
	`Reading test file`
	
	`Checking the integrity of the data read.`
	
	`Appending data into the test file after data already found.`
	
	`The test file is full of data.`
	`It is ready to be inspected for data integrity.`

検査のタイミングは、ファイル・システムの動作によって異なります。 例えば、電源異常の後でファイル・システムが、最初のプログラムによって取得されていたファイル・ロックを解放するまでには、通常 30 から 90 秒の時間が必要です。 時間が短すぎて最初のテスト・プログラムがファイルへの書き込みを終了する前に障害が起きなかった場合は、**amqmfsck** の -x オプションを使用して、テスト・ファイルを削除してください。 さらに大きなテスト・ファイルを使用して、テストを最初から実行してください。

c) テスト・ファイル内のデータの整合性を検査します。

IBM MQ サーバー 2:

```
amqmfsck -i /shared/qmdata
```

サーバーは、次のメッセージで応答します。

```
File lock acquired

Reading test file checking the integrity of the data read.
```

```
The data read was consistent.

The tests on the directory completed successfully.
```

6. テスト・ファイルを削除します。

   IBM MQ サーバー 2:

   ```
   amqmfsck -x /shared/qmdata
   Test files deleted.
   ```

   サーバーは、次のメッセージで応答します。

   ```
   Test files deleted.
   ```

タスクの結果

テストが正常に完了すると、プログラムは終了コードとしてゼロを戻します。そうでない場合はゼロ以外を戻します。

例

3 つの例の最初のものでは、最小出力を生成するコマンドを示します。

1 台のサーバー上の基本ファイル・ロックのテストが正常終了

```
> amqmfsck /shared/qmdata
The tests on the directory completed successfully.
```

1 台のサーバー上の基本ファイル・ロックのテストが失敗

```
> amqmfsck /shared/qmdata
AMQ6245: Error Calling 'write()[2]' on file '/shared/qmdata/amqmfsck.lck' error '2'.
```

2 台のサーバー上のロックのテストが正常終了

表 *11. 2* 台のサーバー上のロックが正常終了

IBM MQ サーバー 1	**IBM MQ サーバー 2**

`> amqmfsck -w /shared/qmdata`
`Please start this program on a second`
`machine with the same parameters.`
`Lock acquired.`
`Press Return`
`or terminate the program to release the lock.`	

	`> amqmfsck -w /shared/qmdata` `Waiting for lock...`
`[ Return pressed ]` `Lock released.`	

	`Lock acquired.`
	`The tests on the directory completed`
	`successfully`

3 つの例の 2 番目のものでは、冗長モードを使用する同じコマンドを示します。

1 台のサーバー上の基本ファイル・ロックのテストが正常終了

```
> amqmfsck -v /shared/qmdata
System call: stat("/shared/qmdata")'
System call: fd = open("/shared/qmdata/amqmfsck.lck", O_RDWR, 0666)
System call: fchmod(fd, 0666)
System call: fstat(fd)
System call: fcntl(fd, F_SETLK, F_WRLCK)
System call: write(fd)
System call: close(fd)
System call: fd = open("/shared/qmdata/amqmfsck.lck", O_RDWR, 0666)
System call: fcntl(fd, F_SETLK, F_WRLCK)
System call: close(fd)
System call: fd1 = open("/shared/qmdata/amqmfsck.lck", O_RDWR, 0666)
System call: fcntl(fd1, F_SETLK, F_RDLCK)
System call: fd2 = open("/shared/qmdata/amqmfsck.lck", O_RDWR, 0666)
System call: fcntl(fd2, F_SETLK, F_RDLCK)
System call: close(fd2)
System call: write(fd1)
System call: close(fd1)
The tests on the directory completed successfully.
```

1 台のサーバー上の基本ファイル・ロックのテストが失敗

```
> amqmfsck -v /shared/qmdata
System call: stat("/shared/qmdata")
System call: fd = open("/shared/qmdata/amqmfsck.lck", O_RDWR, 0666)
System call: fchmod(fd, 0666)
System call: fstat(fd)
System call: fcntl(fd, F_SETLK, F_WRLCK)
System call: write(fd)
System call: close(fd)
System call: fd = open("/shared/qmdata/amqmfsck.lck", O_RDWR, 0666)
System call: fcntl(fd, F_SETLK, F_WRLCK)
System call: close(fd)
System call: fd = open("/shared/qmdata/amqmfsck.lck", O_RDWR, 0666)
System call: fcntl(fd, F_SETLK, F_RDLCK)
System call: fdSameFile = open("/shared/qmdata/amqmfsck.lck", O_RDWR, 0666)
System call: fcntl(fdSameFile, F_SETLK, F_RDLCK)
System call: close(fdSameFile)
System call: write(fd)
AMQxxxx: Error calling 'write()[2]' on file '/shared/qmdata/amqmfsck.lck', errno 2
(Permission denied).
```

2 台のサーバー上のロックのテストが正常終了

表 *12. 2* 台のサーバー上のロックが正常終了 - 冗長モード

IBM MQ サーバー 1	**IBM MQ サーバー 2**

`> amqmfsck -wv /shared/qmdata`
`Calling 'stat("/shared/qmdata")'`
`Calling 'fd = open("/shared/qmdata/`
`amqmfsck.lkw",`
`O_EXCL | O_CREAT | O_RDWR, 0666)'`
`Calling 'fchmod(fd, 0666)'`
`Calling 'fstat(fd)'`
`Please start this program on a second`
`machine with the same parameters.`
`Calling 'fcntl(fd, F_SETLK, F_WRLCK)'`
`Lock acquired.`
`Press Return`
`or terminate the program to release the lock.`	

	`> amqmfsck -wv /shared/qmdata`
	`Calling 'stat("/shared/qmdata")'`
	`Calling 'fd = open("/shared/qmdata/`
	`amqmfsck.lkw",`
	`O_EXCL | O_CREAT | O_RDWR,0666)'`
	`Calling 'fd = open("/shared/qmdata/amqmfsck.lkw,`
	`O_RDWR, 0666)'`
	`Calling 'fcntl(fd, F_SETLK, F_WRLCK)`
	`'Waiting for lock...`

表 12. 2 台のサーバー上のロックが正常終了 - 冗長モード (続き)

IBM MQ サーバー 1	IBM MQ サーバー 2

`[ Return pressed ]`
`Calling 'close(fd)'`
`Lock released.`	

	`Calling 'fcntl(fd, F_SETLK, F_WRLCK)'`
	`Lock acquired.`
	`The tests on the directory completed successfully`

関連資料
高可用性のサンプル・プログラム

Multi

amqsfhac を実行してメッセージの整合性を検査する

IBM MQ MQI client のサンプル・プログラム **amqsfhac** を **amqmfsck** と並行して実行し、障害時にキュー・マネージャーがメッセージの整合性を保持することを実証します。

始める前に

この検査には、4 台のサーバーが必要です。 複数インスタンス・キュー・マネージャー用に 2 つのサーバー、ファイル・システム用に 1 つのサーバー、および **amqsfhac** を IBM MQ MQI client アプリケーションとして実行するために 1 つのサーバー。

119 ページの『共有ファイル・システムの動作の検証 (Multiplatforms)』のステップ 119 ページの『1』に従って、複数インスタンス・キュー・マネージャー用にファイル・システムをセットアップします。

このタスクについて

IBM MQ MQI client のサンプル・プログラム **amqsfhac** は、ネットワーク・ストレージを使用しているキュー・マネージャーが、障害の発生後にデータの整合性を保持していることを検査します。 **amqsfhac** を **amqmfsck** と並行して実行して、障害時にキュー・マネージャーがメッセージ保全性を維持することを示します。

手順

1. 手順のステップ 119 ページの『1』で作成したファイル・システムを使用して、別のサーバー QM1 上に複数インスタンス・キュー・マネージャーを作成します。
   複数インスタンス・キュー・マネージャーの作成を参照してください。
2. 両方のサーバー上でキュー・マネージャーを開始し、可用性を高くします。

   サーバー 1:

   ```
   strmqm -x QM1
   ```

   サーバー 2:

   ```
   strmqm -x QM1
   ```

3. **amqsfhac** を実行するためにクライアント接続をセットアップします。
   a) 企業で使用しているプラットフォーム用の手順 (*IBM MQ* のインストールの検査を参照) を使用してクライアント接続をセットアップするか、再接続可能クライアントのサンプルにあるサンプル・スクリプトを使用します。
   b) QM1 を実行している 2 台のサーバーに対応する 2 つの IP アドレスを使用するように、クライアント・チャネルを変更します。

サンプル・スクリプトで、以下を変更します。

```
DEFINE CHANNEL(CHANNEL1) CHLTYPE(CLNTCONN) TRPTYPE(TCP) +
CONNAME('LOCALHOST(2345)') QMNAME(QM1) REPLACE
```

変換後:

```
DEFINE CHANNEL(CHANNEL1) CHLTYPE(CLNTCONN) TRPTYPE(TCP) +
CONNAME('server1(2345),server2(2345)') QMNAME(QM1) REPLACE
```

ここで、server1 および server2 は 2 台のサーバーのホスト名、2345 はチャネル・リスナーが listen しているポートです。 通常、このデフォルトは 1414 です。 デフォルトのリスナー構成では、1414 を使用できます。

4. テスト用として、QM1 に 2 つのローカル・キューを作成します。
   以下の MQSC スクリプトを実行します。

```
DEFINE QLOCAL(TARGETQ) REPLACE
DEFINE QLOCAL(SIDEQ) REPLACE
```

5. **amqsfhac** を使用して構成を検査します。

```
amqsfhac QM1 TARGETQ SIDEQ 2 2 2
```

6. ファイル・システムの整合性を検査するのと同時に、メッセージの整合性を検査します。

   119 ページの『共有ファイル・システムの動作の検証 (Multiplatforms)』のステップ 120 ページの『5』 で **amqsfhac** を実行します。

```
amqsfhac QM1 TARGETQ SIDEQ 10 20 0
```

   アクティブなキュー・マネージャーのインスタンスを停止すると、**amqsfhac** は、もう 1 つのキュー・マネージャーのインスタンスがアクティブになったときにそのインスタンスに再接続します。 次の検査で障害をリバースできるようにするため、停止したキュー・マネージャーのインスタンスを再始動します。 フェイルオーバーが発生するのに十分な時間、検査プログラムが実行を続けるようにするため、ご使用の環境での試行に基づいて、反復の数を増やす必要があるかもしれません。

タスクの結果

ステップ 125 ページの『6』 での **amqsfhac** の実行例を以下に示します。 この例では、テストは成功します。

```
Sample AMQSFHAC start
qmname = QM1
qname = TARGETQ
sidename = SIDEQ
transize = 10
iterations = 20
verbose = 0
Iteration 0
Iteration 1
Iteration 2
Iteration 3
Iteration 4
Iteration 5
Iteration 6
Resolving MQRC_CALL_INTERRUPTED
MQGET browse side tranid=14 pSideinfo->tranid=14
Resolving to committed
Iteration 7
Iteration 8
Iteration 9
Iteration 10
Iteration 11
Iteration 12
Iteration 13
Iteration 14
Iteration 15
Iteration 16
Iteration 17
```

```
Iteration 18
Iteration 19
Sample AMQSFHAC end
```

検査で問題が検出されると、出力で障害が報告されます。 検査を実行したときに、MQRC_CALL_INTERRUPTED によって "Resolving to backed out" が報告される場合があります。 これが結果に影響することはありません。 これが報告されるかどうかは、ディスクへの書き込みがネットワーク・ファイル・ストレージによって、障害発生の前にコミットされたか、後にコミットされたかに応じて異なります。

関連資料

amqmfsck (ファイル・システム検査)

高可用性のサンプル・プログラム

Multi IBM MQ ファイルの共有 (Multiplatforms)

IBM MQ ファイルの中には、アクティブ・キュー・マネージャーだけがアクセスするファイルもあれば、共有されるファイルもあります。

IBM MQ ファイルは、プログラム・ファイルとデータ・ファイルに分かれています。 プログラム・ファイルは通常、IBM MQ を実行している各サーバー上にローカルにインストールされます。 キュー・マネージャーは、デフォルト・データ・ディレクトリー内のデータ・ファイルおよびディレクトリーへのアクセスを共有します。 キュー・マネージャーは、126 ページの図 32 に示す qmgrs と log の各ディレクトリーに含まれる独自のキュー・マネージャー・ディレクトリー・ツリーに対しては排他的アクセスを要求します。

126 ページの図 32 は、IBM MQ ディレクトリー構造の概要図です。 これは、キュー・マネージャー間で共有し、リモートにすることができるディレクトリーを示しています。 詳細はプラットフォームごとに異なります。 点線は構成可能なパスを表します。

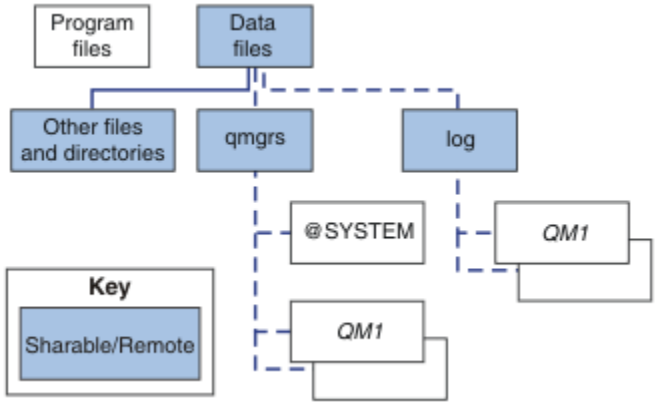


図 *32. IBM MQ* ディレクトリー構造の全体図

プログラム・ファイル

プログラム・ファイル・ディレクトリーは通常、デフォルトの位置に残され、ローカルであり、サーバー上のすべてのキュー・マネージャーによって共有されます。

データ・ファイル

通常、データ・ファイル・ディレクトリーはデフォルトの場所である、AIX and Linux システム上の /var/mqm にあり、Windows 上のインストールに構成可能です。 これはキュー・マネージャー間で共有されます。 デフォルトの位置をリモートにすることはできますが、IBM MQ の異なるインストール済み環境間では共有されません。 IBM MQ 構成内の DefaultPrefix 属性は、このパスを指します。

qmgrs

キュー・マネージャー・データの場所を指定するには、2 つの代替方法があります。

Prefix 属性の使用

Prefix 属性は、qmgrs ディレクトリーの場所を指定します。 IBM MQ は、キュー・マネージャー名からキュー・マネージャー・ディレクトリー名を構成し、それを qmgrs ディレクトリーのサブディレクトリーとして作成します。

Prefix 属性は、 mqs.ini ファイルの QueueManager スタンザ にあり、 「すべてのキュー・マネージャー」スタンザの **DefaultPrefix** 属性の値から継承されます。 デフォルトでは管理を簡素化するために、キュー・マネージャーは通常、同じ qmgrs ディレクトリーを共有します。

キュー・マネージャーの qmgrs ディレクトリーの場所を変更する場合、 その **Prefix** 属性の値を変更する必要があります。

AIX and Linux プラットフォームの場合、 126 ページの図 32 内の QM1 ディレクトリーの **Prefix** 属性は以下のようになります。

```
Prefix=/var/mqm
```

DataPath 属性の使用

DataPath 属性は、キュー・マネージャー・データ・ディレクトリーの場所を指定します。

DataPath 属性は、キュー・マネージャー・データ・ディレクトリーの名前を含めて完全なパスを指定します。 **DataPath** 属性は、キュー・マネージャー・データ・ディレクトリーへの不完全なパスを指定する **Prefix** 属性とは異なります。

DataPath 属性が指定されている場合は、mqs.ini ファイルの QueueManager スタンザ に配置されます。 この属性が指定されている場合、**Prefix** 属性のどの値よりも優先されます。

キュー・マネージャーのキュー・マネージャー・データ・ディレクトリーの場所を変更する場合、DataPath 属性の値を変更する必要があります。

Linux または AIX プラットフォームの場合、 126 ページの図 32 内の QM1 ディレクトリーの DataPath 属性は以下のとおりです。

```
DataPath=/var/mqm/qmgrs/QM1
```

log

ログ・ディレクトリーは、キュー・マネージャー構成の 「ログ・スタンザ」 でキュー・マネージャーごとに個別に指定されます。 キュー・マネージャーの構成は qm.ini にあります。

DataPath/*QmgrName*/@IPCC サブディレクトリー

DataPath/*QmgrName*/@IPCC サブディレクトリーは、共用ディレクトリー・パスにあります。 これらは IPC ファイル・システム・オブジェクト用のディレクトリー・パスを構成するために使用されます。 複数のシステム間でキュー・マネージャーを共有する場合、キュー・マネージャーの名前空間を区別する必要があります。

IPC ファイル・システム・オブジェクトは、システムによって区別する必要があります。 キュー・マネージャーが実行されるシステムごとに、1 つのサブディレクトリーが ディレクトリー・パスに追加されます (127 ページの図 33 を参照)。

```
DataPath/QmgrName/@IPCC/esem/myHostName/
```

図 *33. IPC* サブディレクトリーの例

myHostName は、オペレーティング・システムによって返されるホスト名の先頭の最大 20 文字です。システムによっては、切り捨て前のホスト名は最大 64 文字の長さの場合があります。 *myHostName* の生成値が原因で、以下の 2 つの理由で問題が発生する場合があります。

1. 最初の 20 文字が固有のものでない。
2. ホスト名が、同じホスト名をシステムに割り振るとは限らない DHCP アルゴリズムによって生成された。

このような場合は、環境変数 **MQS_IPC_HOST** を使用して *myHostName* を設定します。 128 ページの図 34 を参照してください。

```
export MQS_IPC_HOST= myHostName
```

図 *34.* 例: ***MQS_IPC_HOST*** の設定

その他のファイル およびディレクトリー

その他のファイルおよびディレクトリー (トレース・ファイルを含むディレクトリーや共通エラー・ログなど) は通常、ローカル・ファイル・システムで共有され、保持されます。

共有ファイル・システムのサポートにより、 IBM MQ はファイル・システム・ロックを使用してこれらのファイルへの排他的アクセスを管理します。 ファイル・システム・ロックでは、 特定のキュー・マネージャーのインスタンスのうち、一度に 1 つのみをアクティブにできます。

特定のキュー・マネージャーの最初のインスタンスを開始すると、 そのインスタンスは自身のキュー・マネージャー・ディレクトリーの所有権を獲得します。 2 番目のインスタンスを開始する場合、そのインスタンスは最初のインスタンスが停止している場合にのみ所有権を取得できます。 最初のキュー・マネージャーがまだ実行中の場合、2 番目のインスタンスは開始に失敗し、キュー・マネージャーが他の場所で実行されていることが報告されます。 最初のキュー・マネージャーがすでに停止していれば、2 番目のキュー・マネージャーが キュー・マネージャーのファイルの所有権を引き継ぎ、実行中のキュー・マネージャーになります。

2 番目のキュー・マネージャーが最初のキュー・マネージャーから引き継ぐ手順は、自動化できます。 別のキュー・マネージャーが引き継ぐことを許可する `strmqm -x` オプションを指定して、最初のキュー・マネージャーを開始します。 その場合、2 番目のキュー・マネージャーは、 最初のキュー・マネージャーのファイルがアンロックされるまで待機してから、キュー・マネージャーのファイルの所有権を引き継いで始動を試みます。

Linux AIX AIX and Linux システムでのディレクトリー構造

AIX and Linux システム上の IBM MQ ディレクトリー構造は、管理を容易にし、パフォーマンスを向上させ、信頼性を向上させるために、異なるファイル・システムにマップすることができます。

IBM MQ の柔軟なディレクトリー構造を使用して、複数インスタンス・キュー・マネージャーの実行にファイル共有システムを活用します。

コマンド **crtmqm** *QM1* を使用して、129 ページの図 35 に示すディレクトリー構造を作成します。ここで、R は製品のリリースです。 これは、 IBM MQ システムで作成されるキュー・マネージャーの標準的なディレクトリー構造です。 一部のディレクトリー、ファイル、および .ini 属性設定は、分かりやすくするために省略しています。また、キュー・マネージャー名はマングリングによって変更される場合があります。ファイル・システムの名前は、各種のシステムで異なります。

標準インストールでは、作成するすべてのキュー・マネージャーが、ローカル・ファイル・システム上の共通 `log` ディレクトリーおよび `qmgrs` ディレクトリーを指しています。 複数インスタンス構成では、`log` ディレクトリーと `qmgrs` ディレクトリーは、IBM MQ の別のインストールと共有されるネットワーク・ファイル・システム上にあります。

V 9.4.2 ネイティブ HA CRR 構成には、リベース操作中に使用されるバックアップログディレクトリもあります。 リベースが発生すると、既存のログディレクトリは *QMname*-backup にリネームされ、同じ名前と権限でログディレクトリが再作成され、リベースのログデータが保存されます。 キューマネージャーの作成時にカスタムログディレクトリが指定されていた場合（またはその後）、バックアップログディレクトリは *customlogname*-backup という名前になります。 バックアップディレクトリはリベースが成功し、もはや不要となった後に削除されます。手動で削除しないでください。

129 ページの図 35 は、 IBM MQ v7.R on AIX (R は製品のリリース)。 代わりの複数インスタンス構成の例については、 134 ページの『AIX and Linux システムでのディレクトリー構造の例』を参照してください。

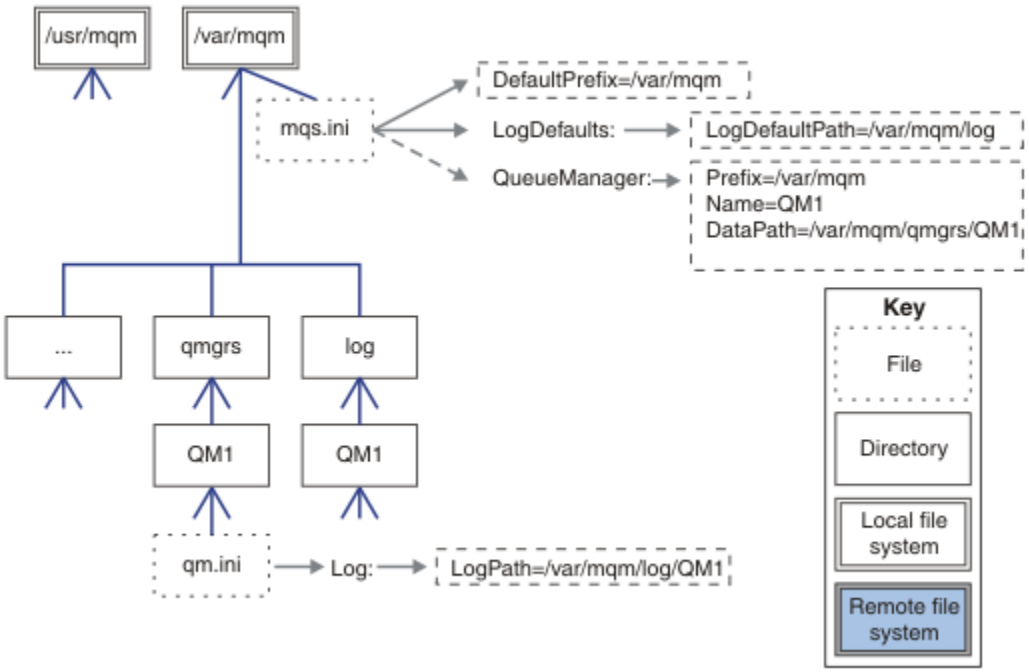


図 *35. AIX and Linux* システムのデフォルトの *IBM MQ* ディレクトリー構造の例

製品の実行ファイルは、デフォルトで、 AIX の /usr/mqm と、その他のシステムの /opt/mqm にインストールされます。 作業ディレクトリは、 /var/mqm ディレクトリの下に作成される。

注 : IBM MQ をインストールする前に /var/mqm ファイル・システムを作成した場合は、mqm ユーザーに完全なディレクトリー許可 (例えば、ファイル・モード 755) があることを確認してください。

注 : /var/mqm/errors ディレクトリーは、キュー・マネージャーによって生成される FFDC で /var/mqm を含むファイル・システムが満杯になるのを防ぐために、別個のファイル・システムにする必要があります。

詳しくは、AIX and Linux システムでのファイル・システムの作成を参照してください。

log および qmgrs ディレクトリーは、mqs.ini ファイル内の LogDefaultPath 属性および DefaultPrefix 属性のデフォルト値によって定義されているデフォルトの場所に表示されます。 キュー・マネージャーが作成されると、デフォルトで、キュー・マネージャーのデータ・ディレクトリーが *DefaultPrefix*/qmgrs 内に作成され、ログ・ファイル・ディレクトリーが *LogDefaultPath*/log に作成されます。 LogDefaultPath および DefaultPrefix は、デフォルトでキュー・マネージャーおよびログ・ファイルが作成される位置にのみ影響します。 キュー・マネージャー・ディレクトリーの実際の場所は、mqs.ini ファイルに保存され、ログ・ファイル・ディレクトリーの場所は qm.ini ファイルに保存されます。

キュー・マネージャーのログ・ファイル・ディレクトリーは、LogPath 属性の qm.ini ファイル内に定義されています。 **crtmqm** コマンドで -ld オプションを使用して、キュー・マネージャーの LogPath 属性を設定します (例: **crtmqm** -ld *LogPath* QM1)。 ld パラメーターを省略すると、代わりに LogDefaultPath の値が使用されます。

キュー・マネージャーのデータ・ディレクトリーは、mqs.ini ファイル内の QueueManager スタンザの DataPath 属性に定義されています。 **crtmqm** コマンドで -md オプションを使用して、キュー・マネージャーの DataPath を設定します (例: **crtmqm** - md *DataPath* QM1)。 md パラメーターを省略すると、代わりに DefaultPrefix または Prefix 属性の値が使用されます。 Prefix は DefaultPrefix より優先されます。

通常、ログ・ディレクトリーとデータ・ディレクトリーの両方を 1 つのコマンドで指定して、QM1 を作成します。

```
 crtmqm
-md DataPath -ld
LogPath QM1
```

キュー・マネージャーの停止時に qm.ini ファイル内の DataPath 属性および LogPath 属性を編集することにより、既存のキュー・マネージャーのキュー・マネージャー・ログ・ディレクトリーとデータ・ディレクトリーの場所を変更できます。

/var/mqm の他のすべてのディレクトリーのパスなど、errors ディレクトリーのパスは変更できません。ただし、ディレクトリーを別のファイル・システムにマウントしたり、 別のディレクトリーにシンボリック・リンクしたりできます。

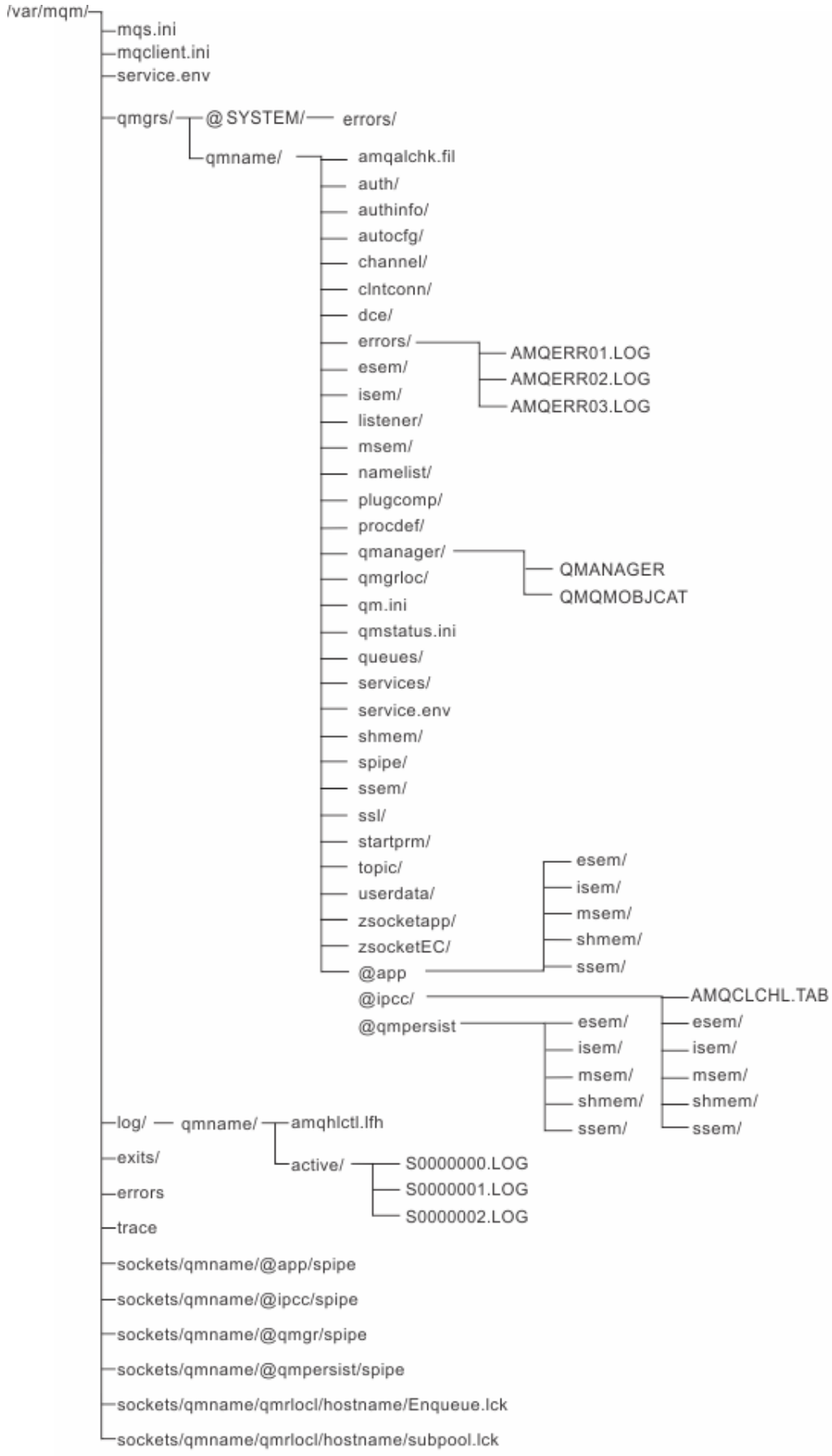
Linux AIX *AIX and Linux* システムでのディレクトリーの内容

キュー・マネージャーと関連付けられたディレクトリーの内容。

製品ファイルの場所については、インストール場所の選択を参照してください。

その他のディレクトリー構成についての詳細は、116 ページの『ファイル・システム・サポートの計画 (Multiplatforms)』を参照してください。

以下のディレクトリー構造は、キュー・マネージャーがしばらく使用された後の IBM MQ の代表的な構造です。 実際の構造は、キュー・マネージャーに対して行われた操作によって異なります。


/var/mqm/
mqs.ini
mqclient.ini
service.env
qmgrs/
@SYSTEM/
errors/
qmname/
amqalchk.fil
auth/
authinfo/
autocfg/
channel/
clntconn/
dce/
errors/
AMQERR01.LOG
AMQERR02.LOG
AMQERR03.LOG
esem/
isem/
listener/
msem/
namelist/
plugcomp/
procdef/
qmanager/
QMANAGER
QMQMOBJCAT
qmgrloc/
qm.ini
qmstatus.ini
queues/
services/
service.env
shmem/
spipe/
ssem/
ssl/
startprm/
topic/
userdata/
zsocketapp/
zsocketEC/
@app
esem/
isem/
msem/
shmem/
ssem/
@ipcc/
AMQCLCHL.TAB
@qmpersist
esem/
isem/
msem/
shmem/
ssem/
esem/
isem/
msem/
shmem/
ssem/
log/
qmname/
amqhlctl.lfh
active/
S0000000.LOG
S0000001.LOG
S0000002.LOG
exits/
errors
trace
sockets/qmname/@app/spipe
sockets/qmname/@ipcc/spipe
sockets/qmname/@qmgr/spipe
sockets/qmname/@qmpersist/spipe
sockets/qmname/qmrlocl/hostname/Enqueue.lck
sockets/qmname/qmrlocl/hostname/subpool.lck

/var/mqm/

/var/mqm ディレクトリーには、個々のキュー・マネージャーには適用されないが、IBM MQ のインストール全体に適用される構成ファイルと出力ディレクトリーが含まれています。

表 *13. AIX and Linux の /var/mqm* ディレクトリーの内容の説明	
ディレクトリーまたはファイル名	**内容**
mqs.ini	キュー・マネージャーの開始時に読み取られる、IBM MQ のインストール済み環境全体に関する構成ファイル。 ファイル・パスは、**AMQ_MQS_INI_LOCATION** 環境変数を使用して変更可能です。 これが、**strmqm** コマンドが実行されるシェルで設定され、エクスポートされていることを確認します。
mqclient.ini	IBM MQ MQI client・プログラムが読み取るデフォルトのクライアント構成ファイル。 ファイル・パスは、**MQCLNTCF** 環境変数を使用して変更可能です。
service.env	マシンを有効範囲とした、サービス・プロセスに対する環境変数が含まれています。 ファイル・パスは固定です。
errors/	マシンを有効範囲としたエラー・ログ、および FFST ファイル。 ディレクトリー・パスは固定です。 FFST: IBM MQ for UNIX および Linux システムも参照してください。
sockets/	各キュー・マネージャーに関する情報が含まれています。システムのみが使用します。
トレース/	トレース・ファイル。 ディレクトリー・パスは固定です。
Web/	mqweb サーバー・ディレクトリー。
出口/	ユーザー・チャネル出口プログラムが含まれているデフォルトのディレクトリー。 場所は mqs.ini ファイルの ApiExit スタンザで変更可能です。
exits64/	

/var/mqm/qmgrs/*qmname*/

/var/mqm/qmgrs/*qmname*/ には、キュー・マネージャーのディレクトリーおよびファイルが含まれています。 このディレクトリーは、アクティブなキュー・マネージャー・インスタンスからの排他的アクセスによりロックされます。 このディレクトリー・パスは、mqs.ini ファイルで直接変更することも、**crtmqm** コマンドの **md** オプションを使用して変更することもできます。

表 *14. AIX and Linux の /var/mqm/qmgrs/qmname* ディレクトリーの内容の説明	
ディレクトリーまたはファイル名	**内容**
qm.ini	キュー・マネージャーの開始時に読み取られるキュー・マネージャー構成ファイル。

表 14. AIX and Linux の /var/mqm/qmgrs/qmname ディレクトリーの内容の説明 (続き)	
ディレクトリーまたはファイル名	**内容**
errors/	キュー・マネージャーを有効範囲としたエラー・ログ。 *qmname* = `@system` には、不明または使用不可のキュー・マネージャーのチャネル関連メッセージが含まれます。
@ipcc/ **AMQCLCHL.TAB**	IBM MQ サーバーが作成し、IBM MQ MQI client・プログラムが読み取るデフォルトのクライアント・チャネル制御テーブル。 ファイル・パスは、**MQCHLLIB** 環境変数および **MQCHLTAB** 環境変数を使用して変更可能です。
qmanager	キュー・マネージャー・オブジェクト・ファイル: QMANAGER キュー・マネージャー・オブジェクト・カタログ: QMQMOBJCAT
authinfo/	キュー・マネージャー内で定義された各オブジェクトは、これらのディレクトリーにあるファイルに関連付けられています。 ファイル名は、定義名とほぼ一致します。『IBM MQ ファイル名についての理解』を参照してください。
チャネル/	
clntconn/	
リスナー/	
namelist/	
procdef/	
キュー/	
services/	
トピック/	
...	IBM MQ が使用するその他のディレクトリー (@ipcc など) は、IBM MQ のみが変更できます。
ユーザー・データ/	アプリケーションの永続的状態を保管するために使用できます (キュー・マネージャーを別のノードに移動するときに RDQM で使用できます。アプリケーションの永続ステータスの保管を参照。)
***DataPath*\autocfg**	自動構成に使用される

/var/mqm/log/*qmname*/

/var/mqm/log/*qmname*/ には、キュー・マネージャーのログ・ファイルが含まれています。 このディレクトリーは、アクティブなキュー・マネージャー・インスタンスからの排他的アクセスによりロックされます。 このディレクトリー・パスは、qm.ini ファイルで変更することも、**crtmqm** コマンドの **ld** オプションを使用して変更することもできます。

表 15. AIX and Linux の /var/mqm/log/qmname ディレクトリーの内容の説明	
ディレクトリーまたはファイル名	**内容**
amqhlctl.lfh	ログ制御ファイル。
active/	このディレクトリーは、S0000000.LOG、S0000001.LOG、S0000002.LOG などの番号が付いたログ・ファイルを含んでいます。

/opt/mqm

/opt/mqm は、デフォルトで、ほとんどのプラットフォームのインストール・ディレクトリーです。 自社で使用しているプラットフォームで必要なインストール・ディレクトリーのスペース量について詳しくは、113 ページの『ディスク・スペース要件 (Multiplatforms)』を参照してください。

Linux AIX *AIX and Linux* システムでのディレクトリー構造の例

AIX and Linux システムでの代替ファイル・システム構成の例。

さまざまな方法で IBM MQ ディレクトリー構造をカスタマイズすることで、多くの異なる目標を達成することができます。

- qmgrs ディレクトリーと log ディレクトリーをリモート共有ファイル・システム上に配置して、マルチインスタンス・キュー・マネージャーを構成します。
- データ・ディレクトリーおよびログ・ディレクトリーに別個のファイル・システムを使用し、それぞれのディレクトリーを異なるディスクに割り振り、入出力競合を削減することでパフォーマンスを向上させます。
- パフォーマンスにより大きな影響を与えるディレクトリーに高速ストレージ・デバイスを使用します。 多くの場合、デバイスの取り付けがローカルか、リモートかではなく、 物理デバイスの待ち時間が、持続メッセージングのパフォーマンスにおけるさらに重要な要素となります。 以下に、 パフォーマンスが重要視されるディレクトリーをその重要度の高い順にリストします。

  1. log
  2. qmgrs
  3. /usr/mqm を含むその他のディレクトリー

- 例えば、冗長ディスク・アレイなどの良好な回復力を備えたストレージに割り振られるファイル・システム上に、qmgrs ディレクトリーおよび log ディレクトリーを作成します。
- ネットワーク・ファイル・システムに関連したエラーをログに記録するように、ネットワーク・ファイル・システムではなく、var/mqm/errors に共通エラー・ログをローカルに保管することをお勧めします。

135 ページの図 36 は、代替 IBM MQ ディレクトリー構造の派生元となるテンプレートです。 テンプレートでは、点線によって構成可能なパスを表しています。 この例の点線は、AMQ_MQS_INI_LOCATION 環境変数に保管されている構成情報、および mqs.ini ファイルと qm.ini ファイルに対応する実線に置き換えられます。

注: パス情報は、mqs.ini ファイルまたは qm.ini ファイルに表示されるとおりに示されます。 **crtmqm** コマンドでパス・パラメーターを指定する場合は、キュー・マネージャー・ディレクトリーの名前を省略します。キュー・マネージャー名は、IBM MQ によってパスに追加されます。

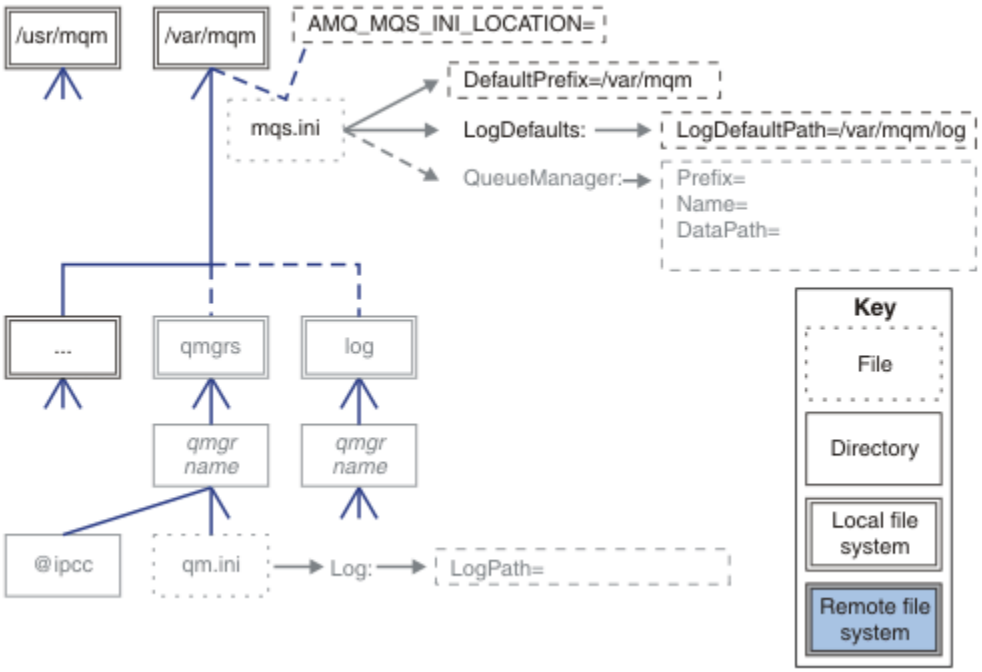


図 *36.* ディレクトリー構造パターン・テンプレート

IBM MQ の標準的なディレクトリー構造

136 ページの図 37 は、IBM MQ でコマンド **crtmqm** *QM1* を発行して作成されたデフォルト・ディレクトリー構造です。

mqs.ini ファイルは、DefaultPrefix の値を参照することによって作成される QM1 キュー・マネージャーのスタンザがあります。 qm.ini ファイルの Log スタンザには、mqs.ini の LogDefaultPath への参照によって設定された LogPath の値が含まれています。

DataPath および LogPath のデフォルト値を指定変更するには、 オプションの **crtmqm** パラメーターを使用します。

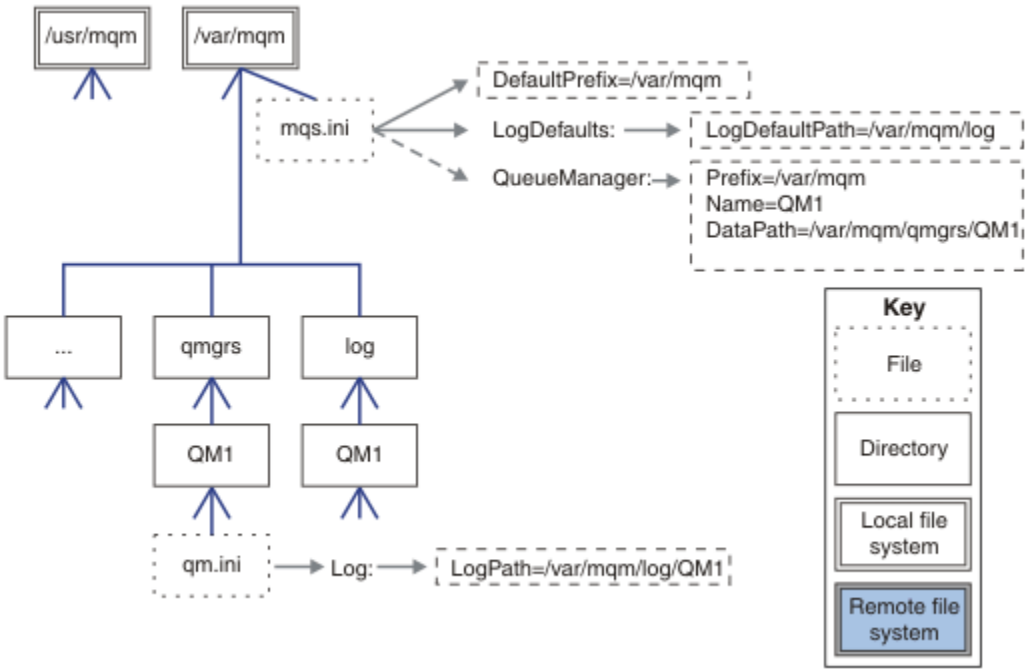


図 *37. AIX and Linux* システムのデフォルトの *IBM MQ* ディレクトリー構造の例

デフォルトの qmgrs ディレクトリーと log ディレクトリーを共有します。

137 ページの『すべて共有』の代わりに、qmgrs ディレクトリーと log ディレクトリーを別々に共有することもできます (136 ページの図 38)。 この構成では、デフォルトの mqs.ini がローカル /var/mqm ファイル・システムに保管されるため、AMQ_MQS_INI_LOCATION を設定する必要はありません。 ファイルおよびディレクトリー (mqclient.ini や mqserver.ini など) も共有されません。

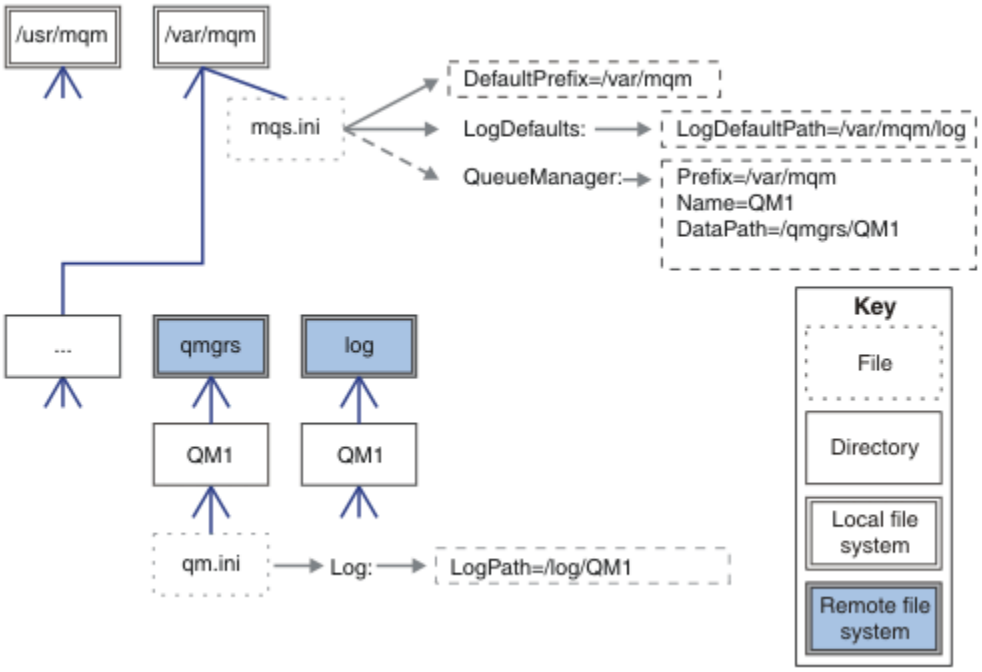


図 *38. qmgrs* ディレクトリーおよび *log* ディレクトリーの共有

名前付き qmgrs と log ディレクトリーの共有

137 ページの図 39 の構成では、/ha という名前の共通名付きリモート共有ファイル・システムに log および qmgrs が配置されています。 2 つの異なる方法で同じ物理構成を作成できます。

1. LogDefaultPath=/ha を設定してから、コマンド **crtmqm** -md */ha/qmgrs* QM1 を実行します。 結果は、137 ページの図 39 に示すとおりになります。
2. デフォルト・パスを未変更のまま、コマンド **crtmqm** -ld */ha/log* -md */ha/qmgrs* QM1 を実行します。

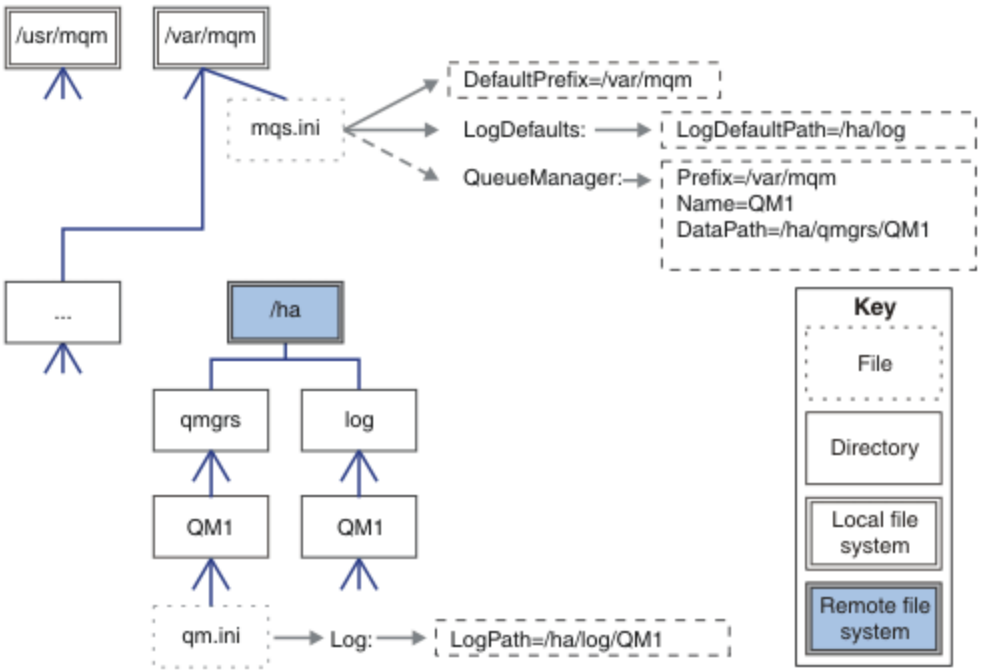


図 *39.* 名前付き *qmgrs* と *log* ディレクトリーの共有

すべて共有

138 ページの図 40 は、 高速ネットワーク・ファイル・ストレージを備えたシステム用の簡単な構成です。

/var/mqm をリモート共有ファイル・システムとしてマウントします。 デフォルトでは、QM1 を開始すると、/var/mqm が検索され、これが共有ファイル・システム上で検出されて、/var/mqm 内の mqs.ini ファイルが読み取られます。 すべてのサーバー上のキュー・マネージャーに単一の /var/mqm/mqs.ini ファイルを使用するのではなく、 別個の mqs.ini ファイルを指すように、 各サーバーで AMQ_MQS_INI_LOCATION 環境変数を設定できます。

注 : /var/mqm/errors/ 内の汎用エラー・ファイルの内容は、異なるサーバー上のキュー・マネージャー間で共有されます。

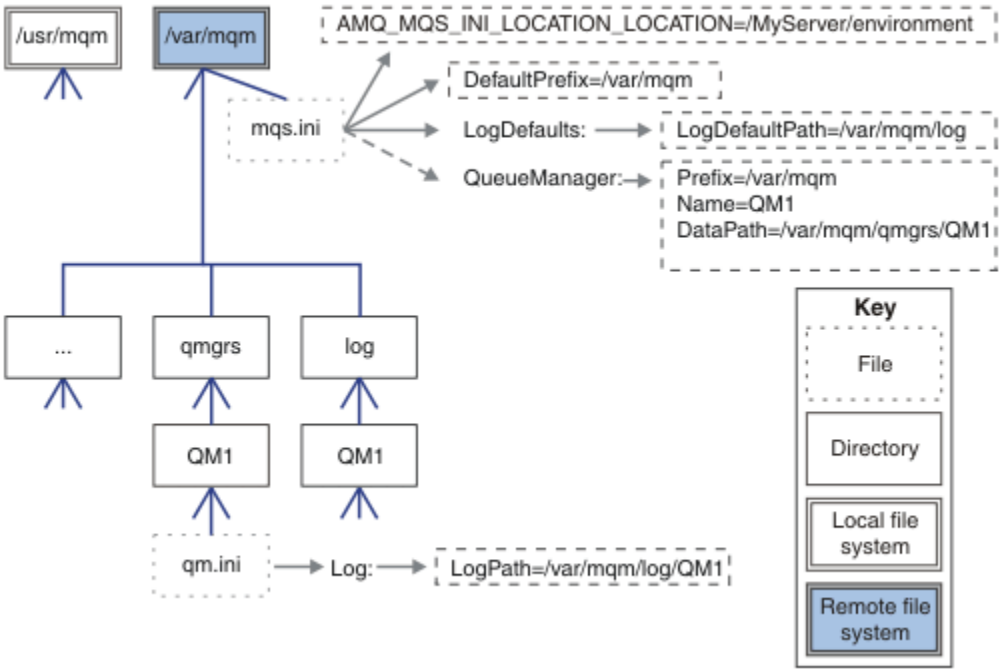


図 *40.* すべて共有

これは、複数インスタンス・キュー・マネージャーに使用することはできません。 これは、複数インスタンス・キュー・マネージャー内の各ホストが、ローカル・データ (セマフォーや共有メモリーなど) を追跡するために独自の `/var/mqm` のローカル・コピーを持つ必要があるためです。 これらのエンティティーは、ホスト間で共有できません。

Windows Windows システムでのディレクトリー構造

Windows 上のキュー・マネージャー構成情報およびディレクトリーを検出する方法を示します。

IBM MQ for Windows インストール済み環境のデフォルト・ディレクトリーは、次のとおりです。

プログラム・ディレクトリー
```
C:&#xa5;Program Files&#xa5;IBM&#xa5;MQ
```
データ・ディレクトリー
```
C:\ProgramData\ IBM \ MQ
```

重要: Windows Windows インストールの場合、ディレクトリーは記載されているとおりです。 ただし、レジストリー項目またはキュー・マネージャー、あるいはその両方が現在も含まれている、以前の製品のインストール済み環境が存在する場合を除きます。 この場合、新しいインストールでは、それまでのデータ・ディレクトリーの場所が使用されます。 詳しくは、プログラムおよびデータのディレクトリーの場所を参照してください。

使用されているインストール・ディレクトリーとデータ・ディレクトリーを調べるには、dspmqver コマンドを実行します。

インストール・ディレクトリーは **InstPath** フィールドに記載され、データ・ディレクトリーは **DataPath** フィールドに記載されます。

dspmqver コマンドを実行すると、例えば次のような情報が表示されます。

```
>dspmqver
Name:        IBM MQ
Version:     9.0.0.0
Level:       p900-L160512.4
BuildType:   IKAP - (Production)
Platform:    IBM MQ for Windows (x64 platform)
Mode:        64-bit
O/S:         Windows 7 Professional x64 Edition, Build 7601: SP1
```

```
InstName:    Installation1
InstDesc:
Primary:     Yes
InstPath:    C:\Program Files\IBM\MQ
DataPath:    C:\ProgramData\IBM\MQ
MaxCmdLevel: 900
LicenseType: Production
```

複数インスタンス・キュー・マネージャー

複数インスタンス・キュー・マネージャーを構成するには、ログ・ディレクトリーおよびデータ・ディレクトリーをネットワーク・ストレージ、できればキュー・マネージャーのインスタンスを実行しているサーバーとは別のサーバー上に置く必要があります。

crtmqm コマンドでは、2 つのパラメーター、**-md** および **-ld** が提供されており、キュー・マネージャー・データ・ディレクトリーおよびログ・ディレクトリーの場所を指定するのを容易にしています。 **-md** パラメーターを指定すると、以下の 4 つの効果があります。

1. mqs.ini スタンザ QueueManager*QmgrName* には、キュー・マネージャーのデータ・ディレクトリーを指す *DataPath* という新しい変数が含まれています。 *Prefix* 変数とは異なり、パスにはキュー・マネージャー・ディレクトリーの名前が含まれます。
2. mqs.ini ファイルに格納されるキュー・マネージャー構成情報が、*Name*、*Prefix*、*Directory*、および *DataPath* に減らされます。

Windows ディレクトリーの内容

IBM MQ ディレクトリーの場所および内容をリストします。

IBM MQ 構成には、主に次の 3 つのファイルおよびディレクトリーのセットがあります。

1. 実行可能ファイル、およびその他の読み取り専用ファイル (保守が適用される場合のみ更新されます)。以下に例を示します。
   - README ファイル
   - IBM MQ エクスプローラーのプラグイン・ファイルおよびヘルプ・ファイル
   - ライセンス・ファイル

   これらのファイルについては、139 ページの表 16 で説明されています。
2. 特定のキュー・マネージャーに固有でない、潜在的に変更可能なファイルおよびディレクトリー。 これらのファイルおよびディレクトリーについては、140 ページの表 17 で説明されています。
3. サーバー上の各キュー・マネージャーに固有のファイルおよびディレクトリー。 これらのファイルおよびディレクトリーについては、141 ページの表 18 で説明されています。

リソース・ディレクトリーおよびファイル

リソース・ディレクトリーおよびファイルには、キュー・マネージャーを実行するためのすべての実行可能コードとリソースが含まれています。 インストール固有の IBM MQ 設定レジストリ・キーにある変数 *FilePath* には、リソース・ディレクトリへのパスが格納されています。

表 *16. FilePath* ディレクトリー内のディレクトリーおよびファイル

ファイル・パス	内容
FilePath\bin	コマンドおよび DLL
FilePath\bin64	コマンドおよび DLL (64 ビット)
FilePath\conv	データ変換テーブル
FilePath\doc	ウィザード・ヘルプ・ファイル
FilePath\MQExplorer	エクスプローラーおよびエクスプローラー・ヘルプ Eclipse プラグイン

表 16. *FilePath* ディレクトリー内のディレクトリーおよびファイル (続き)	
ファイル・パス	**内容**
FilePath\gskit8	グローバル・セキュリティー・キット
FilePath\java	Java などのリソースがある。 JRE
FilePath\licenses	ライセンス情報
FilePath\Non_IBM_License	ライセンス情報
FilePath\properties	内部的に使用
FilePath\Tivoli	
FilePath\tools	開発リソースおよびサンプル
FilePath\web	IBM MQ Console および REST API のインストール・コンポーネントの編集不可ファイルのファイル構造で説明しています。
FilePath\Uninst	内部的に使用
FilePath\README.TXT	README ファイル

キュー・マネージャーに固有でないディレクトリー

ディレクトリーの中には、特定のキュー・マネージャーに固有でないファイル (トレース・ファイルやエラー・ログなど) が含まれているものがあります。 *DefaultPrefix* 変数には、そうしたディレクトリーへのパスが含まれています。 *DefaultPrefix* は、AllQueueManagers スタンザの一部です。

表 17. *DefaultPrefix* ディレクトリー内のディレクトリーおよびファイル	
ファイル・パス	**内容**
DefaultPrefix\config	内部的に使用
DefaultPrefix\conv	ccsid_part2.tbl および ccsid.tbl data 変換制御ファイルについて詳しくは、『データ変換』 で説明されています。
DefaultPrefix\errors	非キュー・マネージャーのエラー・ログ AMQERR *nn*.LOG
DefaultPrefix\exits	チャネル出口プログラム
DefaultPrefix\exits64	チャネル出口プログラム (64 ビット)
DefaultPrefix\ipc	使用されません
DefaultPrefix\qmgrs	141 ページの表 18 で説明されています
DefaultPrefix\trace	トレース・ファイル
DefaultPrefix\web	IBM MQ Console および REST API のインストール・コンポーネントのユーザー編集可能ファイル用のファイル構造で説明しています。
DefaultPrefix\amqmjpse.txt	内部的に使用

キュー・マネージャー・ディレクトリー

キュー・マネージャーを作成するときに、キュー・マネージャーに固有の新規のディレクトリーのセットが作成されます。

-md *filepath* パラメーターを使用してキュー・マネージャーを作成する場合、パスは mqs.ini ファイルのキュー・マネージャー・スタンザの *DataPath* 変数に保管されます。 **-md** *filepath* パラメータを設定せずにキューマネージャを作成した場合、キューマネージャディレクトリは *DefaultPrefix* に格納されてい

るパスに作成され、そのパスは mqs.ini ファイルのキューマネージャスタンザ内の *Prefix* 変数にコピーされます。

表 *18. DataPath* ディレクトリーおよび *Prefix**qmgrs**QmgrName* ディレクトリー内のディレクトリーおよびファイル

ファイル・パス	**内容**
DataPath\@ipcc	AMQCLCHL.TAB のデフォルトの場所 (クライアント接続テーブル)。
DataPath\authinfo	内部的に使用
DataPath\channel	
DataPath\clntconn	
DataPath\errors	エラー・ログ AMQERR *nn*.LOG
DataPath\listener	内部的に使用
DataPath\namelist	
DataPath\plugcomp	
DataPath\procdef	
DataPath\qmanager	
DataPath\queues	
DataPath\services	
DataPath\ssl	
DataPath\startprm	
DataPath\topic	
DataPath\active	
DataPath\active.dat	
DataPath\amqalchk.fil	
DataPath\master	
DataPath\master.dat	
DataPath\qm.ini	キュー・マネージャー構成
DataPath\qmstatus.ini	キュー・マネージャー状況
DataPath\userdata	アプリケーションの永続状態を保管するために使用できます。
Prefix\qmgrs*QmgrName*	内部的に使用
Prefix\qmgrs\@SYSTEM	使用されません
Prefix\qmgrs\@SYSTEM\errors	
*DataPath**autocfg*	自動構成に使用される

IBM i IBM i でのディレクトリー構造

IFS について説明し、サーバー、クライアント、および Java の IBM MQ IFS ディレクトリー構造について説明します。

統合ファイル・システム (IFS) は、IBM i の一部分であり、このサーバーに保管される全情報に対して統合的な構造を提供すると同時に、 パーソナル・コンピューターや AIX and Linux オペレーティング・システム同様の ストリーム入出力およびストレージ管理をサポートします。

IBM i では、ディレクトリー名は文字 @ (at) ではなく文字 & (ampersand) で始まります。 例えば、IBM i 上の @system は &system です。

IBM MQ サーバーの IFS ルート・ファイル・システム

IBM MQ Server for IBM i をインストールすると、IFS ルート・ファイル・システム内に以下のディレクトリーが作成されます。

ProdData:

概要

```
QIBM
   '-- ProdData
      '-- mqm
      '-- doc
      '-- inc
      '-- lib
      '-- samp
      '-- licenses
      '-- LicenseDoc
      '-- 5724H72_V8R0M0
```

/QIBM/ProdData/mqm
この下のサブディレクトリーに、C++ クラス、トレース・フォーマット・ファイル、ライセンス・ファイルなどの全製品データが入っています。 このディレクトリー内のデータは、製品をインストールするごとに削除され、置き換えられます。

/QIBM/ProdData/mqm/doc
CL コマンドに関する HTML 形式のコマンド解説書が、ここにインストールされます。

/QIBM/ProdData/mqm/inc
使用する C または C++ プログラムをコンパイルするためのヘッダー・ファイル。

/QIBM/ProdData/mqm/lib
MQ が使用する補助ファイル。

/QIBM/ProdData/mqm/samp
その他のサンプル。

/QIBM/ProdData/mqm/licenses
ライセンス・ファイル。 言語ごとに、LA_ *xx* および LI_ *xx* のような名前のファイルがあります。ここで、 *xx* は提供される各言語を表す 2 文字の言語 ID です。

以下のディレクトリーにも、使用許諾契約書ファイルが 入っています。

/QIBM/ProdData/LicenseDoc/5724H72_V8R0M0
ライセンス・ファイル。 これらのファイルには 5724H72_V8R0M0_ *xx* のような名前が付けられています。ここで、 *xx* は提供される各言語を表す 2 文字または 5 文字の言語 ID です。

UserData:

概要

```
QIBM
   '-- UserData
      '-- mqm
      '-- errors
      '-- trace
```

```
      '-- qmgrs
      '-- &system
      '-- qmgrname1
      '-- qmgrname2
      '-- and so on
```

/QIBM/UserData/mqm
この下のサブディレクトリーには、キュー・マネージャーに関する全ユーザー・データが 入っています。

製品をインストールすると、ディレクトリー /QIBM/UserData/mqm/ 内に mqs.ini ファイルが作成されます (ただし、このファイルが前のインストールによって既存の場合を除きます)。

キュー・マネージャーを作成すると、ディレクトリー /QIBM/UserData/mqm/qmgrs/*QMGRNAME*/ (*QMGRNAME* はキュー・マネージャー名) に qm.ini ファイルが作成されます。

ディレクトリー内のデータは、製品を削除しても維持されます。

IBM MQ MQI client 用の IFS ルート・ファイル・システム

IBM MQ MQI client for IBM i をインストールすると、IFS ルート・ファイル・システム内に以下のディレクトリーが作成されます。

ProdData:

概要

```
QIBM
   '-- ProdData
      '-- mqm
      '-- lib
```

/QIBM/ProdData/mqm
このディレクトリー下のサブディレクトリーに、全製品データが入っています。 このディレクトリー内のデータは、 製品を置換するごとに削除され、置き換えられます。

UserData:

概要

```
QIBM
   '-- UserData
      '-- mqm
      '-- errors
      '-- trace
```

/QIBM/UserData/mqm
このディレクトリー下のサブディレクトリーに、全ユーザー・データが入っています。

IBM MQ Java 用の IFS ルート・ファイル・システム

IBM MQ Java を IBM i にインストールすると、IFS ルート・ファイル・システム内に以下のディレクトリーが作成されます。

ProdData:

概要

```
QIBM
   '-- ProdData
      '-- mqm
      '-- java
      '--samples
```

```
        '-- bin
        '-- lib
```

/QIBM/ProdData/mqm/java
このディレクトリー下のサブディレクトリーに、Java クラスを含め、全製品データが入っています。 このディレクトリー内のデータは、 製品を置換するごとに削除され、置き換えられます。

/QIBM/ProdData/mqm/java/samples
この下のサブディレクトリーには、すべてのサンプル Java クラスおよびデータが入っています。

サーバーおよびクライアントのインストールによって作成されるライブラリー

IBM MQ サーバーまたはクライアントをインストールすると、 以下のライブラリーが作成されます。

- QMQM

  製品ライブラリー。

- QMQMSAMP

  サンプル・ライブラリー (サンプルのインストールを 選択した場合)。

- QMxxxx

  サーバーのみ。

  キュー・マネージャーを作成するたびに、IBM MQ は、自動的にその関連ライブラリーを QMxxxx (ここで、xxxx はそのキュー・マネージャーの名前から派生します) のような名前で作成します。 このライブラリーには、ジャーナルや関連する受信側など、 そのキュー・マネージャーに特有のオブジェクトが入っています。 デフォルトでは、このライブラリーの名前は、 キュー・マネージャーの名前に接頭部として文字 QM を付けて生成されます。 例えば、TEST という名前のキュー・マネージャーの場合、ライブラリーの名前は QMTEST になります。

  注: キュー・マネージャーを作成するときに、ライブラリー名を指定できます。 以下に例を示します。

  ```
  CRTMQM MQMNAME(TEST) MQMLIB(TESTLIB)
  ```

  WRKLIB コマンドを使用すると、 IBM MQ for IBM i の作成した全ライブラリーをリストできます。 キュー・マネージャーのライブラリーには、テキスト QMGR: QMGRNAME が表示されます。 コマンドの形式は次のとおりです。

  ```
  WRKLIB LIB(QM*)
  ```

  キュー・マネージャーに関連するこれらのライブラリーは、 製品を削除しても維持されます。

Multi MFT on Multiplatforms でのファイル・システム・サポートの計画

IBM MQ Managed File Transfer MFT エージェントを使用して、ファイル・システム上のファイルとの間でデータを転送できます。 さらに、エージェント内で実行されるリソース・モニターは、ファイル・システム上のファイルをモニターするように構成できます。

MFT には、これらのファイルがロックをサポートするファイル・システムに保管されるという要件があります。 これには次の 2 つの理由があります。

- エージェントは、ファイルからのデータの読み取り、またはファイルへのデータの書き込みを開始した後に、ファイルが変更されないようにロックします。
- リソース・モニターは、ロック・ファイルを使用して、他のプロセスがそれらを現在使用していないことを確認します。

エージェントおよびリソース・モニターは、 Java メソッド **FileChannel.tryLock()** を使用してロックを実行します。ファイル・システムは、この呼び出しを使用してロックを要求されたときにファイルをロックできなければなりません。

重要: 以下のファイル・システムは、 MFT の技術要件を満たしていないため、サポートされません。

- GlusterFS

- NFS バージョン 3

Multi 循環ロギングまたはリニア・ロギングの選択 (Multiplatforms)

IBM MQ」では、リカバリーログのロギング方法として、サーキュラー・ロギングとリニア・ロギングを選択することができる。 以下の情報は、両方のタイプの概要を示しています。

リカバリーログの詳細については、'ロギング：メッセージが失われないようにするを参照のこと。 IBM MQ、クラッシュのリカバリー後に必要なログ再生の量を制限するためにチェックポイントを使用する。 ログファイルをアクティブと呼ぶかどうかを制御する重要なイベントは、'checkpoint である。 トランザクションとチェックポイントの詳細については、完全なリカバリーを保証するためのチェックポイントの利用を参照のこと。

循環ロギングの利点

循環ロギングを使用する主な利点は、循環ロギングに以下の特長があることです。

- 管理が容易。

  一度ワークロードに対して循環ロギングを正しく構成すれば、その後の管理は不要です。 一方、リニア・ロギングの場合は、メディア・イメージを記録し、不要になったログ・エクステントをアーカイブまたは削除する必要があります。

- 優れたパフォーマンス

  循環ロギングではフォーマット済みのログ・エクステントを再使用できるので、リニア・ロギングよりもパフォーマンスが高くなります。 一方、リニア・ロギングの場合には、新しいログ・エクステントを割り当てて、それらをフォーマットする必要があります。

詳しくは、ログの管理を参照してください。

リニア・ロギングの利点

リニア・ロギングの主な利点は、より多くの障害に対して保護を提供できることです。

循環ロギングもリニア・ロギングも、破損したログや削除されたログ、アプリケーションや管理者によって削除されたメッセージやキューに対する保護は提供しません。

リニア・ロギングでは、被害を受けたオブジェクトをリカバリーすることが可能です (循環ロギングでは不可)。 つまり、リニア・ロギングは、破損したり削除されたりしたキュー・ファイルに対して保護を提供します。それらの被害を受けたキューをリニア・ログからリカバリーできるからです。 リニアログの自動メディアイメージングと自動ログ管理を設定することで、管理のオーバーヘッドを減らすことができます。 自動メディアイメージング および自動ログ管理を参照。

『停電や通信障害からのリカバリー』で説明しているように、循環もリニアも、停電や通信障害に対する保護を提供します。

その他の考慮事項

リニアまたは循環のどちらを選択するかは、必要な冗長性の程度によって決まります。

冗長性が高いほう (リニア・ロギング) を選択すると、パフォーマンス・コストと管理コストによるコストがかかります。

詳しくは、ログのタイプを参照してください。

AIX AIX 上の共有メモリー

AIX メモリー制限のために特定のアプリケーション・タイプで接続できない場合、大抵は環境変数 EXTSHM=ON を設定することによって解決できます。

AIX 上のいくつかの 32 ビット・プロセスで、IBM MQ キュー・マネージャーへ接続する機能に影響を及ぼすオペレーティング・システムの制限が存在する場合があります。 IBM MQ への各標準接続では共有メモ

リーを使用しますが、他の UNIX プラットフォームとは異なり、AIX では、32 ビット・プロセスで接続できる共有メモリー・セットは 11 個だけです。

ほとんどの 32 ビット・プロセスではこの制限は発生しませんが、メモリー所要量が多いアプリケーションは、理由コード 2102: MQRC_RESOURCE_PROBLEM で IBM MQ への接続に失敗する可能性があります。以下のアプリケーション・タイプで、このようなエラーが生じる場合があります。

- 32 ビット Java 仮想マシンで実行しているプログラム
- 大きいまたは非常に大きいメモリー・モデルを使用しているプログラム
- 多くのキュー・マネージャーまたはデータベースに接続しているプログラム
- それ自身の共有メモリー・セットに接続しているプログラム

AIX で提供されている、32 ビット・プロセス用の拡張共有メモリー・フィーチャーを使用することにより、より多くの共有メモリーを接続できます。 このフィーチャーを使用してアプリケーションを実行するには、キュー・マネージャーおよびプログラムを開始する前に、環境変数 EXTSHM=ON をエクスポートします。 ほとんどの場合、EXTSHM=ON フィーチャーを使用することによってこのエラーを防ぐことができますが、shmctl 関数の SHM_SIZE オプションを使用するプログラムとの互換性はありません。

IBM MQ MQI client・アプリケーションおよびすべての 64 ビット・プロセスは、この制限の影響を受けません。 それらは EXTSHM が設定されているかどうかに関係なく、IBM MQ キュー・マネージャーに接続できます。

Linux AIX IBM MQ と UNIX System V IPC リソース

キュー・マネージャーはいくつかの IPC リソースを使用します。 **ipcs -a** を使用して、どのリソースが使用されているかを調べます。

この情報は、 AIX and Linux システム上で稼働する IBM MQ にのみ適用されます。

IBM MQ は System V プロセス間通信 (IPC) リソース (セマフォー および共有メモリー・セグメント) を使用して、システム・コンポーネント間のデータを保管したり、渡したりします。 これらのリソースは、キュー・マネージャー・プロセスおよびキュー・マネージャーに接続するアプリケーションが使用します。 IBM MQ MQI clients は、IBM MQ トレース制御を除き、IPC リソースを使用しません。 UNIX コマンド **ipcs -a** を使用すると、マシンで現在使用されている IPC リソースの数とサイズの全情報を取得できます。

Linux AIX IBM MQ および UNIX のプロセス優先順位

プロセス優先順位の *nice* 値を設定する際の良い方法。

この情報は、 AIX and Linux システム上で稼働する IBM MQ にのみ適用されます。

プロセスをバックグラウンドで実行する場合、呼び出し側シェルによって、そのプロセスの *nice* 値が高くなる場合があります (従って優先順位は下がります)。 これによって、全般的に IBM MQ のパフォーマンスへの影響が見られる場合があります。 負荷の大きい状態で、優先順位が高く直ちに実行可能なスレッドが多数あり、いくつかのスレッドの優先順位が低い場合、オペレーティング・システムのスケジューリングの特性によって、優先順位の低いスレッドからプロセッサー時間が奪われる可能性があります。

runmqlsr など、キュー・マネージャーに関連付けられたプロセスで別個に開始されたものには、それらが関連付けられているキュー・マネージャーと同じ *nice* 値を持たせることをお勧めします。 シェルがバックグラウンド・プロセスに高い *nice* 値を割り当てることがないようにしてください。 例えば、ksh では、"`set +o bgnice`" 設定を使用して、バックグラウンド・プロセスの *nice* 値を ksh が上げないようにします。 "ps -efl" リストの *NI* 列を調べて、実行中のプロセスの適切な値を確認することができます。

また、IBM MQ アプリケーション・プロセスをキュー・マネージャーと同じ *nice* 値を使用して開始してください。 異なる *nice* 値で実行される場合、アプリケーション・スレッドがキュー・マネージャー・スレッドをブロックするかまたはその逆が生じ、パフォーマンスが 低下する可能性があります。

z/OS Planning your IBM MQ environment on z/OS

When planning your IBM MQ environment, you must consider the resource requirements for data sets, page sets, Db2, Coupling Facilities, and the need for logging, and backup facilities. Use this topic to plan the environment where IBM MQ runs.

Before you plan your IBM MQ architecture, familiarize yourself with the basic IBM MQ for z/OS concepts, see the topics in IBM MQ for z/OS concepts.

When planning your queue manager, you might need to work with different people in your organization. It is usually a good idea to involve those people early, as change control procedures can take a long time. They might also be able to tell you what parameters you need to configure IBM MQ for z/OS.

For example you might need to work with the:

- Storage administrator, to determine the high level qualifier of queue manager data sets, and to allocate enough space for queue manager data sets.
- z/OS system programmer to define the IBM MQ subsystem to z/OS and APF authorize the IBM MQ for z/OS libraries.
- Network administrator to determine which TCP/IP stack and ports should be used for IBM MQ for z/OS.
- Security administrator to set up access to queue manager data sets, security profiles for IBM MQ for z/OS resources, and TLS certificates.
- Db2 administrator to set up Db2 tables when configuring a queue sharing group.

Related concepts

IBM MQ Technical overview

Related tasks

“IBM MQ アーキテクチャーの計画” on page 5

IBM MQ 環境を計画する際、単一および複数キュー・マネージャーのアーキテクチャーについて、また Point-to-Point およびパブリッシュ/サブスクライブのメッセージング・スタイルについて IBM MQ が提供するサポートを考慮します。 また、リソース要件、およびロギングやバックアップの機能の使用方法を計画します。

Configuring z/OS

Administering IBM MQ for z/OS

z/OS Planning for your queue manager

When you are setting up a queue manager, your planning should allow for the queue manager to grow, so that the queue manager meets the needs of your enterprise.

The best way to configure a queue manager is in steps:

1. Configure the base queue manager
2. Configure the channel initiator which does queue manager to queue manager communications, and remote client application communication
3. If you want to encrypt and protect messages, configure Advanced Message Security
4. If you want to use File Transfer over IBM MQ, configure Managed File Transfer for z/OS.
5. If you want to use the administrative or messaging REST API, or the IBM MQ Console to manage IBM MQ from a web browser, configure the mqweb server.

Some enterprises have hundreds of thousands of queue managers in their environment. You need to consider your IBM MQ network now, and in five years time.

On z/OS, some queue managers process thousands of messages a second, and log over 100 MB a second. If you expect very high volumes you may need to consider having more than one queue manager.

On z/OS, IBM MQ can run as part of a queue sharing group (QSG) where messages are stored in the Coupling Facility, and any queue manager in the queue sharing group can access the messages. If you want to run in a queue sharing group you need to consider how many queue managers you need.

Typically, there is one queue manager for each LPAR. You might also have one queue manager to backup CF structures regularly.

Some changes to configuration are easy to do, such as defining a new queue. Some are harder, such as making logs and page sets bigger; and some configuration cannot be changed, such as the name of a queue manager or the queue sharing group name.

There is performance and tuning information available in the MP16 performance SupportPac .

Naming conventions

You need to have a naming convention for the queue manager data sets.

Many enterprises use the release number in the name of the load libraries, and so on. You might want to consider having an alias of `MQM.SCSQAUTH` pointing to the version currently in use, such as `MQM.V930.SCSQAUTH`, so you do not have to change CICS®, Batch, and IMS JCL when you migrate to a new version of IBM MQ.

You can use a symbolic link in z/OS UNIX System Services to reference the installation directory for the version of IBM MQ currently in use.

The data sets used by the queue manager (logs, page sets, JCL libraries) need a naming convention to simplify the creation of security profiles, and the mapping of data sets to SMS storage classes that control where the data sets are placed on disk, and the attributes they have.

Note, that putting the version of IBM MQ into the name of the page sets or logs, is not a good idea. One day you might migrate to a new version, and the data set will have the "wrong" names.

Applications

You need to understand the business applications and the best way to configure IBM MQ. For example if applications have logic to provide recovery and repeat capability, then non persistent messages might be good enough. If you want IBM MQ to handle the recovery, then you need to use persistent messages and put and get messages in syncpoint.

You need to isolate queues from different business transactions. If a queue for one business application fills up, you do not want this impacting other business applications. Isolate the queues in different page sets and buffer pools, or structures, if possible.

You need to understand the profile of messages. For many applications the queues have only a few messages. Other applications can have queues build up during the day, and be processed overnight. A queue which normally has only a few messages on it, might need to hold many hours worth of messages if there is a problem and messages are not processed. You need to size the CF structures and page sets to allow for your expected peak capacity.

Post configuration

Once you have configured your queue manager (and components) you need to plan for:

- Backing up page sets.
- Backing up definitions of objects.
- Automating the backup of any CF structures.
- Monitoring IBM MQ messages, and taking action when a problem is detected.
- Collecting the IBM MQ statistics data.
- Monitoring resource usage, such as virtual storage, and amount of data logged per hour. With this you can see if your resource usage is increasing and if you need to take actions, such as setting up a new queue manager

z/OS Planning your storage and performance requirements on z/OS

You must set realistic and achievable storage, and performance goals for your IBM MQ system. Use this topic help you understand the factors which affect storage, and performance.

This topic contains information about the storage and performance requirements for IBM MQ for z/OS. It contains the following sections:

- z/OS performance options for IBM MQ
- Determining z/OS workload management importance and velocity goals
- “Library storage” on page 149
- “System LX usage” on page 150
- “Storage configuration” on page 150
- “Disk storage” on page 155

See, “Where to find more information about storage and performance requirements” on page 156 for more information.

z/OS performance options for IBM MQ

With workload management, you define performance goals and assign a business importance to each goal. You define the goals for work in business terms, and the system decides how much resource, such as processor and storage, should be given to the work to meet its goal. Workload management controls the dispatching priority based on the goals you supply. Workload management raises or lowers the priority as needed to meet the specified goal. Thus, you need not fine-tune the exact priorities of every piece of work in the system and can focus instead on business objectives.

The three kinds of goals are:

Response time
How quickly you want the work to be processed

Execution velocity
How fast the work should be run when ready, without being delayed for processor, storage, I/O access, and queue delay

Discretionary
A category for low priority work for which there are no performance goals

Response time goals are appropriate for end-user applications. For example, CICS users might set workload goals as response time goals. For IBM MQ address spaces, velocity goals are more appropriate. A small amount of the work done in the queue manager is counted toward this velocity goal but this work is critical for performance. Most of the work done by the queue manager counts toward the performance goal of the end-user application. Most of the work done by the channel initiator address space counts toward its own velocity goal. The receiving and sending of IBM MQ messages, which the channel initiator accomplishes, is typically important for the performance of business applications using them.

Determining z/OS workload management importance and velocity goals

See “z/OS ワークロード管理の重要度の設定” on page 150 for more information.

Library storage

You must allocate disk storage for the product libraries. The exact figures depend on your configuration, and should include both the target and distribution libraries, as well as the SMP/E libraries.

The target libraries used by IBM MQ for z/OS use PDSE formats. Ensure that any PDSE target libraries are not shared outside a sysplex. For more information about the required libraries and their sizes and the required format, see the Program Directory. プログラム・ディレクトリのダウンロードリンクについては、IBM MQ for z/OS Program Directory PDF files を参照。

System LX usage

Each defined IBM MQ subsystem reserves one system linkage index (LX) at IPL time, and a number of non-system linkage indexes when the queue manager is started. The system linkage index is reused when the queue manager is stopped and restarted. Similarly, distributed queuing reserves one non-system linkage index. In the unlikely event of your z/OS system having inadequate system LXs defined, you might need to take these reserved system LXs into account.

If required, the number of system LXs can be increased by setting the *NSYSLX* parameter in SYS1.PARMLIB member IEASYSxx.

z/OS *z/OS ワークロード管理の重要度の設定*

ワークロード管理およびサービス定義による目標定義の詳細については、z/OS 製品ドキュメントを参照してください。

このトピックでは、システム内の他の重要な処理との相対的な位置付けの中で、z/OS ワークロード管理の重要度と速度に関する目標を設定するための推奨事項を取り上げます。 詳しくは、「*z/OS MVS* 計画*:* ワークロード管理」を参照してください。

キュー・マネージャーのアドレス・スペースはサブシステム・サービスを提供するため、高優先度を定義する必要があります。 チャネル・イニシエーターはアプリケーション・アドレス・スペースですが、リモート・キュー・マネージャーに送信されるメッセージが遅延しないようにするために、通常は高優先度が指定されています。Advanced Message Security (AMS) もサブシステム・サービスを提供するため、高優先度を定義する必要があります。

以下のサービス・クラスを使用します。

デフォルトの SYSSTC サービス・クラス

- VTAM アドレス・スペースと TCP/IP アドレス・スペース
- IRLM アドレス・スペース (IRLMPROC)

注 : VTAM、TCP/IP、IRLM のアドレス・スペースでは、すべての DBMS アドレス・スペースとその接続先のアドレス・スペースや従属アドレス・スペースよりも高いディスパッチング優先順位が必要です。 ワークロード管理で、VTAM、TCP/IP、IRLM の優先順位が、他の DBMS アドレス・スペースと同じレベルかそれ以下のレベルに変更されないようにしてください。

ユーザー定義名のサービス・クラス (PRODREGN など)。以下のような対象については、速度に関する高い目標と重要度 1 を設定します。

- IBM MQ キュー・マネージャー、チャネル・イニシエーター、および AMS アドレス・スペース
- Db2 (Db2 で設定するストアード・プロシージャーのアドレス・スペースを除くすべてのアドレス・スペース)
- CICS (すべての領域タイプ)
- IMS (BMP 以外のすべての領域タイプ)

速度の目標を高く設定することは、上記のすべてのアドレス・スペースで始動や再始動をできるだけ短時間で実行するために役立ちます。

CICS 領域と IMS 領域で速度の目標が重要な意味を持つのは、始動時や再始動時に限られます。 トランザクションの実行が始まった後は、ワークロード管理で、CICS や IMS の速度の目標が無視され、それらの領域で実行されているトランザクションの応答時間の目標に基づいて、優先順位が割り当てられます。 それらのトランザクションの目標は、実装先のビジネス・アプリケーションの相対的な優先順位を反映した目標であるべきです。 通常、重要度の値は 2 になります。 同じように、IBM MQ を使用するバッチ・アプリケーションの速度の目標や重要度もまた、実装先のビジネス・アプリケーションの相対的な優先順位を反映する必要があります。 通常、その重要度や速度の目標は、PRODREGN の重要度や速度の目標より小さくなります。

z/OS *Storage configuration*

V 9.4.0 In a 64 bit address space, there is a virtual line called "the bar" that marks the 2GB address. The bar separates storage below the 2GB address, called "below the bar", from storage above the 2GB

address, called "above the bar". Storage below the bar uses 31 bit addressability, storage above the bar uses 64 bit addressability.

V 9.4.0

You can specify the limit of 31-bit storage by using the JCL REGION parameter, and the limit of 64-bit storage by using the MEMLIMIT parameter. These specified values can be overridden by z/OS exits.

Suggested storage configuration

The following table shows suggested **REGION** and **MEMLIMIT** values for the queue manager, channel initiator, and AMS address spaces. These suggestions should be used as a starting point and adjusted using the information in:

- “Queue manager storage configuration” on page 151
- “IBM MQ 9.4.0 からのチャネル・イニシエーター・ストレージ構成” on page 154

Table 19. Suggested definitions for REGION and MEMLIMIT

Address space	Storage configuration
Queue manager	REGION=0M, MEMLIMIT=3G
V 9.4.0 Channel initiator from IBM MQ 9.4.0	REGION=0M, MEMLIMIT=2G
AMS address space	REGION=0M

Managing the MEMLIMIT and REGION size

Other mechanisms, for example the **MEMLIMIT** parameter in the SMFPRMxx member of SYS1.PARMLIB or the IEFUSI exit might be used at your installation to provide a default amount of virtual storage above the bar for z/OS address spaces. See Memory management above the bar for full details about limiting storage above the bar.

z/OS *Queue manager storage configuration*

The queue manager address space is likely to be the major user of 64-bit storage in an IBM MQ installation. Each connection to the queue manager requires common storage to be allocated as described in the following text. In addition to 64-bit storage, you should allow the queue manager to use all available 31-bit storage by specifying REGION=0M on the queue manager JCL.

Common storage

Each IBM MQ for z/OS subsystem has the following approximate storage requirements:

- CSA 4KB
- ECSA 800KB, plus the size of the trace table that is specified in the **TRACTBL** parameter of the CSQ6SYSP system parameter macro. For more information, see Using CSQ6SYSP.

In addition, each concurrent logical connection to the queue manager requires about 5 KB of ECSA. When a task ends, other IBM MQ tasks can reuse this storage.

IBM MQ does not release the storage until the queue manager is shut down, so you can calculate the maximum amount of ECSA required by multiplying the maximum number of concurrent connections by 5KB. The number of concurrent logical connections is the sum of the number of:

- Tasks (TCBs) in Batch, TSO, z/OS UNIX System Services, IMS, and Db2 stored procedure address space (SPAS) regions that are connected to IBM MQ, but not disconnected.
- CICS transactions that have issued an IBM MQ request, but have not terminated

- JMS Connections, Sessions, TopicSessions or QueueSessions that have been created (for bindings connection), but not yet destroyed or garbage collected.
- Active IBM MQ channels

You can set a limit to the common storage, used by logical connections to the queue manager, with the **ACELIM** configuration parameter. The **ACELIM** control is primarily of interest to sites where Db2 stored procedures cause operations on IBM MQ queues.

When driven from a stored procedure, each IBM MQ operation can result in a new logical connection to the queue manager. Large Db2 units of work, for example due to table load, can result in an excessive demand for common storage.

ACELIM is intended to limit common storage use and to protect the z/OS system, by limiting the number of connections in the system. You should only set **ACELIM** on queue managers that have been identified as using excessive quantities of ECSA storage. See the **ACELIM** section in *Using CSQ6SYSP* for more information.

To set a value for **ACELIM**, firstly determine the amount of storage currently in the subpool controlled by the **ACELIM** value. This information is in the SMF 115 subtype 5 records produced by statistics CLASS(3) trace.

IBM MQ SMF data can be formatted using SupportPac MP1B. The number of bytes in use in the subpool controlled by **ACELIM** is displayed in the STGPOOL DD, on the line titled *ACE/PEB*.

For more information about SMF 115 statistics records, see Interpreting IBM MQ for z/OS performance statistics.

Increase the normal value by a sufficient margin to provide space for growth and workload spikes. Divide the new value by 1024 to yield a maximum storage size in KB for use in the **ACELIM** configuration.

Private storage

The queue manager address space uses 64-bit storage for many internal control blocks. The **MEMLIMIT** parameter of the queue manager JCL defines the maximum amount of 64-bit storage available. 3GB of storage, `MEMLIMIT=3G`, is the minimum you should use, however, depending on your configuration significantly more might be required.

You should specify a specific **MEMLIMIT** value rather than `MEMLIMIT=NOLIMIT` to prevent potential problems. If you specify `NOLIMIT` or a very large value, then there is the potential to use up all of the available z/OS virtual storage, which leads to paging in your system. When increasing the value of **MEMLIMIT** you should discuss the new setting with your z/OS system programmer in case there is a system-wide limit on the amount of on storage that can be used.

If you have a large value for **MEMLIMIT** you might need to increase the size of your dump data sets as more data is captured in a dump.

You can monitor the address space storage usage from the CSQY220I message that indicates the amount of 31 and 64-bit private storage in use, and the remaining free amount.

Buffer pools

Buffer pools are a significant user of private storage in the queue manager address space. Each buffer pool size is determined at queue manager initialization time, and storage is allocated for the buffer pool when a page set that is using that buffer pool is connected. The parameter **LOCATION (ABOVE|BELOW)** is used to specify where the buffers are allocated. You can use the ALTER BUFFPOOL command to dynamically change the size of buffer pools.

When calculating a value for **MEMLIMIT** it is critical that you take into account the buffer pool sizes if they are configured with **LOCATION(ABOVE)**. You should perform the calculation as follows.

Calculate the value of **MEMLIMIT** as 2GB plus the size of the buffer pools configured with **LOCATION(ABOVE)**, rounded up to the nearest GB. Set MEMLIMIT to a minimum of 3GB and increase this as necessary when you need to increase the size of your buffer pools.

For example, for three buffer pools configured with **LOCATION(ABOVE)**, buffer pool one has 10,000 buffers, and buffer pools two and three have 50,000 buffers each. Memory usage above the bar equals 110,000 (total number of buffers) * 4096 = 450,560,000 bytes = 430MB.

All buffer pools regardless of **LOCATION** make use of 64-bit storage for control structures. As the number of buffer pools and number of buffers in those pools increase this can become significant. Each buffer requires around an additional 200 bytes of 64-bit storage. For the preceding configuration that would require: 200 * 110,000 = 22,000,000 bytes = 21MB.

Therefore, in this scenario 3GB can be used for the **MEMLIMIT**, which allows scope for growth: 21MB + 430MB + 2GB which rounds up to 3GB.

For some configurations there can be significant performance benefits to using buffer pools that have their buffers permanently backed by real storage. You can achieve this by specifying the FIXED4KB value for the **PAGECLAS** attribute of the buffer pool. However, you should only do this if there is sufficient real storage available on the LPAR, otherwise other address spaces might be affected. For information about when you should use the FIXED4KB value for **PAGECLAS**, see IBM MQ Support Pac MP16: IBM MQ for z/OS - Capacity planning & tuning.

Making the buffer pools so large that there is MVS paging might adversely affect performance. You might consider using a smaller buffer pool that does not page, with IBM MQ moving the message to and from the page set.

Indexed queues

On z/OS, local queues are indexed if the queue has an **INDXTYPE** attribute that has not been set to NONE. The indexes for shared queues are held in a coupling facility, but for private queues the index is held in 64 bit storage. For each message on an indexed queue 136 bytes of data are used to index the message. For very deep queues this can result in a significant amount of 64 bit storage being allocated. For example, 10 million messages on an indexed queue will use 1.27 GB of 64 bit storage in order to maintain the index.

If you expect to have a large number of messages on indexed queues you should allow for this when setting **MEMLIMIT**. To calculate an upper limit for the amount of storage required for indexes, multiply the **MAXDEPTH** attribute for each indexed queue by 136 and sum the value. This value should be added to your existing **MEMLIMIT**.

V 9.4.0 RECOVER CFSTRUCT

From IBM MQ 9.4.0 the **RECOVER CFSTRUCT** command makes greater use of 64-bit storage. In many cases there should be spare 64-bit storage available and so use of the command does not require an increase in the value of **MEMLIMIT**. However, if you are likely to have large structure backups, containing more than a few million messages, you should increase the **MEMLIMIT** for all queue managers which might process the **RECOVER CFSTRUCT** command by 500MB.

For example if you had MEMLIMIT=3G already, you should consider using MEMLIMIT=4G as the **MEMLIMIT** parameter does not allow for decimal points.

Shared Message Data Set (SMDS) buffers and MEMLIMIT

When running messaging workloads using shared message data sets, there are two levels of optimizations that can be achieved by adjusting the DSBUFS and DSBLOCK attributes.

The amount of above bar queue manager storage used by the SMDS buffer is DSBUFS x DSBLOCK. This means that by default, 100 x 256KB (25MB) is used for each CFLEVEL(5) structure in the queue manager.

Although this value is not too high, if your enterprise, or enterprises have many CFSTRUCTs, some of them might allocate a high value of MEMLIMIT for buffer pools, and sometimes they have deep indexed queues, so in total, they might run out of storage above the bar.

V 9.4.0 z/OS *IBM MQ 9.4.0 からのチャネル・イニシエーター・ストレージ構成*

チャネル・イニシエーターは通常、キュー・マネージャーよりはるかに少ない 64 ビット・ストレージを使用します。 ただし、 IBM MQ 9.4.0 以降、使用量が増加しています。 64 ビットストレージに加えて、チャネルイニシエータ JCL で REGION=0M を指定することによって、チャネルイニシエータが利用可能なすべての 31 ビットストレージを使用できるようにする必要があります。

共通ストレージ

通常、チャネル・イニシエーターでは、最大 160KB の ECSA を使用する必要があります。

31 ビット専用ストレージ

チャネル・イニシエーターが使用できる 31 ビット・ストレージは、CHINIT が持つことができる同時接続の数を制限します。

各チャネルは、チャネル・イニシエーターのアドレス・スペースで約 170KB の拡張専用領域を使用します。 メッセージ・チャネル (例えば、送信側チャネルまたは受信側チャネル) の場合、 32KB より大きいメッセージが送信されると、ストレージはメッセージ・サイズだけ増加します。 この増加した分のストレージが解放されるのは、以下の場合です。

- 送信側チャネルまたはクライアント・チャネルで現在のバッファー・サイズの半分に満たないサイズしか必要としないメッセージが 10 件連続した場合。
- ハートビートの送信または受信が行われた場合。

ストレージは Language Environment 内で再使用するために解放されますが、 z/OS 仮想記憶マネージャーはそのストレージをフリーと見なしません。 したがって、チャネル数の上限は、メッセージのサイズと到着パターン、さらには、個々のユーザー・システムで設定されている拡張専用領域サイズの制限値によって決まることになります。

多くのシステムでは、チャネル数の上限は、約 9000 個になります。拡張領域サイズが 1.6GB を超えることは、ほとんどないからです。

チャネル・イニシエーターのトレースはデータ・スペースに書き込まれます。 データ・スペース・ストレージのサイズは、**TRAXTBL** パラメーターによって制御されます。 ALTER QMGR を参照してください。

64 ビット専用ストレージ

チャネルイニシエータ JCL の **MEMLIMIT** パラメータは、利用可能な 64 ビットストレージの最大容量を定義します。 MEMLIMIT=2G 、2GB のストレージは、あなたが使用すべき最小値です。 ご使用の構成によっては、さらに多くのものが必要になる場合があります。

潜在的な問題を防ぐために、 MEMLIMIT=NOLIMIT ではなく、妥当な **MEMLIMIT** 値を指定すべきです。 NOLIMIT または非常に大きな値を指定すると、 z/OS 仮想ストレージのすべてを使用しつくす可能性があり、システムでページングが発生する可能性があります。 **MEMLIMIT** の値を増やす場合は、ストレージの使用量にシステム全体での制限がある可能性があるため、z/OS のシステムプログラマーと新しい設定について話し合う必要があります。

MEMLIMIT の値が大きい場合は、ダンプにキャプチャされるデータが増えるため、ダンプデータセットのサイズを大きくする必要があるかもしれません。

チャネル・イニシエーターには、SMF チャネルとサーバー接続チャネルの 2 人の 64 ビット・ストレージ・ユーザーがいます。

SMF

使用可能にすると、SMF クラス 4 アカウンティングまたは統計に 64 ビット・ストレージが必要になります。 最小 256MB のストレージが必要です。 使用可能なストレージが十分にない場合、チャネル・イニシエーターは CSQX124E メッセージを発行し、クラス 4 のアカウンティングと統計は使用できません。

サーバー接続チャネル

IBM MQ 9.4.0 サーバー接続チャネルからは、32 KB より大きいサイズのメッセージを転送する場合、メッセージ・バッファーを 64 ビット・ストレージに割り振ります。

これらのバッファーが解放されるのは、チャネルが 10 個の連続したメッセージの現行バッファー・サイズの半分未満を必要とする場合、またはハートビートが送信または受信される場合です。

MEMLIMIT の値は、同時に実行できるサーバー接続チャネルの数の上限を設定します。 MEMLIMIT=2G の最小値を使用することで、IBM MQ の以前のバージョンと同じ数のチャンネルを実行できることを保証し、また、成長のための容量も確保できます。

MEMLIMIT の概算値は、同時アクティブなサーバー接続チャネルのピーク最大数と、それらのチャネルで転送が予想される最大メッセージサイズを算出することで計算できます。 MEMLIMIT=2G を出発点として、切り上げてください。

例えば、同時サーバー接続チャネルの最大数を 2,000 に設定し、各チャネルの最大メッセージサイズを 1MB に設定した場合、サーバー接続チャネルは 64 ビットストレージの最大値である 2GB 弱を使用することになります。 これは 2GB に非常に近いので、 MEMLIMIT=3G に切り上げる必要があります。

z/OS *Disk storage*

Use this topic when planning your disk storage requirements for log data sets, Db2 storage, coupling facility storage, and page data sets.

Work with your storage administrator to determine where to put the queue manager data sets. For example, your storage administrator may give you specific DASD volumes, or SMS storage classes, data classes, and management classes for the different data set types.

- Log data sets must be on DASD. These logs can have high I/O activity with a small response time and do not need to be backed up.
- Archive logs can be on DASD or tape. After they have been created, they might never be read again except in an abnormal situation, such as recovering a page set from a backup. They should have a long retention date.
- Page sets might have low to medium activity and should be backed up regularly. On a high use system, they should be backed up twice a day.
- BSDS data sets should be backed up daily; they do not have high I/O activity.

All data sets are similar to those used by Db2, and similar maintenance procedures can be used for IBM MQ.

See the following sections for details of how to plan your data storage:

- **Logs and archive storage**

  “How long do I need to keep archive logs” on page 175 describes how to determine how much storage your active log and archive data sets require, depending on the volume of messages that your IBM MQ system handles and how often the active logs are offloaded to your archive data sets.

- **Db2 storage**

  “Db2 storage” on page 192 describes how to determine how much storage Db2 requires for the IBM MQ data.

- **coupling facility storage**

  “カップリング・ファシリティー・リソースの定義” on page 182 describes how to determine how large to make your coupling facility structures.

- **Page set and message storage**

“ページ・セットとバッファー・プールの計画” on page 156 describes how to determine how much storage your page data sets require, depending on the sizes of the messages that your applications exchange, on the numbers of these messages, and on the rate at which they are created or exchanged.

z/OS *Where to find more information about storage and performance requirements*

Use this topic as a reference to find more information about storage and performance requirements.

You can find more information from the following sources:

Table 20. Where to find more information about storage requirements

Topic	Where to look
System parameters	Using CSQ6SYSP and Customizing your queue managers
Storage required to install IBM MQ	Program Directory. プログラム・ディレクトリのダウンロードリンクについては、 IBM MQ for z/OS Program Directory PDF files を参照。
IEALIMIT and IEFUSI exits	See IEALIMIT and IEFUSI in the *z/OS:MVS Installation Exits* documentation.
Latest information	IBM MQ SupportPac website SupportPacs は'IBM'MQ およびその他のプロジェクト・エリア用.
Workload management and defining goals through the service definition	*z/OS MVS Planning: Workload Management*

z/OS ページ・セットとバッファー・プールの計画

ページ・データ・セットおよびバッファー・プールの初期の数とサイズについて計画する際に役立つ情報を取り上げます。

このトピックには、次のセクションがあります。

- 156 ページの『ページ・セットの計画』
  - ページ・セットの使用法
  - ページ・セットの数
  - ページ・セットのサイズ
  - z/OS データ・セット暗号化の計画
- 157 ページの『ページ・セットのサイズの計算』
  - ページ・セット 0
  - ページ・セット 01 - 99
  - メッセージのストレージ要件の計算
- 160 ページの『Enabling dynamic page set expansion』
- 161 ページの『Defining your buffer pools』

ページ・セットの計画

ページ・セットの使用法

短期のメッセージの場合は、通常、ページ・セットで使用されるページ数はごくわずかであり、始動時、チェックポイント操作時、シャットダウン時を除けば、データ・セットに対する I/O はほとんど、あるいは全くありません。

長期のメッセージの場合は、通常、メッセージが含まれているページがディスクに書き出されます。 この操作を実行するのは、キュー・マネージャーであり、その目的は、再始動時間を短縮することです。

短期のメッセージと長期のメッセージを分離して、それぞれを別々のページ・セット、別々のバッファー・プールに格納してください。

ページ・セットの数

大きなページ・セットをいくつか使用する環境にすれば、IBM MQ 管理者の役割が容易になります。ページ・セットの数が少なく、キューとページ・セットの対応関係がシンプルになるからです。

小さなページ・セットを多数使用する環境にも、いくつかの利点があります。 例えば、バックアップにかかる時間が短くなり、バックアップ時や再始動時に I/O の並列実行が可能になります。 一方、IBM MQ 管理者の役割にかなりのパフォーマンス・コストが加算される、という面もあります。それぞれのキューを多数のページ・セットのいずれかに対応付けることが必要になるからです。

少なくとも以下の 5 つのページ・セットを定義してください。

- オブジェクト定義のための予約ページ・セット (ページ・セット 0)
- システム関連メッセージのためのページ・セット
- パフォーマンスが重要な長期メッセージのためのページ・セット
- パフォーマンスが重要な短期メッセージのためのページ・セット
- 他のすべてのメッセージのためのページ・セット

このような方法でメッセージを各ページ・セットに分散させた場合のパフォーマンス上の利点については、161 ページの『Defining your buffer pools』を参照してください。

ページ・セットのサイズ

メッセージ容量のピーク値を予想して、ページ・セットに十分なスペースを定義してください。 予想外のピーク容量も考慮に入れる必要があります。例えば、キュー・サーバー・プログラムが稼働していない間に大量のメッセージが蓄積される、ということもあり得ます。 そのための対策として、ページ・セットに 2 次エクステントを割り振ったり、ページ・セットの動的拡張を有効にしたりすることができます。 詳細については、160 ページの『Enabling dynamic page set expansion』を参照してください。ページ・セットを小さくすることは難しいので、通常は小さめのページ・セットを割り振って、必要に応じて拡張することをお勧めします。

ページ・セットのサイズを計画するときには、アプリケーション以外のメッセージ・データも含め、生成される可能性のあるすべてのメッセージを考慮に入れます。 例えば、アプリケーションが要求したトリガー・メッセージ、イベント・メッセージ、レポート・メッセージなどがあります。

ページ・セットのサイズによって、バックアップからのリストアを実行するときにページ・セットのリカバリーにかかる時間が変わってきます。ページ・セットのサイズが大きければ、リストアのためにそれだけ多くの時間がかかるからです。

注 : ページ・セットのリカバリーは、キュー・マネージャーがバックアップ作成時以降に書き込まれたログ・レコードの処理に要する時間にも左右されます。この時間は、バックアップ頻度によって異なります。 詳細については、194 ページの『Planning for backup and recovery』を参照してください。

注 : 4 GB を超えるページ・セットでは、SMS の拡張アドレッシング機能を使用する必要があります。

z/OS データ・セット暗号化の計画

z/OS データ・セット暗号化機能は、IBM MQ for z/OS 9.1.4 以降で実行されているキュー・マネージャーのページ・セットに適用できます。

これらのページ・セットは、EXTENDED 属性と、データの AES 暗号化を保証するデータ・セット鍵ラベルを使用して割り振る必要があります。

データ・セット暗号化による IBM MQ for z/OS での保存データの機密性 のセクションを参照してください。 for more information.

ページ・セットのサイズの計算

キュー・マネージャーのオブジェクト定義 (キューやプロセスなど) の場合、ストレージ要件の計算はシンプルです。それらのオブジェクトのサイズは永久的に固定されているからです。 一方、メッセージの計算は複雑になります。それには以下のような理由があります。

- メッセージのサイズは、それぞれ異なります。
- メッセージは、一時的です。
- 取り込まれたメッセージが占めるスペースは、非同期処理によって周期的に取り戻されます。

必要に応じて、4 GB を超える大きなページ・セットを作成し、ネットワークが停止した場合にメッセージを収容するための予備の容量を確保しておくこともできます。 既存のページ・セットを変更することはできません。 その代わりに、拡張アドレッシング機能と拡張フォーマットの属性を設定した新しいページ・セットを作成する必要があります。 新しいページ・セットと古いページ・セットは同じ物理サイズでなければならず、その古いページ・セットを新しいページ・セットにコピーしなければなりません。 逆方向の移行が必要な場合は、ページ・セット 0 を変更しないようにしてください。 4 GB より小さいページ・セットで十分であれば、処置は必要ありません。

ページ・セット 0

ページ・セット 0 はオブジェクト定義用に予約されています。

V 9.4.3 ページセットゼロの場合、必要なストレージはおよそ

```
    (maximum number of local queue definitions x 1364)
       (excluding shared queues)
+   (maximum number of model queue definitions x 1024)
+   (maximum number of alias queue definitions x 512)
+   (maximum number of remote queue definitions x 685)
+   (maximum number of permanent dynamic queue definitions x 1364)
+   (maximum number of process definitions x 680)
+   (maximum number of namelist definitions x 12320)
+   (maximum number of message channel definitions x 2026)
+   (maximum number of client-connection channel definitions x 5170)
+   (maximum number of server-connection channel definitions x 2048)
+   (maximum number of storage class definitions x 287)
+   (maximum number of authentication information definitions x 1010)
+   (maximum number of administrative topic definitions x 15000)
+   (total length of topic strings defined in administrative topic definitions)
```

Long Term Support'と'Continuous Delivery は、'IBM MQ 9.4.3 の前にある。 IBM MQ 9.4.3 から、使用されるスペースの量は、正確なチャンネル定義によって異なる。 2026 バイトは、ストレージ計画上、良い値である。

LTS ページ・セット 0 のために以下のようなストレージが必要です。

```
    (maximum number of local queue definitions x 1010)
       (excluding shared queues)
+   (maximum number of model queue definitions x 746)
+   (maximum number of alias queue definitions x 338)
+   (maximum number of remote queue definitions x 434)
+   (maximum number of permanent dynamic queue definitions x 1010)
+   (maximum number of process definitions x 674)
+   (maximum number of namelist definitions x 12320)
+   (maximum number of message channel definitions x 2026)
+   (maximum number of client-connection channel definitions x 5170)
+   (maximum number of server-connection channel definitions x 2026)
+   (maximum number of storage class definitions x 266)
+   (maximum number of authentication information definitions x 1010)
+   (maximum number of administrative topic definitions x 15000)
+   (total length of topic strings defined in administrative topic definitions)
```

この値を 4096 で除算することによって、ページ・セット・データ・セットのレコード数としてクラスターで指定する値を計算できます。

共有リポジトリーに格納するオブジェクトを考慮に入れる必要はありませんが、ページ・セット 0 に格納するかコピーするオブジェクト (つまり、GROUP または QMGR という属性指定のオブジェクト) は考慮に入れる必要があります。

作成できるオブジェクトの総数は、ページ・セット 0 の容量によって制限されます。 定義できるローカル・キューの数は、524 287 に制限されます。

ページ・セット 01 - 99

ページ・セット 01 - 99 の各ページ・セットで必要なストレージは、そのページ・セットに格納するメッセージの数とサイズによって決まります。 (共有キューに入るメッセージは、ページ・セットに格納されません。)

この値を 4096 で除算することによって、ページ・セット・データ・セットのレコード数としてクラスターで指定する値を計算できます。

メッセージのストレージ要件の計算

このセクションでは、メッセージがページに格納されるしくみについて説明します。 この点を理解すれば、メッセージ用として定義しなければならないページ・セット・ストレージの量を計算できるようになります。 1 つのページ・セットに格納するすべてのメッセージで必要になるスペースの概算値を計算するために、そのページ・セットに対応付けるすべてのキューの最大長と、それらのキューに入るメッセージの平均サイズを考慮に入れる必要があります。

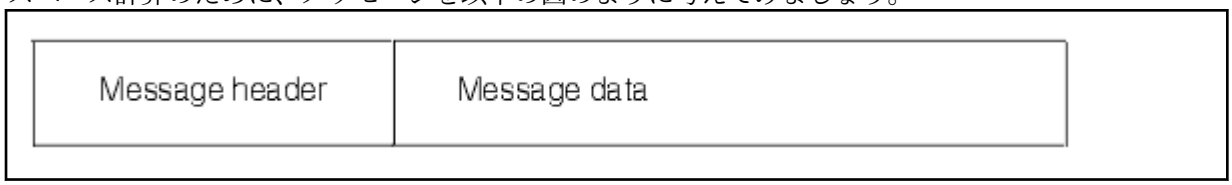
注 : このセクションで取り上げる構造のサイズや制御情報は、メジャー・リリースの登場で変更される可能性があります。 IBM MQ の特定のリリースの詳細については、SupportPac MP16 IBM「MQ z/OS キャパシティ・プランニング&チューニング」と IBM'MQ ファミリー - パフォーマンスレポートを参照してください。

IBM MQ の制御外の理由 (例えば、通信プロトコルの問題など) のためにメッセージの取得が遅れる可能性も考慮に入れる必要があります。 この場合は、メッセージの取得率よりも書き込み率のほうがかなり高くなると考えられます。 その結果、ページ・セットに格納されるメッセージの数が大幅に増え、ストレージ・サイズの所要量が大きくなる可能性があります。

ページ・セット内の各ページの長さは 4096 バイトです。 固定ヘッダー情報の分を差し引くと、各ページにはメッセージを格納するための有効スペースが 4057 バイト分残ることになります。

各メッセージで必要なスペースを計算するときに最初に考慮に入れなければならないのは、メッセージが 1 つのページに収まるか (短メッセージ)、それとも 2 つ以上のページに分割しなければならないか (長メッセージ)、ということです。 このようにメッセージを分割する場合は、スペース計算に追加の制御情報を組み入れなければなりません。

スペース計算のために、メッセージを以下の図のように考えてみましょう。



メッセージ・ヘッダー・セクションには、メッセージ記述子と他の制御情報が入ります。そのサイズは、メッセージのサイズによって異なります。 メッセージ・データ・セクションには、すべての実際のメッセージ・データとその他のヘッダー (伝送ヘッダーや IMS ブリッジ・ヘッダーなど) が入ります。

ページ・セットの制御情報のために最低でも 2 つのページが必要になります。通常、そのスペースは、メッセージで必要な合計スペースの 1% 未満です。

短メッセージ

短メッセージとは、1 つのページに収まるメッセージのことをいいます。

小さいメッセージは、各ページに 1 つずつ保管されます。

長メッセージ

メッセージ・データのサイズが 3596 バイトより大きく、4 MB 以下の場合、そのメッセージは、長メッセージとして分類されます。 長メッセージがあると、IBM MQ は、そのメッセージを一連のページに格納し、それらのページを参照する制御情報を短メッセージの場合と同じ要領で格納します。 160 ページの図 41 を参照してください。

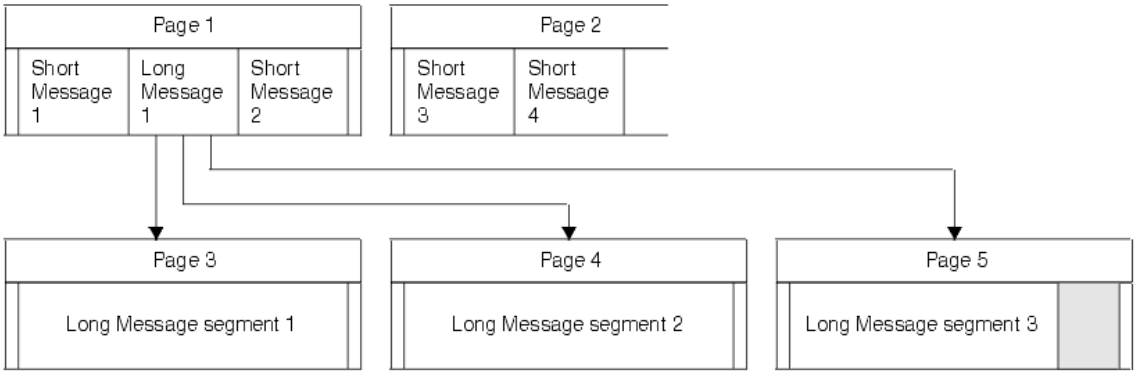


図 *41. IBM MQ* が長メッセージをページ・セットに格納するしくみ

非常に長いメッセージ

非常に長いメッセージは、サイズが 4 MB を超えるメッセージです。 非常に長いメッセージは、それぞれの 4 MB で 1037 ページを使用する、という要領で格納されます。 残りは、前述の長メッセージの場合と同じ要領で格納されます。

z/OS *Enabling dynamic page set expansion*

Page sets can be extended dynamically while the queue manager is running. A page set can have 123 extents, and can be spread over multiple disk volumes.

Each time a page set expands, a new data set extent is used. The queue manager continues to expand a page set when required, until the maximum number of extents has been reached, or until no more storage is available for allocation on eligible volumes.

Once page set expansion fails for one of the reasons above, the queue manager marks the page set for no further expansion attempts. This marking can be reset by altering the page set to EXPAND(SYSTEM).

Page set expansion takes place asynchronously to all other page set activity, when 90% of the existing space in the page set is allocated.

The page set expansion process formats the newly allocated extent and makes it available for use by the queue manager. However, none of the space is available for use, until the entire extent has been formatted. This means that expansion by a large extent is likely to take some time, and putting applications might 'block' if they fill the remaining 10% of the page set before the expansion has completed.

Sample thlqual.SCSQPROC(CSQ4PAGE) shows how to define the secondary extents.

To control the size of new extents, you use one of the following options of the EXPAND keyword of the DEFINE PSID and ALTER PSID commands:

- USER
- SYSTEM
- NONE

USER

Uses the secondary extent size specified when the page set was allocated. If a value was not specified, or if a value of zero was specified, dynamic page set expansion cannot occur.

Page set expansion occurs when the space in the page is 90% used, and is performed asynchronously with other page set activity.

This may lead to expansion by more than a single extent at a time.

Consider the following example: you allocate a page set with a primary extent of 100,000 pages and a secondary extent of 5000 pages. A message is put that requires 9999 pages. If the page set is already

using 85,000 pages, writing the message crosses the 90% full boundary (90,000 pages). At this point, a further secondary extent is allocated to the primary extent of 100,000 pages, taking the page set size to 105,000 pages. The remaining 4999 pages of the message continue to be written. When the used page space reaches 94,500 pages, which is 90% of the updated page set size of 105,000 pages, another 5000 page extent is allocated, taking the page set size to 110,000 pages. At the end of the MQPUT, the page set has expanded twice, and 94,500 pages are used. None of the pages in the second page set expansion have been used, although they were allocated.

At restart, if a previously used page set has been replaced with a data set that is smaller, it is expanded until it reaches the size of the previously used data set. Only one extent is required to reach this size.

SYSTEM

Ignores the secondary extent size that was specified when the page set was defined. Instead, the queue manager sets a value that is approximately 10% of the current page set size. The value is rounded up to the nearest cylinder of DASD.

If a value was not specified, or if a value of zero was specified, dynamic page set expansion can still occur. The queue manager sets a value that is approximately 10% of the current page set size. The new value is rounded up depending on the characteristics of the DASD.

Page set expansion occurs when the space in the page set is approximately 90% used, and is performed asynchronously with other page set activity.

At restart, if a previously used page set has been replaced with a data set that is smaller, it is expanded until it reaches the size of the previously used data set.

NONE

No further page set expansion is to take place.

Related reference
ALTER PSID
DEFINE PSID
DISPLAYUSAGE

Defining your buffer pools

Use this topic to help plan the number of buffer pools you should define, and their settings.

This topic is divided into the following sections:

1. “Decide on the number of buffer pools to define” on page 161
2. “Decide on the settings for each buffer pool” on page 162
3. “Monitor the performance of buffer pools under expected load” on page 163
4. “Adjust buffer pool characteristics” on page 163

Decide on the number of buffer pools to define

You should define four buffer pools initially:

Buffer pool 0
Use for object definitions (in page set zero) and performance critical, system related message queues, such as the SYSTEM.CHANNEL.SYNCQ queue and the SYSTEM.CLUSTER.COMMAND.QUEUE and SYSTEM.CLUSTER.REPOSITORY.QUEUE queues.

However it is important to consider point “7” on page 163 in *Adjust buffer pool characteristics* if a large number of channels, or clustering, is to be used.

Use the remaining three buffer pools for user messages.

Buffer pool 1
Use for important long-lived messages.

Long-lived messages are those that remain in the system for longer than two checkpoints, at which time they are written out to the page set. If you have many long-lived messages, this buffer pool should be relatively small, so that page set I/O is evenly distributed (older messages are written out to DASD each time the buffer pool becomes 85% full).

If the buffer pool is too large, and the buffer pool never gets to 85% full, page set I/O is delayed until checkpoint processing. This might affect response times throughout the system.

If you expect few long-lived messages only, define this buffer pool so that it is sufficiently large to hold all these messages.

Buffer pool 2
Use for performance-critical, short-lived messages.

There is normally a high degree of buffer reuse, using few buffers. However, you should make this buffer pool large to allow for unexpected message accumulation, for example, when a server application fails.

Buffer pool 3
Use for all other (typically, performance noncritical) messages.

Queues such as the dead-letter queue, SYSTEM.COMMAND.* queues and SYSTEM.ADMIN.* queues can also be mapped to buffer pool 3.

Where virtual storage constraints exist, and buffer pools need to be smaller, buffer pool 3 is the first candidate for size reduction.

You might need to define additional buffer pools in the following circumstances:

- If a particular queue is known to require isolation, perhaps because it exhibits different behavior at various times.
  - Such a queue might either require the best performance possible under the varying circumstances, or need to be isolated so that it does not adversely affect the other queues in a buffer pool.
  - Each such queue can be isolated into its own buffer pool and page set.
- You want to isolate different sets of queues from each other for class-of-service reasons.
  - Each set of queues might then require one, or both, of the two types of buffer pools 1 or 2, as described in Suggested definitions for buffer pool settings, necessitating creation of several buffer pools of a specific type.

Decide on the settings for each buffer pool

If you are using the four buffer pools described in “Decide on the number of buffer pools to define” on page 161, then Suggested definitions for buffer pool settings gives two sets of values for the size of the buffer pools.

The first set is suitable for a test system, the other for a production system or a system that will become a production system eventually. In all cases define your buffer pools with the **LOCATION**(*ABOVE*) attribute

Table 21. Suggested definitions for buffer pool settings

Definition setting	Test system	Production system
BUFFPOOL 0	1 050 buffers	50 000 buffers
BUFFPOOL 1	1 050 buffers	20 000 buffers
BUFFPOOL 2	1 050 buffers	50 000 buffers
BUFFPOOL 3	1 050 buffers	20 000 buffers

If you need more than the four suggested buffer pools, select the buffer pool (1 or 2) that most accurately describes the expected behavior of the queues in the buffer pool, and size it using the information in Suggested definitions for buffer pool settings.

Ensure that your MEMLIMIT is set high enough, so that all the buffer pools can be located above the bar.

Monitor the performance of buffer pools under expected load

You can monitor the usage of buffer pools by analyzing buffer pool performance statistics. In particular, you should ensure that the buffer pools are large enough so that the values of QPSTSOS, QPSTSTLA, and QPSTDMC remain at zero.

For further information, see Buffer manager data records.

Adjust buffer pool characteristics

Use the following points to adjust the buffer pool settings from “Decide on the settings for each buffer pool” on page 162, if required.

Use the performance statistics from “Monitor the performance of buffer pools under expected load” on page 163 as guidance.

1. If you are migrating from an earlier version of IBM MQ, only change your existing settings if you have more real storage available.
2. In general, bigger buffer pools are better for performance, and buffer pools can be much bigger if they are above the bar.

   However, at all times you should have sufficient real storage available so that the buffer pools are resident in real storage. It is better to have smaller buffer pools that do not result in paging, than big ones that do.

   Additionally, there is no point having a buffer pool that is bigger than the total size of the page sets that use it, although you should take into account page set expansion if it is likely to occur.
3. Aim for one page set per buffer pool, as this provides better application isolation.
4. If you have sufficient real storage, such that your buffer pools will never be paged out by the operating system, consider using page-fixed buffers in your buffer pool.

   This is particularly important if the buffer pool is likely to undergo much I/O, as it saves the CPU cost associated with page-fixing the buffers before the I/O, and page-unfixing them afterwards.
5. There are several benefits to locating buffer pools above the bar even if they are small enough to fit below the bar. These are:
   - 31 bit virtual storage constraint relief - for example more space for common storage.
   - If the size of a buffer pool needs to be increased unexpectedly while it is being heavily used, there is less impact and risk to the queue manager, and its workload, by adding more buffers to a buffer pool that is already above the bar, than moving the buffer pool to above the bar and then adding more buffers.
6. Tune buffer pool zero and the buffer pool for short-lived messages (buffer pool 2) so that the 15% free threshold is never exceeded (that is, QPSTCBSL divided by QPSTNBUF is always greater than 15%). If more than 15% of buffers remain free, I/O to the page sets using these buffer pools can be largely avoided during normal operation, although messages older than two checkpoints are written to page sets.

   Attention: The optimum value for these parameters is dependent on the characteristics of the individual system. The values given are intended only as a guideline and might not be appropriate for your system.
7. SYSTEM.* queues which get very deep, for example SYSTEM.CHANNEL.SYNCQ, might benefit from being placed in their own buffer pool, if sufficient storage is available.

IBM MQ SupportPac MP16 IBM「MQ z/OS キャパシティ・プランニング&チューニング provides further information about tuning buffer pools.

z/OS z/OS でのロギング環境の計画

このトピックを使用して、IBM MQ によって使用されるログおよびログ・アーカイブの数、サイズ、および配置を計画します。

ログは、以下の目的で使用されます。

- 非持続メッセージに関するリカバリー情報の書き込み
- 非持続メッセージを使用する作業単位に関する情報の記録
- キューの定義など、オブジェクトの変更に関する情報の記録
- CF 構造のバックアップ

さらに、その他の内部情報のためにも使用されます。

IBM MQ ロギング環境は、単一アクティブ・ログまたは二重アクティブ・ログのどちらを指定するか、アーカイブ・ログ・ボリュームに使用するメディア、および使用するログ・バッファーの数などのオプションを指定するシステム・パラメーター・マクロを使用して確立されます。

これらのマクロについては、ブートストラップとログ・データ・セットを作成するとシステム・パラメーター・モジュールの調整を参照してください。

注 : キュー共有グループを使用する場合、SHAREOPTIONS(2 3) を指定してブートストラップ・データ・セットおよびログ・データ・セットを定義していることを確認してください。

このセクションでは、以下のトピックに関する情報を取り上げます。

z/OS *Log data set definitions*

Use this topic to decide on the most appropriate configuration for your log data sets.

This topic contains information to help you answer the following questions:

- Should your installation use single or dual logging?
- How many active log data sets do you need?
- “How large should the active logs be?” on page 165
- Active log placement
- “Active log encryption with z/OS data set encryption” on page 167

Should your installation use single or dual logging?

In general you should use dual logging for production, to minimize the risk of losing data. If you want your test system to reflect production, both should use dual logging, otherwise your test systems can use single logging.

With single logging data is written to one set of log data sets. With dual logging data is written to two sets of log data sets, so in the event of a problem with one log data set, such as the data set being accidentally deleted, the equivalent data set in the other set of logs can be used to recover the data.

With dual logging you require twice as much DASD as with single logging.

If you are using dual logging, then also use dual BSDSs and dual archiving to ensure adequate provision for data recovery.

Dual active logging adds a small performance cost.

Attention: Use of disk mirroring technologies, such as Metro Mirror, are not necessarily a replacement for dual logging and dual BSDS. If a mirrored data set is accidentally deleted, both copies are lost.

If you use persistent messages, single logging can increase maximum capacity by 10-30% and can also improve response times.

Single logging uses 2 - 310 active log data sets, whereas dual logging uses 4 - 620 active log data sets to provide the same number of active logs. Thus single logging reduces the amount of data logged, which might be important if your installation is I/O constrained.

How many active log data sets do you need?

The number of logs depends on the activities of your queue manager. For a test system with low throughput, three active log data sets might be suitable. For a high throughput production system you might want the maximum number of logs available, so, if there is a problem with offloading logs you have more time to resolve the problems.

You must have at least three active log data sets, but it is preferable to define more. For example, if the time taken to fill a log is likely to approach the time taken to archive a log during peak load, define more logs.

Note: Page sets and active log data sets are eligible to reside in the extended addressing space (EAS) part of an extended address volumes (EAV) and an archive log dataset can also reside in the EAS.

You should also define more logs to offset possible delays in log archiving. If you use archive logs on tape, allow for the time required to mount the tape.

Consider having enough active log space to keep a day's worth of data, in case the system is unable to archive because of lack of DASD or because it cannot write to tape. If all the active logs fill up, then IBM MQ is unable to process persistent messages or transactions. It is very important to have enough active log space.

It is possible to dynamically define new active log data sets as a way of minimizing the effect of archive delays or problems. New data sets can be brought online rapidly, using the **DEFINE LOG** command to avoid queue manager 'stall' due to lack of space in the active log.

If you want to define more than 31 active log data sets, you must configure your logging environment to use a version 2 format BSDS. Once a version 2 format BSDS is in use, up to 310 active log data sets can be defined for each log copy ring. See “Planning to increase the maximum addressable log range” on page 176 for information on how you convert to a version 2 format BSDS.

You can tell whether your queue manager is using a version 2 or higher BSDS, either by running the print log map utility (CSQJU004), or from the CSQJ034I message issued during queue manager initialization. An end of log RBA range of FFFFFFFFFFFFFFFF, in the CSQJ034I message, indicates that a version 2, or higher, format BSDS is in use. An end of log RBA range of 0000FFFFFFFFFFFF, in the CSQJ034I message, indicates that a version 1 format BSDS is in use.

When a queue manager is using a version 2, or higher, format BSDS, it is possible to use the **DEFINE LOG** command to dynamically add more than 31 active log data sets to a log copy ring.

How large should the active logs be?

The maximum supported active log size, when archiving to disk or to tape, is 4 GB.

You should create active logs of at least 1 GB in size for production and test systems.

Important: You need to be careful when allocating data sets, because IDCAMS rounds up the size you allocate.

To allocate a 3 GB log specify one of the following options:

- Cylinders(4369)
- Megabytes(3071)
- TRACKS(65535)
- RECORD(786420)

Any one of these allocates 2.99995 GB.

To allocate a 4GB log specify one of the following options:

- Cylinders(5825)
- Megabytes(4095)
- TRACKS(87375)
- RECORD(1048500)

Any one of these allocates 3.9997 GB.

When using striped data sets, where the data set is spread across multiple volumes, the specified size value is allocated on each DASD volume used for striping. So, if you want to use 4 GB logs and four volumes for striping, you should specify:

- CYLinders(1456)
- Megabytes(1023)

Setting these attributes allocates 4*1456 = 5824 Cylinders or 4 * 1023 = 4092 Megabytes.

Note: Striping is supported when using extended format data sets. This is usually set by the storage manager.

See Increasing the size of the active log for information on carrying out the procedure.

Active log placement

You should work with your storage management team to set up storage pools for the queue managers. You need to consider:

- A naming convention, so the queue managers use the correct SMS definitions.
- Space required for active and archive logs. Your storage pool should have enough space for the active logs from a whole day.
- Performance and resilience to failures.

For performance reasons you should consider striping your active log data sets. The I/O is spread across multiple volumes and reduces the I/O response times, leading to higher throughput. See the preceding text for information about allocating the size of the active logs when using striping.

You should review the I/O statistics using reports from RMF or a similar product. Perform the review of these statistics monthly (or more frequently) for the IBM MQ data sets, to ensure there are no delays due to the location of the data sets.

In some situations, there can be much IBM MQ page set I/O, and this can impact the IBM MQ log performance if they are located on the same DASD.

If you use dual logging, ensure that each set of active and archive logs is kept apart. For example, allocate them on separate DASD subsystems, or on different devices.

This reduces the risk of them both being lost if one of the volumes is corrupted or destroyed. If both copies of the log are lost, the probability of data loss is high.

When you create a new active log data, set you should preformat it using CSQJUFMT. If the log is not preformatted, the queue manager formats the log the first time it is used, which impacts the performance.

With older DASD with large spinning disks, you had to be careful which volumes were used to get the best performance.

With modern DASD, where data is spread over many PC sized disks, you do not need to worry so much about which volumes are used.

Your storage manager should be checking the enterprise DASD to review and resolve any performance problems. For availability, you might want to use one set of logs on one DASD subsystem, and the dual logs on a different DASD subsystem.

Active log encryption with z/OS data set encryption

You can apply the z/OS data set encryption feature to active log data sets for queue managers running at IBM MQ for z/OS 9.1.4 or later.

You must allocate these active log data sets with EXTENDED attributes, and a data set key label that ensures the data is AES encrypted.

See the section, confidentiality for data at rest on IBM MQ for z/OS with data set encryption. for more information.

z/OS *Using MetroMirror with IBM MQ*

IBM Metro Mirror, previously known as Synchronous Peer to Peer Remote Copy (PPRC), is a synchronous replication solution between two storage subsystems, where write operations are completed on both the primary and secondary volumes before the write operation is considered to be complete. Metro Mirror can be used in environments that require no data loss in the event of a storage subsystem failure.

Supported data set types

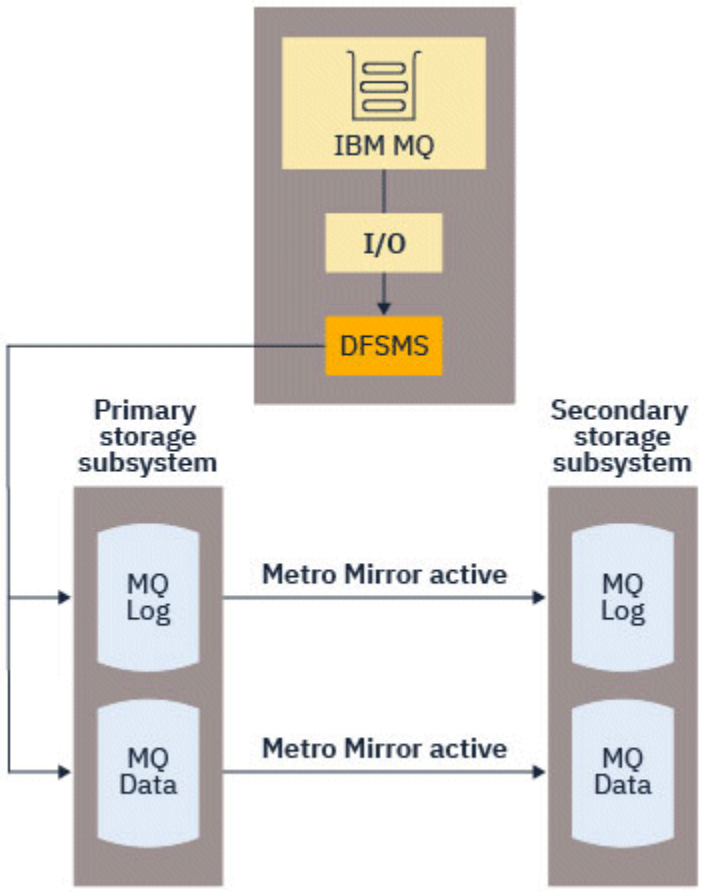
All of the following IBM MQ data set types can be replicated using Metro Mirror. However, exactly which ones are replicated depends on the availability requirements of your enterprise:

- Active logs
- Archive logs
- Bootstrap data set (BSDS)
- Page sets
- Shared message data set (SMDS)
- Data sets used for configuration, for example, in the CSQINP* DD cards on the MSTR JCL

Using zHyperWrite with IBM MQ active logs

When a write is made to a data set that is replicated using Metro Mirror, the write is first made to the primary volume, and then replicated to the secondary volume. This replication is done by the storage subsystem and is transparent to the application that issued the write, for example IBM MQ.

This process is illustrated in the following diagram.

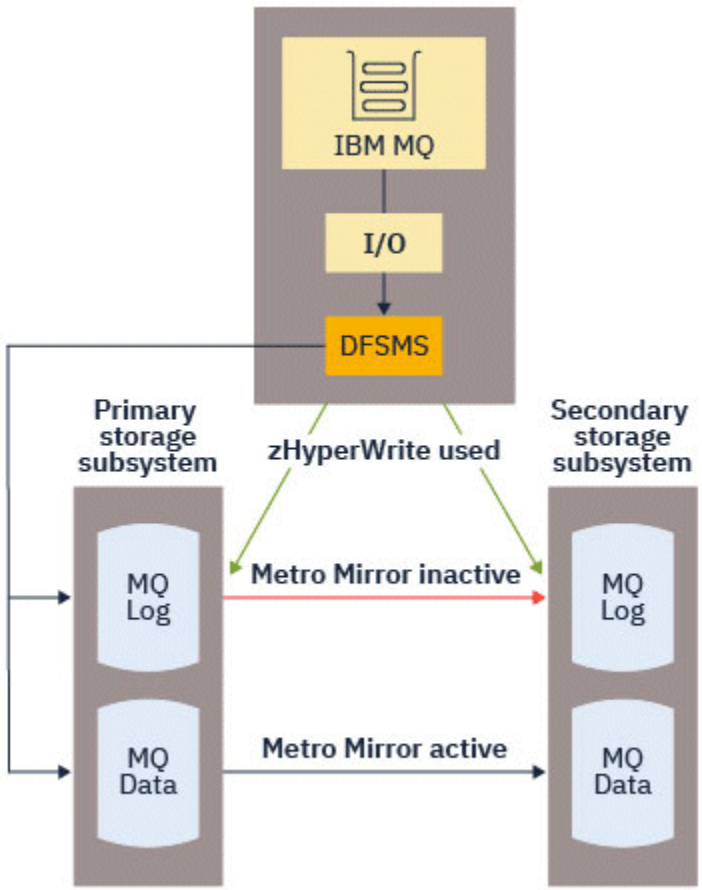


Because both writes to the primary and secondary storage subsystems need to complete before the write returns to IBM MQ, use of Metro Mirror can have a performance impact. You need to balance this performance impact against the availability benefits of using Metro Mirror.

The IBM MQ active logs are most sensitive to the performance impact of using Metro Mirror. IBM MQ allows use of zHyperWrite with the active logs to help reduce this performance impact.

zHyperWrite is a storage subsystem technology that works with z/OS to reduce the performance impact of writes made to data sets that are replicated using Metro Mirror. When zHyperWrite is used, the write to the primary and secondary volumes are issued in parallel at the Data Facility Storage Management Subsystem (DFSMS) level, instead of sequentially at the storage subsystem level, thereby reducing the performance impact.

The following diagram illustrates zHyperWrite being used for the active logs, and Metro Mirror being used for the other IBM MQ data set types. Note that if a zHyperWrite write fails, DFSMS will transparently reissue the write using Metro Mirror.



zHyperWrite on IBM MQ, is supported only on the active log data sets.

In order to use zHyperWrite with the active logs, you need to:

- Configure IBM MQ to use zHyperWrite, and
- The active logs need to be on zHyperWrite capable volumes

You can configure IBM MQ to use zHyperWrite by using one of the following methods:

- Specify ZHYWRITE(YES) in the system parameter module.
- Issue the command SET LOG ZHYWRITE(YES).

Set the following conditions for active log data sets to be on zHyperWrite capable volumes:

- Enable the volumes for Metro Mirror, and the volumes support zHyperWrite
- Ensure that the volumes are HyperSwap enabled
- Specify HYPERWRITE=YES in the IECIOSxx parameter

V 9.4.0 Prior to IBM MQ 9.4.0, if all the preceding conditions are met, then writes to the active logs are enabled for zHyperWrite. If one, or more, of these conditions are not met, IBM MQ writes to the active logs as normal, and Metro Mirror replicates the writes if it is configured.

V 9.4.0 From IBM MQ 9.4.0, if ZHYWRITE(YES) is specified, then IBM MQ always attempts to use zHyperWrite when writing to the active logs, regardless of whether the logs are on zHyperWrite capable volumes. If the logs are not on zHyperWrite capable volumes then Metro Mirror replicates the writes if it is configured. There are no negative effects of attempting to use zHyperWrite if the logs are not on zHyperWrite capable volumes

Notes:

- IBM MQ does not require that all active log data sets are on zHyperWrite capable volumes.

If IBM MQ detects that some active log data sets are on zHyperWrite capable volumes, and others are not, it issues message CSQJ166E and carries on processing.

- IBM MQ checks whether active log data sets are zHyperWrite capable when the data sets are first opened.

  Log data sets are opened either at queue manager start up, or when dynamically adding using the DEFINE LOG command. If the log data sets are made zHyperWrite capable while a queue manager has them open, the queue manager will not detect this until it has been restarted.

You can use the output of the DISPLAY LOG command to indicate whether the current active log data sets are zHyperWrite capable. The following example shows that both of the data sets are zHyperWrite capable. If the queue manager has been configured with ZHYWRITE(YES), writes to these logs would be enabled for zHyperWrite:

```
Copy %Full zHyperWrite DSName
 1      4  CAPABLE     MQTST.SUBSYS.MQDL.LOGCOPY1.DS001
 2      4  CAPABLE     MQTST.SUBSYS.MQDL.LOGCOPY2.DS001
```

z/OS V 9.4.0 *zHyper* リンクを使用したログ・スループットの高速化

zHyper リンク テクノロジーは、CPU と I/O デバイスの間に高速で信頼性の高い直接通信パスを提供することで、入出力 (I/O) のレイテンシを削減するように設計されています。

の概要 zHyper リンク

zHyper リンクは、アクティブ・ログのスループットを向上させ、IBM MQ トランザクション時間を最大 3.5 回短縮することができます。 この目標は、インストールすることで達成されます zHyper リンクアダプタ z/OS ホスト、選択 IBM ストレージハードウェアとそれらを接続して zHyper リンクケーブル。 これにより、CPU と I/O デバイス間のポイントツーポイント接続が作成され、I/O 応答時間が最大 10 倍短縮されます。IBM z 高性能 FICON® (zHPF)。 このような短い応答時間を実現するには、同期入出力要求を使用します。

非同期 I/O に対する同期 I/O の利点

の IBM MQ ロガー タスクは、ログに書き込む必要がある次のデータを待機するループで構成されます。 そのデータが利用可能になると、ロガーは書き込みをスケジュールし、書き込みが完了するまで待機してから、次のデータに進みます。

従来の I/O は CPU よりも遅いため、I/O を非同期で実行して CPU を他のタスクに解放するのが最も効率的です。 したがって、従来の非同期 I/O では、書き込みが完了するまでロガー タスクを一時停止する必要があります。 書き込みが完了すると、ロガー タスクは CPU が使用可能になるまで待機する必要があり、短い再ディスパッチ遅延と、CPU キャッシュの再作成によって発生する遅延が追加されます。

zHyper リンクは CPU の速度に近い、はるかに高速な I/O 時間を提供するため、zHyper リンク、I/O は同期的に実行できるため、書き込み操作中にロガー タスクが中断されることはなく、再ディスパッチとキャッシュ関連の遅延がなくなります。

書き込みが行われている間、ロガー タスクは引き続き CPU をアクティブに使用しているため、従来の I/O と比較して CPU 使用率が増加します。

キューマネージャが使用しようとすると zHyper リンク、そして zHyper たとえば、構成の問題によりリンク書き込みが失敗すると、キュー マネージャーは透過的に従来の I/O にフォールバックします。

最小ハードウェア要件:

- IBM z14 以降
- DS8880 以降

ソフトウェア要件

- zHyperLink Express は z/OS 2.3 以降でサポートされます。

- z/OS イメージは、 IBM z/VM®の下でゲストとしてではなく、LPAR で実行する必要があります。
- zHyper リンクでは、 IBM z High-Performance FICON (zHPF) を使用可能にする必要があります。

IBM MQ アクティブ・ログとの zHyper リンクの使用

使用するには zHyper キュー マネージャーのアクティブ ログにリンクするには、次の操作を行う必要があります。

- 構成、設定 IBM MQ 使用する zHyper リンク、そして
- アクティブログがオンになっていることを確認する zHyper リンク可能なボリューム。

  見るはじめに IBMzHyper リンク z/OS 詳細については。

設定できます IBM MQ 使用する zHyper 次のいずれかの方法でリンクします。

- ログ・パラメーターに ZHYLINK(YES) を指定します。
- コマンド SET LOG ZHYLINK(YES) を発行します。

注 :

- zHyper リンクでは、 zHyper 書き込みがオンになっている必要があります。 つまり、ZHYLINK を使用するには、ログ・パラメーターで ZHYWRITE もオンにする必要があります。 ZHYWRITE (NO) がキュー・マネージャーに設定されている場合にのみ ZHYLINK (YES) を指定すると、ZHYWRITE パラメーターは自動的に YES にオーバーライドされます。
- ZHYWRITE (NO) を指定して ZHYLINK (YES) を明示的に設定しようとすると、SET LOG コマンドが異常終了します。
- ZPRMs で ZHYLINK=YES を設定すると、ZHYWRITE は YES にオーバーライドされます。

問題が発生している場合は、 zHyper のトラブルシューティングのリンク を参照してください。

IBM MQ では、すべてのアクティブ・ログ・データ・セットが zHyper リンク対応ボリューム上にある必要はありませんが、そうすることをお勧めします。 IBM MQ は、一部のアクティブ・ログ・データ・セットが zHyper リンク対応ボリューム上にあることを検出し、その他のアクティブ・ログ・データ・セットがないことを検出すると、メッセージ CSQJ601E を出し、処理を続行します。

IBM MQ は、データ・セットが最初にオープンされたときに、アクティブ・ログ・データ・セットが zHyper リンク可能であるかどうかを検査します。 ログ・データ・セットは、キュー・マネージャーの始動時、または DEFINE LOG コマンドを使用した動的追加時にオープンされます。 キュー・マネージャーがログ・データ・セットをオープンしているときに zHyper リンクを使用可能にすると、キュー・マネージャーは再始動されるまでこれを検出しません。

ZHYLINK (YES) が指定されている場合、IBM MQ は、ログが zHyper リンク対応ボリューム上にあるかどうかに関係なく、アクティブ・ログへの書き込み時に常に zHyper リンクを使用しようとします。 ログが zHyper リンク対応ボリューム上にない場合、 zHyper リンクを使用しようとしても悪影響はありません。

DISPLAY LOG コマンドの出力を使用して、現行アクティブ・ログ・データ・セットの zHyper リンクの状況を示すことができます。

```
Copy %Full zHyperWrite    Encrypted          DSName
  1     81 YES          NO                MQTST.SUBSYS.MQDL.LOGCOPY1.DS001
  2     81 YES            NO                MQTST.SUBSYS.MQDL.LOGCOPY2.DS001
Copy zHyperLink
  1  YES
  2  YES
```

zHyper リンク状況は、以下のいずれかです。

YES
zHyper キュー・マネージャーでリンクが使用可能になっており、すべての書き込みで試行されます。

NO
zHyper リンクがキュー・マネージャーで使用可能になっておらず、データ・セットが zHyper リンク対応ボリューム上に **ありません** 。

CAPABLE
zHyper リンクがキュー・マネージャーで使用可能になっておらず、データ・セット **が zHyper リンク対応ボリューム上の** です。

監視と理解のための SMF 統計が複数追加されています。zHyper リンクパフォーマンス。zHyper リンク統計詳細については。

書き込みセッション

使用する場合 zHyper リンク、1 つ以上 zHyperDASD とのリンク書き込みセッションが確立されます。 現在の DASD は最大 64 の同時書き込みセッションをサポートしているため、どのキューマネージャーを有効にするかを慎重に検討する必要があります。zHyper リンクがオンになっているか、他のサブシステム、例えば Db2 も使用しています zHyper 同じ DASD に書き込むためのリンク。 利用可能な書き込みセッションが不足すると、キュー マネージャーは自動的に従来の非同期 I/O の使用に戻ります。

数を計算することができます zHyper 書き込みセッションを次のようにリンクします。

```
Number of log copies (either 1 or 2) * number of stripes per log copy * 2
if Metro Mirror (PPRC) is used.
```

したがって、1 つのストライプと 1 つの Metro Mirror 単一の書き込みセッションを使用します。 2 つのストライプと PPRC を備えたデュアル ロギング モードのキュー マネージャーは、8 つの書き込みセッションを使用します。

注 : その間 Metro Mirror 書き込みセッションが 2 倍使用される結果、それらの書き込みセッションは 2 つのミラーリングされた DASD 間で均等に分割されます。

z/OS *Planning your log archive storage*

Use this topic to understand the different ways of maintaining your archive log data sets.

You can place archive log data sets on standard-label tapes, or DASD, and you can manage them by data facility hierarchical storage manager (DFHSM). Each z/OS logical record in an archive log data set is a VSAM control interval from the active log data set. The block size is a multiple of 4 KB.

Archive log data sets are dynamically allocated, with names chosen by IBM MQ. The data set name prefix, block size, unit name, and DASD sizes needed for such allocations are specified in the system parameter module. You can also choose, at installation time, to have IBM MQ add a date and time to the archive log data set name.

It is not possible to specify with IBM MQ, specific volumes for new archive logs, but you can use Storage Management routines to manage this. If allocation errors occur, offloading is postponed until the next time offloading is triggered.

If you specify dual archive logs at installation time, each log control interval retrieved from the active log is written to two archive log data sets. The log records that are contained in the pair of archive log data sets are identical, but the end-of-volume points are not synchronized for multivolume data sets.

Should your archive logs reside on tape or DASD?

When deciding whether to use tape or DASD for your archive logs, there are a number of factors that you should consider:

- Review your operating procedures before deciding about tape or disk. For example, if you choose to archive to tape, there must be enough tape drive when they are required. After a disaster, all subsystems might want tape drives and you might not have as many free tape drives as you expect.
- During recovery, archive logs on tape are available as soon as the tape is mounted. If DASD archives have been used, and the data sets migrated to tape using hierarchical storage manager (HSM), there is a delay while HSM recalls each data set to disk. You can recall the data sets before the archive log is used. However, it is not always possible to predict the correct order in which they are required.

- When using archive logs on DASD, if many logs are required (which might be the case when recovering a page set after restoring from a backup) you might require a significant quantity of DASD to hold all the archive logs.
- In a low-usage system or test system, it might be more convenient to have archive logs on DASD to eliminate the need for tape mounts.
- Both issuing a RECOVER CFSTRUCT command, and backing out a persistent unit of work, result in the log being read backwards. Tape drives with hardware compression perform badly on operations that read backwards. Plan sufficient log data on DASD to avoid reading backwards from tape.

Archiving to DASD offers faster recoverability but is more expensive than archiving to tape. If you use dual logging, you can specify that the primary copy of the archive log go to DASD and the secondary copy go to tape. This increases recovery speed without using as much DASD, and you can use the tape as a backup.

See “Changing the storage medium for archive logs” on page 174 for details of how you archive your logs from tape to DASD, and how you carry out the reverse process.

Archiving to tape

If you choose to archive to a tape device, IBM MQ can extend to a maximum of 20 volumes.

If you are considering changing the size of the active log data set so that the set fits on one tape volume, note that a copy of the BSDS is placed on the same tape volume as the copy of the active log data set. Adjust the size of the active log data set downward to offset the space required for the BSDS on the tape volume.

If you use dual archive logs on tape, it is typical for one copy to be held locally, and the other copy to be held off-site for use in disaster recovery.

Archiving to DASD volumes

IBM MQ requires that you catalog all archive log data sets allocated on non-tape devices (DASD). If you choose to archive to DASD, the CATALOG parameter of the CSQ6ARVP macro must be YES. If this parameter is NO, and you decide to place archive log data sets on DASD, you receive message CSQJ072E each time an archive log data set is allocated, although IBM MQ still catalogs the data set.

If the archive log data set is held on DASD, the archive log data sets can extend to another volume; multivolume is supported.

If you choose to use DASD, make sure that the primary space allocation (both quantity and block size) is large enough to contain either the data coming from the active log data set, or that from the corresponding BSDS, whichever is the larger of the two.

This minimizes the possibility of unwanted z/OS X'B37' or X'E37' abend codes during the offload process. The primary space allocation is set with the PRIQTY (primary quantity) parameter of the CSQ6ARVP macro.

Archive log data sets can exist on large or extended-format sequential data sets. SMS ACS routines now use DSNTYPE(LARGE) or DSNTYPE(EXT).

IBM MQ supports allocation of archive logs as extended format data sets. When extended format is used, the maximum archive log size is increased from 65535 tracks to the maximum active log size of 4GB. Archive logs are eligible for allocation in the extended addressing space (EAS) of extended address volumes (EAV).

Where the required hardware and software levels are available, allocating archive logs to a data class defined with COMPACTION using zEDC might reduce the disk storage required to hold archive logs. For more information, see IBM MQ for z/OS: Reducing storage occupancy with IBM zEnterprise Data Compression (zEDC) and zEnterprise Data Compression (zEDC) for more information.

The z/OS data set encryption feature can be applied to archive logs for queue managers running on IBM MQ. These archive logs must be allocated through Automatic Class Selection (ACS) routines to a

data class defined with EXTENDED attributes, and a data set key label that ensures the data is AES encrypted.

Using SMS with archive log data sets

If you have MVS/DFP storage management subsystem (DFSMS) installed, you can write an Automatic Class Selection (ACS) user-exit filter for your archive log data sets, which helps you convert them for the SMS environment.

Such a filter, for example, can route your output to a DASD data set, which DFSMS can manage. You must exercise caution if you use an ACS filter in this manner. Because SMS requires DASD data sets to be cataloged, you must make sure the CATALOG DATA field of the CSQ6ARVP macro contains YES. If it does not, message CSQJ072E is returned; however, the data set is still cataloged by IBM MQ.

For more information about ACS filters, see Data sets that DFSMShsm dynamically allocates during aggregate backup processing.

z/OS *Changing the storage medium for archive logs*

The procedure for changing the storage medium used by archive logs.

About this task

This task describes how to change the storage medium used for archive logs, for example moving from archiving to tape to archiving to DASD.

You have a choice of how to make the changes:

1. Make the changes only using the CSQ6ARVP macro so that they are applied from the next time the queue manager restarts.
2. Make the changes using the CSQ6ARVP macro, and dynamically using the SET ARCHIVE command. This means that the changes apply from the next time the queue manager archives a log file, and persist after the queue manager restarts.

Procedure

1. Changing so archive logs are stored on DASD instead of tape:
   a) Read the section "Archiving to DASD volumes" on page 173 and review the CSQ6ARVP parameters.
   b) Make changes to the following parameters in CSQ6ARVP
      - Update the UNIT and, if necessary, the UNIT2 parameters.
      - Update the BLKSIZE parameter, as the optimal setting for DASD differs from tape.
      - Set the PRIQTY and SECQTY parameters to be large enough to hold the largest of the active log or BSDS.
      - Set the CATALOG parameter to be YES.
      - Confirm the ALCUNIT setting is what you want. You should use BLK, because it is independent of the device type.
      - Set the ARCWTOR parameter to NO if it is not already.
2. Changing so archive logs are stored on tape instead of DASD:
   a) Read the section "Archiving to tape" on page 173, and review the CSQ6ARVP parameters.
   b) Make changes to the following parameters in CSQ6ARVP:
      - Update the UNIT and, if necessary, the UNIT2 parameters.
      - Update the BLKSIZE parameter, as the optimal setting for tape differs from DASD.
      - Confirm the ALCUNIT setting is what you want. You should use BLK, because it is independent of the device type.
      - Review the setting of the ARCWTOR parameter.

z/OS *How long do I need to keep archive logs*

Use the information in this section to help you plan your backup strategy.

You specify how long archive logs are kept in days , using the ARCRETN parameter in USING CSQ6ARVP or the SET SYSTEM command. After this period the data sets can be deleted by z/OS.

You can manually delete archive log data sets when they are no longer needed.

- The queue manager might need the archive logs for recovery.

  The queue manager can only keep the most recent 1000 archives in the BSDS, When the archive logs are not in the BSDS they cannot be used for recovery, and are only of use for audit, analysis, or replay type purposes.
- You might want to keep the archive logs so that you can extract information from the logs. For example, extracting messages from the log, and reviewing which user ID put or got the message.

The BSDS contains information on logs and other recovery information. This data set is a fixed size. When the number of archive logs reaches the value of MAXARCH in CSQ6LOGP, or when the BSDS fills up, the oldest archive log information is overwritten.

There are utilities to remove archive log entries from the BSDS, but in general, the BSDS wraps and overlays the oldest archive log record.

When is an archive log needed

You need to back up your page sets regularly. The frequency of backups determines which archive logs are needed in the event of losing a page set.

You need to back up your CF structures regularly. The frequency of backups determines which archive logs are needed in the event of losing data in the CF structure.

The archive log might be needed for recovery. The following information explains when the archive log might be needed, where there are problems with different IBM MQ resources.

Loss of a page set

You must recover your system from your backup and restart the queue manager.

You need the logs from when the backup was taken, as well as up to three log data sets prior to the backup being taken.

All LPARs lose connectivity to a CF structure, or the structure is unavailable

Use the RECOVER CFSTRUCT command to recover the structure.

Structure recovery requires the logs from all queue managers that have accessed the structure since the last backup (back to the time when the backup was taken) plus the structure backup itself in the log of the queue manager that took the backup.

If you have been doing frequent backups of the CF structures, the data should be in active logs, and you should not need archive logs.

If there is no recent backup of the CF structure, you might need archive logs.

Note: All non persistent messages will be lost; all persistent messages will be re-created by performing the following tasks:

1. Reading the last CF structure backup from the log
2. Reading the logs from all queue managers that have used the structure
3. Merging updates since the backup

Administration structure rebuild

If you need to rebuild the administration structure, the information is read from the last checkpoint of the log for each queue manager in the QSG.

If a queue manager is not active, another queue manager in the QSG reads the log.

You should not need archive logs.

Loss of an SMDS data set

If you lose an SMDS data set, or the data set gets corrupted, the data set becomes unusable and the status for it is set to FAILED. The CF structure is unchanged.

In order to restore the SMDS data set, you need to:

1. Redefine the SMDS data set, and
2. Recover the CF structure by issuing the RECOVER CFSTRUCT command.

   Note: All non persistent messages on the CF structure will be lost; all persistent messages will be restored.

The requirement for queue manager logs is the same as for recovering from a structure that is unavailable.

z/OS *Planning to increase the maximum addressable log range*

You can increase the maximum addressable log range by configuring your queue manager to use a larger log relative byte address (RBA).

The log RBA size was increased from IBM MQ for z/OS 8.0. For an overview of this change, see Larger log Relative Byte Address.

Queue managers created at IBM MQ 9.3.0 or later, have 8 byte log RBA enabled by default and, therefore, do not require conversion.

You can convert your queue managers to use 8 byte log RBA values at any time. A queue sharing group can contain some queue managers with 8 byte log RBA enabled, and some queue managers with 6 byte log RBA enabled.

Undoing the change

The change cannot be backed out.

How long does it take?

The change requires a queue manager restart. Stop the queue manager, run the CSQJUCNV utility against the bootstrap data set (BSDS), or data sets, to create new data sets, rename these bootstrap data sets, and restart the queue manager. The CSQJUCNV utility usually takes a few seconds to run.

What impact does this have?

- With 8 byte log RBA in use, every write of data to the log data sets has additional bytes. Therefore, for a workload consisting of persistent messages there is a small increase in the amount of data written to the logs.
- Data written to a page set, or coupling facility (CF) structure, is not affected.

Related tasks

Implementing the larger log Relative Byte Address

z/OS チャネル・イニシエーターの計画

チャネル・イニシエーターは、キュー・マネージャー間の通信を提供する機能であり、独自のアドレス・スペースで稼働します。

接続には次の 2 つのタイプがあります。

1. ネットワークを経由したキュー・マネージャーへのアプリケーション接続。 クライアント・チャネルといいます。
2. キュー・マネージャー間の接続。 MCA チャネルといいます。

リスナー

チャネル・リスナー・プログラムは、着信ネットワーク要求を listen し、該当するチャネルが必要になった時にそのチャネルを開始します。 チャネル・イニシエーターがインバウンド接続を処理するために、少なくとも 1 つの IBM MQ リスナー・タスクを構成しておく必要があります。 リスナーは、TCP リスナーか LU 6.2 リスナーのいずれかになります。

各リスナーに TCP ポートか LU 名が必要です。

ただし、1 つのチャネル・イニシエーターに複数のリスナーを設定することも可能です。

TCP/IP

チャネル・イニシエーターは、同じ z/OS イメージ上の複数の TCP スタックで作動できます。 例えば、内部接続用に 1 つの TCP スタック、外部接続用にもう 1 つの TCP スタックを設定する、といった具合です。

出力チャネルを定義する場合は、以下のようにします。

1. 接続の宛先のホストとポートを設定します。 これは次のいずれかになります。
   - IP アドレス (例: `10.20.4.6`)
   - ホスト名 (例: `mvs-prod.myorg.com`)

   ホスト名を使用して宛先を指定すると、IBM MQ はドメイン・ネーム・システム (DNS) を使用して宛先の IP アドレスを解決します。
2. 複数の TCP スタックを使用する場合は、チャネル定義で **LOCLADDR** パラメーターを指定できます。そのパラメーターで、使用する IP スタックのアドレスを指定します。

高可用性の DNS サーバーの計画を立ててください。 DNS が使用不可になると、アウトバウンド・チャネルを開始できないことがあります。ホスト名を使用して着信接続をマップするチャネル認証規則も処理できません。

APPC および LU 6.2

APPC を使用する場合は、チャネル・イニシエーターに LU 名と APPC の構成が必要です。

キュー共有グループ

単一システム・イメージを用意し、キュー共有グループ内のどのキュー・マネージャーでも着信 IBM MQ 接続要求に対応できるようにするには、いくらかの構成作業が必要です。 以下に例を示します。

1. ハードウェア・ネットワーク・ルーター。 企業側から見えるこのルーターの IP アドレスは 1 つだけです。 このルーターによって、このハードウェアに接続されている任意のキュー・マネージャーに初期要求をルーティングできます。
2. 仮想 IP アドレス (VIPA)。 エンタープライズ全体の IP アドレスを指定し、そのアドレスをシスプレックス内の任意の TCP スタックにルーティングします。 TCP スタックは、そのアドレスをシスプレックス内の任意のリスニング・キュー・マネージャーにルーティングします。

IBM MQ トラフィックの保護

TLS 接続を使用してワイヤー上のデータを保護するように IBM MQ を構成できます。 TLS を使用するには、デジタル証明書と鍵リングを使用する必要があります。

チャネルのリモート側の担当者と協力し、互換性のある IBM MQ 定義と互換性のある証明書を用意することも必要です。

以下のいずれかに基づいて、IBM MQ に接続できる接続とユーザー ID を制御できます。

- IP アドレス
- クライアント・ユーザー ID
- リモート・キュー・マネージャー

- デジタル証明書 (チャネル認証レコードを参照)

クライアント・アプリケーションを制限するために、クライアント・アプリケーションから有効なユーザー ID とパスワードを提供してもらうように設定することも可能です (接続認証を参照)。

まずチャネル・イニシエーターを作動させてから、TLS を使用するように各チャネルを 1 つずつ構成することもできます。

チャネル・イニシエーターのモニター

チャネル・イニシエーターとチャネルに関する情報を確認するための MQSC コマンドがあります。

- DISPLAY CHINIT コマンドでは、チャネル・イニシエーターとアクティブ・リスナーに関する情報を確認できます。
- DISPLAY CHSTATUS コマンドでは、チャネルのアクティビティーと状況を表示できます。

チャネル・イニシエーターは、チャネル・イニシエーター・タスクおよびチャネルのアクティビティーに関する情報を含む SMF レコードを生成することもできます。 詳しくは、 179 ページの『チャネル・イニシエーター SMF データの計画』 を参照してください。

チャネル・イニシエーターは、チャネルの開始時と停止時にメッセージをジョブ・ログに出力します。 企業の自動化処理でそうしたメッセージを使用して状況を収集できます。 数秒だけアクティブになるチャネルもあるので、そうしたメッセージが多数作成される場合もあります。 これらのメッセージを抑止するには、 z/OS メッセージ処理機能を使用するか、 SET SYSTEM コマンドで **EXCLMSG** を設定します。

IBM MQ チャネル定義の構成

多数のキュー・マネージャーを相互に接続すると、すべてのオブジェクト定義を管理するのが難しくなります。 IBM MQ クラスタリングを使用すれば、その作業をシンプルにできます。

2 つのキュー・マネージャーを完全リポジトリーとして指定します。 その他のキュー・マネージャーでは、1 つのリポジトリーへの接続が 1 つと、もう 1 つのリポジトリーからの接続が 1 つ必要になります。 他のキュー・マネージャーへの接続が必要な場合は、該当キュー・マネージャーがチャネルを自動的に作成して開始します。

クラスターに多数のキュー・マネージャーを組み込むことを計画している場合は、専用リポジトリーとして機能するキュー・マネージャー (アプリケーション・トラフィックのないキュー・マネージャー) を配置するようにしてください。

詳しくは、 20 ページの『分散キューおよびクラスターの計画』 を参照してください。

チャネル・イニシエーターの構成前のアクション

1. TCP/IP または APPC を使用するかを決定します。
2. TCP を使用する場合は、IBM MQ に少なくとも 1 つのポートを割り振ります。
3. DNS サーバーが必要な場合は、必要に応じてサーバーの可用性が高くなるように構成します。
4. APPC を使用する場合は、LU 名を割り振り、APPC を構成します。

チャネル・イニシエーターの構成後のアクション (実動に移行する前)

1. 使用する接続に関する計画を立てます。
   a. リモート・アプリケーションからのクライアント接続。
   b. 他のキュー・マネージャーとの間の MCA チャネル。 通常は、各リモート・キュー・マネージャーとの間のチャネルがあります。
2. クラスタリングをセットアップするか、既存のクラスタリング環境に参加します。
3. チャネル・イニシエーターの前で可用性を実現するために、複数の TCP スタックを使用するか、VIPA を使用するか、外部ルーターを使用するかを検討します。
4. TLS の使用を計画している場合は、以下のようにします。

a. 鍵リングをセットアップします。
b. 証明書をセットアップします。

5. チャネル認証の使用を計画している場合は、以下のようにします。
   a. インバウンド・セッションを MCA ユーザー ID にマップするための基準を決定します。
   b. キュー・マネージャー・パラメーター **REVDNS** を設定して、リバース DNS 参照を有効にします。
   c. セキュリティーについて検討します。 例えば、デフォルト・チャネルを削除したり、チャネルの **MCAUSER** 属性で必要な権限だけを設定したユーザー ID を指定したりします。
6. チャネル・イニシエーターによって作成されたアカウンティングと統計の SMF レコードを取り込んで、後処理を実行します。
7. ジョブ・ログ・メッセージのモニターを自動化します。
8. 必要に応じて、ネットワーク環境を調整してスループットを改善します。 TCP では、大きな送信バッファーと受信バッファーを使用すると、スループットが向上します。 以下のコマンドを使用すれば、MQ で特定の TCP バッファー・サイズを使用するよう、強制設定を行えます。

```
RECOVER QMGR(TUNE CHINTCPRBDYNSZ nnnnn)
RECOVER QMGR(TUNE CHINTCPSBDYNSZ nnnnn)
```

   チャネルの SO_RCVBUF と SO_SNDBUF が nnnnn のサイズ (バイト単位) に設定されます。

関連概念

147 ページの『Planning for your queue manager』
When you are setting up a queue manager, your planning should allow for the queue manager to grow, so that the queue manager meets the needs of your enterprise.

z/OS チャネル・イニシエーター SMF データの計画

チャネル・イニシエーターの SMF データ収集の実装を計画する必要があります。

チャネル・イニシエーターは以下の 2 種類のレコードを生成します。

- 統計データ。チャネル・イニシエーターおよびその中のタスクに関する情報が含まれます。
- チャネル・アカウンティング・データ。DISPLAY CHSTATUS コマンドと同様の情報が含まれます。

統計データの収集を開始するには、以下のコマンドを使用します。

```
START TRACE(STAT) CLASS(4)
```

停止するには、次のコマンドを使用します。

```
STOP TRACE(STAT) CLASS(4)
```

アカウンティング・データの収集を開始するには、次のコマンドを使用します。

```
START TRACE(ACCTG) CLASS(4)
```

停止するには、次のコマンドを使用します。

```
STOP TRACE(ACCTG) CLASS(4)
```

チャネル定義またはキュー・マネージャーの **STATCHL** 属性を使用して、どのチャネルに収集されるアカウンティング・データが含まれるかを制御できます。

- クライアント・チャネルの場合、キュー・マネージャー・レベルで **STATCHL** を設定する必要があります。
- 自動的に定義されたクラスター送信側チャネルの場合、**STATACLS** キュー・マネージャー属性を使用して、アカウンティング・データの収集を制御できます。

キュー・マネージャーの **STATCHL** のデフォルト値は OFF です。 チャネル・アカウンティング・データを収集するには、クラス 4 アカウンティング・トレースを開始することに加え、キュー・マネージャーまたはチャネル定義のいずれかで **STATCHL** の値をデフォルトから変更する必要があります。

SMF レコードは、以下の時点で生成されます。

- IBM MQ for z/OS 9.3.0 以降では、CSQ6SYSP の **STATIME** パラメーターまたは **ACCTIME** パラメーターで指定された時間間隔が経過した場合。または、SMF データ収集ブロードキャストで **STATIME** または **ACCTIME** がゼロである場合。 キュー・マネージャーおよびチャネル・イニシエーターの SMF データを収集する要求が同期されます。
- `STOP TRACE(ACCTG) CLASS(4)` コマンドまたは `STOP TRACE(STAT) CLASS(4)` コマンドが発行された場合、あるいは
- チャネル・イニシエーターがシャットダウンした場合。 この時点で SMF データがすべて書き出されます。

あるチャネルが SMF 間隔内で停止した場合、次回に SMF 処理が実行されるときにアカウンティング・データが SMF に書き込まれます。 クライアントが接続し、何らかの操作を行った後に切断し、さらに再接続して切断した場合には、チャネル・アカウンティング・データが 2 セット生成されます。

通常、統計データは 1 つの SMF レコードに収まりますが、多数のタスクが使用されている場合は複数の SMF レコードが作成されることがあります。

アカウンティング・データは、それが有効化されているチャネルごとに収集され、通常は 1 つの SMF レコードに収まります。 ただし、多数のチャネルがアクティブである場合は複数の SMF レコードが作成されることがあります。

チャネル・イニシエーター SMF データの収集コストはわずかです。 通常、CPU 使用率の増加は数パーセント以下で、ほとんどの場合は測定誤差の範囲内です。

この機能を使用する前に、z/OS システム・プログラマーに連絡を取って、追加のレコードを扱う能力が SMF にあることを確認し、新しい SMF データを含めるよう SMF レコードの抽出プロセスを変更してもらう必要があります。

チャネル・イニシエーター統計データの場合、SMF レコード・タイプは 115、サブタイプは 231 です。

チャネル・イニシエーター・アカウンティング・データの場合、SMF レコード・タイプは 116、サブタイプは 10 です。

このデータを処理する独自のプログラムを作成することができます。または、サポートパック MP1B も使用できます (これに含まれる MQSMF プログラムを使用すると、データを印刷したり、スプレッドシートへのインポートに適したコンマ区切り値 (CSV) 形式のデータを作成したりすることができます)。

チャネル・イニシエーター SMF データの収集で問題が発生する場合は、チャネル・イニシエーター (CHINIT) 用の SMF データを取り込む際の問題の処理を参照してください。

関連タスク

IBM MQ パフォーマンス統計の解釈

チャネル・アカウンティング・データのトラブルシューティング

z/OS z/OS TCP/IP 環境の計画

ネットワークのスループットを最高にするには、64 KB 以上のサイズの TCP/IP 送受信バッファーを使用する必要があります。 このサイズを指定すると、システムによってバッファー・サイズが最適化されます。

Dynamic Right Sizing for High Latency Networks とは何ですか? を参照してください。 for more information.

システム・バッファー・サイズを確認するには、例えば次のような Netstat コマンドを使用します。

```
TSO NETSTAT ALL (CLIENT csq1CHIN
```

結果には、次の 2 つの値を含む、多くの情報が表示されます。

```
ReceiveBufferSize: 0000065536
SendBufferSize: 0000065536
```

65536 は 64 KB です。 バッファー・サイズが 65536 より小さい場合は、ネットワークの担当部署と協力して、TCPIP プロシージャーの PROFILE DDName にある **TCPSENDBFRSIZE** および **TCPRCVBUFRSIZE** の値を増やしてください。 例えば、以下のようなコマンドを使用できます。

```
TCPCONFIG TCPSENDBFRSZE 65536 TCPRCVBUFRSIZE 65536
```

システム全体の'**TCPSENDBFRSIZE** または'**TCPRCVBUFRSIZE** 設定を変更できない場合は、'IBM サポートにお問い合わせください。

z/OS Planning your queue sharing group (QSG)

The easiest way to implement a shared queuing environment, is to configure a queue manager, add that queue manager to a QSG, then add other queue managers to the QSG.

A queue sharing group uses Db2 tables to store configuration information. There is one set of tables used by all QSGs that share the same Db2 data sharing group.

Shared queue messages are stored in a structure in a coupling facility (CF). Each QSG has its own set of CF structures. You need to configure the structures to meet your needs.

Messages over 63KB in size cannot be stored in the CF. You need to use either Shared Message Data Sets (SMDS) or Db2 for these messages.

Message profiles and capacity planning

You should understand the message profile of your shared queue messages. The following are examples of factors that you need to consider:

- Average, and maximum message size
- The typical queue depth, and exception queue depth. For example, you might need to have enough capacity to hold messages for a whole day, and the typical queue depth is under 100 messages.

If the message profile changes, you can increase the size of the structures, or implement SMDS, at a later date.

If you want to be able to handle a large peak volume of messages, you can configure IBM MQ to offload messages to SMDS when the usage of the structure reaches user specified thresholds.

You need to decide if you want to duplex the CF structures. This is controlled by the CF structure definition in the CFRM policy:

1. A duplexed structure uses two coupling facilities. If there is a problem with one CF, there is no interruption to the service, and the structure can be rebuilt on a third CF, if one is available. Duplexed structures can significantly impact the performance of operations on shared queues.
2. If the structure is not duplexed, then a problem with the CF means that shared queues on structures in that CF will become unavailable until the structure can be rebuilt in another CF.

   IBM MQ can be configured to automatically rebuild structures in another CF in this case. Persistent messages will be recovered from the logs of the queue managers.

Note that it is easy to change the CF definitions.

You can define a structure so that it can hold nonpersistent messages only, or so that it can hold persistent and nonpersistent messages.

Structures that can hold persistent messages need to be backed up periodically. Back up your CF structures at least every hour to minimize the time needed to recover the structure in the event of a failure. The backup is stored in the log data set of the queue manager performing the backup.

If you are expecting to have a high throughput of messages on your shared queues, it is best practice to have a dedicated queue manager for backing up the CF structures. This reduces the time needed to recover the structures, as a less data needs to be read from queue manager logs.

Channels

To provide a single system image for applications connecting into an IBM MQ QSG, you can define shared input channels. If these are set up, then a connection coming into the queue sharing group environment, can go to any queue manager in the QSG.

You might need to set up a network router, or Virtual IP address (VIPA) for these channels.

You can define shared output channels. A shared output channel instance can be started from any queue manager in the QSG.

See Shared channels for more information.

Security

You protect IBM MQ resources using an external security manager. If you are using RACF®, the RACF profiles are prefixed with the queue manager name. For example, a queue named APPLICATION.INPUT would be protected using a profile in the MQQUEUE class named `qmgrName.APPLICATION.INPUT`.

When using a queue sharing group you can continue to protect resources with profiles prefixed with the queue manager name, or you can prefix profiles with the queue sharing group name. For example `qsgName.APPLICATION.INPUT`.

You should aim to use profiles prefix with the queue sharing group name because this means there is a single definition for all queue managers, saving you work, and preventing a mismatch in definitions between queue managers.

Related concepts

“Planning for your queue manager” on page 147
When you are setting up a queue manager, your planning should allow for the queue manager to grow, so that the queue manager meets the needs of your enterprise.

z/OS カップリング・ファシリティーおよびオフロード・ストレージ環境の準備

このトピックは、カップリング・ファシリティー (CF) 構造の初期サイズと形式、および共有メッセージ・データ・セット (SMDS) 環境または Db2 環境を計画する場合に使用します。

このセクションでは、以下のトピックに関する情報を取り上げます。

- 182 ページの『カップリング・ファシリティー・リソースの定義』
  - オフロード・ストレージ機構の決定
  - 構造の計画
  - 構造のサイズの計画
  - 共有キューと構造の対応付け
- 188 ページの『Planning your shared message data set (SMDS) environment』
- 191 ページの『Planning your Db2 environment』

カップリング・ファシリティー・リソースの定義

共有キューを使用する場合は、IBM MQ が CFRM ポリシーで使用するカップリング・ファシリティー構造を定義する必要があります。 そのためには、まずそれらの構造に関する情報で CFRM ポリシーを更新してから、そのポリシーをアクティブ化します。

通常、インストール環境には、使用可能なカップリング・ファシリティーが記述されている既存の CFRM ポリシーがあります。 管理データ・ユーティリティーを使用すれば、指定したテキスト・ステートメントに基づいてそのポリシーの内容を変更できます。 新しい構造の名前、それらの構造の定義先のカップリング・ファシリティー、それらの構造のサイズを定義したステートメントをそのポリシーに追加する必要があります。

また、CFRM ポリシーは、IBM MQ 構造が二重構造になるかどうか、および障害の発生時にそれらがどのように再び割り振られるかを決定します。 『共有キューの回復』 には、カップリング・ファシリティーに影響する障害に対する回復力のために CFRM を構成するうえでの推奨事項が記載されています。

オフロード・ストレージ環境の決定

共用キューのメッセージ・データは、カップリング・ファシリティーからオフロードして、 Db2 テーブルまたは 共用メッセージ・データ・セット (SMDS) と呼ばれる IBM MQ 管理対象データ・セットのいずれかに保管することができます。 大きすぎて (つまり、63 KB を超える) カップリング・ファシリティーに保管できないメッセージは、常にオフロードする必要があり、それよりも小さいメッセージは、必要であればオフロードして、カップリング・ファシリティーのスペース使用量を削減することができます。

詳しくは、『共用メッセージのオフロード・オプションの指定』を参照してください。

構造の計画

キュー共有グループ (QSG) の場合は、最低 2 つの構造を定義する必要があります。 最初の構造のことを管理構造といいます。キュー共有グループで IBM MQ の内部アクティビティーを調整するための構造です。この構造には、ユーザー・データは入りません。 *qsg-name*CSQ_ADMIN という固定された名前があります (*qsg-name* は、キュー共有グループの名前です)。 後続の構造はアプリケーション構造と呼ばれ、 IBM MQ 共用キューにメッセージを保持するために使用されます。 それぞれの構造には、最大 512 個の共有キューが入ります。

システム・キューでは、*qsg-name*CSQSYSAPPL という名前のアプリケーション構造を使用します。 この構造を定義するかどうかは任意ですが、特定の機能を使用する場合は必須になります。 デフォルトでは、SYSTEM.QSG.CHANNEL.SYNCQ キューと SYSTEM.QSG.UR.RESOLUTION.QUEUE キューが *qsg-name*CSQSYAPPL 構造に定義されます。

複数の構造の使用

キュー共有グループは、最大で 64 個のカップリング・ファシリティー構造に接続できます。 それらの構造体の 1 つは管理構造体である必要があります。 それを定義する場合は、別の構造を *qsg-name*CSQSYSAPPL 構造にできます。 メッセージ・データ用に最大 63 個 (*qsg-name*CSQSYSAPPL が定義されている場合は 62 個) の構造体を使用できます。 複数のアプリケーション構造を使用する理由としては、以下のような理由が考えられます。

- 多数のメッセージが入る可能性が高いキューがいくつかあるので、カップリング・ファシリティー全体のすべてのリソースが必要になる場合。
- 1 つの構造には 512 個のキューしか入りませんが、多数の共有キューを使用する必要があるので、それらのキューを複数の構造に分割しなければならない場合。
- 構造の使用状況の特性に関する RMF レポートで、構造に含まれているキューをいくつかのカップリング・ファシリティーに分散させるのが望ましいことが明らかになった場合。
- データを分離するために、一部のキュー・データを他のキュー・データとは別の物理的カップリング・ファシリティーに格納する場合。
- 構造レベルの属性とコマンド (BACKUP CFSTRUCT など) を使用して持続共有メッセージのリカバリーを実行する場合。 バックアップとリカバリーを簡略化するために、非持続メッセージを格納するキューを持続メッセージを格納する構造とは別の構造に割り当てることがあります。

構造をどのカップリング・ファシリティーに割り振るかを選択するときに考慮に入れるポイントを以下にまとめます。

- データ分離の要件。
- カップリング・ファシリティーの揮発性 (つまり、電源異常が発生してもデータを維持できるかどうか)。
- アクセス元のシステムとカップリング・ファシリティーの間、またはカップリング・ファシリティー同士の間の障害の独立性。
- カップリング・ファシリティーにインストールされているカップリング・ファシリティー制御コード (CFCC) のレベル (IBM MQ ではレベル 9 以上が必要です)。

構造のサイズの計画

管理構造

管理構造 (*qsg-name*CSQ_ADMIN) は、キュー共有グループ内のキュー・マネージャー 1 つにつき 1000 個のリスト項目を格納できる大きさにする必要があります。 キュー・マネージャーの始動時に、その構造が検査され、キュー共有グループに現時点で定義 されているキュー・マネージャーの数に対応するだけの大きさになっているかどうかが確認されます。 キュー・マネージャーがキュー共有グループに対して定義されていると見なされるのは、キュー・マネージャーが CSQ5PQSG ユーティリティーで追加された場合です。 MQSC DISPLAY GROUP コマンドを使用すれば、グループに対して定義されているキュー・マネージャーを確認できます。

注 : 構造体のサイズを計算する際には、キュー共有グループ内のキュー・マネージャーの数に加えて、大きな作業単位のサイズを考慮する必要があります。

キュー共有グループで定義されているキュー・マネージャーの数に応じて管理構造で必要になる最小サイズを 184 ページの表 22 にまとめます。 ここで示すサイズは、CFCC レベル 14 のカップリング・ファシリティー構造を対象にしています。さらに高いレベルの CFCC の場合は、通常、もっと大きなサイズが必要になります。

表 *22.* 管理構造の最小サイズ

キュー共有グループで定義されているキュー・マネージャーの数	必要なストレージ
1	6144 KB
2	6912 KB
3	7976 KB
4	8704 KB
5	9728 KB
6	10496 KB
7	11520 KB
8	12288 KB
9	13056 KB
10	14080 KB
11	14848 KB
12	15616 KB
13	16640 KB
14	17408 KB
15	18176 KB
16	19200 KB

表 22. 管理構造の最小サイズ (続き)	
キュー共有グループで定義されているキュー・マネージャーの数	**必要なストレージ**
17	19968 KB
18	20736 KB
19	21760 KB
20	22528 KB
21	23296 KB
22	24320 KB
23	25088 KB
24	25856 KB
25	27136 KB
26	27904 KB
27	28672 KB
28	29696 KB
29	30464 KB
30	31232 KB
31	32256 KB

既存のキュー共有グループにキュー・マネージャーを追加するときに、ストレージ要件が 184 ページの表 22 の推奨サイズより大きくなっている場合があります。 その場合は、以下の手順で *qsg-name*CSQ_ADMIN 構造で必要になるストレージを見積もってください。

1. キュー共有グループの既存のメンバーで MQSC コマンド **DISPLAY CFSTATUS(CSQ_ADMIN)** を実行します。
2. CSQ_ADMIN 構造の ENTSMAX 情報を抽出します。
3. この数値が、キュー共有グループで定義したいと思っているキュー・マネージャーの総数の 1000 倍より小さければ、構造のサイズを大きくしてください。

アプリケーション構造

IBM MQ メッセージを格納するために必要なアプリケーション構造のサイズは、構造に同時に格納される可能性があるメッセージの数とサイズによって左右されます。

共有キューに入るメッセージを格納するための CF 構造で必要になるサイズを 186 ページの図 42 のグラフにまとめます。 割り振りサイズを計算するには、以下の情報が必要です。

- キューに入るメッセージの平均サイズ。
- 構造に格納される可能性があるメッセージの総数。

水平軸でメッセージの数を探してください。 メッセージ・サイズに対応するカーブを選択し、垂直軸で必要な値を確認します。 例えば、長さが 1 KB のメッセージが 200 000 件であれば、256 から 512 MB の範囲の値になります。

この同じ情報を表形式でまとめたのが 186 ページの表 23 です。

図 *42.* カップリング・ファシリティー構造のサイズの計算

カップリング・ファシリティー構造のサイズの計算値をまとめた表を以下に示します。

表 *23.* カップリング・ファシリティー構造のサイズの計算

メッセージの数	1 KB	2 KB	4 KB	8 KB	16 KB	32 KB	63 KB
100	6 MB	6 MB	7 MB	7 MB	8 MB	10 MB	14 MB
1000	8 MB	9 MB	12 MB	17 MB	27 MB	48 MB	88 MB
10000	25 MB	38 MB	64 MB	115 MB	218 MB	423 MB	821 MB
100000	199 MB	327 MB	584 MB	1097 MB	2124 MB	4177 MB	8156 MB

CFRM ポリシーに以下のステートメントを組み込みます。

- INITSIZE は、最初のキュー・マネージャーが構造に接続するときにその構造に割り振られるサイズ (KB) です。
- SIZE は、その構造の最大サイズです。
- FULLTHRESHOLD では、構造が満杯になりつつあることを示すメッセージ IXC585E が z/OS によって生成されるしきい値 (パーセント値) を設定します。

ベスト・プラクティスは、INITSIZE と SIZE が 2 倍以内であるようにすることです。 例えば、先ほど取り上げた数値の場合は、以下のようなステートメントを追加できます。

```
STRUCTURE NAME(structure-name)
INITSIZE(value from graph in KB, that is, multiplied by 1024)
SIZE(something larger)
FULLTHRESHOLD(85)
```

```
STRUCTURE NAME(QSG1APPLICATION1)
INITSIZE(262144) /* 256 MB */
SIZE(524288)     /* 512 MB */
FULLTHRESHOLD(85)
```

構造の使用率が警告メッセージ生成のしきい値に達した場合は、何らかの操作が必要になります。 IBM MQ を使用して、構造内のいくつかのキューに対する MQPUT 操作を禁止することによってアプリケーションがそれ以上メッセージを書き込まないようにしたり、それらのキューからメッセージを取り出すアプリケーションをさらに始動したり、キューにメッセージを書き込んでいるアプリケーションのうちのいくつかを静止したりすることができます。

あるいは、z/OS 機能を使用して、構造のサイズを適切に変更することも可能です。 例えば、以下の z/OS コマンドを使用できます。

```
SETXCF START,ALTER,STRNAME=structure-name,SIZE=newsize
```

これは、構造のサイズを *newsize* に変更するコマンドです。*newsize* は、CFRM ポリシーで指定されている構造の SIZE の値より小さく、かつカップリング・ファシリティーの現行サイズより大きい値です。

MQSC DISPLAY CFSTATUS コマンドを使用すれば、カップリング・ファシリティー構造の使用状況をモニターできます。

何の操作も実行しないで、キュー構造が満杯になると、MQRC_STORAGE_MEDIUM_FULL 戻りコードがアプリケーションに返されます。 管理構造が満杯になった場合の具体的な症状は、どのプロセスがエラーになったのかによって異なりますが、通常、以下のような問題が発生する可能性があります。

- コマンドへの応答がなくなります。
- コミット処理時の問題が原因でキュー・マネージャーの障害が発生します。

CSQSYSAPPL 構造

*qsg-name*CSQSYSAPPL 構造は、システム・キューのためのアプリケーション構造です。 *qsg-name*CSQSYSAPPL 構造で定義されるデフォルト・キューのメッセージ・データ・サイズを見積もる方法を表 3 にまとめます。

表 *24.* サイズを見積もるための *CSQSYSAPPL* の使用法を示す表。

***qsg-name*CSQSYSAPPL の使用**	**サイズ見積もり**
SYSTEM.QSG.CHANNEL.SYNCQ	共有チャネルのアクティブ・インスタンス 1 つにつき 500 バイトのメッセージ 2 件
SYSTEM.QSG.UR.RESOLUTION.QUEUE	2 KB のメッセージ 1000 件

構造定義の推奨初期値は、以下のとおりです。

```
STRUCTURE NAME(qsg-nameCSQSYSAPPL)
INITSIZE(20480)        /* 20 MB */
SIZE(30720)            /* 30 MB */
FULLTHRESHOLD(85)
```

これらの値は、共有チャネルとグループ・リカバリー単位の使用状況に応じて調整できます。

共有キューと構造の対応付け

IBM MQ に対してアプリケーション構造を定義するには、DEFINE CFSTRUCT コマンドを使用します。 IBM MQ に対して構造を定義する時には、構造名に QSG 名の接頭部を含めないでください。 例えば、IBM MQ に対して *qsg-name*APPLICATION1 という名前のアプリケーション構造を CFRM ポリシーで定義するには、以下のコマンドを実行します。

```
DEFINE CFSTRUCT(APPLICATION1)
```

キュー定義の CFSTRUCT 属性を使用して、キューを構造に対応付けることができます。 CF 構造の名前を QSG 名前接頭部なしで属性に指定します。 例えば、次のコマンドを使用すると、APPLICATION1 構造で共有キューが定義されます。

```
DEFINE QLOCAL(myqueue) QSGDISP(SHARED) CFSTRUCT(APPLICATION1)
```

z/OS *Planning your shared message data set (SMDS) environment*

If you are using queue sharing groups with SMDS offloading, IBM MQ needs to connect to a group of shared message data sets. Use this topic to help understand the data set requirements, and configuration required to store IBM MQ message data.

A *shared message data set* (described by the keyword SMDS) is a data set used by a queue manager to store offloaded message data for shared messages stored in a coupling facility structure.

Note: When defining SMDS data sets for a structure, you must have one for each queue manager.

When this form of data offloading is enabled, the **CFSTRUCT** requires an associated group of shared message data sets, one data set for each queue manager in the queue sharing group. The group of shared message data sets is defined to IBM MQ using the **DSGROUP** parameter on the **CFSTRUCT** definition. Additional parameters can be used to supply further optional information, such as the number of buffers to use and expansion attributes for the data sets.

Each queue manager can write to the data set which it owns, to store shared message data for messages written through that queue manager, and can read all of the data sets in the group.

A list describing the status and attributes for each data set associated with the structure is maintained internally as part of the **CFSTRUCT** definition, so each queue manager can check the definition to find out which data sets are currently available.

This data set information can be displayed using the **DISPLAY CFSTATUS TYPE(SMDS)** command to display current status and availability, and the **DISPLAY SMDS** command to display the parameter settings for the data sets associated with a specified **CFSTRUCT**.

Individual shared message data sets are effectively identified by the combination of the owning queue manager name (usually specified using the **SMDS** keyword) and the **CFSTRUCT** structure name.

This section describes the following topics:

- The DSGROUP parameter
- The DSBLOCK parameter
- Shared message data set characteristics
- Shared message data set space management
- Access to shared message data sets
- Creating a shared message data set
- Shared message data set performance and capacity considerations
- Activating a shared message data set

See DEFINE CFSTRUCT for details of these parameters.

For information on managing your shared message data sets, see Managing shared message data sets for further details.

The DSGROUP parameter

The **DSGROUP** parameter on the **CFSTRUCT** definition identifies the group of data sets in which large messages for that structure are to be stored. Additional parameters may be used to specify the logical block size to be used for space allocation purposes and values for the buffer pool size and automatic data set expansion options.

The **DSGROUP** parameter must be set up before offloading to data sets can be enabled.

- If a new **CFSTRUCT** is being defined at **CFLEVEL(5)** and the option **OFFLOAD(SMDS)** is specified or assumed, then the **DSGROUP** parameter must be specified on the same command.
- If an existing **CFSTRUCT** is being altered to increase the **CFLEVEL** to **CFLEVEL(5)** and the option **OFFLOAD(SMDS)** is specified or assumed, then the **DSGROUP** parameter must be specified on the same command if it is not already set.

The DSBLOCK parameter

Space within each data set is allocated to queues as logical blocks of a fixed size (usually 256 KB) specified using the **DSBLOCK** parameter on the **CFSTRUCT** definition, then allocated to individual messages as ranges of pages of 4 KB (corresponding to the physical block size and control interval size) within each logical block. The logical block size also determines the maximum amount of message data that can be read or written in a single I/O operation, which is the same as the buffer size for the SMDS buffer pool.

A larger value of the **DSBLOCK** parameter can improve performance for very large messages by reducing the number of separate I/O operations. However, a smaller value decreases the amount of buffer storage required for each active request. The default value for the **DSBLOCK** parameter is 256 KB, which provides a reasonable balance between these requirements, so specifying this parameter might not normally be necessary.

Shared message data set characteristics

A shared message data set is defined as a VSAM linear data set (LDS). Each offloaded message is stored in one or more blocks in the data set. The stored data is addressed directly by information in the coupling facility entries, like an extended form of virtual storage. There is no separate index or similar control information stored in the data set itself.

The direct addressing scheme means that for messages which fit into one block, only a single I/O operation is needed to read or write the block. When a message spans more than one block, the I/O operations for each block can be fully overlapped to minimize elapsed time, provided that sufficient buffers are available.

The shared message data set also contains a small amount of general control information, consisting of a header in the first page, which includes recovery and restart status information, and a space map checkpoint area which is used to save the free block space map at queue manager normal termination.

Shared message data set space management

As background information for capacity, performance and operational considerations, it might be useful to understand the concepts of how space in shared message data sets is managed by the queue managers.

Free space in each shared message data set is tracked by its owning queue manager using a space map which indicates the number of pages in use within each logical block. The space map is maintained in main storage while the data set is open and saved in the data set when it is closed normally. (In recovery situations the space map is automatically rebuilt by scanning the messages in the coupling facility structure to find out which data set pages are currently in use).

When a shared message with offloaded message data is being written, the queue manager allocates a range of pages for each message block. If there is a partly used current logical block for the specified queue, the queue manager allocates space starting at the next free page in that block, otherwise it

allocates a new logical block. If the whole message does not fit within the current logical block, the queue manager splits the message data at the end of the logical block and allocates a new logical block for the next message block. This is repeated until space has been allocated for the whole message. Any unused space in the last logical block is saved as the new current logical block for the queue. When the data set is closed normally, any unused pages in current logical blocks are returned to the space map before it is saved.

When a shared message with offloaded message data has been read and is ready to be deleted, the queue manager processes the delete request by transferring the coupling facility entry for the message to a clean-up list monitored by the owning queue manager (which may be the same queue manager). When entries arrive on this list, the owning queue manager reads and deletes the entries and returns the freed ranges of pages to the space map. When all used pages in a logical block have been freed the block becomes available for reuse.

Access to shared message data sets

Each shared message data set must be on shared direct access storage which is accessible to all queue managers in the queue sharing group.

During normal running, each queue manager opens its own shared message data set for read/write access, and opens any active shared message data sets for other queue managers for read-only access, so it can read messages stored by those queue managers. This means that each queue manager userid requires at least UPDATE access to its own shared message data set and READ access to all other shared message data sets for the structure.

If it is necessary to recover shared message data sets using **RECOVER CFSTRUCT**, the recovery process can be executed from any queue manager in the queue sharing group. A queue manager which may be used to perform recovery processing requires UPDATE access to all data sets that it may need to recover

Creating a shared message data set

Each shared message data set should normally be created before the corresponding **CFSTRUCT** definition is created or altered to enable the use of this form of message offloading, as the **CFSTRUCT** definition changes will normally take effect immediately, and the data set will be required as soon as a queue manager attempts to access a shared queue which has been assigned to that structure. A sample job to allocate and pre-format a shared message data set is provided in SCSQPROC(CSQ4SMDS). The job must be customized and run to allocate a shared message data set for each queue manager which uses a CFSTRUCT with OFFLOAD(SMDS).

If the queue manager finds that offload support has been enabled and tries to open its shared message data set but it has not yet been created, the shared message data set will be flagged as unavailable. The queue manager will then be unable to store any large messages until the data set has been created and the queue manager has been notified to try again, for example using the **START SMDSCONN** command.

A shared message data set is created as a VSAM linear data set using an Access Method Services **DEFINE CLUSTER** command. The definition must specify **SHAREOPTIONS(2 3)** to allow one queue manager to open it for write access and any number of queue managers to read it at the same time. The default control interval size of 4 KB must be used. If the data set may need to expand beyond 4 GB, it must be defined using an SMS data class which has the VSAM extended addressability attribute. A shared message data set is eligible to reside in the extended addressing space (EAS) part of an extended address volumes (EAV).

Each shared message data set can either be empty or pre-formatted to binary zeros (using **CSQJUFMT** or a similar utility such as the sample job SCSQPROC(CSQ4SMDS)), before its initial use. If it is empty or only partly formatted when it is opened, the queue manager automatically formats the remaining space to binary zeros.

Shared message data set performance and capacity considerations

Each shared message data set is used to store offloaded data for shared messages written to the associated **CFSTRUCT** by the owning queue manager, from regions within the same system. Each message that is offloaded takes up to 768 bytes of CF storage, made up of 256 bytes for the entry and 512 bytes for the two elements of header and descriptor. Each offloaded message is stored in one or more pages (physical blocks of size 4 KB) in the data set.

The data set space required for a given number of offloaded messages can therefore be estimated by rounding up the overall message size (including the descriptor) to the next multiple of 4 KB and then multiplying by the number of messages.

As for a page set, when a shared message data set is almost full, it can optionally be expanded automatically. The default behavior for this automatic expansion can be set using the **DSEXPAND** parameter on the **CFSTRUCT** definition. This setting can be overridden for each queue manager using the **DSEXPAND** parameter on the **ALTER SMDS** command. Automatic expansion is triggered when the data set reaches 90% full and more space is required. If expansion is allowed but an expansion attempt is rejected by VSAM because no secondary space allocation was specified when the data set was defined, expansion is retried using a secondary allocation of 20% of the current size of the data set.

Provided that the shared message data set is defined with the extended addressability attribute, the maximum size is only limited by VSAM considerations to a maximum of 16 TB or 59 volumes. This is significantly larger than the 64 GB maximum size of a local page set.

Activating a shared message data set

When a queue manager has successfully connected to an application coupling facility structure, it checks whether that structure definition specifies offloading using an associated **DSGROUP** parameter. If so, the queue manager allocates and opens its own shared message data set for write access, then it opens for read access any existing shared message data sets owned by other queue managers.

When a shared message data set is opened for the first time (before it has been recorded as active within the queue sharing group), the first page will not yet contain a valid header. The queue manager fills in header information to identify the queue sharing group, the structure name and the owning queue manager.

After the header has been completed, the queue manager registers the new shared message data set as active and broadcasts an event to notify any other active queue managers about the new data set.

Every time a queue manager opens a shared message data set it validates the header information to ensure that the correct data set is still being used and that it has not been damaged.

z/OS *Planning your Db2 environment*

If you are using queue sharing groups, IBM MQ needs to attach to a Db2 subsystem that is a member of a data sharing group. Use this topic to help understand the Db2 requirements used to hold IBM MQ data.

IBM MQ needs to know the name of the data sharing group that it is to connect to, and the name of a Db2 subsystem (or Db2 group) to connect to, to reach this data sharing group. These names are specified in the QSGDATA parameter of the CSQ6SYSP system parameter macro (described in Using CSQ6SYSP).

Within the data sharing group, shared Db2 tables are used to hold:

- Configuration information for the queue sharing group.
- Properties of IBM MQ shared and group objects.
- Optionally, data relating to offloaded IBM MQ messages.

IBM MQ provides a single set of sample jobs for defining the necessary Db2 table spaces, tables, and indexes. These jobs make use of Universal Table Spaces (UTS). Earlier versions of the product had two sets of jobs, one for UTS, and one for older types of table space, which have been deprecated by the most recent versions of Db2.

IBM MQ can still be used with older types of table space, and this might be appropriate if you already have an existing queue sharing group. However, if you are creating a new queue sharing group, it should use UTS.

Db2 V12 Function level 508 provides a non disruptive migration process for migrating multi-table table spaces to universal table spaces. You can use this approach to migrate the multi-table table spaces, used by existing queue sharing groups, to universal table spaces without taking an outage of the whole queue sharing group.

In Db2 V13, use the MOVE TABLE option of the ALTER TABLESPACE statement. See Moving tables from multi-table table spaces to partition-by-growth table spaces for more information.

By default Db2 uses the user ID of the person running the jobs as the owner of the Db2 resources. If this user ID is deleted then the resources associated with it are deleted, and so the table is deleted. Consider using a group ID to own the tables, rather than an individual user ID. You can do this by adding `GROUP=groupname` onto the JOB card, and specifying `SET CURRENT SQLID='groupname'` before any SQL statements.

IBM MQ uses the RRS Attach facility of Db2. This means that you can specify the name of a Db2 group that you want to connect to. The advantage of connecting to a Db2 group attach name (rather than a specific Db2 subsystem), is that IBM MQ can connect (or reconnect) to any available Db2 subsystem on the z/OS image that is a member of that group. There must be a Db2 subsystem that is a member of the data sharing group active on each z/OS image where you are going to run a queue-sharing IBM MQ subsystem, and RRS must be active.

Db2 storage

For most installations, the amount of Db2 storage required is about 20 or 30 cylinders on a 3390 device. However, if you want to calculate your storage requirement, the following table gives some information to help you determine how much storage Db2 requires for the IBM MQ data. The table describes the length of each Db2 row, and when each row is added to or deleted from the relevant Db2 table. Use this information together with the information about calculating the space requirements for the Db2 tables and their indexes in the *Db2 for z/OS Installation Guide*.

Table 25. Planning your Db2 storage requirements

Db2 table name	Length of row	A row is added when:	A row is deleted when:
CSQ.ADMIN_B_QSG	252 bytes	A queue sharing group is added to the table with the ADD QSG function of the CSQ5PQSG utility.	A queue sharing group is removed from the table with the REMOVE QSG function of the CSQ5PQSG utility. (All rows relating to this queue sharing group are deleted automatically from all the other Db2 tables when the queue sharing group record is deleted.)
CSQ.ADMIN_B_QMGR	Up to 3828 bytes	A queue manager is added to the table with the ADD QMGR function of the CSQ5PQSG utility.	A queue manager is removed from the table with the REMOVE QMGR function of the CSQ5PQSG utility.
CSQ.ADMIN_B_STRUCTURE	1454 bytes	The first local queue definition, specifying the QSGDISP(SHARED) attribute, that names a previously unknown structure within the queue sharing group is defined.	The last local queue definition, specifying the QSGDISP(SHARED) attribute, that names a structure within the queue sharing group is deleted.

Table 25. Planning your Db2 storage requirements (continued)

Db2 table name	Length of row	A row is added when:	A row is deleted when:
CSQ.ADMIN_B_SCST	342 bytes	A shared channel is started.	A shared channel becomes inactive.
CSQ.ADMIN_B_SSKT	254 bytes	A shared channel that has the NPMSPEED(NORMAL) attribute is started.	A shared channel that has the NPMSPEED(NORMAL) attribute becomes inactive.
CSQ.ADMIN_B_STRBACKUP	514 bytes	A new row is added to the CSQ.ADMIN_B_STRUCTURE table. Each entry is a dummy entry until the BACKUP CFSTRUCT command is run, which overwrites the dummy entries.	A row is deleted from the CSQ.ADMIN_B_STRUCTURE table.
CSQ.OBJ_B_AUTHINFO	3400 bytes	An authentication information object with QSGDISP(GROUP) is defined.	An authentication information object with QSGDISP(GROUP) is deleted.
CSQ.OBJ_B_QUEUE	Up to 3707 bytes	• A queue with the QSGDISP(GROUP) attribute is defined. • A queue with the QSGDISP(SHARED) attribute is defined. • A model queue with the DEFTYPE(SHAREDYN) attribute is opened.	• A queue with the QSGDISP(GROUP) attribute is deleted. • A queue with the QSGDISP(SHARED) attribute is deleted. • A dynamic queue with the DEFTYPE(SHAREDYN) attribute is closed with the DELETE option.
CSQ.OBJ_B_NAMELIST	Up to 15127 bytes	A namelist with the QSGDISP(GROUP) attribute is defined.	A namelist with the QSGDISP(GROUP) attribute is deleted.
CSQ.OBJ_B_CHANNEL	Up to 14127 bytes	A channel with the QSGDISP(GROUP) attribute is defined.	A channel with the QSGDISP(GROUP) attribute is deleted.
CSQ.OBJ_B_STGCLASS	Up to 2865 bytes	A storage class with the QSGDISP(GROUP) attribute is defined.	A storage class with the QSGDISP(GROUP) attribute class is deleted.
CSQ.OBJ_B_PROCESS	Up to 3347 bytes	A process with the QSGDISP(GROUP) attribute is defined.	A process with the QSGDISP(GROUP) attribute is deleted.
CSQ.OBJ_B_TOPIC	Up to 14520 bytes	A topic object with QSGDISP(GROUP) attribute is defined.	A topic object with QSGDISP(GROUP) attribute is deleted.
CSQ.EXTEND_B_QMGR	Less than 430 bytes	A queue manager is added to the table with the ADD QMGR function of the CSQ5PQSG utility.	A queue manager is removed from the table with the REMOVE QMGR function of the CSQ5PQSG utility.
CSQ.ADMIN_B_MESSAGES	87 bytes	For large message PUT (1 per BLOB).	For large message GET (1 per BLOB).

Table 25. Planning your Db2 storage requirements (continued)			
Db2 table name	**Length of row**	**A row is added when:**	**A row is deleted when:**
CSQ.ADMIN_MSGS_BAUX1 CSQ.ADMIN_MSGS_BAUX2 CSQ.ADMIN_MSGS_BAUX3 CSQ.ADMIN_MSGS_BAUX4		These 4 tables contain message payload for large messages added into one of these 4 tables for each BLOB of the message. BLOBS are up to 511 KB in length, so if the message size is > 711 KB, there will be multiple BLOBs for this message.	

The use of large numbers of shared queue messages of size greater than 63 KB can have significant performance implications on your IBM MQ system. For more information, see SupportPac MP16, Capacity Planning and Tuning for IBM MQ for z/OS, at: SupportPacs for IBM MQ and other project areas.

z/OS Planning for backup and recovery

Developing backup and recovery procedures at your site is vital to avoid costly and time-consuming losses of data. IBM MQ provides means for recovering both queues and messages to their current state after a system failure.

This topic contains the following sections:

- “Recovery procedures” on page 194
- “Tips for backup and recovery” on page 195
- “Recovering page sets” on page 197
- “Recovering CF structures” on page 198
- “リカバリーに関する具体的な目標の達成” on page 198
- “Backup considerations for other products” on page 200
- “Recovery and CICS” on page 200
- “Recovery and IMS” on page 200
- “Preparing for recovery on an alternative site” on page 201
- “Example of queue manager backup activity” on page 201

Recovery procedures

Develop the following procedures for IBM MQ:

- Creating a point of recovery.
- Backing up page sets.
- Backing up CF structures.
- Recovering page sets.
- Recovering from out-of-space conditions (IBM MQ logs and page sets).
- Recovering CF structures.

See IBM MQ for z/OS 管理 for information about these.

Become familiar with the procedures used at your site for the following:

- Recovering from a hardware or power failure.
- Recovering from a z/OS component failure.

- Recovering from a site interruption, using off-site recovery.

z/OS Tips for backup and recovery

Use this topic to understand some backup and recovery tasks.

The queue manager restart process recovers your data to a consistent state by applying log information to the page sets. If your page sets are damaged or unavailable, you can resolve the problem using your backup copies of your page sets (if all the logs are available). If your log data sets are damaged or unavailable, it might not be possible to recover completely.

Consider the following points:

- Periodically take backup copies
- Do not discard archive logs you might need
- Do not change the DDname to page set association

Periodically take backup copies

A *point of recovery* is the term used to describe a set of backup copies of IBM MQ page sets and the corresponding log data sets required to recover these page sets. These backup copies provide a potential restart point in the event of page set loss (for example, page set I/O error). If you restart the queue manager using these backup copies, the data in IBM MQ is consistent up to the point that these copies were taken. Provided that all logs are available from this point, IBM MQ can be recovered to the point of failure.

The more recent your backup copies, the quicker IBM MQ can recover the data in the page sets. The recovery of the page sets is dependent on all the necessary log data sets being available.

In planning for recovery, you need to determine how often to take backup copies and how many complete backup cycles to keep. These values tell you how long you must keep your log data sets and backup copies of page sets for IBM MQ recovery.

When deciding how often to take backup copies, consider the time needed to recover a page set. The time needed is determined by the following:

- The amount of log to traverse.
- The time it takes an operator to mount and remove archive tape volumes.
- The time it takes to read the part of the log needed for recovery.
- The time needed to reprocess changed pages.
- The storage medium used for the backup copies.
- The method used to make and restore backup copies.

In general, the more frequently you make backup copies, the less time recovery takes, but the more time is spent making copies.

For each queue manager, you should take backup copies of the following:

- The archive log data sets
- The BSDS copies created at the time of the archive
- The page sets
- Your object definitions
- Your CF structures

To reduce the risk of your backup copies being lost or damaged, consider:

- Storing the backup copies on different storage volumes to the original copies.
- Storing the backup copies at a different site to the original copies.

- Making at least two copies of each backup of your page sets and, if you are using single logging or a single BSDS, two copies of your archive logs and BSDS. If you are using dual logging or BSDS, make a single copy of both archive logs or BSDS.

Before moving IBM MQ to a production environment, fully test and document your backup procedures.

Backing up your page sets

You need to back up page sets regularly. Some enterprises back up the page sets twice a day.

You need the active and archive logs since a backup to be able to recover using the backup. You need enough log data to go back four checkpoints if the backup was taken when the queue manager was running.

You can use ADRDSSU FastReplication to back up page sets, and you can do this while the queue manager is active. Note that you need to ensure there is enough space in the storage pool.

Backing up your object definitions

Create backup copies of your object definitions. To do this, use the MAKEDEF feature of the COMMAND function of the utility program (described in Using the COMMAND function of CSQUTIL).

You should do this whenever you take backup copies of your queue manager data sets, and keep the most current version.

Backing up your coupling facility structures

If you have set up any queue sharing groups, even if you are not using them, you must take periodic backups of your CF structures. To do this, use the IBM MQ BACKUP CFSTRUCT command. You can use this command only on CF structures that are defined with the RECOVER(YES) attribute. If any CF entries for persistent shared messages refer to offloaded message data stored in a shared message data set (SMDS) or Db2, the offloaded data is retrieved and backed up together with the CF entries. Shared message data sets should not be backed up separately.

It is recommended that you take a backup of all your CF structures about every hour, to minimize the time it takes to restore a CF structure.

You could perform all your CF structure backups on a single queue manager, which has the advantage of limiting the increase in log use to a single queue manager. Alternatively, you could perform backups on all the queue managers in the queue sharing group, which has the advantage of spreading the workload across the queue sharing group. Whichever strategy you use, IBM MQ can locate the backup and perform a RECOVER CFSTRUCT from any queue manager in the queue sharing group. The logs of all the queue managers in the queue sharing group need to be accessed to recover the CF structure.

Backing up your message security policies

If you are using Advanced Message Security to create a backup of your message security policies, create a backup using the message security policy utility (CSQ0UTIL) to run **dspmqspl** with the -export parameter, then save the policy definitions that are output to the EXPORT DD.

You should create a backup of your message security policies whenever you take backup copies of your queue manager data sets, and keep the most current version.

Do not discard archive logs you might need

IBM MQ might need to use archive logs during restart. You must keep sufficient archive logs so that the system can be fully restored. IBM MQ might use an archive log to recover a page set from a restored backup copy. If you have discarded that archive log, IBM MQ cannot restore the page set to its current state. When and how you discard archive logs is described in Discarding archive log data sets.

You can use the `/cpf DIS USAGE TYPE(ALL)` command to display the log RBA, and log range sequence number (LRSN) that you need to recover your queue manager's page sets and the queue sharing group's structures. You should then use the print log map utility (CSQJU004) to print bootstrap data set (BSDS) information for the queue manager to locate the logs containing the log RBA.

For CF structures, you need to run the CSQJU004 utility on each queue manager in the queue sharing group to locate the logs containing the LRSN. You need these logs and any later logs to be able to recover the page sets and structures.

Do not change the DDname to page set association

IBM MQ associates page set number 00 with DDname CSQP0000, page set number 01 with DDname CSQP0001, and so on, up to CSQP0099. IBM MQ writes recovery log records for a page set based on the DDname that the page set is associated with. For this reason, you must not move page sets that have already been associated with a PSID DDname.

z/OS Recovering page sets

Use this topic to understand the factors involved when recovering pages sets, and how to minimize restart times.

A key factor in recovery strategy concerns the time for which you can tolerate a queue manager outage. The total outage time might include the time taken to recover a page set from a backup, or to restart the queue manager after an abnormal termination. Factors affecting restart time include how frequently you back up your page sets, and how much data is written to the log between checkpoints.

To minimize the restart time after an abnormal termination, keep units of work short so that, at most, two active logs are used when the system restarts. For example, if you are designing an IBM MQ application, avoid placing an MQGET call that has a long wait interval between the first in-syncpoint MQI call and the commit point because this might result in a unit of work that has a long duration. Another common cause of long units of work is batch intervals of more than 5 minutes for the channel initiator.

You can use the DISPLAY THREAD command to display the RBA of units of work and to help resolve the old ones.

How often must you back up a page set?

Frequent page set backup is essential if a reasonably short recovery time is required. This applies even when a page set is very small or there is a small amount of activity on queues in that page set.

If you use persistent messages in a page set, the backup frequency should be in hours rather than days. This is also the case for page set zero.

To calculate an approximate backup frequency, start by determining the target total recovery time. This consists of the following:

1. The time taken to react to the problem.
2. The time taken to restore the page set backup copy.

   If you use SnapShot backup/restore, the time taken to perform this task is a few seconds. For information about SnapShot, see the *DFSMSdss Storage Administration Guide*.
3. The time the queue manager requires to restart, including the additional time needed to recover the page set.

   This depends most significantly on the amount of log data that must be read from active and archive logs since that page set was last backed up. All such log data must be read, in addition to that directly associated with the damaged page set.

   Note: When using *fuzzy backup* (where a snapshot is taken of the logs and page sets while a unit of work is active), it might be necessary to read up to three additional checkpoints, and this might result in the need to read one or more additional logs.

When deciding on how long to allow for the recovery of the page set, the factors that you need to consider are:

- The rate at which data is written to the active logs during normal processing depends on how messages arrive in your system, in addition to the message rate.

  Messages received or sent over a channel result in more data logging than messages generated and retrieved locally.
- The rate at which data can be read from the archive and active logs.

  When reading the logs, the achievable data rate depends on the devices used and the total load on your particular DASD subsystem.

  With most tape units, it is possible to achieve higher data rates for archived logs with a large block size. However, if an archive log is required for recovery, all the data on the active logs must be read also.

z/OS Recovering CF structures

Use this topic to understand the recovery process for CF structures.

At least one queue manager in the queue sharing group must be active to process a RECOVER CFSTRUCT command. CF structure recovery does not affect queue manager restart time, because recovery is performed by an already active queue manager.

The recovery process consists of two logical steps that are managed by the RECOVER CFSTRUCT command:

1. Locating and restoring the backup.
2. Merging all the logged updates to persistent messages that are held on the CF structure from the logs of all the queue managers in the queue sharing group that have used the CF structure, and applying the changes to the backup.

The second step is likely to take much longer because a lot of log data might need to be read. You can reduce the time taken if you take frequent backups, or if you recover multiple CF structures at the same time, or both.

The queue manager performing the recovery locates the relevant backups on all the other queue managers' logs using the data in Db2 and the bootstrap data sets. The queue manager replays these backups in the correct time sequence across the queue sharing group, from just before the last backup through to the point of failure.

The time it takes to recover a CF structure depends on the amount of recovery log data that must be replayed, which in turn depends on the frequency of the backups. In the worst case, it takes as long to read a queue manager's log as it did to write it. So if, for example, you have a queue sharing group containing six queue managers, an hour's worth of log activity could take six hours to replay. In general it takes less time than this, because reading can be done in bulk, and because the different queue manager's logs can be read in parallel. As a starting point, we recommend that you back up your CF structures every hour.

All queue managers can continue working with non-shared queues and queues in other CF structures while there is a failed CF structure. If the administration structure has also failed, at least one of the queue managers in the queue sharing group must be started before you can issue the RECOVER CFSTRUCT command.

Backing up CF structures can require considerable log writing capacity, and can therefore impose a large load on the queue manager doing the backup. Choose a lightly loaded queue manager for doing backups; for busy systems, add an additional queue manager to the queue sharing group and dedicate it exclusively for doing backups.

z/OS リカバリーに関する具体的な目標の達成

このトピックでは、バックアップ頻度を調整することによって、リカバリー時間に関する具体的な目標を達成するための指針を取り上げます。

例えば、キュー・マネージャーのリカバリーと再始動の処理を、通常の起動時間も加算して、*xx* 秒以内に完了する、といった、リカバリーに関する具体的な目標を設定する場合は、バックアップ頻度 (時単位) を見積もるために以下の計算式を使用できます。

```
Formula (A)
                     Required restart time * System recovery log read rate
                           (in secs)                  (in MB/sec)
Backup frequency  = ------------------------------------------------------
   (in hours)               Application log write rate (in MB/hour)
```

注 : 以下のそれぞれの例で強調されているのは、ページ・セットのバックアップを頻繁に作成する必要がある、という点です。 計算式は、ほとんどのログ・アクティビティーが多数の持続メッセージから派生していることを前提にしています。 ただし、ログ・アクティビティーの量を簡単に計算できない状況もあります。 例えば、キュー共有グループ環境で、他のリソースに加えて共有キューも更新される作業単位があると、UOW レコードが IBM MQ のログに書き込まれる可能性があります。 したがって、数式 (A) のアプリケーション・ログの書き込み速度の正確な値は、IBM MQ のログが満杯になる実際の速度からのみ算出可能です。

例えば、IBM MQ MQI clients が合計ロードとして 1 秒あたり 100 件の持続メッセージを生成するシステムがあるとします。 この場合、すべてのメッセージがローカルに生成されます。

各メッセージのユーザー長が 1 KB であれば、1 時間ごとにログに記録されるデータの量は、おおよそ以下のような値になります。

```
100 * (1 + 1.3) KB * 3600 = approximately 800 MB

where
      100           = the message rate a second
      (1 + 1.3) KB  = the amount of data logged for
                      each 1 KB of persistent messages
```

リカバリー時間全体の目標を 75 分と設定するとしましょう。 問題に対応し、ページ・セットのバックアップ・コピーをリストアするための時間を 15 分と想定すると、数式 (A) を適用しながら、キュー・マネージャーのリカバリーと再始動を 60 分 (3600 秒) 以内に完了しなければなりません。 必要なすべてのログ・データが RVA2-T82 DASD にあるとしましょう。この DASD のリカバリー速度は、約 2.7 MB/秒なので、ページ・セットのバックアップ頻度は、最低でも以下の時間に 1 回ということになります。

```
3600 seconds * 2.7 MB a second / 800 MB an hour = 12.15 hours
```

IBM MQ アプリケーションの 1 日が約 12 時間だとすれば、毎日 1 回のバックアップが適切な頻度になります。 一方、アプリケーションの 1 日が 24 時間だとすれば、毎日 2 回のバックアップが適切です。

次に、すべてのメッセージが要求/応答型のアプリケーションのメッセージになる実動システム (つまり、受信側チャネルで持続メッセージを受信し、送信側チャネルで持続応答メッセージを生成して送信する、というシステム) の例を取り上げましょう。

この例では、達成するバッチ・サイズは 1 なので、メッセージごとに 1 つのバッチがあることになります。 1 秒あたりの要求応答が 50 件だとすると、合計ロードは、1 秒あたり 100 件の持続メッセージになります。 各メッセージの長さが 1 KB であれば、1 時間ごとにログに記録されるデータの量は、おおよそ以下のような値になります。

```
50((2 * (1+1.3) KB) + 1.4 KB + 2.5 KB) * 3600 = approximately 1500 MB

where:
 50                 = the message pair rate a second
 (2 * (1 + 1.3) KB) = the amount of data logged for each message pair
 1.4 KB             = the overhead for each batch of messages
                      received by each channel
 2.5 KB             = the overhead for each batch of messages sent
                      by each channel
```

キュー・マネージャーのリカバリーと再始動を 30 分 (1800 秒) 以内に完了するという目標を達成するには、この場合も必要なすべてのログ・データが RVA2-T82 DASD にあるという前提で計算すると、ページ・セットのバックアップが最低でも以下の時間に 1 回必要になります。

```
1800 seconds * 2.7 MB a second / 1500 MB an hour = 3.24 hours
```

バックアップ頻度の周期的な見直し

IBM MQ のログの使用量を「MB/時」という単位でモニターしてください。 このチェックを周期的に実行し、必要に応じてページ・セットのバックアップ頻度を変更します。

z/OS Backup considerations for other products

If you are using IBM MQ with CICS or IMS then you must also consider the implications for your backup strategy with those products. The data facility hierarchical storage manager (DFHSM) manages data storage, and can interact with the storage used by IBM MQ.

Backup and recovery with DFHSM

The data facility hierarchical storage manager (DFHSM) does automatic space-availability and data-availability management among storage devices in your system. If you use it, you need to know that it moves data to and from the IBM MQ storage automatically.

DFHSM manages your DASD space efficiently by moving data sets that have not been used recently to alternative storage. It also makes your data available for recovery by automatically copying new or changed data sets to tape or DASD backup volumes. It can delete data sets, or move them to another device. Its operations occur daily, at a specified time, and allow for keeping a data set for a predetermined period before deleting or moving it.

You can also perform all DFHSM operations manually. For more information on DFHSM, see the z/OS DFSMS product documentation. If you use DFHSM with IBM MQ, note that DFHSM does the following:

- Uses cataloged data sets.
- Operates on page sets and logs.
- Supports VSAM data sets.

Recovery and CICS

The recovery of CICS resources is not affected by the presence of IBM MQ. CICS recognizes IBM MQ as a non-CICS resource (or external resource manager), and includes IBM MQ as a participant in any syncpoint coordination requests using the CICS resource manager interface (RMI). For more information about CICS recovery and the CICS resource manager interface, see the CICS product documentation.

Recovery and IMS

IMS recognizes IBM MQ as an external subsystem and as a participant in syncpoint coordination. IMS recovery for external subsystem resources is described in the IMS product documentation.

z/OS Preparing for recovery on an alternative site

If a total loss of an IBM MQ computing center, you can recover on another IBM MQ system at a recovery site.

To recover an IBM MQ system at a recovery site, you must regularly back up the page sets and the logs. As with all data recovery operations, the objectives of disaster recovery are to lose as little data, workload processing (updates), and time as possible.

At the recovery site:

- The recovery IBM MQ queue manager **must** have the same name as the lost queue manager.
- Ensure the system parameter module used on the recovery queue manager contains the same parameters as the lost queue manager.

See Administering IBM MQ for z/OS and Troubleshooting IBM MQ for z/OS problems for more information.

z/OS Example of queue manager backup activity

This topic shows as an example of queue manager backup activity.

When you plan your queue manager backup strategy, a key consideration is retention of the correct amount of log data. Managing the logs describes how to determine which log data sets are required, by reference to the system recovery RBA of the queue manager. IBM MQ determines the system recovery RBA using information about the following:

- Currently active units of work.
- Page set updates that have not yet been flushed from the buffer pools to disk.
- CF structure backups, and whether this queue manager's log contains information required in any recovery operation using them.

You must retain sufficient log data to be able to perform media recovery. While the system recovery RBA increases over time, the amount of log data that must be retained only decreases when subsequent backups are taken. CF structure backups are managed by IBM MQ, and so are taken into account when reporting the system recovery RBA. This means that in practice, the amount of log data that must be retained only reduces when page set backups are taken.

Figure 43 on page 202 shows an example of the backup activity on a queue manager that is a member of a queue sharing group, how the recovery RBA varies with each backup, and how that affects the amount of log data that must be retained. In the example the queue manager uses local and shared resources: page sets, and two CF structures, STRUCTURE1 and STRUCTURE2.

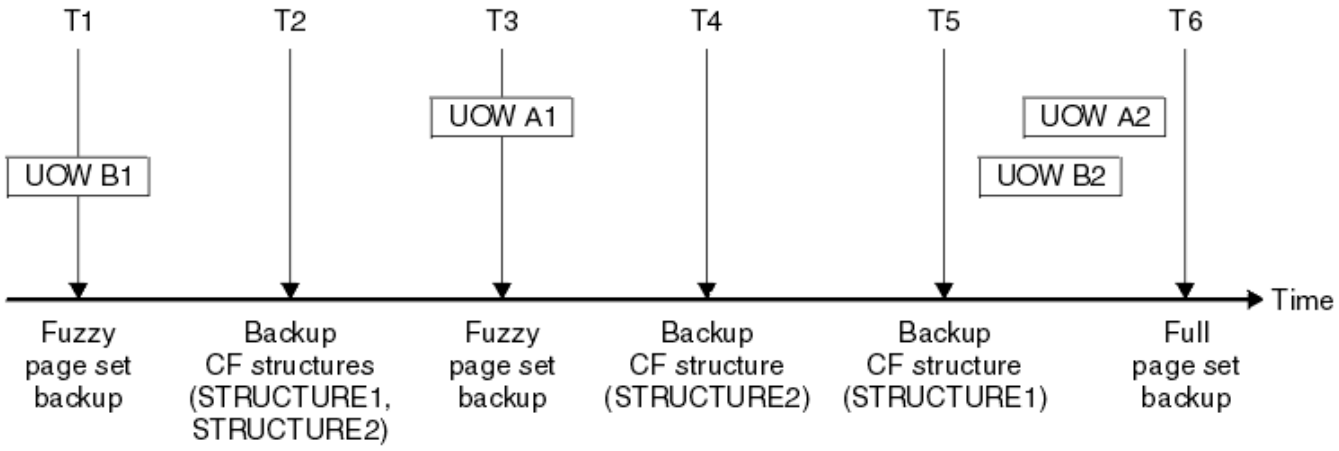


Figure 43. Example of queue manager backup activity

This is what happens at each point in time:

Point in time T1

A fuzzy backup is created of your page sets, as described in How to back up and recover page sets.

The system recovery RBA of the queue manager is the lowest of the following:

- The recovery RBAs of the page sets being backed up at this point.
- The lowest recovery RBA required to recover the CF application structures. This relates to the recovery of backups of STRUCTURE1 and STRUCTURE2 created earlier.
- The recovery RBA for the oldest currently active unit of work within the queue manager (UOWB1).

The system recovery RBA for this point in time is given by messages issued by the DISPLAY USAGE command, which is part of the fuzzy backup process.

Point in time T2

Backups of the CF structures are created. CF structure STRUCTURE1 is backed up first, followed by STRUCTURE2.

The amount of log data that must be retained is unchanged, because the same data as determined from the system recovery RBA at T1 is still required to recover using the page set backups taken at T1.

Point in time T3

Another fuzzy backup is created.

The system recovery RBA of the queue manager is the lowest of the following:

- The recovery RBAs of the page sets being backed up at this point.
- The lowest recovery RBA required to recover CF structure STRUCTURE1, because STRUCTURE1 was backed up before STRUCTURE2.
- The recovery RBA for the oldest currently active unit of work within the queue manager (UOWA1).

The system recovery RBA for this point in time is given by messages issued by the DISPLAY USAGE command, which is part of the fuzzy backup process.

You can now reduce the log data retained, as determined by this new system recovery RBA.

Point in time T4

A backup is taken of CF structure STRUCTURE2. The recovery RBA for the recovery of the oldest required CF structure backup relates to the backup of CF structure STRUCTURE1, which was backed up at time T2.

The creation of this CF structure backup has no effect on the amount of log data that must be retained.

Point in time T5

A backup is taken of CF structure STRUCTURE1. The recovery RBA for recovery of the oldest required CF structure backup now relates to recovery of CF structure STRUCTURE2, which was backed up at time T4.

The creation of this CF structure backup has no effect on amount of log data that must be retained.

Point in time T6

A full backup is taken of your page sets as described in How to back up and recover page sets.

The system recovery RBA of the queue manager is the lowest of the following:

- The recovery RBAs of the page sets being backed up at this point.
- The lowest recovery RBA required to recover the CF structures. This relates to recovery of CF structure STRUCTURE2.
- The recovery RBA for the oldest currently active unit of work within the queue manager. In this case, there are no current units of work.

The system recovery RBA for this point in time is given by messages issued by the DISPLAY USAGE command, which is part of the full backup process.

Again, the log data retained can be reduced, because the system recovery RBA associated with the full backup is more recent.

z/OS Planning your z/OS UNIX environment

Certain processes within the IBM MQ queue manager, channel initiator, and mqweb server use z/OS UNIX System Services (z/OS UNIX) for their normal processing.

The queue manager and channel initiator started task user IDs need an OMVS segment with a UID defined in order to be able to access z/OS UNIX. The user IDs require no special permissions in z/OS UNIX.

Note: Although the queue manager and channel initiator make use of z/OS UNIX facilities (for example, to interface with TCP/IP services), they do not need to access any of the content of the IBM MQ installation directory in the z/OS UNIX file system. As a result, the queue manager and channel initiator do not require any configuration to specify the path for the z/OS UNIX file system.

The mqweb server, which hosts the IBM MQ Console and REST API, makes use of files in the IBM MQ installation directory in the z/OS UNIX file system. It also needs access to another file system which is used to store data such as configuration and log files. The mqweb started task JCL needs to be customized to reference these z/OS UNIX file systems.

The content of the IBM MQ directory in the z/OS UNIX file system is also used by applications connecting to IBM MQ. For example, applications using the IBM MQ classes for Java or IBM MQ classes for JMS interfaces.

See the following topics for the relevant configuration instructions:

- Environment variables relevant to IBM MQ classes for Java
- IBM MQ classes for Java libraries
- Setting environment variables
- Configuring the Java Native Interface (JNI) libraries

z/OS 計画中 Advanced Message Security の上 z/OS

TLS (または SSL) を使用して、ネットワークを流れるメッセージを暗号化して保護できますが、キューに入っている (保存状態の) メッセージは保護されません。Advanced Message Security (AMS) では、メッセージが最初にキューに書き込まれる時から取り出される時まで保護されるので、そのメッセージの対象受信者だけがメッセージを読み取れます。 メッセージの書き込み処理では、暗号化と署名が可能ですが、取り出し処理では無保護になります。

AMS では、メッセージ保護の構成方法がいくつかあります。

1. メッセージに署名できます。 メッセージは平文ですが、チェックサムがあり、チェックサムに署名できます。 その結果、メッセージの内容が変更されれば、その変更を検出できます。 署名入りの内容から、データに署名したユーザーを識別できます。
2. メッセージを暗号化できます。 復号鍵のないユーザーは内容を表示できません。 復号鍵は受信者ごとに暗号化されます。
3. メッセージを暗号化して署名を追加できます。 復号鍵は受信者ごとに暗号化されます。署名から、メッセージの送信者を識別できます。

暗号化と署名では、デジタル証明書と鍵リングを使用します。

クライアント側で AMS を使用するためのセットアップを行えば、データをクライアント・チャネルに書き込む前に保護できます。 保護したメッセージをリモート・キュー・マネージャーに送信できますが、そのメッセージを処理するための構成をリモート・キュー・マネージャー側で行う必要があります。

AMS の設定

AMS の処理を実行する時には、AMS のアドレス・スペースを使用します。 これも、鍵リングと証明書のアクセス権を設定してデータを保護するための追加のセキュリティー・セットアップになります。

ユーティリティー・プログラム (CSQ0UTIL) を使用してキューのセキュリティー・ポリシーを定義することによって、保護するキューを構成します。

AMS のセットアップ後の作業

メッセージを書き込むユーザーとメッセージを取り出すユーザーのデジタル証明書と鍵リングをセットアップする必要があります。

ユーザー Alice が z/OS で Bob にメッセージを送信する必要がある場合は、AMS で Bob の公開証明書のコピーが必要になります。

Bob が Alice から受け取ったメッセージを処理するには、AMS で Alice の公開証明書か Alice が使用するのと同じ認証局証明書が必要になります。

重要 : 以下の作業が必要です。

- キューへの書き込みとキューからの取り出しを行えるユーザーを慎重に計画します。
- ユーザー名とユーザーの証明書名を指定します。

ミスが起きやすく、問題を解決するのが難しい場合もあります。

関連概念

147 ページの『Planning for your queue manager』
When you are setting up a queue manager, your planning should allow for the queue manager to grow, so that the queue manager meets the needs of your enterprise.

z/OS 計画中 Managed File Transfer の上 z/OS

このセクションは、z/OS 上で Managed File Transfer (MFT) を実行するためにシステムをセットアップする方法に関するガイダンスとして使用します。

z/OS 計画中 Managed File Transfer- ハードウェアおよびソフトウェアの要件 z/OS

システムのハードウェア要件およびソフトウェア要件の設定方法に関するガイダンスとしてこのトピックを使用し、z/OS で Managed File Transfer (MFT) を実行します。

ソフトウェア要件

Managed File Transfer は Java で作成されており、プログラムを構成して操作するためのシェル・スクリプトと JCL がいくつか含まれています。

重要 : Managed File Transfer を構成するには、 z/OS UNIX System Services (z/OS UNIX) に精通している必要があります。 以下に例を示します。

- ファイル・ディレクトリー構造。`/u/userID/myfile.txt` などの名前が付けられています。
- 以下のような z/OS UNIX コマンド
  - `cd` (ディレクトリー変更)
  - `ls` (リスト)
  - `chmod` (ファイル・アクセス権の変更)
  - `chown` (ファイル所有権、またはファイルやディレクトリーにアクセスできるグループの変更)

MFT を構成および実行できるようにするには、z/OS UNIX で以下の製品が必要です。

1. 例えば、Java はディレクトリー `/java/java80_bit64_GA/J8.0_64/` にあります。
2. 例えば、IBM MQ 9.4.0 はディレクトリー `/mqm/V9R3M0` にあります。
3. 状況および履歴に Db2 を使用する場合は、Db2 JDBC ライブラリーをインストールする必要があります (例えば、ディレクトリー `/db2/db2v10/jdbc/libs` にインストールします)。

製品の登録

始動時に、Managed File Transfer では `sys1.parmlib` (IFAPRDxx) 連結内で登録が検査されます。 以下のコードは、MFT を登録する方法の例を示しています。

```
PRODUCT OWNER('IBM CORP')
NAME('WS MQ FILE TRANS')
ID(5655-MFT)
VERSION(*) RELEASE(*) MOD(*)
FEATURENAME('WS MQ FILE TRANS')
STATE(ENABLED)
```

ディスク・スペース

IBM MQ for z/OS プログラム・ディレクトリーに Managed File Transfer の DASD および zFS のストレージ要件が説明されています。 IBM MQ for z/OS のプログラム・ディレクトリーのダウンロード・リンクについては、 IBM MQ 9.4 製品資料およびプログラム・ディレクトリーの PDF ファイルを参照してください。

z/OS 計画中 Managed File Transfer- トポロジー z/OS

システムに必要なトポロジに関するガイダンスとしてこのトピックを使用して、z/OS で Managed File Transfer (MFT) を実行します。

Managed File Transfer キュー・マネージャー

IBM MQ Managed File Transfer トポロジーは、以下のものから構成されます。

エージェント、およびそれに関連するキュー・マネージャー
エージェントは、エージェント・キュー・マネージャーでホストされているシステム・キューを使用して、状況情報を保持し、作業要求を受け取ります。

コマンド・キュー・マネージャー
これは、MFT トポロジーへのゲートウェイとして機能します。 これは、送信側チャネルと受信側チャネル、またはクラスタリングを介して、エージェント・キュー・マネージャーに接続されます。 特定のコマンドが実行されると、コマンド・キュー・マネージャーに直接接続され、指定されたエージェントにメッセージが送信されます。 このメッセージは IBM MQ ネットワークを経由してエージェント・キュー・マネージャーに経路指定されます。そこでメッセージはエージェントによって取り出され、処理されます。

調整キュー・マネージャー
これは、トポロジー全体について把握する中央のハブです。 調整キュー・マネージャーは、トポロジー内のすべてのエージェント・キュー・マネージャーに対して、送信側チャネルと受信側チャネル、ま

たはクラスタリングを使用して接続されます。 エージェントは定期的にステータス情報を調整キュー・マネージャーに公開し、そこに転送テンプレートを保管します。

1 つのキュー・マネージャーが 1 つのトポロジー内で複数のロールを実行することが可能です。 例えば、同じキュー・マネージャーを調整キュー・マネージャーと、トポロジーのコマンド・キュー・マネージャーの両方として構成することができます。

複数のキュー・マネージャーを使用する場合、キュー・マネージャー間のチャネルをセットアップする必要があります。 これを行うには、クラスター化を使用するか、Point-to-Point 接続を使用できます。

IBM MQ Managed File Transfer for z/OS を使用する場合、トポロジー内の異なるロールに使用するキュー・マネージャーを決定する際に、考慮すべき事柄がいくつかあります。

エージェント・キュー・マネージャー

IBM MQ Managed File Transfer for z/OS エージェントのエージェント・キュー・マネージャーが z/OS 上で実行されている必要があります。

次の場合

- エージェントが IBM MQ 9.1 以降で Managed File Transfer for z/OS を実行している
- かつ、エージェント・キュー・マネージャーが IBM MQ Advanced for z/OS Value Unit Edition (Advanced VUE) のライセンス交付を受けている

エージェントは、CLIENT トランスポートを使用してキュー・マネージャーに接続することができます。

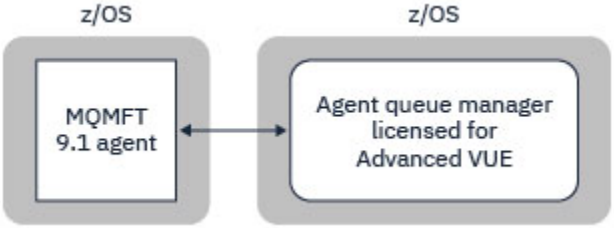


図 *44. z/OS* 上の *MFT 9.1* エージェントは、キュー・マネージャーが *Advanced VUE* のライセンス交付を受けていることを前提として、*CLIENT* トランスポートを使用してキュー・マネージャーに接続できます。

次の場合

- エージェントが IBM MQ 9.0 以前で Managed File Transfer for z/OS を実行している
- または、エージェント・キュー・マネージャーが IBM MQ 9.0 以降で Managed File Transfer for z/OS を実行していて、エージェント・キュー・マネージャーが MFT、IBM MQ Advanced for z/OS、または Advanced VUE のいずれかに対してライセンス交付を受けている

エージェントは BINDINGS トランスポートを使用してキュー・マネージャーに接続する必要があります。

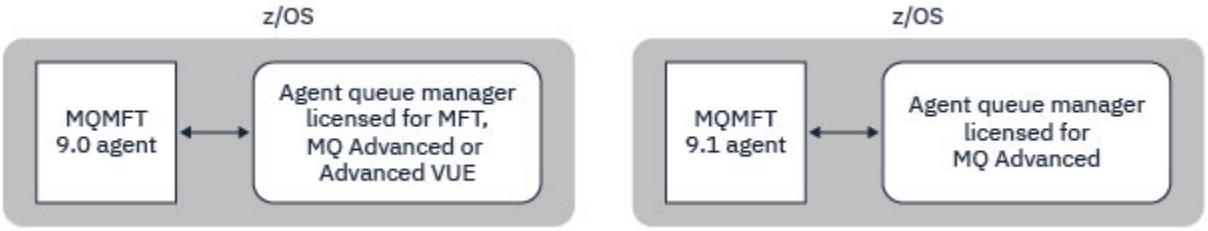


図 *45. MFT* または *IBM MQ Advanced* のいずれかのライセンス交付を受けたエージェント・キュー・マネージャーを持つ *z/OS* および *9.1* エージェント上の *MFT 9.0* エージェントは、*BINDINGS* トランスポートを使用して接続する必要があります。

コマンド・キュー・マネージャー

『MFT のコマンドおよびプロセスと、その接続先のキュー・マネージャー』 トピックには、Managed File Transfer トポロジーのコマンド・キュー・マネージャーに接続するすべてのコマンドが表示されます。

注 : これらのコマンドを z/OS 上で実行する場合は、コマンド・キュー・マネージャーも z/OS 上になければなりません。

そのコマンド・キュー・マネージャーが Advanced VUE 用にライセンス交付を受けている場合、それらのコマンドは CLIENT トランスポートを使用してそのキュー・マネージャーに接続することができます。 それ以外の場合、コマンドは BINDINGS トランスポートを使用してコマンド・キュー・マネージャーに接続する必要があります。

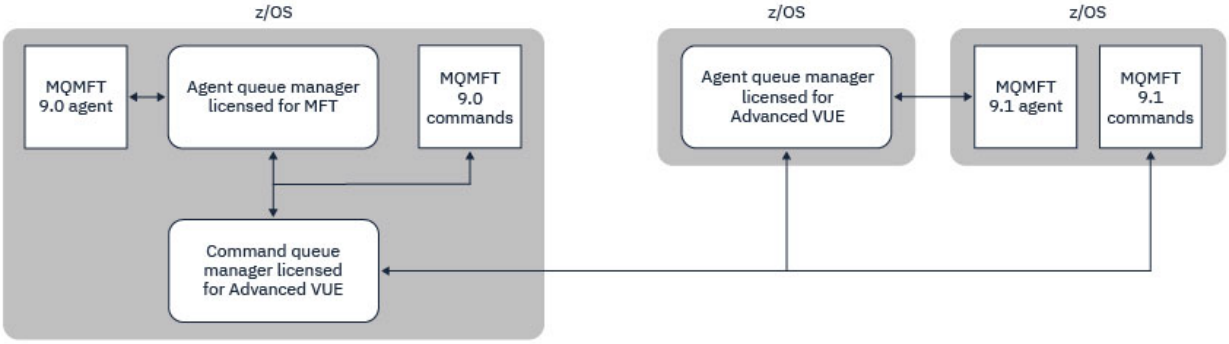


図 *46.* コマンドは *MFT* トポロジーのコマンド・キュー・マネージャーに接続します。 これらのコマンドを *z/OS* 上で実行する場合は、コマンド・キュー・マネージャーも *z/OS* 上になければなりません

調整キュー・マネージャー

IBM MQ Managed File Transfer for z/OS エージェントは、調整キュー・マネージャーが z/OS 上で実行されているか、またはマルチプラットフォーム上で実行されているトポロジーの一部とすることができます。

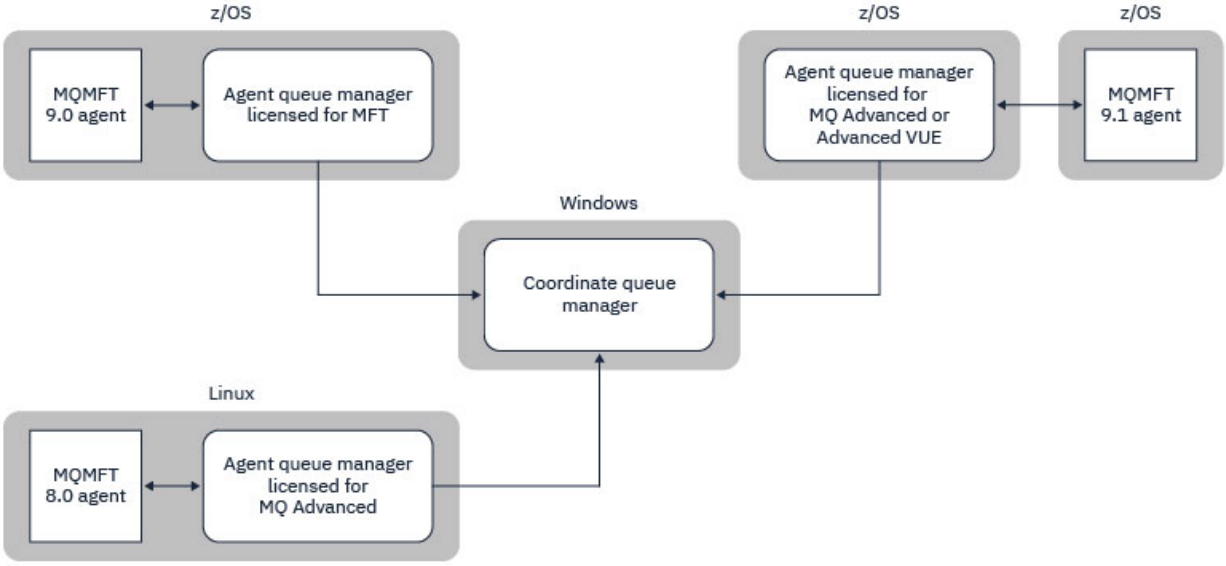


図 *47. z/OS* 上で実行される *MFT* エージェントは、調整キュー・マネージャーが *IBM MQ* マルチプラットフォーム上で実行される *MFT* トポロジーの一部にすることができます。

『MFT のコマンドおよびプロセスと、その接続先のキュー・マネージャー』トピックには、Managed File Transfer トポロジーの調整キュー・マネージャーに接続するコマンドが示されています。 これらのコマンドを z/OS で実行してから、別のプラットフォームで実行されている調整キュー・マネージャーに接続することができます。

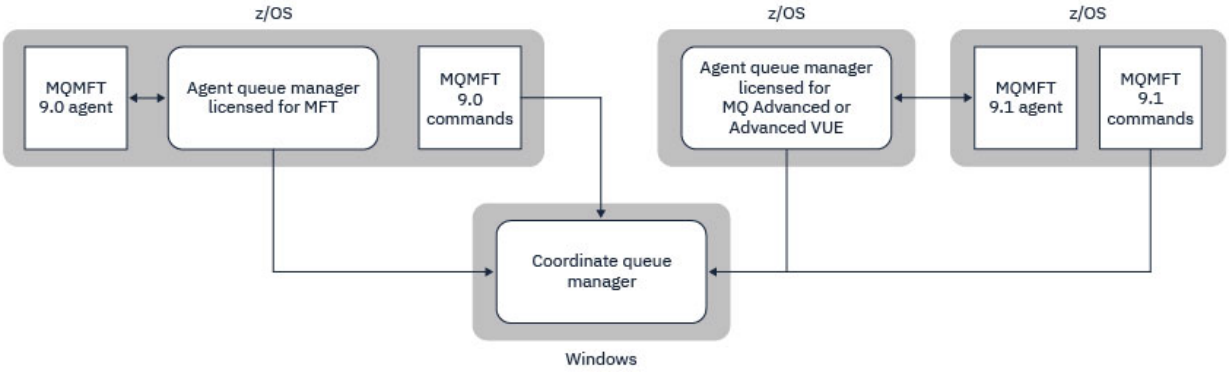


図 *48.* ***fteListAgents*** などの特定のコマンドは、*MFT* トポロジーの調整キュー・マネージャーに直接接続します。

必要なエージェントの数はいくつか

エージェントはデータ転送を処理します。データの転送要求を出すときには、エージェントの名前を指定します。

デフォルトでは、1 つのエージェントは 25 個の送信要求と 25 個の受信要求を同時に処理できます。 これらの処理は構成可能です。 詳しくは、 Managed File Transfer configuration options on z/OS を参照してください。

エージェントがビジー状態の場合は、処理がキューに入れられます。 1 つの要求を処理するのにかかる時間は、例えば送信されるデータの量、ネットワーク帯域幅、ネットワーク遅延などの複数の要因によって決まります。

複数のエージェントを使って並列的に処理を行うことができます。

また、あるエージェントがどのリソースにアクセスできるかを制御することにより、いくつかのエージェントの処理対象を特定のデータ・サブセットに限定することもできます。

さまざまな優先度を使って要求を処理する必要がある場合には、複数のエージェントを使用し、ワークロード・マネージャーを使ってジョブの優先度を設定することができます。

エージェントの実行

通常、エージェントは長時間にわたって実行されるプロセスです。 バッチで実行される複数のジョブとして、または開始済みタスクとして、これらのプロセスを実行依頼できます。

z/OS 計画中 Managed File Transfer- セキュリティに関する考慮事項 z/OS

セキュリティーに関する考慮事項のガイダンスとしてこのトピックを使用して、z/OS で、Managed File Transfer (MFT) を実行します。

セキュリティー

MFT 構成および MFT 操作にどのユーザー ID が使用されるかを識別する必要があります。

転送するファイルやキューを識別して、どのユーザー ID が MFT への転送要求の送信を行うかを識別する必要があります。

エージェントおよびロガーをカスタマイズするときには、MFT サービスまたは MFT 管理の実行を許可されるユーザーのグループを指定します。

MFT のカスタマイズを始める前に、このグループをセットアップしてください。 MFT が IBM MQ キューを使用するとき、キュー・マネージャーでセキュリティーが有効になっていると、MFT では以下のリソースのアクセス権限が必要になります。

表 *26. MQADMIN* リソース・クラス	
名前	**必要なアクセス権限**
QUEUE.SYSTEM.FTE.EVENT.agent_name	更新
QUEUE.SYSTEM.FTE.COMMAND.agent_name	更新
CONTEXT.SYSTEM.FTE.COMMAND.agent_name	更新
QUEUE.SYSTEM.FTE.STATE.agent_name	更新
QUEUE.SYSTEM.FTE.DATA.agent_name	更新
QUEUE.SYSTEM.FTE.REPLY.agent_name	更新
QUEUE.SYSTEM.FTE.AUTHAGT1.agent_name	更新
QUEUE.SYSTEM.FTE.AUTHTRN1.agent_name	更新
QUEUE.SYSTEM.FTE.AUTHOPS1.agent_name	更新
QUEUE.SYSTEM.FTE.AUTHSCH1.agent_name	更新
QUEUE.SYSTEM.FTE.AUTHMON1.agent_name	更新
QUEUE.SYSTEM.FTE.AUTHADM1.agent_name	更新

表 *27. MQQUEUE* リソース・クラス	
名前	**必要なアクセス権限**
SYSTEM.FTE.AUTHAGT1.agent_name	更新
SYSTEM.FTE.AUTHTRN1.agent_name	更新
SYSTEM.FTE.AUTHOPS1.agent_name	更新
SYSTEM.FTE.AUTHSCH1.agent_name	更新
SYSTEM.FTE.AUTHMON1.agent_name	更新

転送を要求するユーザーがファイル・システムのどの部分にアクセスできるかを決定するには、ユーザー・サンドボックス機能を使用できます。

ユーザー・サンドボックス機能を有効にするには、制限の対象となるエージェントの *agent.properties* ファイルに `userSandboxes=true` ステートメントを追加して、`MQ_DATA_PATH/mqft/config/coordination_qmgr_name/agents/agent_name/UserSandboxes.xml` ファイルに適切な値を追加します。

詳しくは、ユーザー・サンドボックスでの処理を参照してください。

このユーザー ID は、`UserSandboxes.xml` ファイルで構成されます。

この XML ファイルには、ユーザー ID またはユーザー ID*、使用可能 (包含)/使用不可 (除外) リソース・リストなどの情報が含まれます。 特定のユーザー ID がどのリソースにアクセスできるかを定義する必要があります。以下に例を示します。

表 *28.* ユーザー *ID* および特定のリソースへのアクセス権限の例			
ユーザー ID	**アクセス**	**含めるか除外するか**	**Resource**
Admin*	読み取り	含める	`/home/user/**`
Admin*	読み取り	除外する	`/home/user/private/**`
Sysprog	読み取り	含める	`/home/user/**`

表 28. ユーザー ID および特定のリソースへのアクセス権限の例 (続き)			
ユーザー ID	**アクセス**	**含めるか除外するか**	**Resource**
Admin*	読み取り	含める	Application.reply.queue

注 :

1. type=queue が指定されている場合、リソースはキュー名または queue@qmgr のいずれかです。
2. リソースが // で始まる場合、リソースはデータ・セットです。そうでない場合、リソースは z/OS UNIX 内のファイルです。
3. ユーザー ID は MQMD 構造からのユーザー ID です。このため、実際にメッセージを書き込むユーザー ID を反映していない可能性があります。
4. ローカル・キュー・マネージャーでの要求には、MQADMIN CONTEXT.* を使用できます。 この値を設定できるユーザーを制限します。
5. リモート・キュー・マネージャーを介して受信される要求に関しては、MQMD 構造内のユーザー ID が無許可で設定されることを防ぐために、分散キュー・マネージャーでセキュリティーが有効になっていることを前提とする必要があります。
6. Linux マシン上の SYSPROG1 というユーザー ID は、z/OS 上のセキュリティー検査のための同じユーザー ID SYSPROG1 です。

z/OS Planning to use the IBM MQ Console and REST API on z/OS

The IBM MQ Console and REST API are applications that run in a WebSphere Liberty (Liberty) server known as mqweb. The mqweb server runs as a started task. The IBM MQ Console allows a web browser to be used to administer queue managers. The REST API provides a simple programmatic interface for applications to do queue manager administration, and to perform messaging.

Installation and configuration files

You need to install the IBM MQ for z/OS UNIX System Services Web Components feature, which will install the files needed to run the mqweb server in z/OS UNIX System Services (z/OS UNIX). You need to be familiar with z/OS UNIX to be able to configure and manage the mqweb server.

See IBM MQ for z/OS Program Directory PDF files for information on installing IBM MQ for z/OS UNIX System Services Components.

The IBM MQ files in z/OS UNIX are installed with various attributes set that are required for the correct operation of the mqweb server. If you need to copy the IBM MQ z/OS UNIX installation files, for example if you have installed IBM MQ on one system, and run IBM MQ on a different system, you should copy the IBM MQ ZFS created during the installation, and mount it read only at the destination. Copying the files in other ways might cause some file attributes to be lost.

You need to decide upon the location for, and create, a Liberty user directory when you create the mqweb server. This directory contains configuration and log files, and the location can be something similar to `/var/mqm/mqweb`.

Using the IBM MQ Console and REST API with queue managers at different levels

The REST API can directly interact only with queue managers that run at the same Version, Release, and Modification (VRM) as the mqweb server which runs the REST API. For example, the IBM MQ 9.4.0 REST API can directly interact only with local queue managers at IBM MQ 9.4.0, and the IBM MQ 9.3.5 REST API can directly interact only with local queue managers at IBM MQ 9.3.5.

You can use the REST API to administer a queue manager at a different version from the mqweb server by configuring a gateway queue manager. However, you need at least one queue manager at the same version as the mqweb server to act as the gateway queue manager. For more information, see Remote administration using the REST API.

The IBM MQ Console can be used to manage local queue managers that run at the same version as the IBM MQ Console. From IBM MQ 9.3.0, you can also use the IBM MQ Console to administer a queue manager running on a remote system, or at a different version to the IBM MQ Console. For more information, see IBM MQ Console: Adding a remote queue manager.

Migration

If you have only one queue manager, you can run the mqweb server as a single started task, and change the libraries it uses when you migrate your queue manager.

If you have more than one queue manager, during migration you can start mqweb servers at different versions by using started tasks with different names. These names can be any name you want. For example, you can start an IBM MQ 9.3.0 mqweb server using a started task named MQWB0930, and an IBM MQ 9.3.5 mqweb server using a started task named MQWB0935.

Then, when you migrate the queue managers from one version to a later version, the queue managers become available in the mqweb server for the later version, and are no longer available in the mqweb server for the earlier version.

After you have migrated all the queue managers to the later version, you can delete the mqweb server for the earlier version.

HTTP ports

The mqweb server uses up to two ports for HTTP:

- One for HTTPS, with a default value of 9443.
- One for HTTP. HTTP is not enabled by default, but if enabled, has a default value of 9080.

If the default port values are in use, you must allocate other ports. If you have more than one mqweb server running simultaneously for more than one version of IBM MQ, you must allocate separate ports for each version. For more information on setting the ports that the mqweb server uses, see Configuring the HTTP and HTTPS ports.

You can use the following TSO command to display information about a port:

```
NETSTAT TCP tcpip (PORT portNumber)
```

where *tcpip* is the name of the TCP/IP address space, and *portNumber* specifies the number of the port to display information about.

Security - starting the mqweb server

The mqweb server user ID needs certain authorities. For more information, see Authority required by the mqweb server started task user ID.

Security - using the IBM MQ Console and REST API

When you use the IBM MQ Console and REST API, you must authenticate as a user that is included in a configured registry. These users are assigned specific roles that determine the actions the users can perform. For example, to use the messaging REST API, a user must be assigned the `MQWebUser` role. For more information about the available roles for the IBM MQ Console and REST API, and the access that these roles grant, see Roles on the IBM MQ Console and REST API.

For more information about configuring security for the IBM MQ Console and REST API, see IBM MQ Console and REST API security.

特記事項

本書は米国 IBM が提供する製品およびサービスについて作成したものです。

IBM 本書に記載の製品、サービス、または機能が日本においては提供されていない場合があります。 日本で利用可能な製品、サービス、および機能については、日本 IBM の営業担当員にお尋ねください。 本書で IBM 製品、プログラム、またはサービスに言及していても、その IBM 製品、プログラム、またはサービスのみが使用可能であることを意味するものではありません。 これらに代えて、IBM の知的所有権を侵害することのない、機能的に同等の製品、プログラム、またはサービスを使用することができます。 ただし、IBM 以外の製品とプログラムの操作またはサービスの 評価および検証は、お客様の責任で行っていただきます。

IBM は、本書に記載されている内容に関して特許権 (特許出願中のものを含む) を保有している場合があります。 本書の提供は、お客様にこれらの特許権について 実施権を許諾することを意味するものではありません。 実施権についてのお問い合わせは、書面にて下記宛先にお送りください。

〒 103-8510
東京都中央区日本橋箱崎町 19 番 21 号
日本アイ・ビー・エム株式会社
日本アイ・ビー・エム株式会社
法務・知的財産
U.S.A.

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

Intellectual Property Licensing
Legal and Intellectual Property Law
〒 103-8510
19-21, Nihonbashi-Hakozakicho, Chuo-ku
Tokyo 103-8510, Japan

以下の保証は、国または地域の法律に沿わない場合は、適用されません。 INTERNATIONAL BUSINESS MACHINES CORPORATION は、法律上の瑕疵担保責任、商品性の保証、特定目的適合性の保証および法律上の瑕疵担保責任を含むすべての明示もしくは黙示の保証責任を負わないものとします。 "" 国または地域によっては、法律の強行規定により、保証責任の制限が 禁じられる場合、強行規定の制限を受けるものとします。

この情報には、技術的に不適切な記述や誤植を含む場合があります。 本書は定期的に見直され、必要な変更は本書の次版に組み込まれます。 IBM は予告なしに、随時、この文書に記載されている製品またはプログラムに対して、改良または変更を行うことがあります。

本書において IBM 以外の Web サイトに言及している場合がありますが、便宜のため記載しただけであり、決してそれらの Web サイトを推奨するものではありません。 それらの Web サイトにある資料は、この IBM 製品の資料の一部では ありません。それらの Web サイトは、お客様の責任でご使用ください。

IBM は、お客様が提供するいかなる情報も、お客様に対してなんら義務も負うことのない、自ら適切と信ずる方法で、使用もしくは配布することができるものとします。

本プログラムのライセンス保持者で、(i) 独自に作成したプログラムとその他のプログラム (本プログラムを含む) との間での情報交換、および (ii) 交換された情報の相互利用を可能にすることを目的として、本プログラムに関する情報を必要とする方は、下記に連絡してください。

東京都中央区日本橋箱崎町 19 番 21 号
日本アイ・ビー・エム株式会社
Software Interoperability Coordinator, Department 49XA
3605 Highway 52 N
Rochester, MN 55901
U.S.A.

© Copyright IBM Corp. 2007 年, 2025

本プログラムに関する上記の情報は、適切な使用条件の下で使用すること ができますが、有償の場合もあります。

本書で説明されているライセンス・プログラムまたはその他の ライセンス資料は、IBM 所定のプログラム契約の契約条項、IBM プログラムのご使用条件、またはそれと同等の条項に基づいて、 IBM より提供されます。

この文書に含まれるいかなるパフォーマンス・データも、管理環境下で 決定されたものです。 そのため、他の操作環境で得られた結果は、異なる可能性があります。 一部の測定が、開発レベルのシステムで行われた可能性がありますが、その測定値が、一般に利用可能なシステムのものと同じである保証はありません。 さらに、一部の測定値が、推定値である可能性があります。 実際の結果は、異なる可能性があります。 お客様は、お客様の特定の環境に適したデータを確かめる必要があります。

IBM 以外の製品に関する情報は、その製品の供給者、出版物、 もしくはその他の公に利用可能なソースから入手したものです。 IBM は、それらの製品のテストは行っておりません。 したがって、他社製品に関する実行性、互換性、またはその他の要求については確証できません。 IBM 以外の製品の性能に関する質問は、それらの製品の供給者にお願いします。

IBM の将来の方向または意向に関する記述については、予告なしに変更または撤回される場合があり、単に目標を示しているものです。

本書には、日常の業務処理で用いられるデータや報告書の例が含まれています。 より具体性を与えるために、それらの例には、個人、企業、ブランド、 あるいは製品などの名前が含まれている場合があります。これらの名称はすべて架空のものであり、 名称や住所が類似する企業が実在しているとしても、それは偶然にすぎません。

著作権使用許諾:

本書には、様々なオペレーティング・プラットフォームでのプログラミング手法を例示するサンプル・アプリケーション・プログラムがソース言語で掲載されています。 お客様は、サンプル・プログラムが書かれているオペレーティング・プラットフォームのアプリケーション・プログラミング・インターフェースに準拠したアプリケーション・プログラムの開発、使用、販売、配布を目的として、いかなる形式においても、IBM に対価を支払うことなくこれを複製し、改変し、配布することができます。 このサンプル・プログラムは、あらゆる条件下における完全なテストを経ていません。 従って IBM は、これらのサンプル・プログラムについて信頼性、利便性もしくは機能性が あることをほのめかしたり、保証することはできません。

この情報をソフトコピーでご覧になっている場合は、写真やカラーの図表は表示されない場合があります。

プログラミング・インターフェース情報

プログラミング・インターフェース情報 (提供されている場合) は、このプログラムで使用するアプリケーション・ソフトウェアの作成を支援することを目的としています。

本書には、プログラムを作成するユーザーが IBM MQ のサービスを使用できるようにするためのプログラミング・インターフェースに関する情報が記載されています。

ただし、この情報には、診断、修正、および調整情報が含まれている場合があります。 診断、修正、調整情報は、お客様のアプリケーション・ソフトウェアのデバッグ支援のために提供されています。

重要 : この診断、修正、およびチューニング情報は、変更される可能性があるため、プログラミング・インターフェースとして使用しないでください。

商標

IBM、 IBM ロゴ、 ibm.com®は、世界の多くの国で登録された IBM Corporation の商標です。 現時点での IBM の商標リストについては、 "Copyright and trademark information"www.ibm.com/legal/copytrade.shtml をご覧ください。 他の製品名およびサービス名等は、それぞれ IBM または各社の商標である場合があります。

Microsoft および Windows は、Microsoft Corporation の米国およびその他の国における商標です。

UNIX は The Open Group の米国およびその他の国における登録商標です。

Linux は、Linus Torvalds 氏の米国およびその他の国における登録商標です。

この製品には、 Eclipse Project (https://www.eclipse.org/) により開発されたソフトウェアが含まれています。

Java およびすべての Java 関連の商標およびロゴは Oracle やその関連会社の米国およびその他の国における商標または登録商標です。

部品番号:

(1P) P/N: