

7 メッセージング 2

7.1 概要	2
7.1.1 メッセージング	2
7.1.2 JMS	3
7.1.2.1 JMS のタイプ	3
7.1.2.2 JMS の特徴	4
7.1.2.3 利用可能な JMS プロバイダー	5
7.1.3 MDB (MESSAGE-DRIVEN BEAN)	5
7.2 SIBus 構成	6
7.2.1 トポロジ設定	6
7.2.1.1 基本トポロジ	6
7.2.1.2 高可用性構成	9
7.2.1.3 パーティション構成	13
7.2.1.4 メッセージング・エンジン・ポリシー・アシスタンス	22
7.2.2 その他の設定	23
7.2.2.1 宛先の設定	24
7.2.2.2 外部バス	24
7.2.2.3 メディエーション	32
7.3 障害時の動作	35
7.3.1 HA マネージャー	35
7.3.2 高可用性構成における障害時の動作	38
7.3.3 パーティション構成における障害時の動作	51
7.3.4 ネットワーク障害時固有の考慮点	55
7.4 JMS 接続設定	56
7.4.1 デフォルト・メッセージング・プロバイダー	56
7.4.1.1 JMS 接続ファクトリーの設定	57
7.4.1.2 JMS キュー接続ファクトリーの設定	59
7.4.1.3 JMS トピック接続ファクトリーの設定	59
7.4.1.4 JMS キューの設定	59
7.4.1.5 JMS トピックの設定	62
7.4.1.6 JMS アクティベーション・スペックの設定	63
7.4.1.7 MDB を使用する際の構成	66
7.4.1.8 接続プールの設定	68
7.4.2 WEBSphere MQ メッセージング・プロバイダー	70
7.4.2.1 WebSphere MQ 接続ファクトリーの設定	71
7.4.2.2 WebSphere MQ キュー接続ファクトリーの設定	74
7.4.2.3 WebSphere MQ トピック接続ファクトリーの設定	74
7.4.2.4 WebSphere MQ キューの設定	74
7.4.2.5 WebSphere MQ トピックの設定	76
7.4.2.6 JMS アクティベーション・スペックの設定	77
7.4.2.7 MDB を使用する際の構成	80
7.4.2.8 接続プールの設定	82
7.4.2.9 セッション・プールの設定	83

7 メッセージング

この章では、メッセージングについて説明します。WebSphere Application Server (WAS) V8.5 では、SIBus の機能によりメッセージングを WAS 単独で実行することができます。また、WebSphere MQ に代表される Message Oriented Middleware (MOM) 製品と連携することも可能です。この章では、SIBus の構成方法と、JMS の接続先として SIBus と WebSphere MQ を利用する場合の設定方法について説明します。

7.1 概要

7.1.1 メッセージング

メッセージングとは、メッセージを介したソフトウェア・コンポーネントやアプリケーション間の通信方式の1つであり、疎結合の分散コミュニケーションを実現します。つまり、メッセージの送信者と受信者は、通信のために同時に利用可能状態にある必要は必ずしもありません。送信者は受信者に関して何も知る必要はなく、また受信者も送信者に関して知っている必要はありません。送信者と、受信者が知っておかなければならないのは、メッセージのフォーマットと宛先 (キュー) のみです。この点が、EJB のクライアントとサーバーのように、相手のリモート・メソッドを直接呼び出すような緊密な結合をするテクノロジーと異なります。

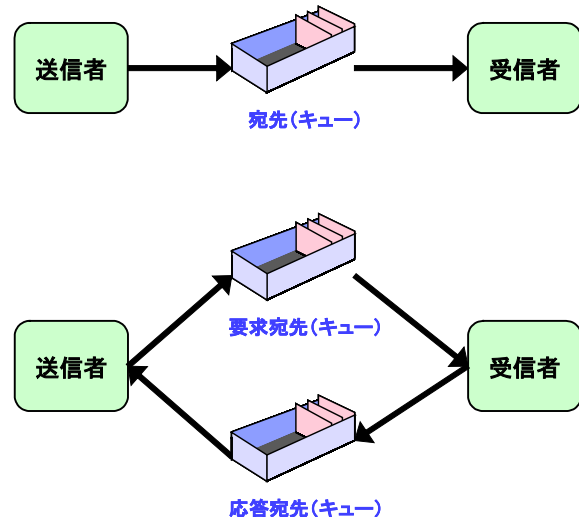
メッセージングを利用したアプリケーションのタイプには、非同期タイプと同期タイプの 2 種類が存在します。各タイプの特徴は以下のとおりです。

- 非同期タイプ

送信者が受信者に対してメッセージを送信することで処理が完了するタイプの処理方法で、一方向型通信と呼ばれることもあります。バッチ処理やメール送信など、実際の処理を後で実施すればよいアプリケーションに活用できます。

- 同期 (要求／応答) タイプ

通常のアプリケーションのように、送信者がメッセージを送信し、その処理結果を受信者から受け取る同期タイプのオンライン処理も可能です。ただし、この同期タイプでは要求宛先と応答宛先は別になります。そのため要求処理と応答処理は関連していないので、まったく別の処理として実行されます。



7.1.2 JMS

JMS (Java Message Service) は、Java EE 仕様におけるメッセージングを実現する API として位置付けられています。アプリケーションは、JMS を使用し MOM 製品と連携することによって、メッセージングシステムを構築することが可能です。代表的な MOM 製品としては、IBM WebSphere MQ があります。

JMS においては、「宛先」があり、送信アプリケーション（プロデューサー）が宛先に対して、「メッセージ」を PUT します。もう一方の受信アプリケーション（コンシューマー）は、処理可能なタイミングでその「メッセージ」を「宛先」から GET し、処理を実行します。

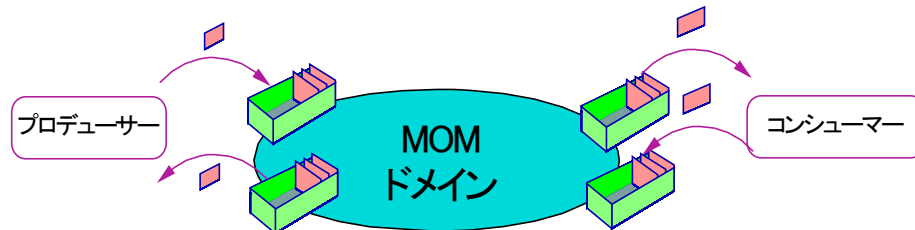


図 7-1 メッセージング概要

JMS は、JMS API と JMS プロバイダー、JMS クライアントから構成されます。

- JMS プロバイダー
JMS のインターフェース、管理、制御の機能を提供するメッセージング・サーバー・コンポーネント
- JMS クライアント

JMS インターフェースを利用して、JMS サーバー・コンポーネントを利用するクライアント・コンポーネント

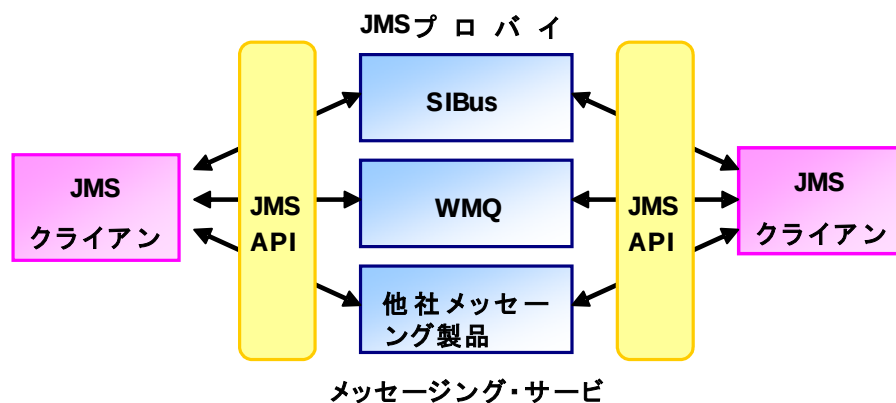


図 7-2 メッセージング・サービスの構成要素

7.1.2.1 JMS のタイプ

JMS には、2 種類の処理モデルがあります。

- Point-to-Point (PTP)

PTP では、宛先をキューと呼びます。メッセージを生成する側をセNDER、メッセージを消費（使用）する側をレシーバーと呼び、セNDERとレシーバーの関係が、1 対 1 となります。PTP はキューをベースにしたモデルで、クライアントは特定のキューにメッセージを送信し、特定のキューからメッセージを受信します。

Point to point (PTP) モデル

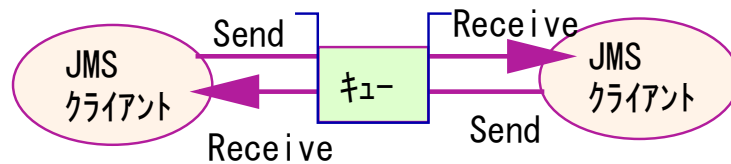


図 7-3 Point-to-Point (PTP) モデル

- Publish/Subscribe (Pub/Sub)

Pub/Sub では、宛先をトピックと呼びます。メッセージを生成する側はパブリッシャー、メッセージを消費する側はサブスクライバーと呼ばれます。あるパブリッシャーがトピックに対してメッセージをパブリッシュします。次にサブスクライブ状態で待っているすべてのクライアントがそのメッセージをサブスクライブします。Pub/Sub の処理モデルは、サブスクライバーの数が不特定であり、パブリッシャーとサブスクライバーの関係が 1 対多の関係にあります。Pub/Sub ドメインは、複数のアプリケーションで、同一のメッセージを共有利用するという要件にあったメッセージング・ドメインです。

Publish/Subscribe (Pub/Sub) モデル

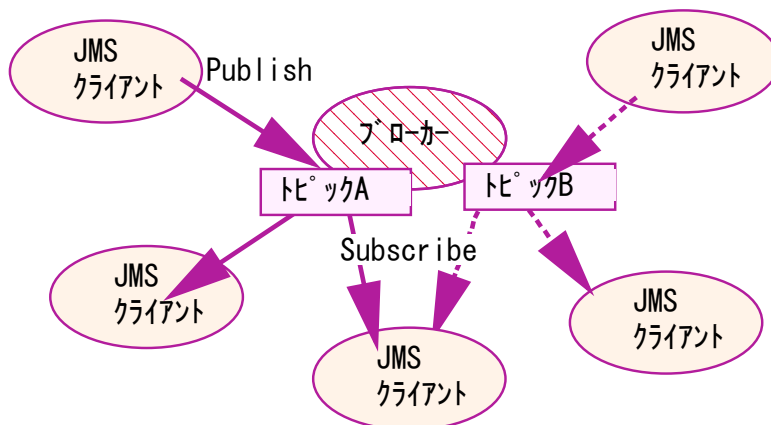


図 7-4 Publish/Subscribe (Pub/Sub) モデル

7.1.2.2 JMS の特徴

- XA 対応

X/Open における XA インターフェースを使用することによって、複数リソースにおける 2 フェーズコミット処理を実現することが可能です。

- 高信頼性

アプリケーションは最寄りの MOM 製品と通信を行った段階で処理が完了します。そのため、MOM 製品内での通信障害等に影響されることが無く動作することが出来ます。

- 高開発生産性

上記の各特徴から、アプリケーションは通信環境上でのエラーを意識せずに開発することができ、シンプルな構造となります。

7.1.2.3 利用可能な JMS プロバイダー

WAS V8.5 は Java EE 6 準拠のアプリケーション・サーバー製品のため、JMS/MDB を稼働させるために必要なフル機能の JMS プロバイダーが内蔵されています。その他にも WAS V8.5 では WAS 以外の JMS プロバイダーも利用可能です。

- SIBus

SIBus の基本機能であるメッセージング・エンジンは、WAS 上で非同期通信を提供する基盤です。

WAS V5 では、WebSphere MQ をベースとした組み込みメッセージング・サービスがデフォルトのメッセージング・サービスとして提供されていました。WAS V6.0 からは、新規に作成された SIBus がデフォルトのメッセージング・エンジンとなっています。この SIBus は、JMS 環境で最高のパフォーマンスを出せるよう設計され Java で実装されています。

SIBus は、クラスタリング構成をとることにより負荷分散および高可用性構成を構築することが可能となります。また、WebSphere MQ 環境とリンクすることができ、WebSphere MQ と連携したメッセージング環境を構築することが可能となります。

SIBus を使用するには、メッセージング・エンジンを構成し、デフォルトの JMS プロバイダーを設定する必要があります。

- WebSphere MQ

WebSphere MQ は、異なる環境間でメッセージングを扱うためのメッセージング専用製品です。WebSphere MQ を使用することにより、異なる OS 間でメッセージングを行うことが可能となります。さらに、クラスター環境を構成することにより負荷分散および高可用性を実現できます。例えば、Linux 上の WAS で JMS を使用した Java EE アプリケーションから、z/OS ネイティブのアプリケーションにメッセージを渡しシステム連携することができます。

WAS V8.5 は、WebSphere MQ V6.0.2 / WebSphere MQ V7.0 / WebSphere MQ V7.5 / WebSphere MQ V6.0.2 for z/OS / WebSphere MQ V7.0 for z/OS / WebSphere MQ V7.1 と接続することが可能です。

【参考】「Using WebSphere Application Server Version 8.5 with WebSphere MQ」

http://pic.dhe.ibm.com/infocenter/prodconn/v1r0m0/index.jsp?topic=/com.ibm.scenarios.wmqwasusing.doc/topics/swg21498708_85.htm

WebSphere MQ を使用するためには、別途 WebSphere MQ 環境を構築し、WAS の JMS プロバイダーを設定する必要があります。詳細は、「7.4.2 WebSphere MQ メッセージング・プロバイダー」を参照して下さい。

- その他の JMS プロバイダー

WAS V8.5 では、上記以外にも複数の JMS プロバイダーがサポートされます。

「V5 デフォルト・メッセージング」JMS プロバイダーを使用し、WAS V5.x で使用していた組み込みメッセージング・サービスも使用することが可能です。ただし、これは WAS V5.x からのマイグレーション時のみ使用可能です。

「汎用」JMS プロバイダーは WebSphere MQ 以外の MOM 製品を利用する場合に使用されます。「汎用」JMS プロバイダーは特定の JMS プロバイダー向けに設定されていないので、ユーザーが使用する MOM 製品にあわせて内容を設定する必要があります。

7.1.3 MDB (Message-Driven Bean)

MDB (Message-Driven Bean) は、EJB 2.0 で追加された JMS API を利用する EJB です。WAS V8.5 では EJB 3.1 を使用することができます。MDB はイベント駆動タイプの処理方式であり、EJB コンテナがメッセージの受信を検知すると、あらかじめ定義された処理が実行されます。この機能を利用

用することで、メッセージングシステムにアクセスする非同期の EJB アプリケーションを容易に開発できます。

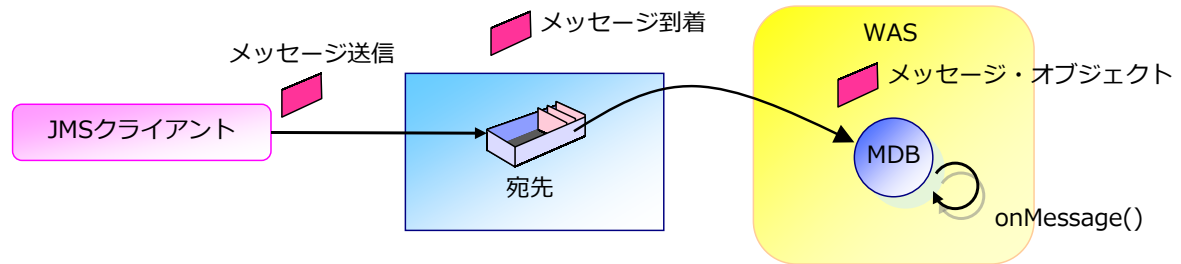


図 7-5 イベント駆動型処理

7.2 SIBus 構成

この節では、SIBus 上に WAS のデフォルトの JMS プロバイダーとして、メッセージング・エンジンを構成する方法を説明します。

7.2.1 トポロジー設定

SIBus を構成し、バス・メンバーとしてアプリケーション・サーバーやクラスターを追加することで、メッセージング・エンジンが構成されます。トポロジーとしては、以下の構成があり、次項以降でそれぞれの詳細について説明します。

- 基本トポロジー
- 高可用性構成
- パーティション構成

7.2.1.1 基本トポロジー

概要

単体サーバーでメッセージング・エンジンを構成するが、基本トポロジーです。1 つのメッセージング・エンジンのみのため、サーバー障害時の可用性を実現することができません。

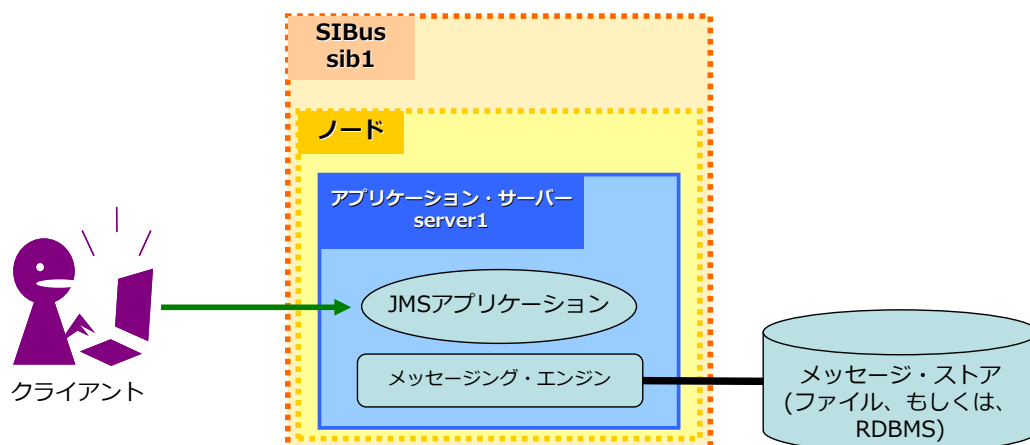


図 7-6 メッセージング・エンジンの基本トポロジー

メッセージング・エンジンは構成情報やパーシスタント・メッセージを保持する為に、メッセージ・ストアを利用します。WAS V6.0 では、メッセージ・ストアとしてデータベースのみ使用可能でしたが、V6.1以降からフラットファイルも指定可能になりました。データベースのメッセージ・ストアを「データ・ストア」と呼び、フラットファイルのメッセージ・ストアを「ファイル・ストア」と呼びます。

構成方法

図 7-6に示した基本トポロジは、以下の手順で設定します。

- 1). データ・ソースを作成する (メッセージ・ストアにデータベースを使用する場合のみ)
- 2). バスを作成する
- 3). バス・メンバーを登録し、メッセージ・ストアの設定をする

SIBus (sib1) にアプリケーション・サーバー (server1) を、バス・メンバーとして追加すると、メッセージング・エンジン (MQ の Queue Managerに相当) が自動で作成されます。メッセージング・エンジンは構成情報やパーシスタント・メッセージを保持するために、メッセージ・ストアを利用します。メッセージ・ストアにはフラットファイル、あるいはデータベースを指定出来ます。データベースの場合には JDBC で接続し、一つのメッセージング・エンジンはデータベース内の一つのスキーマに関連付けられます。

アプリケーションは、SIBus 内に定義された宛先に対してメッセージの send や receive を行います。宛先は JMS 1.1 の Destination に相当するもので、実体がトピック・スペースであってもキューであっても操作上は区別しません。

具体的な設定内容を以下に記載します。

1) データ・ソースを作成する (メッセージ・ストアにデータベースを使用する場合のみ)

メッセージング・エンジンでデータ・ストアを利用するために、データ・ソースを定義します。

JDBC の設定方法の詳細については、「2-9 データベース接続設定」を参照してください。

2) バスを作成する

管理コンソールのメニューから、[サービス統合]→[バス]を選択し、バスを新規作成します。バス作成時および作成後にバス・セキュリティを有効にすることで、バス・リソースに対する認証やロールベースのアクセス制御を行うことが可能です。

設定は、バス・セキュリティ・ウィザードを使用します。前提条件として管理セキュリティを有効にする必要があり、その他オプションとして SSL/NonSSL、バス・セキュリティ・ドメインを設定します。

表 7-1 バスの設定

名前	値	概要
名前	sib1	SIBus の名前です。
UUID		管理上の目的でシステムによってこのバスに割り当てられる汎用固有 ID。
説明		管理上の目的で使用されるバスの説明 (オプション)。
エンジン間トランスポート・チェーン		同一 SIBus 内のメッセージング・エンジン間で利用するトランスポート・チェーン名。デフォルトでは InboundBasicMessaging です。
メッセージを破棄する	(チェックなし)	キュー・ポイントなどが削除された時の処理。チェックした場合はメッセージを削除、チェックしない場合はシステム例外宛先に保持します。

構成の再ロードを使用可能にする	(チェックあり)	再起動しないで設定変更を反映するかどうか。
デフォルト・メッセージング・エンジンのメッセージの高しきい値	50000	キュー・ポイントなどの閾値です。メッセージング・エンジンを追加するとこの値がデフォルト値になります。
使用可能なブートストラップ・メンバーの範囲を以下に制限する	サービス統合バス・サービスが使用可能なセルの全メンバー	使用可能なブートストラップ・メンバーの範囲制限。

3) バス・メンバーを登録し、メッセージ・ストアの設定をする

i) アプリケーション・サーバーをバス・メンバーとして登録する

作成したバスの設定画面を開き、[トポロジー:バス・メンバー]を選択して、遷移した画面で[追加]ボタンを選択します。

バス・メンバーはウィザード形式で追加出来ます。バス・メンバーの選択画面では、バス・メンバーに追加するアプリケーション・サーバー（今回は、アプリケーション・サーバー server1）を選択し、次へ進みます。

ii) メッセージ・ストアを設定する

バス・メンバー追加ウィザードで、アプリケーション・サーバーをバス・メンバーとして登録すると、メッセージ・ストアのタイプの選択の画面へ移ります。メッセージ・ストアにファイルを使用する場合は、「ファイル・ストア」を、データベースを使用する場合は、「データ・ストア」を選択し、次へと進みます。

● 「ファイル・ストア」の場合

ログ・ファイル、ストア・ファイル（永続ストアと一時ストア）の3つのファイルのサイズとディレクトリー・パスを指定します。デフォルトでは以下の値となっていますが、サイジングして値を指定してください。

表 7-2 ファイル・ストアの設定

	名前	値	概要
ログ	ログ・サイズ	100 (MB)	ログ・ファイルのサイズ (MB)。固定サイズ。
	デフォルトのログ・ディレクトリー・パス	(チェックあり)	デフォルトのログ・ファイル・ディレクトリー・パスを使用するかどうか。 デフォルトでは、 <code>\${USER_INSTALL_ROOT}/filestores/com.ibm.ws.sib/<メッセージ・エンジン名>-<メッセージ・エンジン UUID>/log</code>
	ログ・ディレクトリー・パス		ログ・ファイル・ディレクトリーのパス名。明示的に指定したい場合に使用。
ストア	永続ストアおよび一時ストアに対して同一の設定	(チェックあり)	選択されている場合、永続ストアと同じ設定が一時ストアに適用。
永続	最小永続ストア・サイズ	200 (MB)	永続ストア・ファイルの最小サイズ (MB)。

ストア	永続ストア・サイズ無制限	(チェックなし)	永続ストア・ファイルのサイズが無制限かどうか。
	最大永続ストア・サイズ	500 (MB)	永続ストア・ファイルの最大サイズ(MB)。
	デフォルトの永続ストア・ディレクトリー・パス	(チェックあり)	デフォルトの永続ストア・ファイル・ディレクトリー・パスを使用するかどうか。 デフォルトでは、 <code>\${USER_INSTALL_ROOT}/filestores/com.ibm.ws.sib/<メッセージ・エンジン名>-<メッセージ・エンジン UUID>/store</code>
	永続ストア・ディレクトリー・パス		永続ストア・ファイル・ディレクトリーのパス名。明示的に使用したい場合に使用。
一時ストア	最小一時ストア・サイズ	200 (MB)	一時ストア・ファイルの最小サイズ(MB)。
	一時ストア・サイズ無制限	(チェックなし)	一時ストア・ファイルのサイズが無制限かどうか。
	最大一時ストア・サイズ	500 (MB)	一時ストア・ファイルの最大サイズ(MB)。
	デフォルトの一時ストア・ディレクトリー・パス	(チェックあり)	デフォルトの一時ストア・ファイル・ディレクトリー・パスを使用するかどうか。 デフォルトでは、 <code>\${USER_INSTALL_ROOT}/filestores/com.ibm.ws.sib/<メッセージ・エンジン名>-<メッセージ・エンジン UUID>/store</code>
	一時ストア・ディレクトリー・パス		一時ストア・ファイル・ディレクトリーのパス名。明示的に使用したい場合に使用。

- 「データ・ストア」の場合
データ・ソース JNDI 名を入力します。なお一つのデータ・ソースを複数のメッセージング・エンジンで利用するには、それぞれのメッセージング・エンジンで異なるスキーマ名を指定します。

バス・メンバーの登録が完了すると、メッセージング・エンジンが自動で作成されます。これと同時に、システム例外宛先 (System Exception Destination、WebSphere MQ の Dead Letter Queue に相当) が自動で作成されます。

7.2.1.2 高可用性構成

概要

SIBus のバス・メンバーとしてクラスターを追加することで、クラスターメンバーにメッセージ・エンジンが構成されます。この状態でクラスターを起動することで、クラスター内にメッセージング・エンジンが1つ起動され、他のクラスターメンバーのメッセージング・エンジンが待機状態になる構成が、高可用性構成です。この構成の特徴は、メッセージング・エンジンが起動しているサーバーにプロセス障害やネットワーク障害が発生しても、HA マネージャーによりメッセージング・エンジンがフェイルオーバーされることで、サービスの提供を継続できることにあります。また、稼働系と待機系のメッセージング・エンジンは、同じメッセージ・ストアを利用することで、構成情報やデータの引継ぎが行われます。

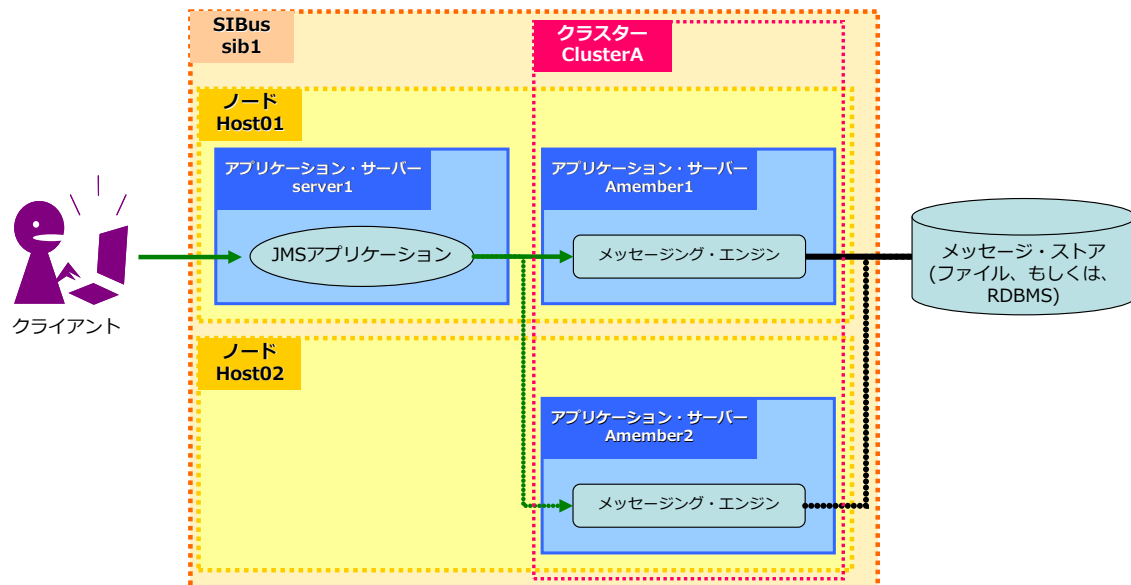


図 7-7 メッセージング・エンジンの高可用性構成

構成方法

図 7-7に示した高可用性構成は、以下の手順で設定します。

- 1). クラスターを作成する
 - 2). データ・ソースを作成する（メッセージ・ストアにデータベースを使用する場合のみ）
 - 3). クラスターをバス・メンバーとして登録し、メッセージ・ストアの設定をする
 - 4). メッセージング・エンジンの稼働状況（HA 機能稼働状況）を確認する
- 具体的な設定内容を以下に記載します。

1) クラスターを作成する

SIBus に追加するクラスターを作成します。作成手順は、「2-7 サーバー管理」のクラスター作成手順を参照してください。ここでは、ClusterA というクラスターを作成したとします。

2) データ・ソースを作成する（メッセージ・ストアにデータベースを使用する場合のみ）

メッセージ・ストアにデータベースを使用する場合は、データ・ソースを定義します。JDBC の設定方法の詳細については、「2-9 データベース接続設定」を参照してください。

3) クラスターをバス・メンバーとして登録し、メッセージ・ストアの設定をする

クラスターをバス・メンバーとして追加すると、デフォルトでは HA 機能が有効になります。バス・メンバーへの登録手順は、以下のとおりです。

- i) バス・メンバーにクラスターを追加する

管理コンソールのメニューから、[サービス統合]→[パス]を選択して SIB1 を開き、[追加プロパティ:バス・メンバー]を開きます。遷移した画面で[追加]ボタンを選択します。バス・メンバーの選択画面では、バス・メンバーに追加するクラスターを選択し、次に進みます。

ii) メッセージング・エンジン・ポリシー・アシスタンスを設定する

ポリシー・タイプとして「高可用性」を選択し、次に進みます。メッセージング・エンジン・ポリシー・アシスタンスについては、「7.2.1.4メッセージング・エンジン・ポリシー・アシスタンス」を参照してください。

iii) メッセージ・ストアを設定する

メッセージ・ストアのタイプの選択の画面へ移ります。メッセージ・ストアにファイルを使用する場合は、「ファイル・ストア」を、データベースを使用する場合は、「データ・ストア」を選択し、次へと進みます。

• 「ファイル・ストア」の場合

ログ・ファイル、ストア・ファイル(永続ストアと一時ストア)の 3 つのファイルのサイズとディレクトリー・パスを指定します。

クラスターの場合は、メッセージング・エンジンが起動するクラスターメンバーがログ・ファイル、ストア・ファイルにアクセスできるように、ディレクトリー・パスを指定する必要があります。例えば、複数ノードで稼働するクラスターメンバー間でファイル・ストアの引継ぎを行う場合は、ファイルを NAS などの共有ファイル・システムに配置し、各クラスターメンバーがファイルにアクセスできるように構成する必要があります。デフォルトでは以下の値となっていますが、サイジングして値を指定してください。

表 7-3 ファイル・ストアの設定

	名前	値	概要
ログ	ログ・サイズ	100 (MB)	ログ・ファイルのサイズ (MB)。固定サイズ。
	ログ・ディレクトリー・パス	/share/cluster A.000- SIB1/log	ログ・ファイル・ディレクトリーのパス名。
ストア	永続ストアおよび一時ストアに対して同一の設定	(チェックあり)	選択されている場合、永続ストアと同じ設定が一時ストアに適用。
永続 ストア	最小永続ストア・サイズ	200 (MB)	永続ストア・ファイルの最小サイズ(MB)。
	永続ストア・サイズ無制限	(チェックなし)	永続ストア・ファイルのサイズが無制限かどうか。
	最大永続ストア・サイズ	500 (MB)	永続ストア・ファイルの最大サイズ(MB)。
	永続ストア・ディレクトリー・パス	/share/cluster A.000- SIB1/store	永続ストア・ファイル・ディレクトリーのパス名。
一時 ストア	最小一時ストア・サイズ	200 (MB)	一時ストア・ファイルの最小サイズ(MB)。
	一時ストア・サイズ無制限	(チェックなし)	一時ストア・ファイルのサイズが無制限かどうか。
	最大一時ストア・サイズ	500 (MB)	一時ストア・ファイルの最大サイズ(MB)。

	一時ストア・ディレクトリー・パス	/share/cluster A.000- SIB1/store	一時ストア・ファイル・ディレクトリーのパス名。
--	------------------	--	-------------------------

- 「データ・ストア」の場合
データ・ソース JNDI 名を入力します。なお一つのデータ・ソースを複数のメッセージング・エンジンで利用するには、それぞれのメッセージング・エンジンで異なるスキーマ名を指定します。

4) メッセージング・エンジンの稼働状況を確認する

クラスターを起動し、クラスター内のどのサーバーでメッセージング・エンジンが起動しているかを確認します。

- メニューで[サーバー]→[コア・グループ]→[コア・グループ設定]を選択し、コア・グループの一覧から DefaultCoreGroup を選択します。
- コア・グループのランタイムを表示します
「ランタイム」タブをクリックし、表示される画面から、「グループの表示」ボタンを選択します。
- 設定したメッセージング・エンジンに適用されているポリシーを確認します
「WSAF_SIB_MESSAGING_ENGINE=clusterA.000-sib1」を含む行を確認します。ポリシーとして、「ClusterA.000-SIB1-683705EC10021211Policy」が適用されていることが分かります。なお、ポリシーの中に含まれている「683705EC10021211」の部分の文字列ですが、これはメッセージング・エンジンの UUID となります。

選択	高可用性グループ	クォーラム	ポリシー	状況
管理できるリソース:				
☐	IBM_hc=ClusterA,WSAF_SIB_BUS=SIB1,WSAF_SIB_MESSAGING_ENGINE=ClusterA.000-SIB1.type=WSAF_SIB	使用不可	ClusterA.000-SIB1-683705EC10021211Policy	⊙

図 7-8 高可用性構成のメッセージング・エンジンに適用されているポリシー

DefaultCoreGroup の画面の[追加プロパティ:ポリシー]を選択すると、「ClusterA.000-SIB1-683705EC10021211Policy」は、「N ポリシーの中の 1 つ」というポリシー・タイプとなっています。

このポリシー・タイプは、HA グループのメンバーの中から、1 つのサーバーでサービスを稼働させるポリシーです。サービスが稼働しているサーバーが停止した場合は、クラスター内のほかのアプリケーション・サーバーにサービスが引き継がれ、常に 1 つのサービスが稼働します。このポリシーにより、メッセージング・エンジンの高可用性が保たれています。ポリシー・タイプの詳細については、「3-5-2-4 コア・グループ・ポリシー」を参照してください。

選択	名前	説明	ポリシー・タイプ	一致基準
管理できるリソース:				
☐	ClusterA.000-SIB1-683705EC10021211Policy		N ポリシー中の 1 つ	type=WSAF_SIB, WSAF_SIB_MESSAGING_ENGINE=ClusterA.000-SIB1

図 7-9 「ClusterA.000-SIB1-683705EC10021211Policy」ポリシー

iv) メッセージング・エンジンが稼働しているサーバーを確認します

再度、ii) を実施し、「WSAF_SIB_MESSAGING_ENGINE=clusterA.000-sib1」を含む行を選択します。状況の列を確認することで、メッセージング・エンジンが稼働しているサーバーを確認できます。図 7-10 の例では、Amember2 でメッセージング・エンジンが稼働しています。

選択	名前	ノード	バージョン	状況
管理できるリソース:				
☐	Amember1	mgwas01Node01	8.5.5.0	
☐	Amember2	mgwas02Node01	8.5.5.0	

図 7-10 メッセージング・エンジンが稼働しているサーバー

7.2.1.3 パーティション構成

概要

「7.2.1.2 高可用性構成」に記載した構成では、クラスター内で稼働するメッセージング・エンジンが 1 つしかないため、メッセージの処理が 1 つのメッセージング・エンジンに集中します。

「パーティション構成」は、クラスター内でメッセージング・エンジンを複数稼働させることで、高可用性に加えて、メッセージ処理を分散できる構成です。この構成において、メッセージング・エンジンが起動するサーバーに障害が発生した場合、メッセージング・エンジンごとにフェイルオーバーとメッセージの引き継ぎが行われます。

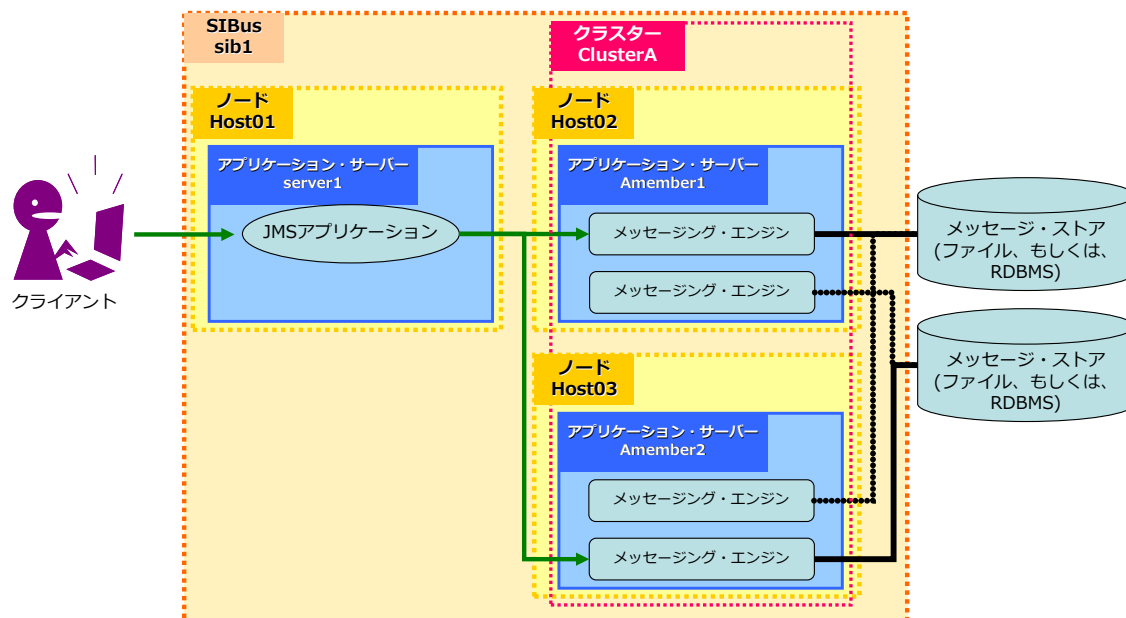


図 7-11 メッセージング・エンジンのパーティション構成

パーティション構成の注意点

パーティション構成を有効にするためには、JMS アプリケーションがメッセージング・エンジンを同じ検索範囲で複数見つけられることが必要となります。複数のメッセージング・エンジンを構成していても、JMS アプリケーションが検索した際に、1つのメッセージング・エンジンしか見つからなかった場合には、パーティション構成が有効になりませんのでご注意ください。

メッセージング・エンジンの検索範囲

JMS アプリケーションが使用するメッセージング・エンジンは、検索範囲内で最初に見つかったメッセージング・エンジンになります。



図 7-12 メッセージング・エンジンの検索順序

図 7-12の場合、Host01 の ClusterA 内の Amember1 にある JMS アプリケーションは、メッセージング・エンジンを次の順序で検索します。

- ① JMS アプリケーションと同じサーバー上で稼動するメッセージング・エンジン
- ② 同じクラスター内で稼動するメッセージング・エンジン
- ③ JMS アプリケーションと同じノード上で稼動する、同一バス・メンバーのクラスターメンバー以外のサーバーのメッセージング・エンジン
- ④ 同一バス・メンバー上で稼動するメッセージング・エンジン

ただし、JMS 接続ファクトリーの設定にある、「接続の接近性」を設定することでどこまでを検索範囲とするかを制限することも可能です。設定は、「バス」、「クラスター」、「ホスト」、「サーバー」で、各設定により下記の範囲が検索範囲になります。

バス	①～④の範囲 (デフォルト設定)
クラスター	①②の範囲
ホスト	①～③の範囲
サーバー	①

パーティション構成でのメッセージの割り振り

パーティション構成として、以下に示す 4 つのパターンが考えられます。各パターンにおいて、JMS アプリケーションからの要求がどのように割り振りされるかについて説明します。なお、MDB アプリケーションの場合は、最初にアクセスしたメッセージング・エンジンの宛先のみを使用します。

1) JMS アプリケーションが稼動するクラスターとメッセージング・エンジンが稼動するクラスターが同一クラスター

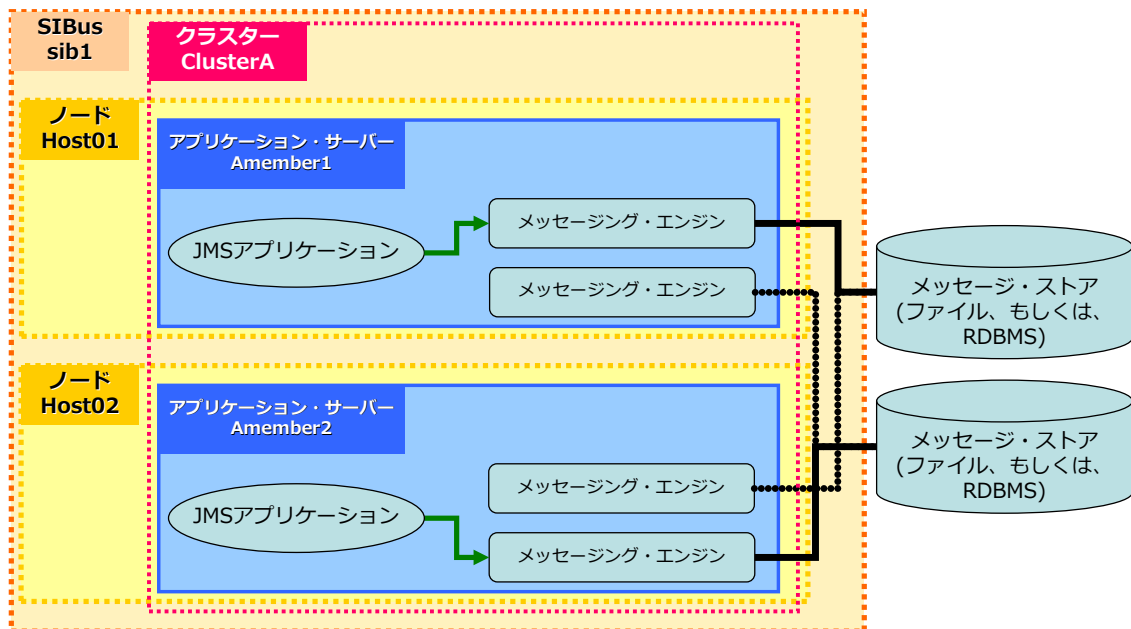


図 7-13 JMS アプリケーションとメッセージング・エンジンが同一クラスターの場合

この構成の場合、JMS アプリケーションは同じサーバーで稼動しているメッセージング・エンジンを発見しますので、そのメッセージング・エンジンのみを使用します。

2) JMS アプリケーションが稼動するクラスターとメッセージング・エンジンが稼動するクラスターが別クラスター

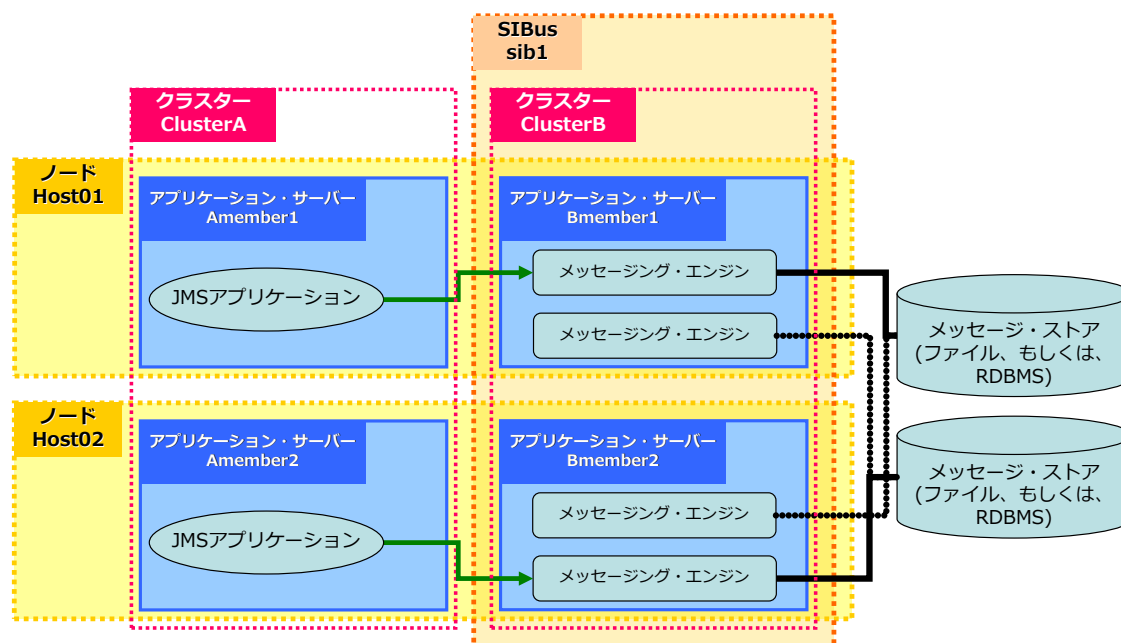


図 7-14 JMS アプリケーションとメッセージング・エンジンが別クラスターの場合

この構成の場合、JMS アプリケーションは自分が稼動しているアプリケーション・サーバー上にメッセージング・エンジンを見つけることができませんので、2 番目の検索範囲である同一クラスターを検索します。同一クラスターでも見つけることができないので、3 番目の検索範囲である同一ホストを検索します。そこにメッセージング・エンジンを見つけることができるので、そのメッセージング・エンジンのみを使用します。

3) JMS アプリケーションが稼動するクラスターとメッセージング・エンジンが稼動するクラスターが別ノード、かつ、同一 SIBus に所属する

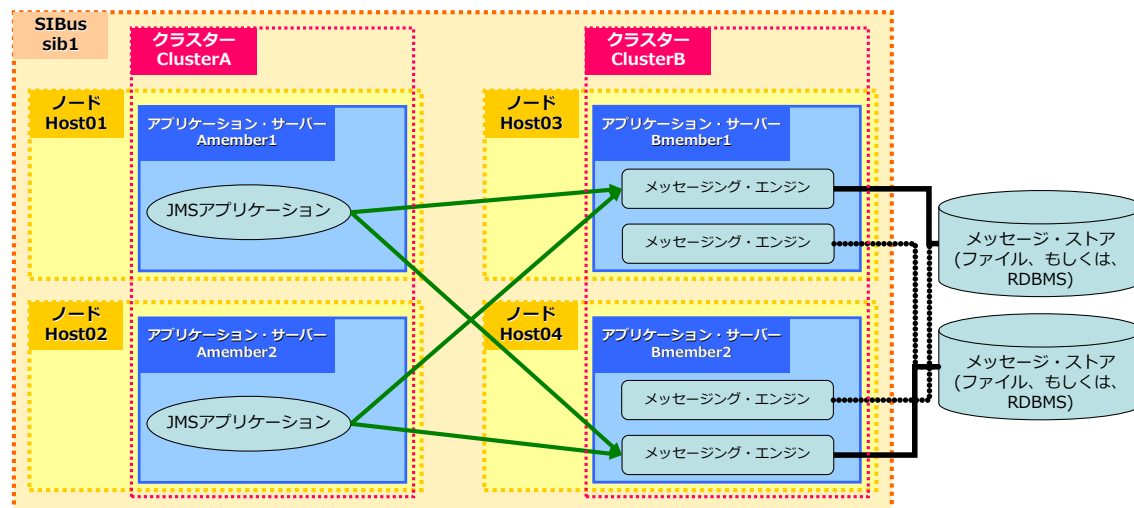


図 7-15 JMS アプリケーションとメッセージング・エンジンが別クラスターかつ別ノードの場合

この構成の場合、JMS アプリケーションは自分が起動しているアプリケーション・サーバー上にメッセージング・エンジンが見つからないため、同じクラスター内を検索します。それでも見つからないため、他のノードにあるメッセージング・エンジンを検索し、同じ検索範囲にメッセージング・エンジンを 2 つ見つけます。そのメッセージング・エンジンに対してラウンドロビンでメッセージを PUT/GET します。

4) JMS アプリケーションが稼動するクラスターとメッセージング・エンジンが稼動するクラスターが別ノード、かつ、同一 SIBus に所属しない

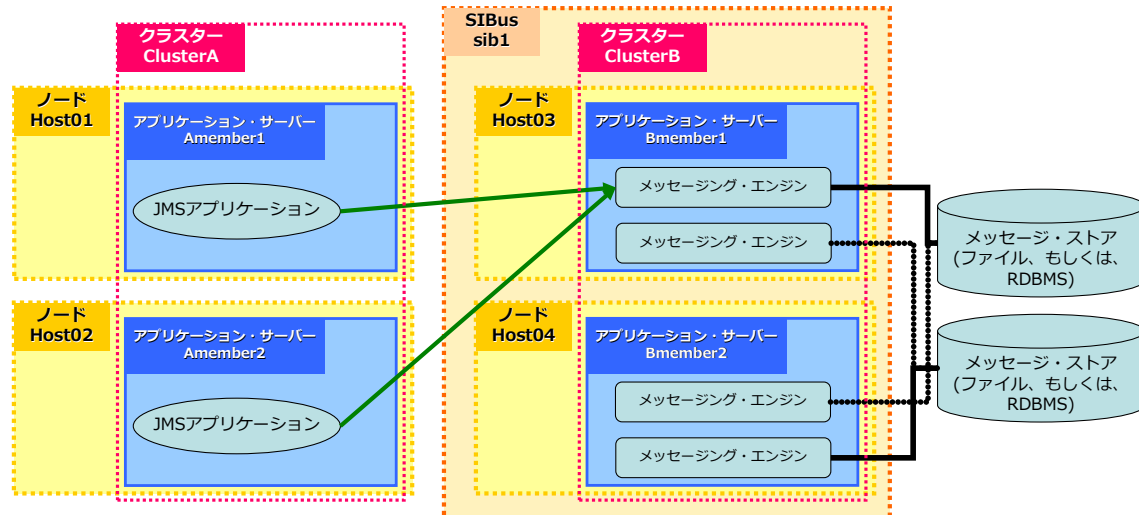


図 7-16 JMS アプリケーションとメッセージング・エンジンが同一の SIBus に所属しない場合

この構成の場合、JMS アプリケーションは自分が起動しているアプリケーション・サーバー上にメッセージング・エンジンが見つからないため、同じクラスター内を検索します。それでも見つからないため、他のノードにあるメッセージング・エンジンを検索し、同じ検索範囲にメッセージング・エンジンを 2 つ見つけます。しかし、SIBus メンバーに属していないため、最初に接続したメッセージング・エンジンに対してのみメッセージを PUT/GET します。

構成方法

図 7-11に示したパーティション構成は、以下の手順で設定します。1 つのバス・メンバー内に 2 つのメッセージング・エンジン「ClusterA.000-sib1」、「ClusterA.001-sib1」を構成し、それぞれの稼動系を「Amember01」と「Amember02」としてパーティショニングします。

- 1). クラスターを作成する
 - 2). データ・ソースを作成する（メッセージ・ストアにデータベースを使用する場合のみ）
 - 3). クラスターをバス・メンバーとして登録し、メッセージ・ストアの設定をする
 - 4). メッセージング・エンジンの稼働状況を確認する
- 具体的な設定内容を以下に記載します。

1) クラスターを作成する

SIBus に追加するクラスターを作成します。作成手順は、「2-7-7-3 静的クラスターの作成」を参照してください。ここでは、ClusterA というクラスターを作成したとします。

2) データ・ソースを作成する（メッセージ・ストアにデータベースを使用する場合のみ）

メッセージ・ストアにデータベースを使用する場合は、データ・ソースを定義します。1つのメッセージング・エンジンには 1つのメッセージ・ストアが必要ですから、2つのメッセージ・エンジンを構成するには

2 つのデータ・ソースを定義する必要があります。JDBC の設定方法の詳細については、「2-9 データベース接続設定」を参照してください。

3) クラスタをバス・メンバーとして登録し、メッセージ・ストアの設定をする

クラスタをバス・メンバーとして追加すると、デフォルトでは HA 機能が有効になります。バス・メンバーへの登録手順は、以下のとおりです。

i) バス・メンバーにクラスタを追加する

管理コンソールのメニューから、[サービス統合]→[バス]を選択して SIB1 を開き、[追加プロパティ:バス・メンバー]を開きます。遷移した画面で[追加]ボタンを選択します。バス・メンバーの選択画面では、バス・メンバーに追加するクラスタを選択し、次に進みます。

ii) メッセージング・エンジン・ポリシー・アシスタンスを設定する

ポリシー・タイプとして「高可用性を備えたスケラビリティ」を選択し、次に進みます。メッセージング・エンジン・ポリシー・アシスタンスについては、「7.2.1.4メッセージング・エンジン・ポリシー・アシスタンス」を参照してください。

iii) メッセージ・ストアを設定する

メッセージ・ストアのタイプの選択の画面へ移ります。メッセージ・ストアにファイルを使用する場合は、「ファイル・ストア」を、データベースを使用する場合は、「データ・ストア」を選択し、次へと進みます。

● 「ファイル・ストア」の場合

ログ・ファイル、ストア・ファイル(永続ストアと一時ストア)の 3 つのファイルのサイズとディレクトリー・パスを指定します。

クラスタの場合は、メッセージング・エンジンが起動するクラスタメンバーがログ・ファイル、ストア・ファイルにアクセスできるように、ディレクトリー・パスを指定する必要があります。例えば、複数ノードで稼働するクラスタメンバー間でファイル・ストアの引継ぎを行う場合は、ファイルを NAS などの共有ファイル・システムに配置し、各クラスタメンバーがファイルにアクセスできるように構成する必要があります。

また、1 つのメッセージング・エンジンには 1 つのメッセージ・ストアが必要となるため、今回の例では、2 つのファイル・ストアが必要となります。デフォルトでは以下の値となっていますが、サイジングして値を指定してください。

表 7-4 ファイル・ストアの設定 (ClusterA.000-sib1 用)

	名前	値	概要
ログ	ログ・サイズ	100 (MB)	ログ・ファイルのサイズ (MB)。固定サイズ。
	ログ・ディレクトリー・パス	/share/cluster A.000-SIB1/log	ログ・ファイル・ディレクトリーのパス名。
ストア	永続ストアおよび一時ストアに対して同一の設定	(チェックあり)	選択されている場合、永続ストアと同じ設定が一時ストアに適用。
永続ストア	最小永続ストア・サイズ	200 (MB)	永続ストア・ファイルの最小サイズ (MB)。
	永続ストア・サイズ無制限	(チェックなし)	永続ストア・ファイルのサイズが無制限かどうか。
	最大永続ストア・サイズ	500 (MB)	永続ストア・ファイルの最大サイズ (MB)。

	永続ストア・ディレクトリー・パス	/share/cluster A.000- SIB1/store	永続ストア・ファイル・ディレクトリーのパス名。
一時 ストア	最小一時ストア・サイズ	200 (MB)	一時ストア・ファイルの最小サイズ(MB)。
	一時ストア・サイズ無制限	(チェックなし)	一時ストア・ファイルのサイズが無制限かどうか。
	最大一時ストア・サイズ	500 (MB)	一時ストア・ファイルの最大サイズ(MB)。
	一時ストア・ディレクトリー・パス	/share/cluster A.000- SIB1/store	一時ストア・ファイル・ディレクトリーのパス名。

表 7-5 ファイル・ストアの設定 (ClusterA.001-sib1 用)

	名前	値	概要
ログ	ログ・サイズ	100 (MB)	ログ・ファイルのサイズ (MB)。固定サイズ。
	ログ・ディレクトリー・パス	/share/cluster A.001- SIB1/log	ログ・ファイル・ディレクトリーのパス名。
ストア	永続ストアおよび一時ストアに対して同一の設定	(チェックあり)	選択されている場合、永続ストアと同じ設定が一時ストアに適用。
永続 ストア	最小永続ストア・サイズ	200 (MB)	永続ストア・ファイルの最小サイズ(MB)。
	永続ストア・サイズ無制限	(チェックなし)	永続ストア・ファイルのサイズが無制限かどうか。
	最大永続ストア・サイズ	500 (MB)	永続ストア・ファイルの最大サイズ(MB)。
	永続ストア・ディレクトリー・パス	/share/cluster A.001- SIB1/store	永続ストア・ファイル・ディレクトリーのパス名。
一時 ストア	最小一時ストア・サイズ	200 (MB)	一時ストア・ファイルの最小サイズ(MB)。
	一時ストア・サイズ無制限	(チェックなし)	一時ストア・ファイルのサイズが無制限かどうか。
	最大一時ストア・サイズ	500 (MB)	一時ストア・ファイルの最大サイズ(MB)。
	一時ストア・ディレクトリー・パス	/share/cluster A.001- SIB1/store	一時ストア・ファイル・ディレクトリーのパス名。

- 「データ・ストア」の場合
データ・ソース JNDI 名を入力します。なお一つのデータ・ソースを複数のメッセージング・エンジンで利用するには、それぞれのメッセージング・エンジンで異なるスキーマ名を指定します。

4) メッセージング・エンジンの稼働状況を確認する

クラスターを起動し、クラスター内のどのサーバーでメッセージング・エンジンが起動しているかを確認します。

- i) メニューで[サーバー]→[コア・グループ]→[コア・グループ設定]を選択し、コア・グループの一覧から DefaultCoreGroup を選択します。
- ii) コア・グループのランタイムを表示します
「ランタイム」タブをクリックし、表示される画面から、「グループの表示」ボタンを選択します。
- iii) 設定したメッセージング・エンジンに適用されているポリシーを確認します
以下の2つを含む行を確認します。
 - WSAF_SIB_MESSAGING_ENGINE=clusterA.000-sib1
 - WSAF_SIB_MESSAGING_ENGINE=clusterA.001-sib1
 ポリシーとして、それぞれ以下が適用されていることが分かります。
 - ClusterA.000-SIB1-44EFA291A1C446AEPolicy
 - ClusterA.001-SIB1-4C6B329319CD4BFBPolicy

選択	名前	説明	ポリシー・タイプ	一致基準
管理できるリソース:				
☐	ClusterA.000-SIB1-44EFA291A1C446AEPolicy		N ポリシーの中の 1 つ	type=WSAF_SIB, WSAF_SIB_MESSAGING_ENGINE=ClusterA.000-SIB1
☐	ClusterA.001-SIB1-4C6B329319CD4BFBPolicy		N ポリシーの中の 1 つ	type=WSAF_SIB, WSAF_SIB_MESSAGING_ENGINE=ClusterA.001-SIB1

図 7-17 パーティション構成のメッセージング・エンジンに適用されているポリシー

Default Core Group の画面の[追加プロパティ:ポリシー]を選択すると、先ほど図 7-17で確認したポリシーは、ともに「N ポリシーの中の 1 つ」というポリシー・タイプとなっています。

このポリシー・タイプは、HA グループのメンバーの中から、1 つのサーバーでサービスを稼働させるポリシーです。サービスが稼働しているサーバーが停止した場合は、クラスター内のほかのアプリケーション・サーバーにサービスが引き継がれ、常に 1 つのサービスが稼働します。このポリシーにより、メッセージング・エンジンの高可用性が保たれています。ポリシー・タイプの詳細については、「3-5-2-4 コア・グループ・ポリシー」を参照してください。

選択	高可用性グループ	クォーラム	ポリシー	状況
管理できるリソース:				
☐	IBM_hc=ClusterA.WSAF_SIB_BUS=SIB1.WSAF_SIB_MESSAGING_ENGINE=ClusterA.000-SIB1.type=WSAF_SIB	使用不可	ClusterA.000-SIB1-44EFA291A1C446AEPolicy	⊙
☐	IBM_hc=ClusterA.WSAF_SIB_BUS=SIB1.WSAF_SIB_MESSAGING_ENGINE=ClusterA.001-SIB1.type=WSAF_SIB	使用不可	ClusterA.001-SIB1-4C6B329319CD4BFBPolicy	⊙

図 7-18 適用されているポリシー・タイプ

- iv) メッセージング・エンジンが稼働しているサーバーを確認します
再度、ii) を実施し、「WSAF_SIB_MESSAGING_ENGINE=clusterA.000-sib1」を含む行を選択します。状況の列を確認することで、メッセージング・エンジンが稼働しているサーバーを確認できます。図 7-19の例では、Amember2 でメッセージング・エンジンが稼働しています。

選択	名前	ノード	バージョン	状況
管理できるリソース:				
☐	Amember1	mgwas01Node01	8.5.5.0	⏏
☐	Amember2	mgwas02Node01	8.5.5.0	

図 7-19 clusterA.000-sib1 のメッセージング・エンジンが稼動しているサーバー

同様に、「WSAF_SIB_MESSAGING_ENGINE=clusterA.001-sib1」のメッセージング・エンジンが稼動しているサーバーを確認できます。図 7-20の例では、Amember2 でメッセージング・エンジンが稼動しています。

選択	名前	ノード	バージョン	状況
管理できるリソース:				
☐	Amember1	mgwas01Node01	8.5.5.0	
☐	Amember2	mgwas02Node01	8.5.5.0	

図 7-20 clusterA.001-sib1 のメッセージング・エンジンが稼動しているサーバー

7.2.1.4 メッセージング・エンジン・ポリシー・アシスタンス

WAS V7.0 より、メッセージング・エンジンを構成する際のアドバイザー機能を提供するメッセージング・エンジン・ポリシー・アシスタンスが追加されました。この機能を利用すると、ウィザード形式で、「高可用性」、「スケーラビリティ」、「高可用性を備えたスケーラビリティ」の 3 つのポリシー・タイプから 1 つを選択して、そのポリシー・タイプにしたがったメッセージング・エンジンとポリシーを作成することが可能です。さらに、SIBus メンバーとしてクラスターを追加すると、ポリシーごとにメッセージング・エンジンのトポロジーを選択することができます。また、カスタムを選択すると、このアドバイザー機能を利用できなくなりますが、自由にポリシーを設定することができます。

このメッセージング・エンジン・ポリシー・アシスタンス機能はデフォルトで有効になっていますが、無効にすることも可能です。無効にする場合は、管理コンソールより、[サービス統合]→[バス]→[バス名]→[トポロジー:バス・メンバー] からリソースを選択し、[メッセージング・エンジン・ポリシー・アシスタンスの使用可能化]のチェックボックスをOFF に設定して下さい。

メッセージング・エンジン・ポリシー・アシスタンスで選択できるポリシー・タイプ

1) 高可用性ポリシー

「7.2.1.2高可用性構成」で記載した構成です。このポリシーは、メッセージング・エンジンのフェイルオーバーを実現します。クラスター内の 1 つのアプリケーション・サーバーにてメッセージング・エンジンが稼動し、フェイルオーバー先のメッセージング・エンジンと同じメッセージ・ストアを利用します。

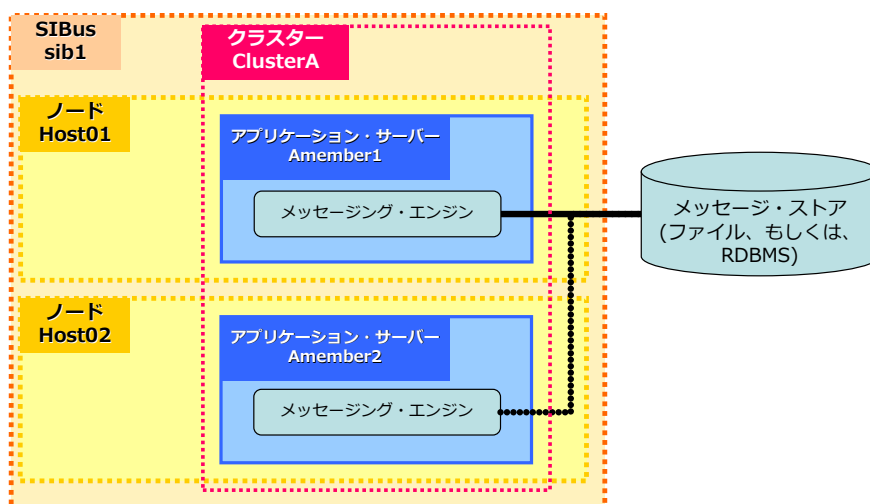


図 7-21 高可用性ポリシー

2) スケーラビリティ・ポリシー

このポリシーは、メッセージング・エンジンの負荷分散を実現します。クラスター内のアプリケーション・サーバーごとにメッセージング・エンジンが稼動し、一つの宛先に対して複数のメッセージング・エンジンで処理を行います。それぞれのメッセージング・エンジンは、個別のメッセージ・ストアを使用します。

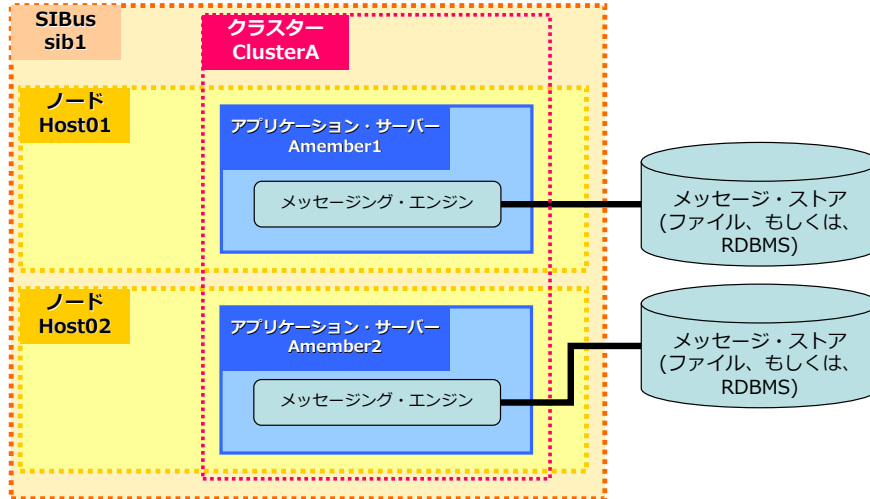


図 7-22 スケーラビリティ・ポリシー

3) 高可用性を備えたスケーラビリティ・ポリシー

「7.2.1.3パーティション構成」で記載した構成です。このポリシーは、メッセージング・エンジンのフェイルオーバーおよび負荷分散を実現します。

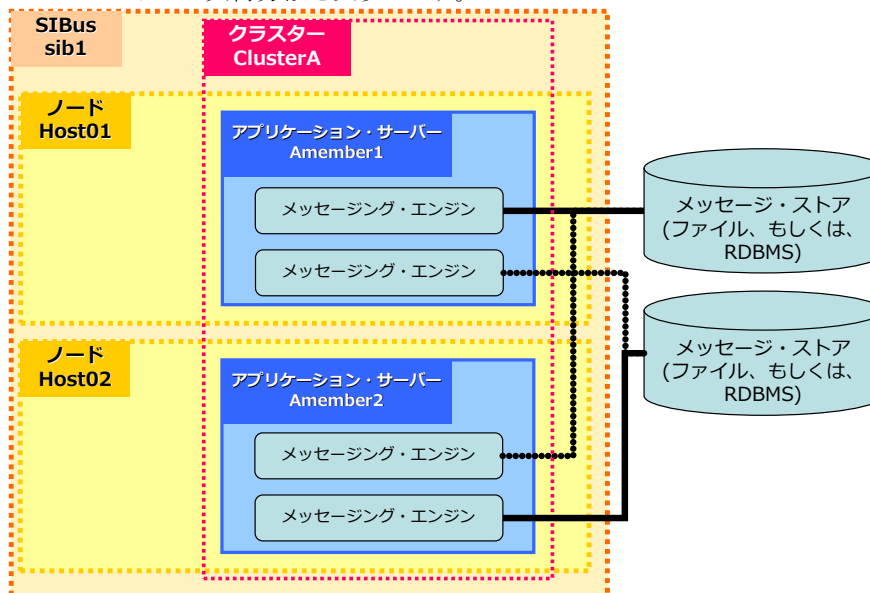


図 7-23 高可用性を備えたスケーラビリティ・ポリシー

7.2.2 その他の設定

「7.2.1 トポロジー設定」で構成したメッセージング・エンジンを利用し、メッセージを送受信するための設定について説明します。

7.2.2.1 宛先の設定

宛先は、アプリケーション間でメッセージを送受信するために使用されます。SIBus の宛先には、以下の4つの種類が存在します。

- キュー宛先
Point-to-Point (PTP)のメッセージングに使用されるメッセージ・キューです。
- トピック・スペース宛先
Publish/Subscribe (Pub/Sub)のメッセージングに使用されるトピックです。
- 外部宛先
別の SIBus や WebSphere MQ (WMQ) のメッセージング・サービスに対してメッセージを送信するために使用されます。Point-to-Point のメッセージングに使用できます。
- 別名宛先
宛先に別名をつけるために使用されます。たとえば、WMQ のアプリケーションが宛先に対してメッセージを送信するときに、宛先の名前が長すぎて WMQ の制約に抵触する場合に、別名宛先を定義することで、WMQ 準拠の名前となり、メッセージの送信ができるようになります。

ここでは、キュー宛先を設定する方法について説明します。

- i) 宛先を新規作成します
管理コンソールのメニューより、[サービス統合]→[バス]→[キュー宛先を作成するバス名]→[宛先リソース:宛先]を選択し、「新規作成」をクリックします。
- ii) 宛先タイプとして、「キュー」を選択します
- iii) 宛先名を設定します
「ID」にキュー宛先の名前を入力します。
- iv) バス・メンバーを選択します
どのバス・メンバーにキュー・ポイントを作成するか選択します。

7.2.2.2 外部バス

SIBus には、同一セルもしくは別セルにある SIBus を接続する「SIBus リンク機能」と、バスと WMQ のキュー・マネージャー (Qmgr) をチャネル接続する「MQ リンク機能」があります。これらの機能では、リンク先のバスや Qmgr を外部バスとして作成します。リンクを作成すると、バス間でメッセージをやり取りできるようになります。

SIBus リンクと MQ リンクの作成方法について、それぞれ個別に説明します。

SIBus リンクの作成方法

SIBus リンクは、2 つの SIBus のそれぞれに含まれるメッセージング・エンジン同士をリンクすることで、バス間でのメッセージのやり取りを可能にする機能です。SIBus リンクは、リンクする相互のバス上に同じ名前で作成する必要があります。

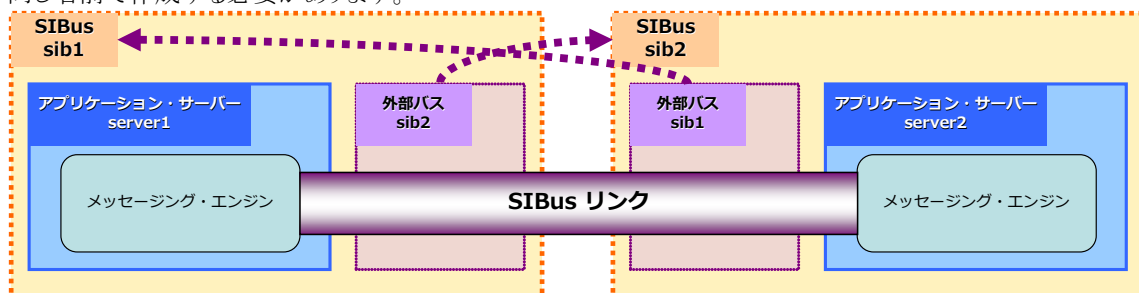


図 7-24 SIBus リンク概要

SIBus リンクを作成し、外部バス上の宛先に対する別名宛先を作成する手順は以下のとおりです。ここで作成する別名宛先に対してメッセージを送信することで、外部バスにメッセージを送信できます。

- i) 外部バス接続を新規作成します
管理コンソールのメニューより、**[サービス統合]→[バス]→[バス名]→[トポロジー:外部バス接続]**を選択し、「新規作成」をクリックします。
- ii) バス接続タイプとして、「直接接続」を選択します
- iii) 外部バス・タイプとして「サービス統合バス」を選択します
- iv) ローカル・バス詳細を以下のとおり設定します

表 7-6 ローカル・バス詳細

名前	値	概要
接続をホストするメッセージング・エンジン	ClusterA.000-SIB1	接続元となるローカルのメッセージング・エンジン名です。SIBus リンクの実体は、メッセージング・エンジン間の接続です。
インバウンド・ユーザーID		外部バスから届くメッセージのユーザーIDをこのIDに置換します。届いたメッセージが宛先にアクセスできるかどうか判定するために使用されます。

- v) 外部バス詳細を以下のとおり設定します

表 7-7 外部バス詳細

名前	値	概要
接続先（外部バス）のサービス統合バスの名前	SIB2	接続相手のSIBusの名称を設定します。接続相手の名称と必ず一致する必要があります。
外部バス内のゲートウェイ・メッセージング・エンジン	ClusterB.000-SIB2	接続相手のメッセージング・エンジンの名称を設定します。
この接続に対するパブリッシュ/サブスクライブ・メッセージングの構成		パブリッシュ/サブスクライブのメッセージングを構成するか選択します
サービス統合バス・リンク名	SIB1-SIB2	サービス統合バス・リンクの名称を設定します。
ターゲット・インバウンド・トランスポート・チェーン	InboundBasicMessaging	外部バスとの通信に使用されるトランスポート・チェーンのタイプを設定します。 InboundBasicMessaging 標準のTCP/IP接続です InboundSecureMessaging InboundBasicMessagingがSSLで保護されたプロトコルです

ブートストラップ・サービス統合バスのプロバイダー・エンドポイント	サーバー名:ポート番号	ブートストラップ・サーバーのリストを”,”(カンマ) 区切りで指定します。指定方法は、「host:port:protocol」です。 protocol には、InboundBasicMessaging の場合は BootstrapBasicMessaging を、InboundSecureMessaging の場合は BootstrapSecureMessaging をそれぞれ指定します。 ブートストラップ・サーバーは、管理コンソールの[サービス統合]→[バス]→[バス名]→[トポロジー:ブートストラップ・メンバー]より確認できます。
セキュア接続		セキュア接続にするかどうかを指定します
認証別名		外部バスに接続する際に使用する J2C 認証エイリアスを指定します。

vi) 作成したサービス統合バス・リンクの内容を確認します

管理コンソールのメニューより、[サービス統合]→[バス]→[バス名]→[トポロジー:外部バス接続]→[外部バス名]→[関連項目:サービス統合バス・リンク]→[サービス統合バス・リンク名]を選択します。先ほどの手順で設定した値に加え、以下の項目が追加となっていますので、必要に応じて修正します。

表 7-8 サービス統合バス・リンク詳細

名前	値	概要
例外宛先	システム	エラー・メッセージの例外宛先を指定します。 エラー・メッセージを破棄する場合は、「なし」を選択します。 エラー・メッセージをデフォルトのローカルのメッセージング・エンジン例外宛先に送信する場合は、「システム」を選択します。 任意の例外宛先にエラー・メッセージを送信するには、「指定」を選択し、例外宛先を入力します。
インバウンド・メッセージの動作	ClusterB.000-SIB2	サービス統合バス・リンクと関連しているメッセージング・エンジンのターゲット宛先のキュー・ポイントに優先的にメッセージを送信するのか、ターゲット宛先の全てのキュー・ポイントにメッセージを送信するのかを指定します。

vii) 別名宛先を作成します

管理コンソールのメニューより、[サービス統合]→[バス]→[バス名]→[宛先リソース:宛先]を選択し、「新規作成」をクリックします。宛先タイプの選択は「別名」とし、以下の内容を設定します。

表 7-9 別名宛先の設定内容

名前	値	概要
ID	SAMPLEQ	バス上に作成する別名宛先名です。
バス	SIB1	別名宛先を作成するバス名です。
ターゲット ID	その他指定 SIB2 上のキュー宛先名	接続相手に存在する宛先の宛先名です。
ターゲット・バス	SIB2	接続相手の SIBus 名です。
デフォルトの転送ルーティン グ・パス		転送ルーティング・パス (Forward Routing Path) が含まれていない場合に、設定される値です。「bus_name:destination_name (同一バス上の場合は「:destination_name」でも指定可能)」の形式で指定します。

MQ リンクの作成方法

WMQ の Qmgr が存在するとき、チャネル接続を作成することでメッセージのやり取りが可能になります。Qmgr はバス上に外部バスとして作成されます。チャネルは、バスから Qmgr にメッセージを送送するための送信チャネルと、Qmgr からバスにメッセージを送送するための受信チャネルを作成します。メッセージング・エンジンは Qmgr 上で外部 Qmgr として作成されます。そのため、MQ リンク作成時には、メッセージング・エンジンに仮想 Qmgr 名を設定します。

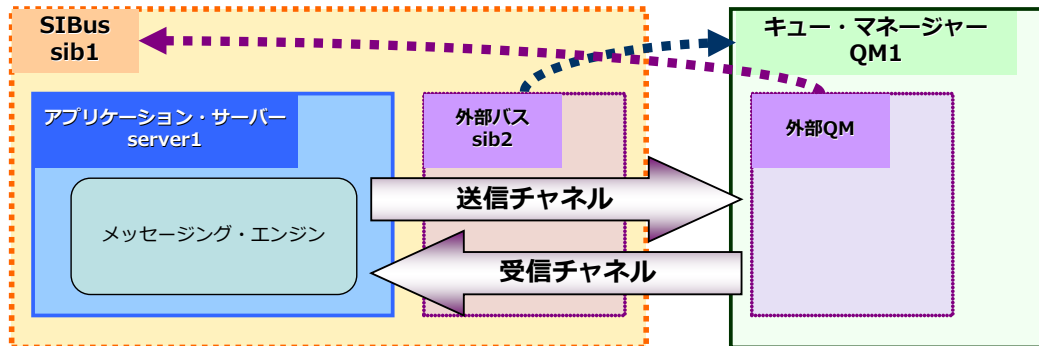


図 7-25 MQ リンク概要

MQ リンクを作成し、QMgr 上のローカル・キューに対する別名宛先を作成する手順は以下のとおりです。ここで作成する別名宛先に対してメッセージを送信することで、Qmgr にメッセージを送信できます。なお、Qmgr 上には送信チャネルと受信チャネルが作成済みであるものとします。

- i) 外部バス接続を新規作成します
管理コンソールのメニューより、[サービス統合]→[バス]→[バス名]→[トポロジー:外部バス接続]を選択し、「新規作成」をクリックします。
- ii) バス接続タイプとして、「直接接続」を選択します
- iii) 外部バス・タイプとして「WebSphere MQ」を選択します
- iv) ローカル・バス詳細を以下のとおり設定します

表 7-10 ローカル・バス詳細

名前	値	概要
接続をホストするメッセージング・エンジン	ClusterA.000-SIB1	接続元となるローカルのメッセージング・エンジン名です。
仮想キュー・マネージャー名	SIB1	接続相手となる Qmgr によって認識される、メッセージング・エンジンの仮想 Qmgr 名です。

v) 外部バス詳細を以下のとおり設定します

表 7-11 WebSphere MQ 詳細

名前	値	概要
外部バス名	MQB1	外部バスの名称を設定します。
MQリンク名	SIB1-QM1	MQ リンクの名称を設定します。
この接続に対するパブリッシュ/サブスクライブ・メッセージングの構成		パブリッシュ/サブスクライブのメッセージングを構成するか選択します
サービス統合バス・リンク名	SIB1-SIB2	サービス統合バス・リンクの名称を設定します。
サービス統合バスから WebSphere MQ へのメッセージ・フローを使用可能にする	チェック	サービス統合バスから WMQ へのメッセージ・フローを使用可能にするかを指定します。
WebSphere MQ 受信側チャンネル名	受信チャンネル名	メッセージの送信先の WMQ 受信側チャンネルの名前を設定します。この値は、MQ リンク送信側チャンネルの名前として使用されます。
ホスト名	Qmgr のホスト名	Qmgr が稼動しているホスト名を指定します。
ポート	1414	Qmgr が稼動しているポートを指定します。
WebSphere MQ 受信側チャンネルはセキュアか		WMQ 受信側チャンネルがセキュアであるかどうかを指定します。チェックした場合には、「OutboundSecureMQLink」をトランスポート・チェーンとして使用し、MQ リンク送信側チャンネルが作成されます。
WebSphere MQ からサービス統合バスへのメッセージ・フローを使用可能にする	チェック	WMQ からサービス統合バスへのメッセージ・フローを使用可能にするかを指定します。

WebSphere MQ 送信側 チャンネル名	送信チャンネル名	メッセージを受信する WMQ 送信側チャンネルの名前を設定します。この値は、MQ リンク受信側チャンネルの名前として使用されます。
SIB インバウンド・ユーザー ID		Qmgr から届くメッセージのユーザーID をこの ID に置換します。届いたメッセージが宛先にアクセスできるかどうか判定するために使用されます。

vi) 作成した MQ リンクの内容を確認します

管理コンソールのメニューより、[サービス統合]→[バス]→[バス名]→[トポロジー:外部バス接続]→[外部バス名]→[関連項目:WebSphere MQ リンク]→[MQ リンク名]を選択します。先ほどの手順で設定した値に加え、以下の項目が追加となっていますので、必要に応じて修正します。

表 7-12 MQ リンク詳細

名前	値	概要
バッチ・サイズ	デフォルト値	チェックポイントを取ることなく、チャンネルによって送信できるメッセージの最大数を設定します。
最大メッセージサイズ	デフォルト値	MQ リンクのチャンネルで伝送可能なメッセージの最大の長さ (バイト) を設定します。
ハートビート間隔	デフォルト値	MQ リンク送信側チャンネルがサービスしている伝送ポイントにメッセージがない場合に、MQ リンク送信側チャンネルから WMQ 受信側チャンネルに渡される、ハートビートの間隔を秒単位で設定します。
シーケンス折り返し値	デフォルト値	メッセージのシーケンス番号をラップする値を設定します。この値を超過すると、再び1からシーケンス番号が始まります。
採用可能	チェック	MQ リンクに関連付けられている WMQ リンク受信側チャンネルの実行インスタンスを採用するかどうかを選択します。 チェックすると、既に実行中である WMQ リンク受信側チャンネルのインスタンスが停止され、新しいインスタンスが開始されます。

宛先例外	システム	エラー・メッセージの例外宛先を指定します。 エラー・メッセージを破棄する場合は、「なし」を選択します。 エラー・メッセージをデフォルトのローカルのメッセージング・エンジン例外宛先に送信する場合は、「システム」を選択します。 任意の例外宛先にエラー・メッセージを送信するには、「指定」を選択し、例外宛先を入力します。
初期状態	始動済み	メッセージング・エンジンが初めて開始されたときに、MQ リンクが開始されているか停止されているかを、始動済み/停止済みから選択します。
非パーシステント・メッセージ速度	高速	MQ リンクのチャネルに対する非パーシスタント・メッセージのサービス・クラスを、高速/通常から選択します。

vii) 別名宛先を作成します

管理コンソールのメニューより、[サービス統合]→[バス]→[バス名]→[宛先リソース:宛先]を選択し、「新規作成」をクリックします。宛先タイプの選択は「別名」とし、以下の内容を設定します。

表 7-13 別名宛先の設定内容

名前	値	概要
ID	SAMPLEQ	バス上に作成する別名宛先名です。
バス	SIB1	別名宛先を作成するバス名です。
ターゲット ID	その他指定 SAMPLEQ@QM1	接続相手の Qmgr のローカル・キュー名です。「Qlocal_name@Qmgr_name」の形式で指定します。
ターゲット・バス	MQB1	接続相手の宛先があるバス名です。

viii) リモート・キューを作成する

SIBus 上の宛先に対して、Qmgr 上にリモート・キューを定義します。このリモート・キューにメッセージを PUT することで、メッセージが SIBus に転送されます。

```
DEFINE QREMOTE('Qremote_name') RNAME('Dest_name') RQMNAME('ME_Qmgr_name')
```

Qremote_name=Qmgr 上に定義するリモート・キュー名

Dest_name=SIBus 上の宛先名

ME_Qmgr_name=メッセージング・エンジンの仮想 Qmgr 名

外部バスの管理

外部バス接続で作成した SIBus リンクと MQ リンクについて、管理コンソールから以下を操作することが可能です。

- 外部バスの状況確認
- 保留メッセージの操作
- 例外宛先の管理

ここでは、MQ リンクについて説明します。

外部バスの状況確認

管理コンソールのメニューより、[サービス統合]→[バス]→[バス名]→[トポロジー: 外部バス接続]→[外部バス接続名]→[追加プロパティ: WebSphere MQ リンク]を選択します。下記のように、外部バスの状況、現在のアウトバウンド・メッセージ数、通信メッセージ数、受信メッセージ数を確認できます。

最新表示		新規作成	削除	開始	停止モード: 静止		ターゲットの状態: 非アクティブ		停止
選択	名前	説明	ローカル・メッセージング・エンジン	仮想キュー・マネージャー名	状況	現在のアウトバウンド・メッセージ数		送信メッセージ数	受信メッセージ数
☐	MQLink01		host1Node01.Amember11-sib1	MEQM	RUNNING: 報告する問題はありません。	送信側チャネル送信元	100	送信側チャネル送信元	0
						リンク送信側	0	リンク送信側	0
合計 1									

図 7-26 外部バスの状況確認

保留メッセージの操作

保留メッセージは、図 7-26の「現在のアウトバウンド・メッセージ数」に表示されます。送信側チャネル送信元、もしくはリンク送信側を選択すると、図 7-27のように、現在のアウトバウンド・メッセージ数、通信メッセージ数、最後の受信メッセージ以降の時間を確認できます。ここで、[全メッセージを移動]ボタンを押すと、すべての保留メッセージを例外宛先に移動できます。また、[全メッセージを削除]ボタンを押すと、すべての保留メッセージを削除できます。

全メッセージを移動

全メッセージを削除

選択

WebSphere MQ リンク送信側チャネル

状況

現在のアウトバウンド・メッセージ数

送信メッセージ数

最後の送信メッセージ以降の時間

既知のリンク送信側

次のリソースを管理できます。

MEQM.MQQM

RUNNING:
報告する問題はありません。

100

200

2 分

既知のリンク送信側を表示

合計 1

図 7-27 送信側チャネル送信元の確認

WAS V7.0 から保留メッセージの移動、削除が可能になりました。図 7-27にて、WebSphere MQ リンク送信側チャンネルを選択し、保留メッセージごとに、移動、削除、コミット、ロールバックを実行します。

移動		削除	保留確認 バッチのコミット	保留確認 バッチのロールバック	メッセージの最大表示数	256	リフレッシュ
選択	位置	ID	状態	トランザクション ID	ターゲット・キュー・マネージャー	ターゲット・キュー	メッセージ長の概算 (バイト)
☐	001	7000207	保留送信		MQQM	MQ.QL01	1663
☐	002	7000208	保留送信		MQQM	MQ.QL01	1663
☐	003	7000209	保留送信		MQQM	MQ.QL01	1663

図 7-28 保留メッセージの確認

例外宛先の管理

エラー・メッセージの例外宛先として、デフォルトのローカルのメッセージング・エンジン例外宛先を指定することも、任意の宛先を指定することも可能です。また、例外宛先に送信せず、メッセージを破棄することも指定可能です。例外宛先に送信されたメッセージは、メッセージ・プロパティにて例外宛先タイムスタンプや例外宛先理由等を確認できます。

7.2.2.3 メディエーション

概要

メディエーション・プロセスは、アプリケーションが送信したメッセージが受信される前に、メッセージを処理します。メディエーション処理をコーディングすることで、次のような処理が可能です。

- ある形式から別の形式へのメッセージ変換
- 送信側アプリケーションによって指定されなかった 1 つ以上のターゲット宛先へのメッセージのルーティング
- データ・ソースからデータを追加することによる、メッセージの拡張
- 複数のターゲット宛先へのメッセージの配布

メディエーションを利用する場合、事前にメディエーション・ハンドラーをコーディングしておく必要があります。ここでは、メディエーション・ハンドラーのコーディング方法は解説しませんが、メディエーションを理解するには、いくつかのキーワードを理解する必要があります。以下にそれらのキーワードを列挙します。

メディエーション

SIBus 上の定義です。実体は EJB デプロイメント記述子に定義されている、メディエーション・ハンドラー・リストです。つまりメディエーションは、メディエーション・ハンドラー・リストへのポインターです。メディエーションを宛先に設定すると、宛先に入ったメッセージは、メディエーション・ハンドラー・リストに登録されているハンドラーを経由するようになります。

このとき、メディエーションされる前にメッセージが蓄積されるキューをメディエーション・ポイント (もしくは Pre-mediated)、メディエーションされた後にメッセージが蓄積されるキューをキュー・ポイント (もしくは Post-mediated) と呼びます。

なお、アプリケーションは、キュー・ポイントからメッセージを受信できますが、メディエーション・ポイントからは受信できません。

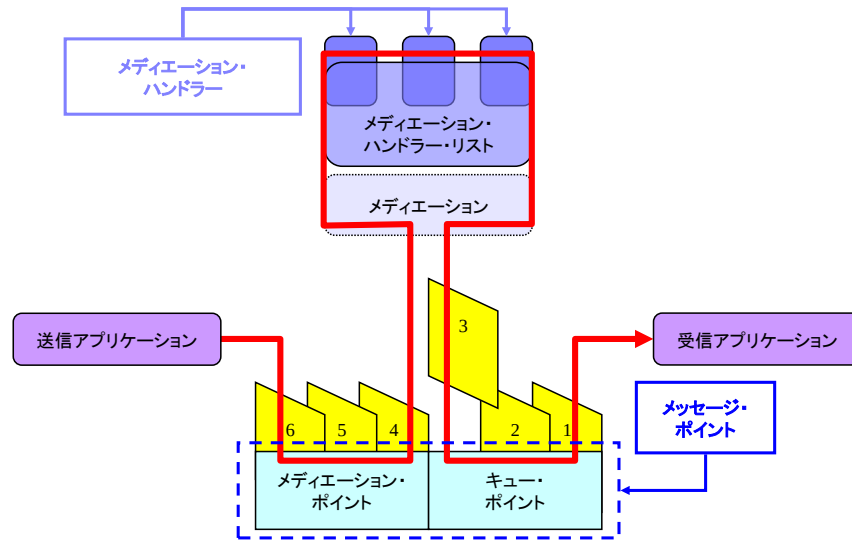


図 7-29 メディエーションの概要

メディエーション・ハンドラー

実体はロジックが記述された `Stateless Session Bean` です。`com.ibm.websphere.sib.mediation.handler.MediationHandler` クラスを継承して作成します。ロジックは、`handle (MessageContext context)` メソッドに記述します。この `MessageContext` を使って、メッセージを操作します。

メディエーション・ハンドラー・リスト

複数のメディエーション・ハンドラーをまとめた定義です。どのハンドラーを、どのような順番で呼び出していくかを定義してあります。メディエーション・ハンドラー・リストは、EJB デプロイメント記述子に定義します。メディエーション・ハンドラー・リストには、一意の名前を割り当てる必要があります。メディエーションはこの名前に対するポインターです。

メディエーション・ハンドラーの `handle` メソッドが `true` を返すと、次のメディエーション・ハンドラーを実行します。これに対して `false` を返すと、次のメディエーション・ハンドラーは呼び出されず、メッセージ「再デリバリー回数」が 1 回インクリメントされ、メディエーション・ポイントに戻ります。メディエーション・ポイントに戻ったメッセージの「再デリバリー回数」が、宛先に設定された「最大デリバリー失敗数」を上回ると、メッセージは例外宛先に転送されます。例外宛先を設定していない場合は、メディエーションはリトライを繰り返し続けます。

設定方法

メディエーション・ハンドラー・リストを定義済みの EAR ファイルが用意できていることを前提に、管理コンソールによるメディエーションの設定方法を説明します。

- i) エンタープライズ・アプリケーションをデプロイします

メディエーション・ハンドラー・リストを含むエンタープライズ・アプリケーションをデプロイします。

- ii) メディエーションを作成します

管理コンソールのメニューより、[サービス統合]→[バス]→[バス名]→[宛先リソース:メディエーション]を選択し、「新規作成」ボタンをクリックします。設定項目の内容は、以下のとおりです。

表 7-14 メディエーション登録時の設定項目

項目	説明
メディエーション名	このメディエーションに割り当てる名前です。
ハンドラー・リスト名	デプロイメント記述子に設定した、メディエーション・ハンドラー・リストの名前を入力してください。
グローバル・トランザクション	グローバル・トランザクションを利用可能です。このオプションを有効にすると、メディエーションで処理された各メッセージに対してグローバル・トランザクションが開始されます。これにより、メディエーションとリソース・マネージャーのトランザクションの保全性が確保できます。
並行メディエーションを許可	スレッドを利用して、複数の処理を同時に進行出来ます。並行処理をしたい場合、チェックして下さい。 なお、並行メディエーションに利用するスレッド・プール mediationThreadPool は、次のコマンドで作成できます。 <code>\$AdminConfig create ThreadPool \$messagingEngine {{name stitch.server1-bus2-mediationThreadPool}} mediationThreadPool</code>
セレクトター	メディエーションが仲介するメッセージを選択できます。セレクトターは JMS の仕様で定義されています。
判別プログラム	指定された規則に一致すると、メディエーションがメッセージに適用されます。

iii) メディエーションを宛先に設定します

管理コンソールのメニューより、[サービス統合]→[バス]→[バス名]→[宛先リソース:宛先] を選択します。宛先一覧が表示されるので、メディエーションを設定したい宛先のチェックボックスをチェックして、[仲介]ボタン(※)をクリックします。設定項目の内容は、以下のとおりです。

(※) 管理コンソール上には、「メディエーション」という表記と、メディエーションの訳語である「仲介」という表記が混在しています。

表 7-15 メディエーションを宛先に登録する際の設定項目

項目	説明
この宛先に適用するメディエーション	この宛先に適用するメディエーションを選択します。
メディエーション・ポイントの割り当て先のバス・メンバー	メディエーションがインストールされているバス・メンバーを選択してください。

iv) メディエーションを始動/停止します

管理コンソールのツリー・ビュー上で、[サービス統合]→[バス]→[バス名]→[宛先リソース:宛先] →[メッセージ・ポイント:メディエーション・ポイント]を選択します。ここで、メディエーション・ポイントを選択し、[始動]/[停止]ボタンをクリックすると、メディエーションが始動/停止されます。

7.3 障害時の動作

7.3.1 HA マネージャー

SIBus を高可用性構成およびパーティション構成のトポロジーで構成した場合、メッセージング・エンジンをどのアプリケーション・サーバーで稼動させるか、障害を検知した場合の動作をどうするかなどは、HA マネージャーにより管理されます。HA マネージャーの詳細については、「3-5 HA マネージャー」を参照してください。

障害検知と検知時の動作

HA マネージャーの機能により、他サーバーのプロセス障害やネットワーク障害を検知した場合、ポリシーに基づいてメッセージング・エンジンのフェイルオーバーが行われます。

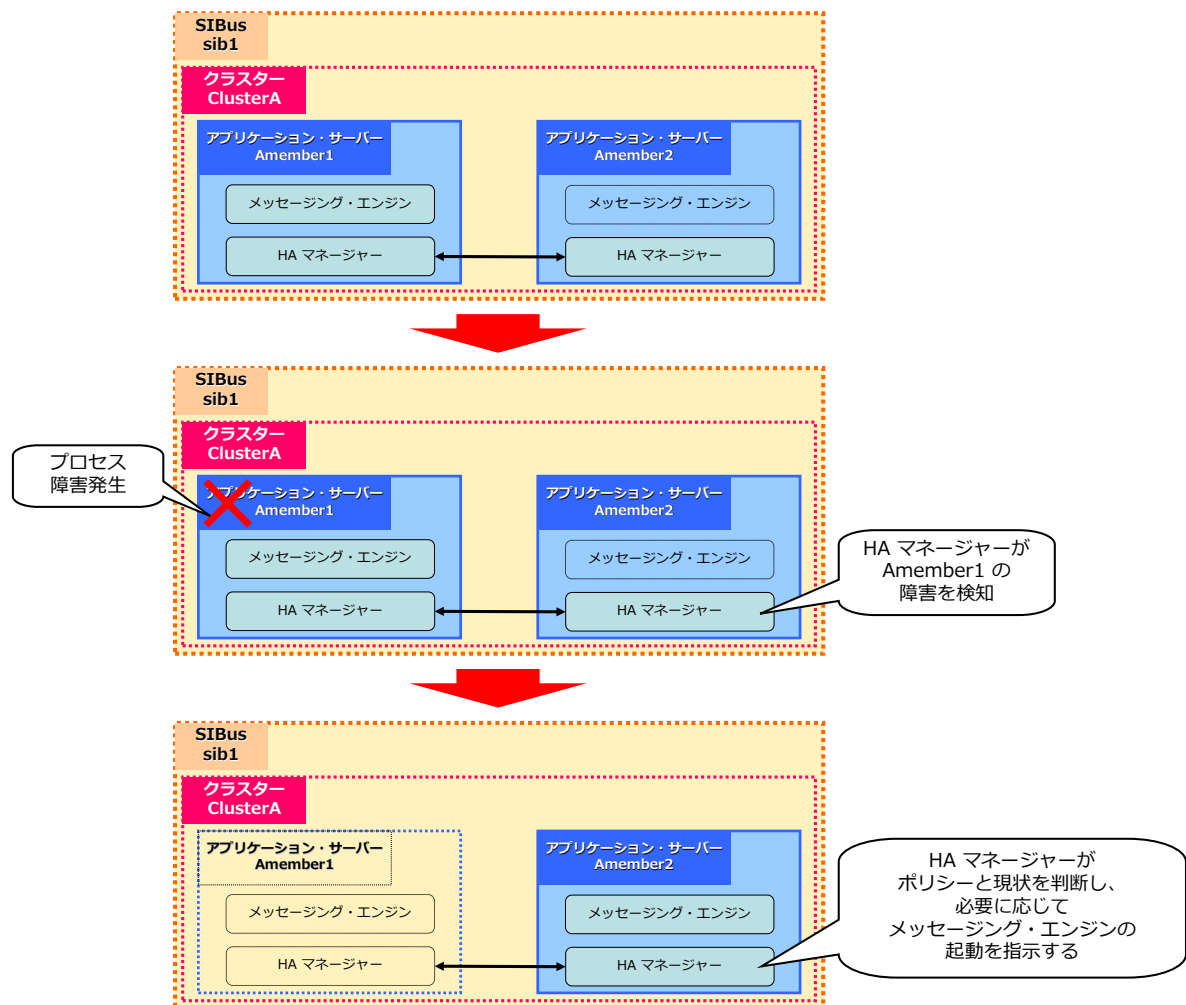


図 7-30 HA マネージャーによるメッセージング・エンジンのフェイルオーバー

ポリシーの設定と挙動

HA マネージャーがどのようにメッセージング・エンジンを起動するかは、ポリシー設定に基づいて行われます。管理コンソールのメニューより、[サーバー]→[コア・グループ]→[コア・グループ設定]→[ポリシー名]→[追加プロパティ:ポリシー]→[ポリシー名]を選択します。

どのメッセージング・エンジンにポリシーを設定するかは、追加プロパティの「一致基準」から指定することができます。

メッセージング・エンジンをどのサーバーで起動するかを優先順位を、追加プロパティの「優先サーバー」からサーバー単位で指定することができます。特定のメッセージング・エンジンに対してポリシーを設定する場合は、下記のように設定します。

- 対象となるメッセージング・エンジンの指定
名前: WSAF_SIB_MESSAGING_ENGINE
値: クラスターメンバー名.メッセージング・エンジン番号-SIBus 名
- メッセージング・エンジンに対する指定
名前: type
値: WSAF_SIB

一般プロパティ内の「クォーラム」、「フェイルバック」および「優先サーバーのみ」のそれぞれを設定した場合の挙動を以下に記載します。

ポリシーの挙動として「クォーラム」を指定した場合

クラスター構成時に、クラスターメンバーが半数以上起動するまで、一致基準で指定されるメッセージング・エンジンを起動しません。

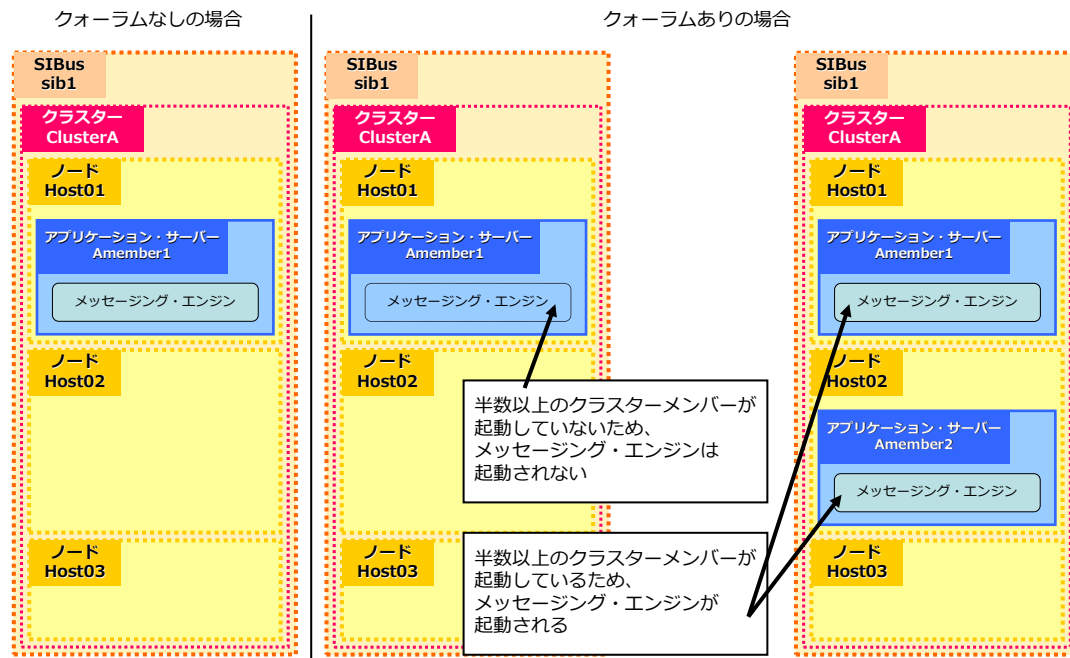


図 7-31 クォーラムが設定されている場合の動作

ポリシーの挙動として「フェイルバック」を指定した場合

優先サーバーで指定されている、優先順位の高いサーバーが障害から復帰したら、そのサーバーのメッセージング・エンジンが起動し、優先順位の低いサーバーのメッセージング・エンジンが停止します。

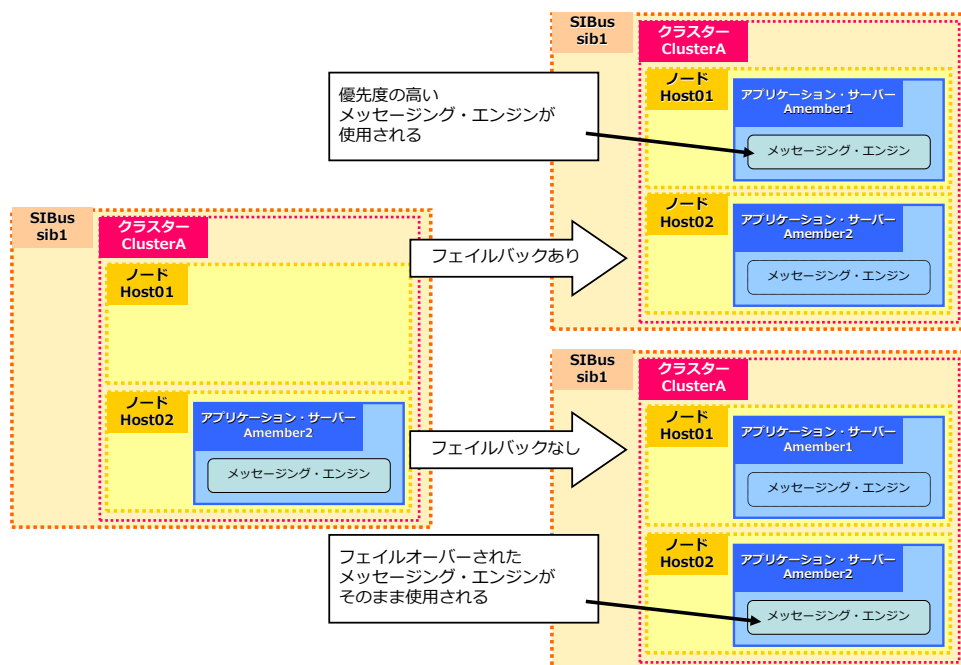


図 7-32 フェイルバックが設定されている場合の動作

ポリシーの挙動として「優先サーバーのみ」を指定した場合

優先サーバーで指定されたサーバーでのみ、メッセージング・エンジンを起動します。

7.3.2 高可用性構成における障害時の動作

説明を行う構成は、「7.2.1.2 高可用性構成」と同じ構成です。

一般的な JMS アプリケーションの動作フロー

この説明の中で使用している JMS アプリケーションの動作フローを説明します。

1) メッセージを PUT するアプリケーション

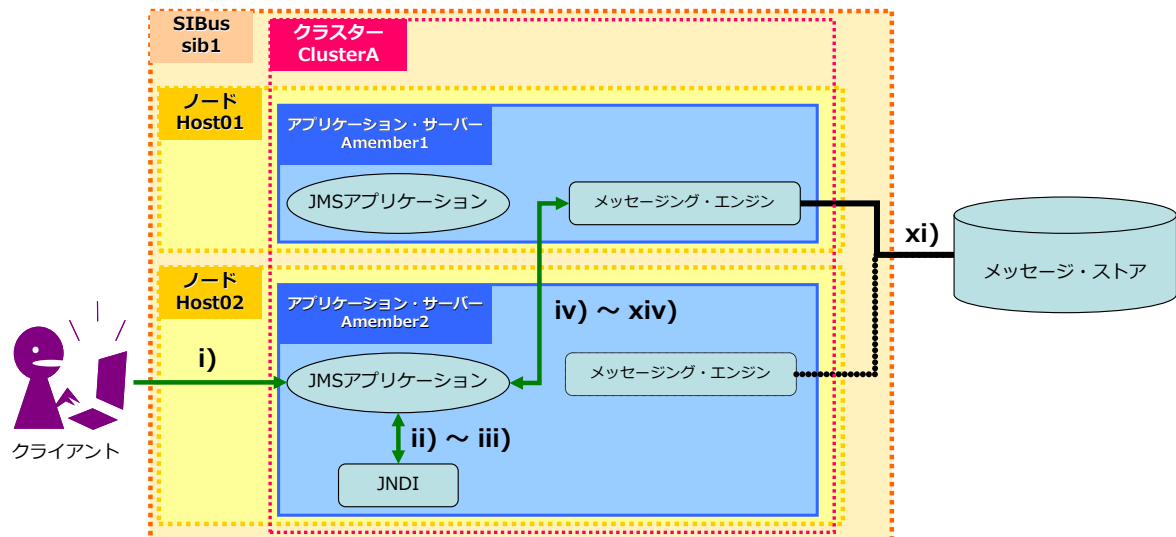


図 7-33 メッセージを PUT するアプリケーションの概要

- i) クライアントは、JMS アプリケーションを実行します
- ii) JMS アプリケーションは、JMS 接続ファクトリーおよびキューの情報を JNDI に問い合わせます
- iii) JNDI は、JMS 接続ファクトリーおよびキューの情報を返します
- iv) JMS 接続ファクトリーを使用して、Connection を作成します
- v) Session を作成します
- vi) メッセージを送信するための MessageProducer を作成します
- vii) JMS メッセージを作成します
- viii) JMS メッセージに返信先をセットします
- ix) JMS 接続に対して、操作開始を指示します
- x) メッセージを PUT します
- xi) メッセージング・エンジンは PUT されたメッセージをメッセージ・ストアに格納します
- xii) MessageProducer をクローズします
- xiii) Session をクローズします
- xiv) Connection を切断します

2) メッセージを GET するアプリケーション

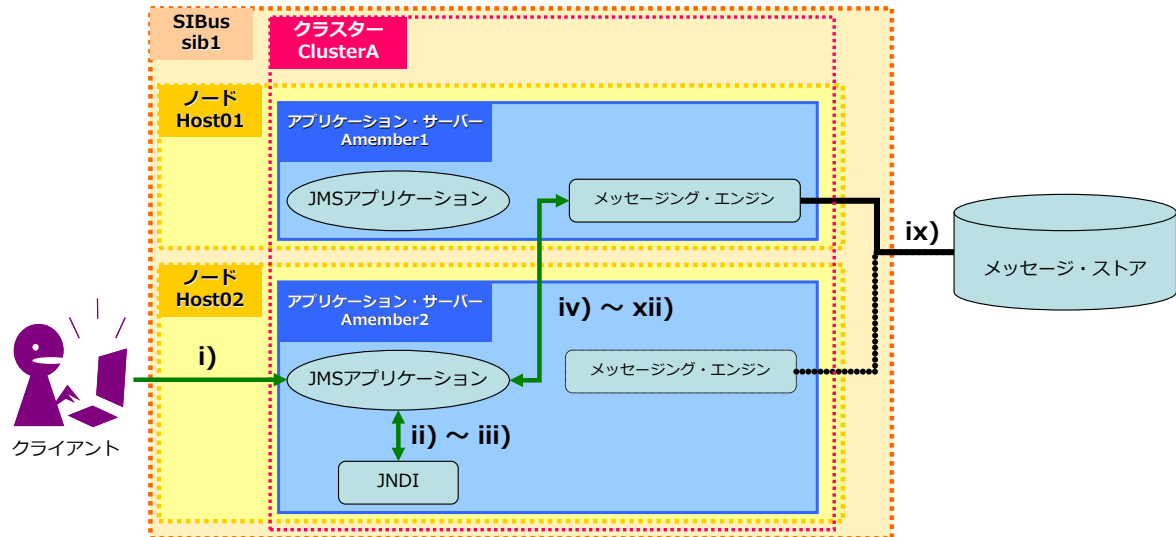


図 7-34 メッセージを GET するアプリケーションの概要

- i) クライアントは、JMS アプリケーションを実行します
- ii) JMS アプリケーションは、JMS 接続ファクトリーおよびキューの情報を JNDI に問い合わせます
- iii) JNDI は、JMS 接続ファクトリーおよびキューの情報を返します
- iv) JMS 接続ファクトリーを使用して、Connection を作成します
- v) Session を作成します
- vi) メッセージを受信するための MessageConsumer を作成します
- vii) JMS 接続に対して、操作開始を指示します
- viii) メッセージを GET します
- ix) メッセージング・エンジンは GET されたメッセージをメッセージ・ストアから削除します
- x) MessageConsumer をクローズします
- xi) Session をクローズします
- xii) Connection を切断します

障害発生時の挙動

メッセージング・エンジンが稼動するアプリケーション・サーバーにプロセス障害やネットワーク障害が発生した場合は、HA マネージャーにより即時に検知され、同一 HA グループに所属するメッセージング・エンジンが起動されます。障害が発生した時点で、JMS アプリケーションが JMS のコネクションを作成している状態であれば、そのリクエストを処理しているアプリケーションでは、アプリケーションの実行フローに応じた `JMSEException` を受け取ります。ただし、メッセージング・エンジンのフェイルオーバーはほぼ即時に完了しますので、再度リクエストを送信するか、エラーハンドリングの中で、再度 PUT や GET を行うことでリクエストは正常に完了することができます。

PUT を行うアプリケーションの場合

PUT を行うアプリケーションのフローごとに、メッセージング・エンジンが起動しているアプリケーション・サーバー (AMember01) にプロセス障害かネットワーク障害が発生した場合の挙動を説明します。

- i). クライアントは、JMS アプリケーションを実行します

- ii). JMS アプリケーションは、JMS 接続ファクトリーおよびキューの情報を JNDI に問い合わせます
この時点で、メッセージング・エンジンが稼働しているプロセスやネットワークに障害が発生すると、HA マネージャーにより検知され、AMember02 のメッセージング・エンジンが起動します。メッセージング・エンジンへのアクセスはまだ発生していないため、JMS アプリケーションに対して、例外は発生しません。
- iii). JNDI は、JMS 接続ファクトリーおよびキューの情報を返します
この時点で、メッセージング・エンジンが稼働しているプロセスやネットワークに障害が発生すると、HA マネージャーにより検知され、AMember02 のメッセージング・エンジンが起動します。メッセージング・エンジンへのアクセスはまだ発生していないため、JMS アプリケーションに対して、例外は発生しません。
- iv). JMS 接続ファクトリーを使用して、Connection を作成します
この時点で、メッセージング・エンジンが稼働しているプロセスやネットワークに障害が発生すると、HA マネージャーにより検知され、AMember02 のメッセージング・エンジンが起動します。
障害が発生した時点でアクセスが行われている JMS アプリケーションに対して、`javax.jms.JMSEException` が返されアプリケーションの実行は失敗します。返される `JMSEException` メッセージの全文は以下のとおりです。

`javax.jms.JMSEException:`

CWSIA0241E: メソッド `JmsManagedConnectionFactoryImpl.createConnection` の呼び出し中に例外を受信しました:

`com.ibm.websphere.sib.exception.SIResourceException:`

CWSIT0019E: バス JMS Test に使用可能な適切なメッセージング・エンジンがありません。

メッセージング・エンジンのフェイルオーバーが完了した後のクライアントからのアクセスは、正常に行われます。

- v). Session を作成します
この時点で、メッセージング・エンジンが稼働しているプロセスやネットワークに障害が発生すると、HA マネージャーにより検知され、AMember02 のメッセージング・エンジンが起動します。
障害が発生した時点でアクセスが行われている JMS アプリケーションに対して、`javax.jms.JMSEException` が返されアプリケーションの実行は失敗します。返される `JMSEException` メッセージの全文は以下のとおりです。

プロセス障害の場合:

`javax.jms.JMSEException:`

CWSIA0053E: メソッド `JmsSessionImpl.` の呼び出し中に例外を受信しました

`com.ibm.wsspi.sib.core.exception.SIConnectionDroppedException:`

CWSIJ0044E: すでにクローズされた接続での操作が行われようとしていました。

ネットワーク障害の場合:

`javax.jms.JMSEException:`

CWSIA0053E: メソッド `JmsSessionImpl.` の呼び出し中に例外を受信しました

`com.ibm.ws.sib.jfapchannel.JFapConnectionBrokenException:`

CWSIJ0056E: 予期しない条件により、ネットワーク接続がクローズしました。

メッセージング・エンジンのフェイルオーバーが完了した後のクライアントからのアクセスは、正常に行われます。

vi). メッセージを送信するための **MessageProducer** を作成します

この時点で、メッセージング・エンジンが稼働しているプロセスやネットワークに障害が発生すると、HA マネージャーにより検知され、AMember02 のメッセージング・エンジンが起動します。

障害が発生した時点でアクセスが行われている JMS アプリケーションに対して、`javax.jms.JMSEException` が返されアプリケーションの実行は失敗します。返される `JMSEException` メッセージの全文は以下のとおりです。

プロセス障害の場合：

`javax.jms.JMSEException:`

`CWSIA0067E: メソッド JmsMsgProducerImpl. の呼び出し中に例外を受信しました:`

`com.ibm.wsspi.sib.core.exception.SIConnectionDroppedException:`

`CWSIJ0047E: すでにクローズされた接続での操作が行われようとしていました。`

ネットワーク障害の場合：

`javax.jms.JMSEException:`

`CWSIA0067E: メソッド JmsMsgProducerImpl. の呼び出し中に例外を受信しました:`

`com.ibm.ws.sib.jfapchannel.JFapConnectionBrokenException:`

`CWSIJ0056E: 予期しない条件により、ネットワーク接続がクローズしました。`

メッセージング・エンジンのフェイルオーバーが完了した後のクライアントからのアクセスは、正常に行われます。

vii). JMS メッセージを作成します

この時点で、メッセージング・エンジンが稼働しているプロセスやネットワークに障害が発生すると、HA マネージャーにより検知され、AMember02 のメッセージング・エンジンが起動します。

メッセージ作成時点での障害が原因での例外は発生しませんが、メッセージ PUT 時に `javax.jms.JMSEException` が返され、障害が発生した時点でアクセスが行われている JMS アプリケーションに対して、アプリケーションの実行が失敗します。

メッセージング・エンジンのフェイルオーバーが完了した後のクライアントからのアクセスは、正常に行われます。

viii). JMS メッセージに返信先をセットします

この時点で、メッセージング・エンジンが稼働しているプロセスやネットワークに障害が発生すると、HA マネージャーにより検知され、AMember02 のメッセージング・エンジンが起動します。

メッセージの返信先をセットした時点での障害が原因での例外は発生しませんが、メッセージ PUT 時に `javax.jms.JMSEException` が返され、障害が発生した時点でアクセスが行われている JMS アプリケーションに対して、アプリケーションの実行が失敗します。

メッセージング・エンジンのフェイルオーバーが完了した後のクライアントからのアクセスは、正常に行われます。

ix). JMS 接続に対して、操作開始を指示します

この時点で、メッセージング・エンジンが稼働しているプロセスやネットワークに障害が発生すると、HA マネージャーにより検知され、AMember02 のメッセージング・エンジンが起動します。

操作開始の指示をした時点での障害が原因での例外は発生しませんが、メッセージ PUT 時に `javax.jms.JMSEException` が返され、障害が発生した時点でアクセスが行われている JMS アプリケーションに対して、アプリケーションの実行が失敗します。

メッセージング・エンジンのフェイルオーバーが完了した後のクライアントからのアクセスは、正常に行われます。

x). メッセージを PUT します

この時点で、メッセージング・エンジンが稼働しているプロセスやネットワークに障害が発生すると、HA マネージャーにより検知され、AMember02 のメッセージング・エンジンが起動します。

障害が発生した時点でアクセスが行われている JMS アプリケーションに対して、`javax.jms.JMSEException` が返されアプリケーションの実行は失敗します。返される `JMSEException` メッセージの全文は以下のとおりです。

プロセス障害の場合:

javax.jms.JMSEException:

WSIA0067E: メソッド JmsMsgProducerImpl.sendMessage (#4) の呼び出し中に例外を受信しました:

com.ibm.wsspi.sib.core.exception.SIConnectionDroppedException:

CWSIJ0047E: すでにクローズされた接続での操作が行われようとしていました。

ネットワーク障害の場合:

javax.jms.JMSEException:

CWSIA0067E: メソッド JmsMsgProducerImpl.sendMessage (#4) の呼び出し中に例外を受信しました:

com.ibm.ws.sib.jfapchannel.JFapConnectionBrokenException:

CWSIJ0056E: 予期しない条件により、ネットワーク接続がクローズしました。

メッセージング・エンジンのフェイルオーバーが完了した後でのクライアントからのアクセスは、正常に行われます。

xi). MessageProducer をクローズします

この時点で、メッセージング・エンジンが稼働しているプロセスやネットワークに障害が発生すると、HA マネージャーにより検知され、AMember02 のメッセージング・エンジンが起動します。

障害が発生した時点でアクセスが行われている JMS アプリケーションに対して、javax.jms.JMSEException が返されアプリケーションの実行は失敗します。返される JMSEException メッセージの全文は以下のとおりです。

プロセス障害の場合:

javax.jms.JMSEException:

CWSIA0067E: メソッド JmsMsgProducerImpl.close の呼び出し中に例外を受信しました:

com.ibm.wsspi.sib.core.exception.SIConnectionDroppedException:

CWSIJ0047E: すでにクローズされた接続での操作が行われようとしていました。

ネットワーク障害の場合:

javax.jms.JMSEException:

CWSIA0067E: メソッド JmsMsgProducerImpl.close の呼び出し中に例外を受信しました:

com.ibm.ws.sib.jfapchannel.JFapConnectionBrokenException:

CWSIJ0056E: 予期しない条件により、ネットワーク接続がクローズしました。

メッセージング・エンジンのフェイルオーバーが完了した後でのクライアントからのアクセスは、正常に行われます。

xii). Session をクローズします

この時点で、メッセージング・エンジンが稼働しているプロセスやネットワークに障害が発生すると、HA マネージャーにより検知され、AMember02 のメッセージング・エンジンが起動します。

障害が発生した時点でアクセスが行われている JMS アプリケーションに対して、`javax.jms.JMSEException` が返されアプリケーションの実行は失敗します。返される `JMSEException` メッセージの全文は以下のとおりです。

プロセス障害の場合:

`javax.jms.JMSEException:`

CWSIA0067E: メソッド `JmsMsgProducerImpl.close` の呼び出し中に例外を受信しました:

`com.ibm.wsspi.sib.core.exception.SIConnectionDroppedException:`

CWSIJ0047E: すでにクローズされた接続での操作が行われようとしてしました。

ネットワーク障害の場合:

javax.jms.JMSEException:

CWSIA0067E: メソッド JmsMsgProducerImpl.close の呼び出し中に例外を受信しました:

com.ibm.ws.sib.jfapchannel.JFapConnectionBrokenException:

CWSIJ0056E: 予期しない条件により、ネットワーク接続がクローズしました。

メッセージング・エンジンのフェイルオーバーが完了した後のクライアントからのアクセスは、正常に行われます。

xiii). Connection を切断します

この時点で、メッセージング・エンジンが稼働しているプロセスやネットワークに障害が発生すると、HA マネージャーにより検知され、AMember02 のメッセージング・エンジンが起動します。

障害が発生した時点でアクセスが行われている JMS アプリケーションに対して、javax.jms.JMSEException が返されアプリケーションの実行は失敗します。返される JMSEException メッセージの全文は以下のとおりです。

javax.jms.JMSEException:

CWSIA0022E: メソッド JmsConnectionImpl.close の呼び出し中に例外を受信しました:

com.ibm.wsspi.sib.core.exception.SIConnectionDroppedException:

CWSIJ0047E: すでにクローズされた接続での操作が行われようとしてしました。

メッセージング・エンジンのフェイルオーバーが完了した後のクライアントからのアクセスは、正常に行われます。

GET を行うアプリケーションの場合

GET を行うアプリケーションのフローごとに、メッセージング・エンジンが起動しているアプリケーション・サーバー (AMember01) にプロセス障害かネットワーク障害が発生した場合の挙動を説明します。

i). クライアントは、JMS アプリケーションを実行します

ii). JMS アプリケーションは、JMS 接続ファクトリーおよびキューの情報を JNDI に問い合わせます

この時点で、メッセージング・エンジンが稼働しているプロセスやネットワークに障害が発生すると、HA マネージャーにより検知され、AMember02 のメッセージング・エンジンが起動します。メッセージング・エンジンへのアクセスはまだ発生していないため、JMS アプリケーションに対して、例外は発生しません。

iii). JNDI は、JMS 接続ファクトリーおよびキューの情報を返します

この時点で、メッセージング・エンジンが稼働しているプロセスやネットワークに障害が発生すると、HA マネージャーにより検知され、AMember02 のメッセージング・エンジンが起動します。メッセージング・エンジンへのアクセスはまだ発生していないため、JMS アプリケーションに対して、例外は発生しません。

iv). JMS 接続ファクトリーを使用して、Connection を作成します

この時点で、メッセージング・エンジンが稼働しているプロセスやネットワークに障害が発生すると、HA マネージャーにより検知され、AMember02 のメッセージング・エンジンが起動します。

障害が発生した時点でアクセスが行われている JMS アプリケーションに対して、javax.jms.JMSEException が返されアプリケーションの実行は失敗します。返される JMSEException メッセージの全文は以下のとおりです。

javax.jms.JMSEException:

CWSIA0241E: メソッド JmsManagedConnectionFactoryImpl.createConnection の呼び出し中に例外を受信しました:

com.ibm.websphere.sib.exception.SIResourceException:

CWSIT0019E: バス PME_JMSTest に使用可能な適切なメッセージング・エンジンがありません。

メッセージング・エンジンのフェイルオーバーが完了した後でのクライアントからのアクセスは、正常に行われます。

v). Session を作成します

この時点で、メッセージング・エンジンが稼働しているプロセスやネットワークに障害が発生すると、HA マネージャーにより検知され、AMember02 のメッセージング・エンジンが起動します。

障害が発生した時点でアクセスが行われている JMS アプリケーションに対して、javax.jms.JMSEException が返されアプリケーションの実行は失敗します。返される JMSEException メッセージの全文は以下のとおりです。

プロセス障害の場合:

javax.jms.JMSEException:

CWSIA0053E: メソッド JmsSessionImpl. の呼び出し中に例外を受信しました

com.ibm.wsspi.sib.core.exception.SIConnectionDroppedException:

CWSIJ0044E: すでにクローズされた接続での操作が行われようとしてしました。

ネットワーク障害の場合:

javax.jms.JMSEException:

CWSIA0053E: メソッド JmsSessionImpl.<constructor> の呼び出し中に例外を受信しました:

com.ibm.ws.sib.jfapchannel.JFapConnectionBrokenException:

CWSIJ0056E: 予期しない条件により、ネットワーク接続がクローズしました。

メッセージング・エンジンのフェイルオーバーが完了した後でのクライアントからのアクセスは、正常に行われます。

vi). メッセージを受信するための MessageProducer を作成します

この時点で、メッセージング・エンジンが稼働しているプロセスやネットワークに障害が発生すると、HA マネージャーにより検知され、AMember02 のメッセージング・エンジンが起動します。

障害が発生した時点でアクセスが行われている JMS アプリケーションに対して、javax.jms.JMSEException が返されアプリケーションの実行は失敗します。返される JMSEException メッセージの全文は以下のとおりです。

プロセス障害の場合:

javax.jms.JMSEException:

CWSIA0067E: メソッド JmsMsgProducerImpl. の呼び出し中に例外を受信しました:

com.ibm.wsspi.sib.core.exception.SIConnectionDroppedException:

CWSIJ0044E: すでにクローズされた接続での操作が行われようとしてしました。

ネットワーク障害の場合:

javax.jms.JMSEException:

CWSIA0067E: メソッド JmsMsgProducerImpl. の呼び出し中に例外を受信しました:

com.ibm.ws.sib.jfapchannel.JFapConnectionBrokenException:

CWSIJ0056E: 予期しない条件により、ネットワーク接続がクローズしました。

メッセージング・エンジンのフェイルオーバーが完了した後のクライアントからのアクセスは、正常に行われます。

vii). JMS 接続に対して、操作開始を指示します

この時点で、メッセージング・エンジンが稼働しているプロセスやネットワークに障害が発生すると、HA マネージャーにより検知され、AMember02 のメッセージング・エンジンが起動します。

操作開始の指示をした時点での障害が原因での例外は発生しませんが、メッセージ GET 時に `javax.jms.JMSEException` が返され、障害が発生した時点でアクセスが行われている JMS アプリケーションに対して、アプリケーションの実行が失敗します。

メッセージング・エンジンのフェイルオーバーが完了した後のクライアントからのアクセスは、正常に行われます。

viii). メッセージを GET します

この時点で、メッセージング・エンジンが稼働しているプロセスやネットワークに障害が発生すると、HA マネージャーにより検知され、AMember02 のメッセージング・エンジンが起動します。

障害が発生した時点でアクセスが行われている JMS アプリケーションに対して、`javax.jms.JMSEException` が返されアプリケーションの実行は失敗します。返される `JMSEException` メッセージの全文は以下のとおりです。

プロセス障害の場合：

`javax.jms.JMSEException:`

CWSIA0103E: メッセージの受信中に例外が発生しました:

`com.ibm.wsspi.sib.core.exception.SIConnectionDroppedException:`

CWSIJ0044E: すでにクローズされた接続での操作が行われようとした。

ネットワーク障害の場合：

`javax.jms.JMSEException:`

WSIA0067E: メソッド `JmsMsgProducerImpl.sendMessage (#4)` の呼び出し中に例外を受信しました:

`com.ibm.ws.sib.jfapchannel.JFapConnectionBrokenException:`

CWSIJ0056E: 予期しない条件により、ネットワーク接続がクローズしました。

メッセージング・エンジンのフェイルオーバーが完了した後のクライアントからのアクセスは、正常に行われます。

ix). MessageProducer をクローズします

この時点で、メッセージング・エンジンが稼働しているプロセスやネットワークに障害が発生すると、HA マネージャーにより検知され、AMember02 のメッセージング・エンジンが起動します。

障害が発生した時点でアクセスが行われている JMS アプリケーションに対して、`javax.jms.JMSEException` が返されアプリケーションの実行は失敗します。返される `JMSEException` メッセージの全文は以下のとおりです。

プロセス障害の場合：

`javax.jms.JMSEException:`

CWSIA0053E: メソッド `JmsSessionImpl.` の呼び出し中に例外を受信しました

`com.ibm.wsspi.sib.core.exception.SIConnectionDroppedException:`

CWSIJ0044E: すでにクローズされた接続での操作が行われようとした。

ネットワーク障害の場合：

`javax.jms.JMSEException:`

CWSIA0067E: メソッド JmsMsgProducerImpl.close の呼び出し中に例外を受信しました:

com.ibm.ws.sib.jfapchannel.JFapConnectionBrokenException:

CWSIJ0056E: 予期しない条件により、ネットワーク接続がクローズしました。

メッセージング・エンジンのフェイルオーバーが完了した後のクライアントからのアクセスは、正常に行われます。

x). Session をクローズします

この時点で、メッセージング・エンジンが稼働しているプロセスやネットワークに障害が発生すると、HA マネージャーにより検知され、AMember02 のメッセージング・エンジンが起動します。

障害が発生した時点でアクセスが行われている JMS アプリケーションに対して、`javax.jms.JMSEException` が返されアプリケーションの実行は失敗します。返される `JMSEException` メッセージの全文は以下のとおりです。

プロセス障害の場合:

```
javax.jms.JMSEException:
CWSIA0053E: メソッド JmsSessionImpl. の呼び出し中に例外を受信しました
com.ibm.wsspi.sib.core.exception.SIConnectionDroppedException:
CWSIJ0044E: すでにクローズされた接続での操作が行われようとした。
```

ネットワーク障害の場合:

```
javax.jms.JMSEException:
CWSIA0067E: メソッド JmsMsgProducerImpl.close の呼び出し中に例外を受信しました:
com.ibm.ws.sib.jfapchannel.JFapConnectionBrokenException:
CWSIJ0056E: 予期しない条件により、ネットワーク接続がクローズしました。
```

メッセージング・エンジンのフェイルオーバーが完了した後のクライアントからのアクセスは、正常に行われます。

xi). Connection を切断します

この時点で、メッセージング・エンジンが稼働しているプロセスやネットワークに障害が発生すると、HA マネージャーにより検知され、AMember02 のメッセージング・エンジンが起動します。

障害が発生した時点でアクセスが行われている JMS アプリケーションに対して、`javax.jms.JMSEException` が返されアプリケーションの実行は失敗します。返される `JMSEException` メッセージの全文は以下のとおりです。

```
javax.jms.JMSEException:
CWSIA0022E: メソッド JmsConnectionImpl.close の呼び出し中に例外を受信しました
com.ibm.wsspi.sib.core.exception.SIConnectionDroppedException:
CWSIJ0044E: すでにクローズされた接続での操作が行われようとした。
```

メッセージング・エンジンのフェイルオーバーが完了した後のクライアントからのアクセスは、正常に行われます。

障害発生時に返されたエラーのハンドリング

JMS アプリケーションでは、メッセージング・エンジンが原因で起きる例外はすべて粒度の高い「`JMSEException`」として返されます。実際にアプリケーション的にリトライが可能かどうかは、`JMSEException` からエラー・コードを取得し、エラー・コード毎に判断する必要があります。

例外処理のコード例は以下のとおりです。

```
catch(JMSEException e) {
    String errorCode = e.getErrorCode();
```

```
    if (errorCode.equal("CWSIA0067")) {  
        //エラーハンドリングのコード  
    } else if(errorCode.equal("CWSIA0053")) {  
        //エラーハンドリングのコード  
    }  
}
```

7.3.3 パーティション構成における障害時の動作

パーティション構成で稼働している際に、プロセス障害やネットワーク障害が発生した場合どのように動作するのかについて説明します。

使用するメッセージング・エンジンと、同一 SIBus に所属するメッセージング・エンジンがある JMS アプリケーションであれば、メッセージング・エンジンが障害から復旧することで、ワークロードシェアリングも正常に稼働しますが、同一 SIBus に所属するメッセージング・エンジンがない場合、メッセージの偏りが発生する可能性があります。

ここでは、どのようにフェイルオーバーとメッセージの PUT や GET が行われるかについてのみ説明し、アプリケーションの動作フローにそってどのような例外が返されるかについては、クラスター構成時での解説と同じですので割愛します。また、説明を行う構成は、「7.2.1.3パーティション構成」と同じ構成です。

1) JMS アプリケーションが稼働するクラスターとメッセージング・エンジンが稼働するクラスターが同一クラスター

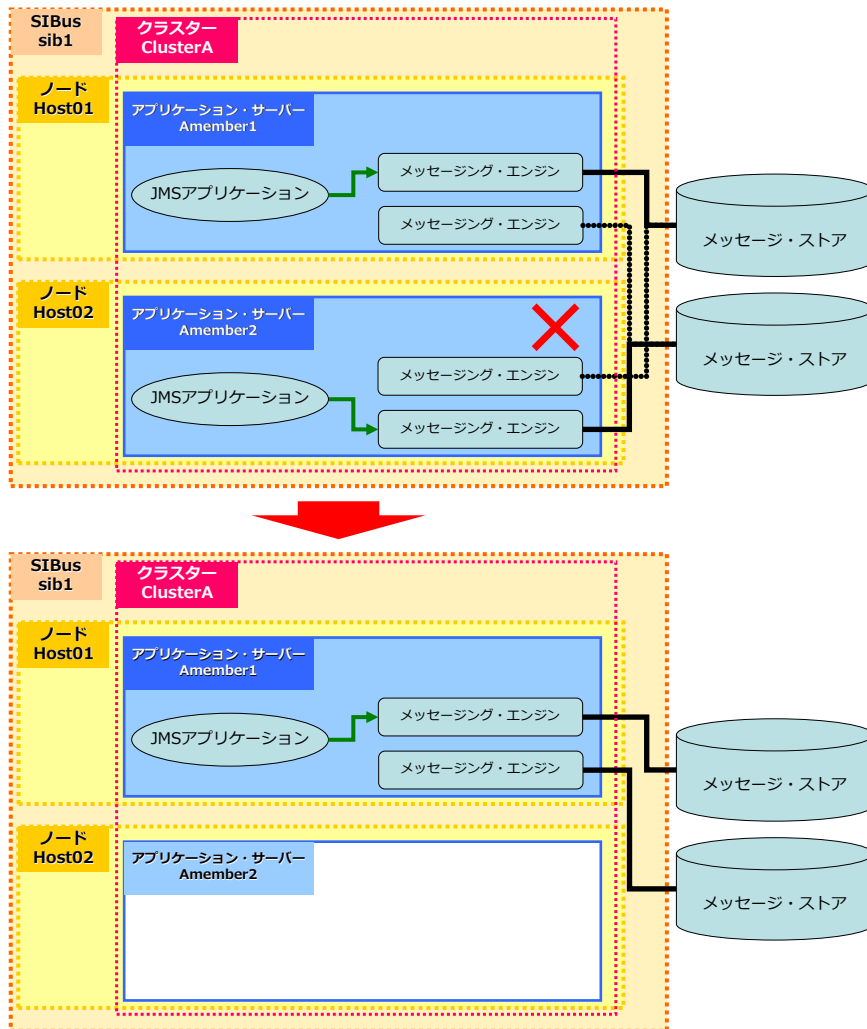


図 7-35 アプリケーションとメッセージング・エンジンが同一クラスター上で稼働している場合の障害時の挙動

JMS アプリケーションとメッセージング・エンジンが起動しているサーバーの1つがダウンすると、HA マネージャーにより検知され、稼働しているサーバー上でメッセージング・エンジンが起動します。このようにメッセージング・エンジンが、1サーバーで複数稼働することになりますが、最初に接続されているメッセージング・エンジンのみを使用します。

なお、障害が発生したサーバーが復旧した場合の動作は、メッセージング・エンジンに適用されているポリシーの設定により異なります。「フェイルバック」の設定が有効になっている場合、優先度の高いサーバーが復旧した時点で、そのサーバーにメッセージング・エンジンの稼働が戻ります。

2) JMS アプリケーションが稼動するクラスターとメッセージング・エンジンが稼動するクラスターが別クラスター

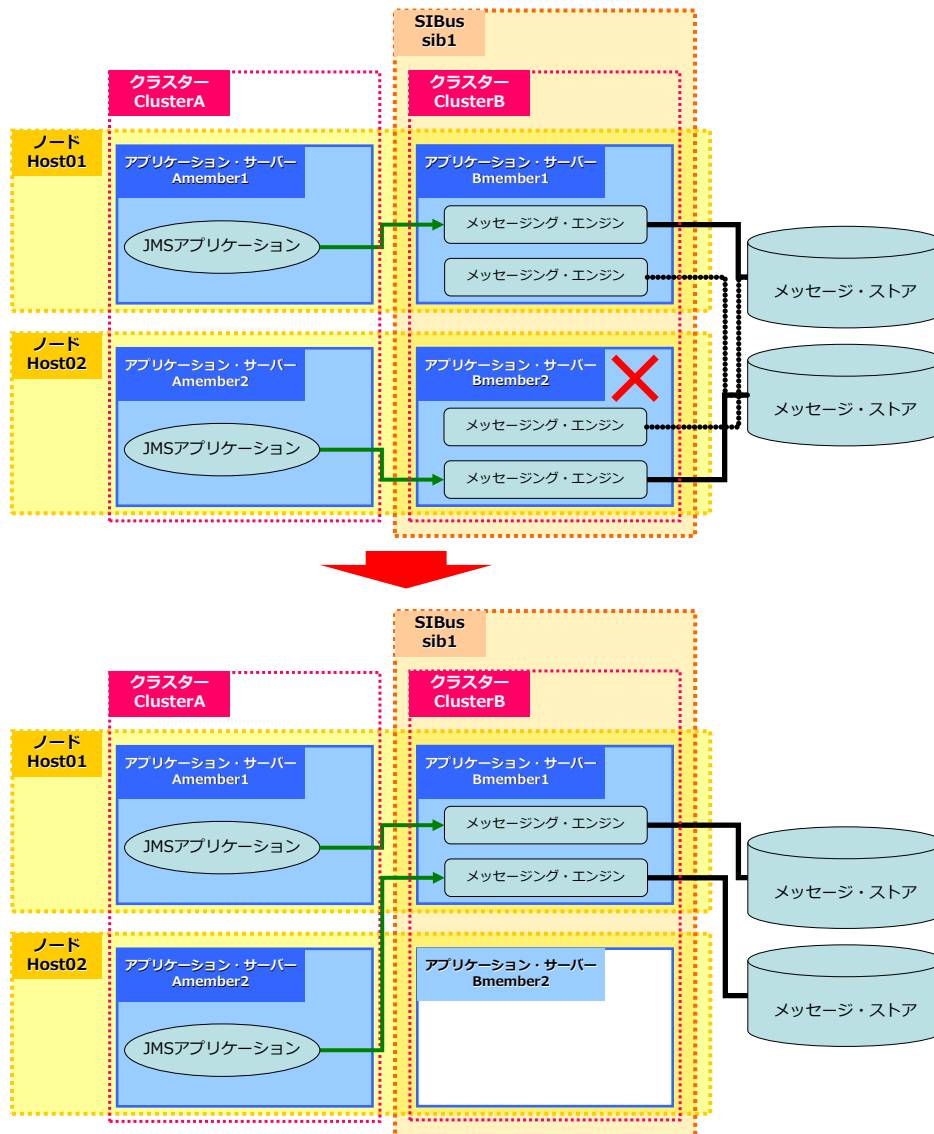


図 7-36 アプリケーションとメッセージング・エンジンが異なるクラスター上で稼働している場合の障害時の挙動

この構成の場合、最初の時点では JMS アプリケーションは、稼動しているサーバーと同じノードにあるメッセージング・エンジンを使用します。障害が発生すると、障害が発生したメッセージング・エンジンを使用していたアプリケーションは、障害が発生していない側のサーバーが使用しているメッセージング・エンジンを使用します。

なお、障害が発生したサーバーが復旧した場合の動作は、メッセージング・エンジンに適用されているポリシーの設定により異なります。「フェイルバック」の設定が有効になっている場合、優先度の高いサーバーが復旧した時点で、そのサーバーにメッセージング・エンジンの稼働が戻ります。

3) JMS アプリケーションが稼動するクラスターとメッセージング・エンジンが稼動するクラスターが別ノード、かつ、同一 SIBus に所属する

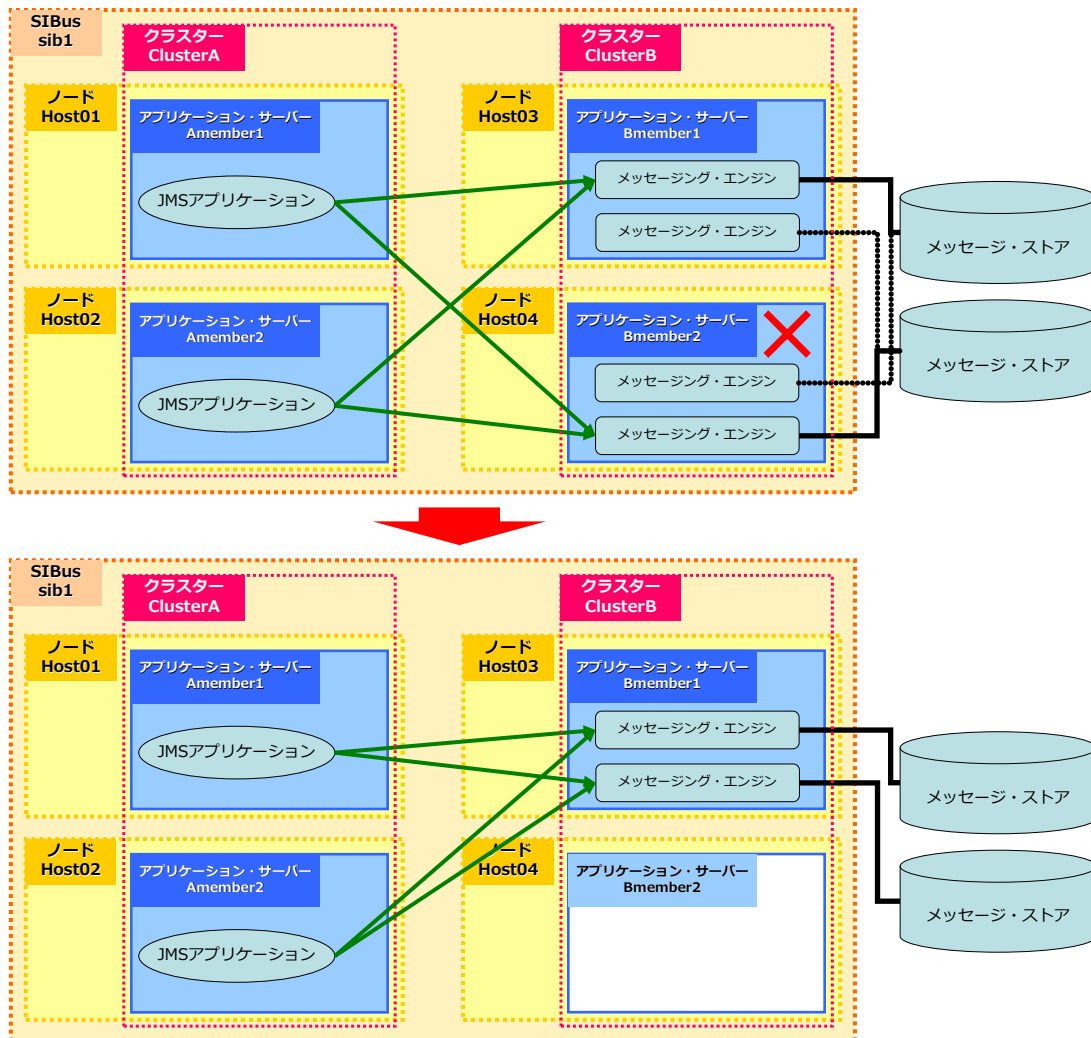


図 7-37 アプリケーションとメッセージング・エンジンが同一 SIBus 上の異なるクラスター上で稼働している場合の障害時の挙動

この構成の場合、それぞれの JMS アプリケーションは、すべてのメッセージング・エンジンに対して、ラウンドロビンでメッセージを PUT/GET します。障害が発生することにより、1サーバーに複数メッセージング・エンジンが起動してもラウンドロビンにてメッセージを PUT/GET します。

なお、障害が発生したサーバーが復旧した場合の動作は、メッセージング・エンジンに適用されているポリシーの設定により異なります。「フェイルバック」の設定が有効になっている場合、優先度の高いサーバーが復旧した時点で、そのサーバーにメッセージング・エンジンの稼働が戻ります。

4) JMS アプリケーションが稼動するクラスターとメッセージング・エンジンが稼動するクラスターが別ノード、かつ、同一 SIBus に所属しない

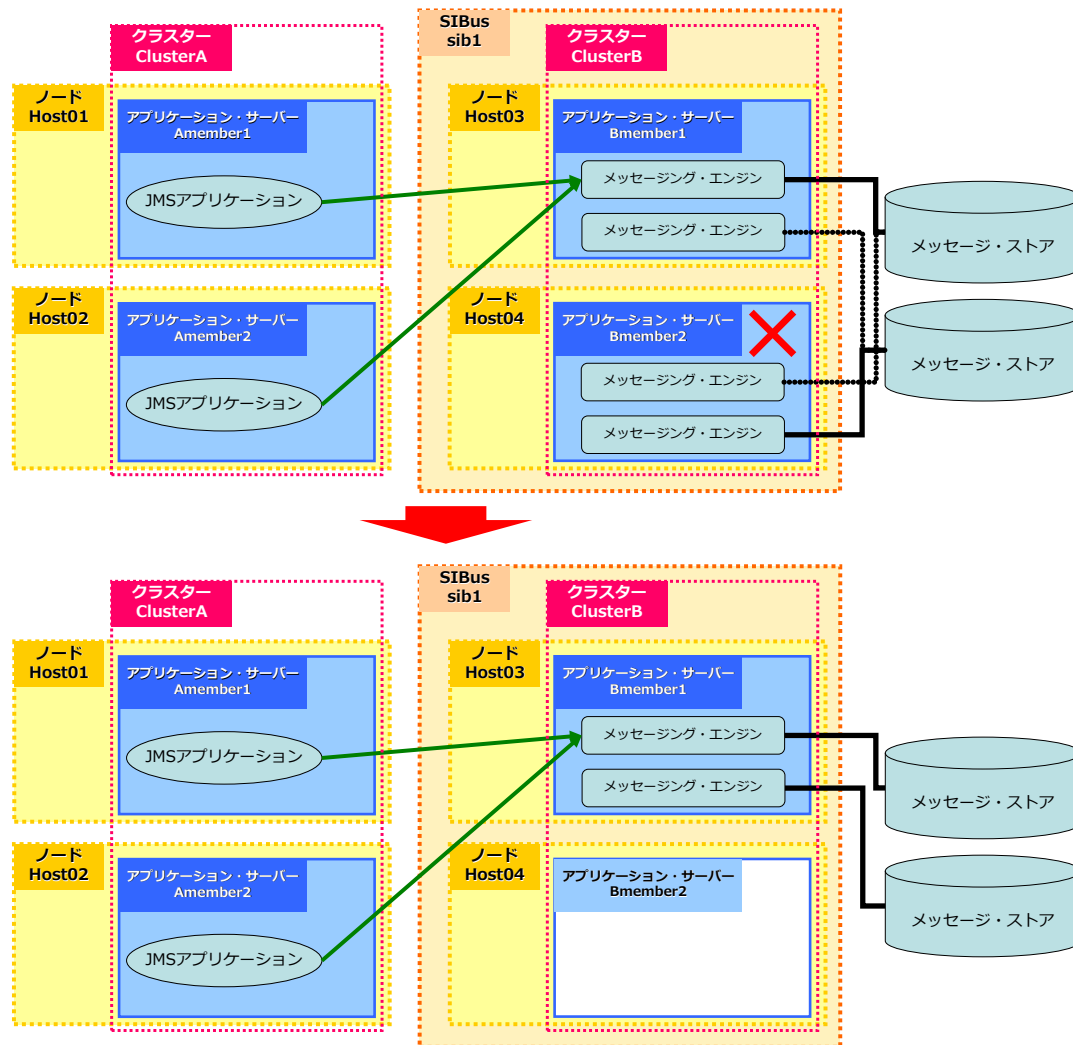


図 7-38 アプリケーションとメッセージング・エンジンが異なるクラスター、異なるノード上で稼働している場合の障害時の挙動

この構成の場合、最初の時点ではすべての JMS アプリケーションは、最初に使用されたメッセージング・エンジンのみを使用します。障害が発生すると、障害が発生したメッセージング・エンジンを使用していたアプリケーションは、障害が発生していない側のサーバーが使用しているメッセージング・エンジンを使用します。

なお、障害が発生したサーバーが復旧した場合の動作は、メッセージング・エンジンに適用されているポリシーの設定により異なります。「フェイルバック」の設定が有効になっている場合、優先度の高いサーバーが復旧した時点で、そのサーバーにメッセージング・エンジンの稼働が戻ります。

7.3.4 ネットワーク障害時固有の考慮点

メッセージ・ストアにデータベース(DB2)を使用している場合、ネットワーク障害発生時に DB2 はネットワーク障害を検知できません。そのためテーブルロックが解除されず、フェイルオーバーにより起動するメッセージング・エンジンがパーシスタンス・データベースを使用できません。その結果、メッセージング・エンジンのフェイルオーバーが失敗することがあります。

DB2 V9.7 FP5 以上/DB2 V10.1 FP2 以上に相当するバージョンの JDBC ドライバーを使用している場合は、JDBC ドライバーのパラメーターである `keepAliveTimeout`(デフォルト 15 秒)の値を調整することで、WAS でデータベース障害を検知するまでのタイムアウトを設定できます。詳細は、以下のガイドを参照してください。

【参考】「WAS-DB2 接続における TCP KeepAlive タイムアウト設定について(WAS-13-002)」

<http://www-01.ibm.com/support/docview.wss?uid=jpn1J1010585>

上記以外の DB2 のバージョンの JDBC ドライバーを使用する場合は、DB2 がネットワーク障害を検知してテーブルロックを解除するためには、DB2 が稼働しているノードの TCP/IP タイムアウトの値を、調整する必要があります。そのため、ネットワーク障害を検知するには、パースistent・データベースが稼働しているノードの TCP/IP タイムアウトの値を、HA マネージャーによるネットワーク障害検知とほぼ同じ時間になるように調整します。

AIX の場合、ネットワーク障害検知までの時間は下記の様に構成されます。

$$(\text{tcp_keepcnt} \times \text{tcp_keepintvl}) + \text{tcp_keepidle} \div 2 = \text{ネットワーク障害検知までの時間}$$

各項目の意味は下記ようになります。

表 7-16 ネットワーク障害検知までの時間を求めるための各項目の意味

項目名	意味
tcp_keepcnt	キープアライブ・プローブをいくつ送信したら接続を終了するか (デフォルト設定:8)
tcp_keepidle	アイドル TCP 接続をアクティブにしておく時間 (デフォルト設定:14400 <2 時間>)
tcp_keepintvl	TCP 接続を検査するため、何秒おきにパケットを送信するか (デフォルト設定:150 <75 秒>)

この設定は、下記のコマンドにて行います。

```
# no -o [tcp_keepcnt/tcp_keepidle/tcp_keepintvl] = value
```

HA マネージャーによるネットワーク障害検知までの時間は、ハートビート・タイムアウト期間にて設定します。設定方法については、「3-5 HA マネージャー」を参照してください

7.4 JMS 接続設定

7.4.1 デフォルト・メッセージング・プロバイダー

メッセージング・エンジンの宛先に対してメッセージを送受信する場合に必要な、メッセージ・リソースを設定します。メッセージ・リソースの設定は、JMS のバージョンにより異なります。使用する設定と JMS のバージョンは、下記ようになります。

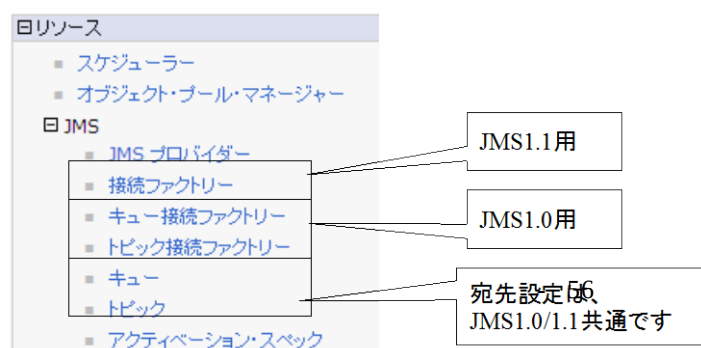


図 7-39 デフォルトのメッセージング・プロバイダーの設定箇所

- JMS 接続ファクトリー
 - ✓ JMS1.0 の場合

JMS 接続ファクトリーは、キュー接続用の「JMS キュー接続ファクトリー」とトピック接続用の「JMS トピック接続ファクトリー」に分かれており、用途に応じて構成する必要があります。
 - ✓ JMS1.1 の場合

JMS 接続ファクトリーは、キュー接続用およびトピック接続用に関わらず、「JMS 接続ファクトリー」を使用することもできます。
- 宛先

宛先の設定は、JMS1.0/1.1 ともに共通です。

7.4.1.1 JMS 接続ファクトリーの設定

JMS 1.1 を使用してメッセージング・エンジン接続を行う際に使用する、JMS 接続ファクトリーを設定します。JMS 1.1 では、キューを使用する場合でも、トピックを使用する場合でも、使用する接続ファクトリーは共通です。

- 管理コンソールのメニューより、[リソース]→[JMS]→[接続ファクトリー]を選択します
- 有効範囲として、ここでは作成したクラスター「ClusterA」を指定します

接続ファクトリー

接続ファクトリーは、関連した JMS プロバイダーへの接続を作成するために使用されます。これらの接続ファクトリーは、JMS キューおよび JMS トピック宛先にアクセスするために使用できます。

☐ 有効範囲: セル=**guideCell01**, クラスター=**ClusterA**

☒ すべての有効範囲オプションがある「有効範囲選択」ドロップダウン・リストを表示

有効範囲は、リソース定義が可視となるレベルを指定します。有効範囲とその機能に関する詳細は、[有効範囲設定のヘルプを参照してください。](#)

クラスター=ClusterA

設定

新規作成 削除

選択	名前	JNDI 名	プロバイダー	説明	有効範囲
なし。					
合計 0					

図 7-40 接続ファクトリーの有効範囲設定

- 有効範囲に「ClusterA」が表示されていることを確認したら、「新規作成」をクリックします

- iv) JMS リソース・プロバイダーの選択では、「デフォルトのメッセージング・プロバイダー」を選択し、「OK」をクリックします

図 7-41 JMS リソース・プロバイダーの選択

- v) JMS 接続ファクトリーを下記のとおり設定します

表 7-17 JMS 接続ファクトリーの設定 (管理)

項目	説明
名前	JMS 接続ファクトリーの識別名称を設定します
JNDI 名	JMS 接続ファクトリーの JNDI 名を設定します

表 7-18 JMS 接続ファクトリーの設定 (接続)

項目	説明
バス名	接続先のバス名を指定します
ターゲット	メッセージング・エンジンのグループを識別するターゲットの名前を指定します
ターゲット・タイプ	「ターゲット」で指定されたターゲットのタイプ
ターゲット重要度	「ターゲット」で指定されるメンバーのみを使用するかを指定します
ターゲット・インバウンド・トランスポート・チェーン	メッセージング・エンジンの名前解決に使用されるトランスポート・チェーンを指定します
プロバイダー・エンドポイント	メッセージング・エンジンのためのブート・ストラップ・ポートを指定します
接続の接近性	メッセージング・エンジンを検索する範囲を指定します

表 7-19 JMS 接続ファクトリーの設定 (永続サブスクリプション)

項目	説明
クライアント ID	永続トピック・サブスクリプションに必要な固有の JMS クライアント ID を指定します
永続サブスクリプション・ホーム	永続サブスクリプションに配信されるメッセージを保管するために使用されるメッセージング・エンジンの名前を指定します

表 7-20 JMS 接続ファクトリーの設定 (サービスの品質)

項目	説明
非パーシスタント・メッセージの信頼性	非永続 JMS メッセージに適用される信頼性を指定します
パーシスタント・メッセージの信頼性	永続 JMS メッセージに適用される信頼性を指定します

7.4.1.2 JMS キュー接続ファクトリーの設定

JMS1.0 で Point-to-Point のメッセージングを使用する場合は、キュー接続ファクトリーを取得する必要があります。

- 管理コンソールのメニューより、[リソース]→[JMS]→[キュー接続ファクトリー]を選択します
- 「7.4.1.1 JMS 接続ファクトリーの設定」と同様に設定内容を入力し、キュー接続ファクトリーを作成します

7.4.1.3 JMS トピック接続ファクトリーの設定

JMS1.0 で Publish/Subscribe のメッセージングを使用する場合は、トピック接続ファクトリーを取得する必要があります。

- 管理コンソールのメニューより、[リソース]→[JMS]→[トピック接続ファクトリー]を選択します
- 「7.4.1.1 JMS 接続ファクトリーの設定」と同様に設定内容を入力し、トピック接続ファクトリーを作成します

7.4.1.4 JMS キューの設定

JMS 1.0/1.1 で Point-to-Point のメッセージを送るときに使用するキューを設定します。

- 管理コンソールのメニューより、[リソース]→[JMS]→[キュー]を選択します
- 有効範囲として、ここでは作成したクラスター「ClusterA」を指定します

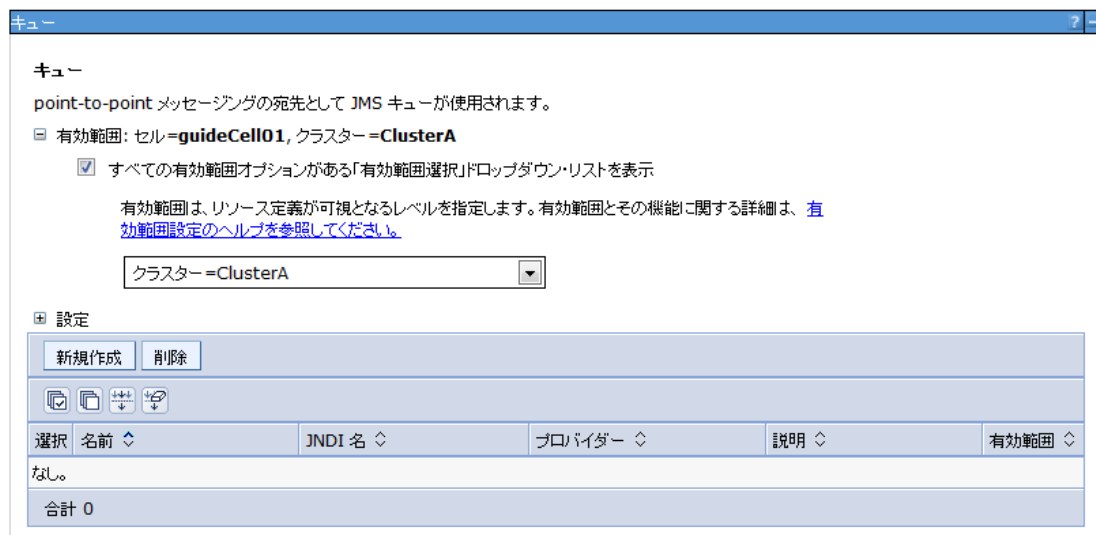


図 7-42 キューの有効範囲設定

- 有効範囲に「ClusterA」が表示されていることを確認したら、「新規作成」をクリックします

- iv) JMS リソース・プロバイダーの選択では、「デフォルトのメッセージング・プロバイダー」を選択し、「OK」をクリックします

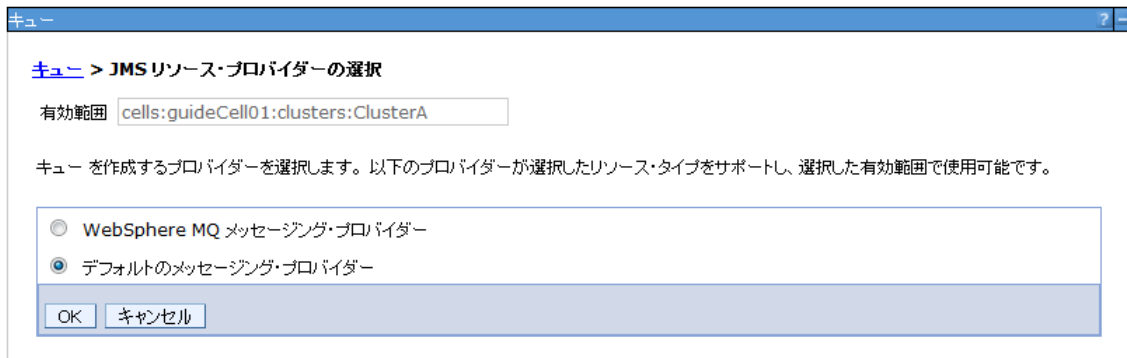


図 7-43 JMS リソース・プロバイダーの選択

- v) JMS キューを下記のとおり設定します

表 7-21 JMS キューの設定 (管理)

項目	説明
名前	JMS キューの識別名称を設定します
JNDI 名	JMS キューの JNDI 名を設定します

表 7-22 JMS キューの設定 (接続)

項目	説明
キュー名	メッセージング・エンジンに定義した宛先を指定します
バス名	キュー名で指定する宛先を持つバス名を指定します
デリバリー・モード	この宛先に送信されるメッセージのデリバリー・モードを指定します
存続期間	メッセージが到着してからの存続時間を指定します
優先順位	この宛先に送信されるメッセージの優先順位を指定します

表 7-23 JMS キューの設定 (拡張)

項目	説明
先読み	メッセージ配信時に先読み最適化を行うかどうかを指定します

表 7-24 JMS キューの設定 (複数のキュー・ポイント間でのメッセージ制御)

項目	説明
接続されたメッセージング・エンジンにキュー・ポイントが構成されている場合、メッセージをローカル・キュー・ポイントに制限する	同じ JVM のメッセージ・プロデューサーとメッセージング・エンジン、または、メッセージ・コンシューマーとメッセージング・エンジンを紐付け、それ以外への送信・受信を行わないかどうかを指定します

表 7-25 JMS キューの設定 (MessageProducer ごとに複数のキュー・ポイント間で制御)

項目	説明
ローカル・キュー・ポイント設定	
ローカル・キュー・ポイントへのメッセージ送信を優先	ローカルのキュー・ポイントへの送信を優先するかどうかを指定します。ローカルへ正常に送信できない場合に、ローカル以外のキュー・ポイントへ送信されます。
他のキュー・ポイントよりもローカル・キュー・ポイントを優先しない	ローカルのキュー・ポイントだけでなく、それ以外のキュー・ポイントに対しても均等に送信するかどうかを指定します
キュー・ポイント間でのメッセージのアフィニティ	
すべてのメッセージを同じキュー・ポイントに送信	メッセージのロード・バランスが設定され、同じメッセージ・プロデューサーのインスタンスが複数回続けてメッセージを送信する場合、メッセージを同じキュー・ポイントに送信するかどうかを指定します
メッセージは別のキュー・ポイントに送信可能	メッセージのロード・バランスが設定され、同じメッセージ・プロデューサーのインスタンスが複数回続けてメッセージを送信する場合、メッセージを別のキュー・ポイントに送信するかどうかを指定します

表 7-26 JMS キューの設定 (MessageConsumer または QueueBrowser ごとに複数のキュー・ポイント間で制御)

項目	説明
メッセージの可視性	
単一のキュー・ポイントのメッセージのみが表示される	ローカルのキュー・ポイントだけの処理を行うかどうかを指定します
すべてのキュー・ポイントのメッセージが表示される	ローカル以外のキュー・ポイントの処理を行うかどうかを指定します

「複数のキュー・ポイント間でのメッセージ制御」の設定項目について説明します。

「接続されたメッセージング・エンジンにキュー・ポイントが構成されている場合、メッセージをローカル・キュー・ポイントに制限する」のチェックボックスを ON に設定した場合、表 7-25や表 7-26で設定する内容よりも、ローカル・キュー・ポイントでの設定が優先されます。ローカル・キュー・ポイントとは、接続先のメッセージング・エンジン上で構成される SIBus のキュー・ポイントのことです。

WAS V7.0 以降では、「他のキュー・ポイントよりもローカル・キュー・ポイントを優先しない」を設定すると、ローカルのキュー・ポイントだけでなく、それ以外のキュー・ポイントに対しても均等にメッセージが割り振られます。

「キュー・ポイント間でのメッセージのアフィニティ」は、「ローカル・キュー・ポイント設定」にて「他のキュー・ポイントよりもローカル・キュー・ポイントを優先しない」が設定されていることが前提です。この時、「すべてのメッセージを同じキュー・ポイントに送信」を設定すると、メッセージを同

じキュー・ポイントに送信することが可能です。「メッセージは別のキュー・ポイントに送信可能」を設定すると、常にラウンドロビンで送信することができます。

また、「ローカル・キュー・ポイント設定」および「キュー・ポイント間でのメッセージのアフィニティー」は、キューの宛先がローカル・キューだけではなく、別名キューの場合にも有効です。

「すべてのキュー・ポイントのメッセージが表示される」を設定すると、ローカルのキュー・ポイント以外のキュー・ポイントのデータが確認でき、すべてのキュー・ポイントのデータを処理することが可能です。「単一のキュー・ポイントのメッセージのみが表示される」を設定すると、ローカルのキュー・ポイントのデータのみを処理することができます。

7.4.1.5 JMS トピックの設定

JMS 1.0/1.1 で Publish/Subscribe のメッセージを送るときに使用するトピックを設定します。

- i) 管理コンソールのメニューより、[リソース]→[JMS]→[トピック]を選択します
- ii) 有効範囲として、ここでは作成したクラスター「ClusterA」を指定します

図 7-44 トピックの有効範囲設定

- iii) 有効範囲に「ClusterA」が表示されていることを確認したら、「新規作成」をクリックします
- iv) JMS リソース・プロバイダーの選択では、「デフォルトのメッセージング・プロバイダー」を選択し、「OK」をクリックします

図 7-45 JMS リソース・プロバイダーの選択

- v) JMS トピックを下記のとおり設定します

表 7-27 JMS トピックの設定 (管理)

項目	説明
名前	JMS トピックの識別名称を設定します
JNDI 名	JMS トピックの JNDI 名を設定します

表 7-28 JMS トピックの設定 (接続)

項目	説明
トピック名	トピックの割り当て先のトピックの名前を指定します
トピック・スペース	メッセージング・エンジンに定義したトピック・スペースを指定します
バス名	トピック・スペースで指定したトピック・スペース宛先を持つバス名を指定します
デリバリー・モード	この宛先に送信されるメッセージのデリバリー・モードを指定します
存続時間	メッセージが到着してからの存続時間を指定します
メッセージ優先順位	この宛先に送信されるメッセージの優先順位を指定します

表 7-29 JMS トピックの設定 (拡張)

項目	説明
先読み	メッセージ配信時に先読み最適化を行うかどうかを指定します

7.4.1.6 JMS アクティベーション・スペックの設定

JMS 1.1 の MDB を使用するために必要な、「JMS アクティベーション・スペック」を設定する方法を紹介します。

- 管理コンソールのメニューより、[リソース]→[JMS]→[アクティベーション・スペック]を選択します
- 有効範囲として、ここでは作成したクラスター「ClusterA」を指定します

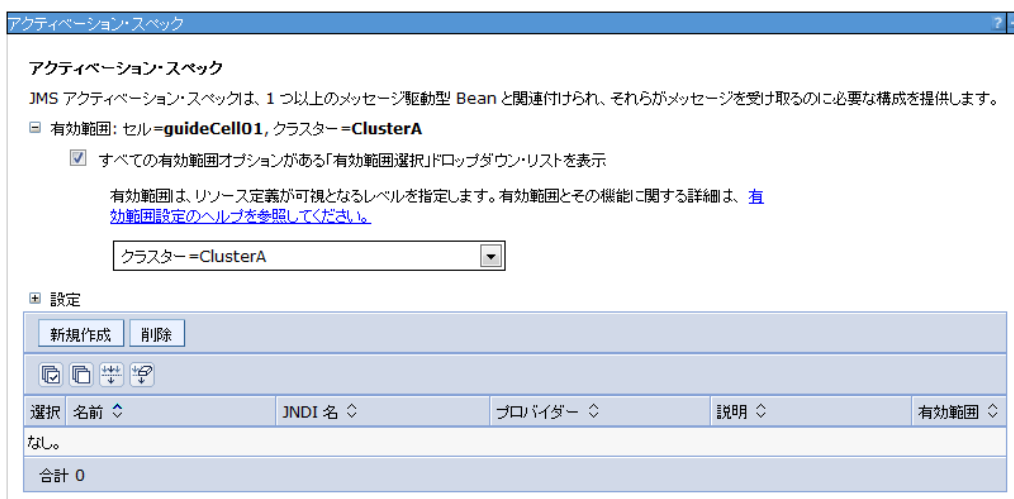


図 7-46 アクティベーション・スペックの有効範囲設定

- 有効範囲に「ClusterA」が表示されていることを確認したら、「新規作成」をクリックします

- iv) JMS リソース・プロバイダーの選択では、「デフォルトのメッセージング・プロバイダー」を選択し、「OK」をクリックします

図 7-47 JMS リソース・プロバイダーの選択

- v) JMS アクティベーション・スペックを下記のとおり設定します

表 7-30 JMS アクティベーション・スペックの設定 (管理)

項目	説明
名前	JMS アクティベーション・スペックの識別名称を設定します
JNDI 名	JMS アクティベーション・スペックの JNDI 名を設定します

表 7-31 JMS アクティベーション・スペックの設定 (宛先)

項目	説明
宛先タイプ	MDB が監視する宛先のタイプを指定します
宛先 JNDI 名	MDB が監視する宛先の JNDI 名を指定します
メッセージ・セレクター	メッセージ・ヘッダー内のフィールド、および、メッセージ・プロパティ内のフィールドを元に、取得するメッセージを選択する場合に設定します
バス名	宛先 JNDI 名で指定される宛先が存在するバス名を指定します
確認応答モード	MDB が受信したメッセージをどのように応答するかを指定します
ターゲット	メッセージング・エンジンのグループを識別する名前を指定します
ターゲット・タイプ	「ターゲット」のタイプを指定します
ターゲット重要度	ターゲット・グループ内のメッセージング・エンジンが使用できない場合、ターゲット・グループ外の同一バス上の他のメッセージング・エンジンに接続するかどうかを指定します
ターゲット・インバウンド・トランスポート・チェーン	監視対象の宛先を持つバス・メンバーとして MDB が稼働するアプリケーション・サーバーが含まれていない場合に、接続先となるメッセージング・エンジンを指定します
プロバイダー・エンドポイント	ローカルのセル内のバスに接続できない場合に、リモートのセルのブートストラップ・サーバーを指定します

表 7-32 JMS アクティベーション・スペックの設定 (追加)

項目	説明
最大バッチ・サイズ	単一バッチで受信するメッセージの最大数を指定します
エンドポイントごとの MDB の最大並行呼び出し数	メッセージが同時に配信されるエンドポイントの最大数を指定します
繰り返されるメッセージの障害時にエンドポイントを自動停止する	
使用可能	エンドポイントを自動的に停止するかどうかを指定します
連続障害メッセージのしきい値	エンドポイントを自動的に停止する際の連続して失敗したメッセージ数を指定します
障害のあるメッセージの再試行間の遅延	連続障害メッセージしきい値に達していない MDB 処理に失敗したメッセージを、再試行するまでの経過時間を指定します

表 7-33 JMS アクティベーション・スペックの設定 (サブスクリプション耐久性)

項目	説明
サブスクリプション耐久性	トピック・サブスクリプションが永続的か非永続的かを指定します
サブスクリプション名	サブスクリプション名を指定します
クライアント ID	永続トピック・サブスクリプションに必要な固有の JMS クライアント ID を指定します
永続サブスクリプション・ホーム	永続サブスクリプションに配信されるメッセージを保管するために使用される、メッセージング・エンジンの名前を指定します

表 7-34 JMS アクティベーション・スペックの設定 (参照によるメッセージ・ペイロードの受け渡し)

項目	説明
この接続ファクトリーを使用してメッセージを送信するアプリケーション	接続ファクトリーを使用してバスに接続しているアプリケーションから送信されたメッセージを、設定時にコピーせず、必要時にシステムによってメッセージデータのみをシリアライズするかを指定します
この接続ファクトリーを使用してメッセージを受信するアプリケーション	接続ファクトリーを使用してバスに接続しているアプリケーションから受信されたメッセージを、必要時にシステムによってメッセージデータのみをシリアライズするかを指定します

表 7-35 JMS アクティベーション・スペックの設定 (拡張)

項目	説明
永続サブスクリプションを共用	永続サブスクリプションが、接続を越えてサーバー・クラスターのメンバーとの間で共用されるかどうかを指定します
CMP とデータ・ソースを共用	JMS と CMP Entity Bean 間の接続の共用を許可するかどうかを指定します

先読み	コンシューマーにメッセージを先制して割り当てる最適化を行うかどうかを指定します
常にすべてのサーバーで MDB を活動化する	同一のバス内にて、メッセージ・エンジンがアクティブであるアプリケーション・サーバー上で稼動している MDB アプリケーションのみメッセージ処理を行うのか、メッセージ・エンジンがアクティブでないアプリケーション・サーバー上で稼動している MDB アプリケーションからもメッセージ処理を行うのかを指定します
再試行間隔	メッセージング・エンジンへ接続した後、最適なメッセージング・エンジンへの接続を試みる経過時間を指定します

表 7-36 JMS アクティベーション・スペックの設定 (セキュリティ)

項目	説明
認証別名	アクティベーション・スペックを使用する際の J2C 認証エイリアスを指定します

7.4.1.7 MDB を使用する際の構成

MDB の構成には、アクティベーション・スペックとリスナー・ポートのどちらかを使用可能です。どちらを使用するかを選択指針に関しては、以下の Information Center を参照してください。ここでは、アクティベーション・スペックを利用した構成について説明します。

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/index.jsp?topic=/com.ibm.websphere.nd.doc/ae/cmb_aslp.html

MDB の管理機能

アプリケーションやリソース・アダプターは起動した状態のまま、メッセージ・エンドポイントの一時停止と再開が可能です。また、アクティベーション・スペックに、障害メッセージのしきい値を設定し、しきい値を超えるとメッセージ・エンドポイントを自動で停止させることが可能です。この機能追加により、例えば、エラー・メッセージが大量に送信された場合に、例外宛先があふれるのを防いだり、他のリソースへの影響を抑えたりすることができます。

メッセージ・エンドポイントの一時停止と再開は、管理コンソールより、[アプリケーション]→[WebSphere エンタープライズ・アプリケーション]→[アプリケーション名]→[ランタイムタブ]→[メッセージ・エンドポイントの管理]を選択し、[一時停止]および[再開]をクリックします。

一時停止 再開		
選択	名前 (アクティベーション・スペック) ⇅	実行中オブジェクトの有効範囲 ⇅
次のリソースを管理できます。		
☐	GetMsgMDB_J2CMessageEndpoint (jms/as1)	セル : guideCell01 ノード: host1Node01 サーバー : Amember11
合計 1		

図 7-48 メッセージ・エンドポイントの管理

- メッセージ・エンドポイントを一時停止した時の SystemOut.log へのログ出力

```
[08/12/03 20:23:48:995 JST] 0000003c ActivationSpec I J2CA0524I: ActivationSpec
jms/as1 (com.ibm.ws.sib.api.jmsra.impl.JmsJcaActivationSpecImpl)および MDB Appli-
cation GetMsg#GetMsgEJB.jar#GetMsgMDB の Message Endpoint が非活動化されま
した。
```

- メッセージ・エンドポイントを再開した時の SystemOut.log へのログ出力

```
[08/12/03 20:24:01:176 JST] 0000003c ActivationSpec I J2CA0523I: ActivationSpec
jms/as1 (com.ibm.ws.sib.api.jmsra.impl.JmsJcaActivationSpecImpl)および MDB Appli-
cation GetMsg#GetMsgEJB.jar#GetMsgMDB の Message Endpoint が活動化されま
した。
```

障害メッセージのしきい値は、管理コンソールより、[リソース]→[JMS]→[アクティベーション・スペック]→[アクティベーション・スペック名]を選択し、[繰り返されるメッセージの障害時にエンドポイントを自動停止する]の[使用可能にする]をチェックし、[連続障害メッセージのしきい値]を指定します。設定方法については、表 7-32を参照してください。

- 連続障害メッセージのしきい値に達した時の SystemOut.log へのログ出力

```
[08/12/03 21:47:29:069 JST] 0000003a SibMessage W [:] CWSIV0902W: 宛先
[ QueueDestination, false, null, sib1 ] を listen していたメッセージ・エンドポイント
MDBApplication#MDBEJB.jar#TestMDB は、システムによって一時停止されました。

[08/12/03 21:47:29:091 JST] 0000003a ActivationSpec I J2CA0524I: ActivationSpec
jms/as1 (com.ibm.ws.sib.api.jmsra.impl.JmsJcaActivationSpecImpl)および MDB Appli-
cation MDBApplication#MDBEJB.jar#TestMDB の Message Endpoint が非活動化され
ました。
```

MDB を使用したトポロジー

同一の SIBus 内にて、メッセージング・エンジンがアクティブでないアプリケーション・サーバー上のアプリケーションからメッセージを取得することが可能です。この構成は図 7-49 のようになり、管理コンソールのツリー・ビュー上で、[リソース]→[JMS]→[アクティベーション・スペック]→[アクティベーション・スペック名]を選択し、[常にすべてのサーバーで MDB を活動化する]をチェックします。設定方法については、表 7-35 を参照して下さい。図 7-49 の場合、アプリケーション・サーバー: A member1 のメッセージ・エンジンがアクティブの場合でも、アプリケーション・サーバー: A member2 で稼動している MDB 用のアプリケーションからメッセージを取得することができます。

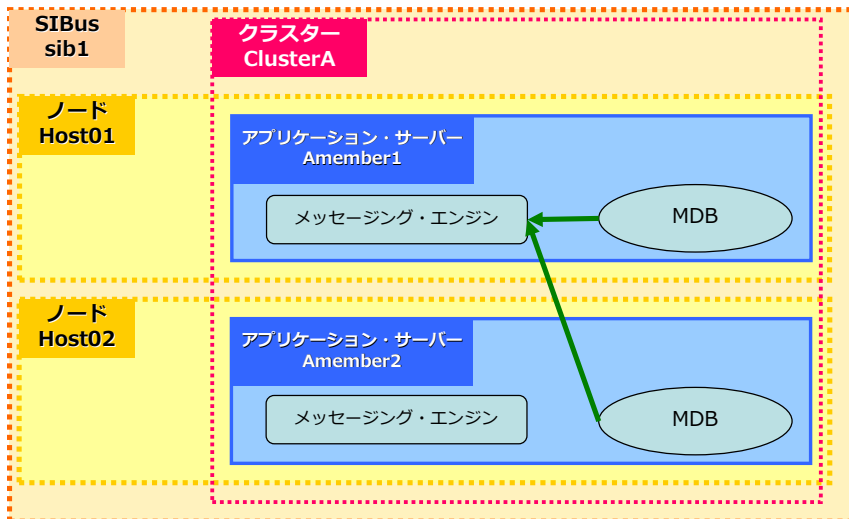


図 7-49 すべての MDB を活動化した場合のトポロジー

7.4.1.8 接続プールの設定

JMS プロバイダーとしてデフォルト・メッセージング・プロバイダーを利用した場合、Java EE Connector Architecture (JCA または J2C) の仕様に基づいたコネクション・プーリング機能が利用可能です。この機能により、JMS クライアントの SIBus 接続処理のオーバーヘッドを緩和できます。

JMS クライアントの場合

[接続ファクトリー](#) > [JMSConnectionFactory](#) > [接続プール](#)

このページを使用して、アプリケーションのパフォーマンスに影響を与える可能性のある、接続管理タスクのタイミングに影響するプロパティを設定します。デフォルト値は慎重に考慮してください。アプリケーション要件によっては、これらの値の変更が必要な場合があります。

構成

一般プロパティ

有効範囲
cells:guideCell01:clusters:ClusterA

* 接続タイムアウト
180 秒間

* 最大接続数
10 接続

* 最小接続数
1 接続

* リープ時間
180 秒間

* 未使用タイムアウト
1800 秒間

* 経過時間タイムアウト
0 秒間

ページ・ポリシー
EntirePool

適用 OK リセット 取り消し

追加プロパティ

- [拡張接続プール・プロパティ](#)
- [接続プール・カスタム・プロパティ](#)

図 7-50 JMS クライアントの場合の接続プールの設定

JMS クライアントの SIBus 接続は「J2C 接続プール」に保持され、その後の接続要求に対して再利用されます。管理コンソールの[リソース]→[JMS]→[接続ファクトリー]→[接続ファクトリー名]→[追加プロパティ:接続プール]から最大接続数などの項目が設定可能です。

MDB の場合

[アプリケーション・サーバー](#) > [Amember1](#) > [スレッド・プール](#) > [Default](#)

このページを使用して、使用するサーバー用のスレッド・プールを指定します。サーバー・コンポーネントは、スレッド・プールによって、実行時に新規のスレッドを作成する代わりに、スレッドを再利用できます。新規スレッドの作成は通常、時間とリソースを多く使用する処理です。

構成

一般プロパティ

* 名前
Default

説明

* 最小サイズ
20 スレッド

* 最大サイズ
20 スレッド

* スレッド非活動タイムアウト
5000 ミリ秒

☐ 最大スレッド・サイズを超えたスレッド割り振り許可

適用 OK リセット キャンセル

追加プロパティ

- [カスタム・プロパティ](#)

図 7-51 MDB の場合の接続プールの設定

MDB と SIBus のコネクションは、サーバーの「スレッド・プール」の「Default」に保持されます。管理コンソールの[WebSphere Application Server]→[サーバー名]→[追加プロパティ:スレッド・プール]→[Default]から最大サイズなどが設定できます。

7.4.2 WebSphere MQ メッセージング・プロバイダー

WebSphere MQ (WMQ) を JMS プロバイダーとして利用することが出来ます。アプリケーションは、WMQ のキュー・マネージャー (Qmgr) に対して接続し、キューにメッセージを送信/受信出来ます。WAS の JMS プロバイダーとして、WMQ を使用するためには、WMQ の製品を別途購入/インストールする必要があります。

WMQ への接続方式

WMQ を JMS プロバイダーとして利用するには、2 種類の接続方法があります。

- バインディング接続

アプリケーション・サーバーと Qmgr が、同じ筐体 (OS 上で動作している場合、バインディング接続が利用出来ます。Qmgr に Inter-Process Communication (IPC) を利用して直接アクセスするため、クライアント接続より高速です。また、MQ クラスターの負荷分散機能が利用可能であるなど、クライアント接続より多くの機能を利用できます。

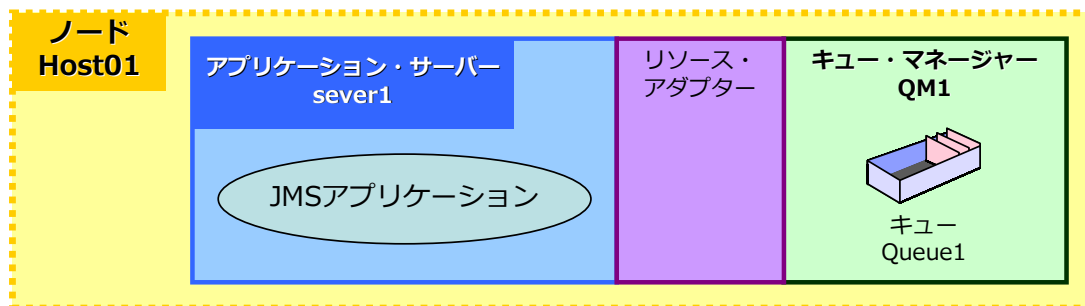


図 7-52 バインディング接続

- クライアント接続

Qmgr に対して、TCP/IP を利用してチャネル経由で接続します。基本的にバインディング接続と比べると低速で、利用できる機能も制限されます。MQ クライアントでは、Qmgr のサーバー・接続チャネル名を指定します。

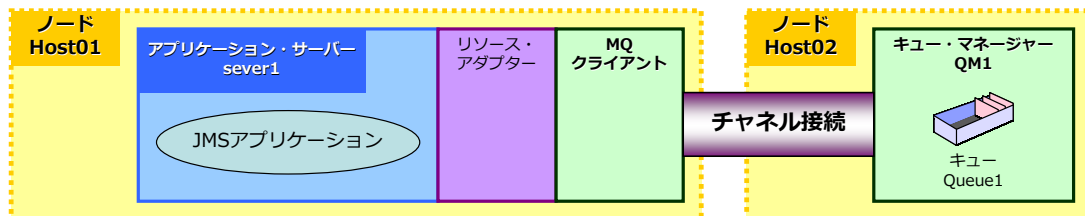


図 7-53 クライアント接続

なお、バインディング接続、クライアント接続ともに設定の方法は共通ですので、本ガイドでは、接続形態による区別はしていません (クライアント接続の場合には、バインディング接続時の構成に加え、ホスト名やポート番号、チャネル名など、クライアント接続のためのプロパティを定義する必要があります)。

【参考】接続方法の設定順序

WMQ 側では、以下のステップが必要になります。

- i) Qmgr 作成
- ii) Qmgr 開始
- iii) ローカル・キュー作成

クライアント接続の場合には、これらに加えて、以下のステップが必要になります。

- iv) サーバー接続チャネル作成
- v) リスナー開始

ここでは、既に WMQ が適切に構成されていることを前提として、作業手順を説明します。

環境変数設定

WMQ に対して接続する場合、WMQ のネイティブ・ライブラリーを利用します。UNIX 系の OS では、プロセスが起動する前に、ネイティブ・ライブラリーを読み取れるように設定しておく必要があります。ネイティブ・ライブラリー・パスを設定する環境変数名は、OS 毎に異なります。この環境変数は、表 7-37のとおりです。なお、Windows ではこの設定は必要ありません。

表 7-37 OS ごとのライブラリー・パス設定用の環境変数

OS	環境変数名
AIX	LIBPATH
Linux	LD_LIBRARY_PATH
Solaris	LD_LIBRARY_PATH
HPUX	SHLIB_PATH

WebSphere 変数の設定

WMQ についての WebSphere 環境変数を設定します。管理コンソールのメニューより、[環境]→[WebSphere 変数]を選択します。複数表示される変数のうち、以下の値を編集してください。

表 7-38 WMQ を利用するための WebSphere 変数

項目	説明
MQ_INSTALL_ROOT	WMQ のインストール先 ディレクトリー (例えば、C:\Program Files\IBM\WebSphere MQ)

WMQ アクセス権限の設定

バインディング接続の場合、WAS 起動ユーザーを WMQ 起動ユーザーが属しているグループ (mqmグループ)に含めてください。

7.4.2.1 WebSphere MQ 接続ファクトリーの設定

JMS1.1 で新しく定義された Common Interface を利用するための接続ファクトリーです。このインターフェースを利用することで、メッセージングのタイプ (Point-to-Point や Publish/Subscribe) に依存しないメッセージング・アプリケーションを作成できます。

- i) 管理コンソールのメニューより、[リソース]→[JMS]→[接続ファクトリー]を選択します

ii) 有効範囲として、ここでは作成したクラスター「ClusterA」を指定します

図 7-54 接続ファクトリーの有効範囲設定

- iii) 有効範囲に「ClusterA」が表示されていることを確認したら、「新規作成」をクリックします
iv) JMS リソース・プロバイダーの選択では、「WebSphere MQ メッセージング・プロバイダー」を選択し、「OK」をクリックします

図 7-55 JMS リソース・プロバイダーの選択

- v) JMS 接続ファクトリーの基本属性を下記のとおり設定します

表 7-39 JMS 接続ファクトリーの設定 (基本属性)

項目	説明
名前	JMS 接続ファクトリーの識別名称を設定します
JNDI 名	JMS 接続ファクトリーの JNDI 名を設定します

- vi) 接続メソッドの選択では、「このウィザードに必要な情報をすべて入力」を選択し、「OK」をクリックします

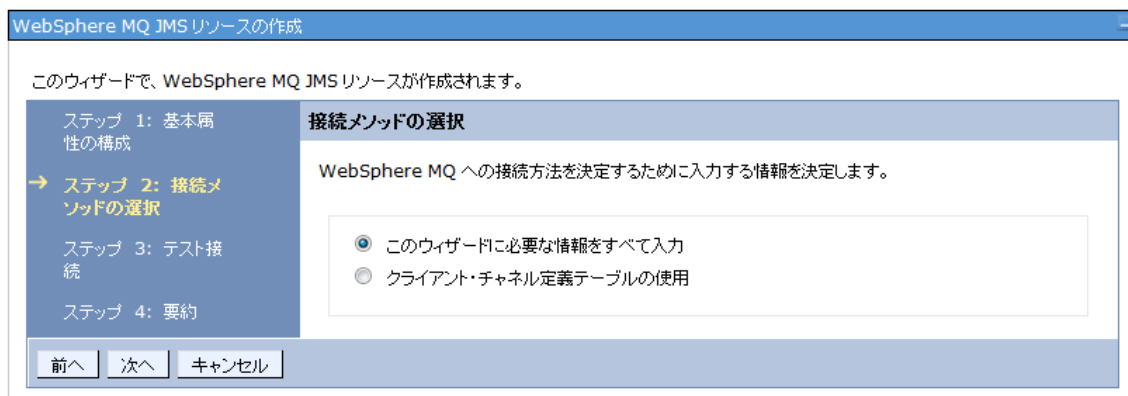


図 7-56 接続メソッドの選択

- vii) Qmgr の情報を下記のとおり設定します

表 7-40 JMS 接続ファクトリーの設定 (キュー・マネージャーの詳細情報)

項目	説明
キュー・マネージャーまたはキュー共有グループ名	Qmgr 名、またはキュー共有グループ名を設定します。

- viii) 接続情報を下記のとおり設定します

表 7-41 JMS 接続ファクトリーの設定 (接続詳細)

項目	説明
トランスポート	接続方法を以下から指定します。 ・バインディングとクライアント ・クライアント ・バインディング
サーバー接続チャンネル	Qmgr 上に定義されている、サーバー接続チャンネル名を設定します。
「個別のホスト名値およびポート値の形式でホストおよびポート情報を入力」を選択した場合	
ホスト名	Qmgr が稼動するホスト名を設定します。
ポート	Qmgr への接続に使用されるポート番号を設定します。
「接続名リストの形式でホストおよびポート情報を入力」を選択した場合	
接続名リスト	Qmgr のホスト名とポート番号をリスト形式で指定します。リストは、「host(port),host(port)」のようにカンマ区切りで表現します。ポート番号を省略した場合は、デフォルトの 1414 番ポートが使用されます。

7.4.2.2 WebSphere MQ キュー接続ファクトリーの設定

JMS1.0 で Point-to-Point のメッセージングを使用する場合は、キュー接続ファクトリーを取得する必要があります。

- i) 管理コンソールのメニューより、[リソース]→[JMS]→[キュー接続ファクトリー]を選択します
- ii) 「7.4.2.1 WebSphere MQ 接続ファクトリーの設定」と同様に設定内容を入力し、キュー接続ファクトリーを作成します

7.4.2.3 WebSphere MQ トピック接続ファクトリーの設定

JMS1.0 で Publish/Subscribe のメッセージングを使用する場合は、トピック接続ファクトリーを取得する必要があります。

- i) 管理コンソールのメニューより、[リソース]→[JMS]→[トピック接続ファクトリー]を選択します

- ii) 「7.4.2.1 WebSphere MQ 接続ファクトリーの設定」と同様に設定内容を入力し、トピック接続ファクトリーを作成します

7.4.2.4 WebSphere MQ キューの設定

JMS 1.0/1.1 で Point-to-Point のメッセージを送るときに使用するキューを設定します。

- i) 管理コンソールのメニューより、[リソース]→[JMS]→[キュー]を選択します
- ii) 有効範囲として、ここでは作成したクラスター「ClusterA」を指定します

図 7-57 キューの有効範囲設定

- iii) 有効範囲に「ClusterA」が表示されていることを確認したら、「新規作成」をクリックします
- iv) JMS リソース・プロバイダーの選択では、「WebSphere MQ メッセージング・プロバイダー」を選択し、「OK」をクリックします

図 7-58 JMS リソース・プロバイダーの選択

v) JMS キューを下記のとおり設定します

表 7-42 JMS キューの設定 (管理)

項目	説明
名前	JMS キューの識別名称を設定します
JNDI 名	JMS キューの JNDI 名を設定します

表 7-43 JMS キューの設定 (WebSphere MQ キュー)

項目	説明
キュー名	WMQ のキューの名前を指定します
キュー・マネージャまたは キュー共有グループ名	キューが存在する Qmgr またはキュー・共有グループの名前を指定します

7.4.2.5 WebSphere MQ トピックの設定

JMS 1.0/1.1 で Publish/Subscribe のメッセージを送るときに使用するトピックを設定します。

- 管理コンソールのメニューより、[リソース]→[JMS]→[トピック]を選択します
- 有効範囲として、ここでは作成したクラスター「ClusterA」を指定します

トピック

パブリッシュ/サブスクライブ・メッセージングを行うための宛先に使用される JMS トピック。

有効範囲: セル=**guideCell01**, クラスター=**ClusterA**

☒ すべての有効範囲オプションがある「有効範囲選択」ドロップダウンリストを表示

有効範囲は、リソース定義が可視となるレベルを指定します。有効範囲とその機能に関する詳細は、[有効範囲設定のヘルプを参照してください。](#)

クラスター=**ClusterA**

設定

新規作成 削除

選択	名前	JNDI 名	プロバイダー	説明	有効範囲
なし。					
合計 0					

図 7-59 トピックの有効範囲設定

- 有効範囲に「ClusterA」が表示されていることを確認したら、「新規作成」をクリックします
- JMS リソース・プロバイダーの選択では、「WebSphere MQ メッセージング・プロバイダー」を選択し、「OK」をクリックします

トピック > JMS リソース・プロバイダーの選択

有効範囲:

トピックを作成するプロバイダーを選択します。以下のプロバイダーが選択したリソース・タイプをサポートし、選択した有効範囲で使用可能です。

☒ WebSphere MQ メッセージング・プロバイダー

☐ デフォルトのメッセージング・プロバイダー

OK キャンセル

図 7-60 JMS リソース・プロバイダーの選択

- v) JMS トピックを下記のとおり設定します

表 7-44 JMS トピックの設定 (管理)

項目	説明
名前	JMS トピックの識別名称を設定します
JNDI 名	JMS トピックの JNDI 名を設定します

表 7-45 JMS トピックの設定 (WebSphere MQ トピック)

項目	説明
トピック名	トピックの割り当て先のトピックの名前を指定します
ブローカー永続サブスクリプション・キュー	永続サブスクリプション・メッセージの検索元となるブローカー・キューの名前を指定します
ブローカー永続サブスクライバーの接続コンシューマー・キュー	接続コンシューマーが永続サブスクリプション・メッセージを受信するキューの名前を指定します
ブローカー公開キュー	メッセージの公開に使用するキューの名前を指定します。 「接続として」を選択した場合は、接続ファクトリーの値が使用されます。 「オーバーライド接続」を選択した場合は、テキストボックスにキューの名前を指定します
ブローカー公開キュー・マネージャー	ブローカーが実行されているキュー・マネージャーの名前を指定します

7.4.2.6 JMS アクティベーション・スペックの設定

JMS 1.1 の MDB を使用するために必要な、「JMS アクティベーション・スペック」を設定する方法を紹介します。

- 管理コンソールのメニューより、[リソース]→[JMS]→[アクティベーション・スペック]を選択します
- 有効範囲として、ここでは作成したクラスター「ClusterA」を指定します

アクティベーション・スペック

アクティベーション・スペック

JMS アクティベーション・スペックは、1 つ以上のメッセージ駆動型 Bean と関連付けられ、それらがメッセージを受け取るのに必要な構成を提供します。

☒ 有効範囲: セル=**guideCell01**, クラスター=**ClusterA**

☒ すべての有効範囲オプションがある「有効範囲選択」ドロップダウン・リストを表示

有効範囲は、リソース定義が可視となるレベルを指定します。有効範囲とその機能に関する詳細は、[有効範囲設定のヘルプを参照してください](#)。

クラスター=**ClusterA**

設定

新規作成 削除

選択 名前 JNDI 名 プロバイダー 説明 有効範囲

なし。

合計 0

図 7-61 アクティベーション・スペックの有効範囲設定

- iii) 有効範囲に「ClusterA」が表示されていることを確認したら、「新規作成」をクリックします
- iv) JMS リソース・プロバイダーの選択では、「WebSphere MQ メッセージング・プロバイダー」を選択し、「OK」をクリックします

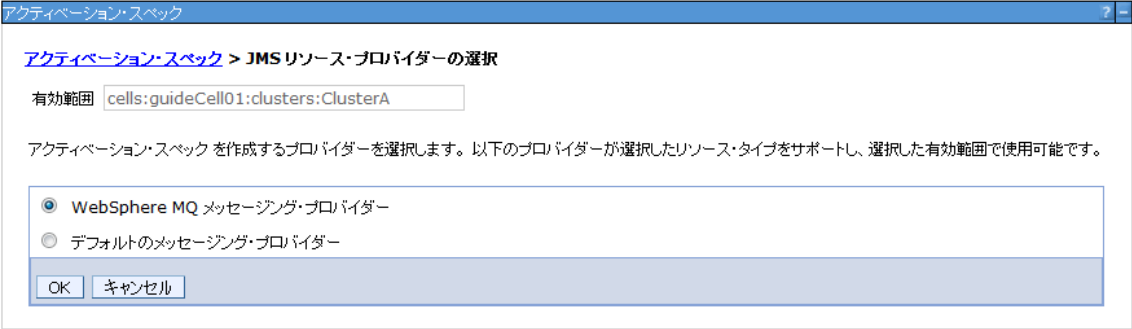


図 7-62 JMS リソース・プロバイダーの選択

- v) JMS アクティベーション・スペックを下記のとおり設定します

表 7-46 JMS アクティベーション・スペックの設定 (管理)

項目	説明
名前	JMS アクティベーション・スペックの識別名称を設定します
JNDI 名	JMS アクティベーション・スペックの JNDI 名を設定します

表 7-47 JMS アクティベーション・スペックの設定 (宛先)

項目	説明
宛先 JNDI 名	MDB が監視する宛先の JNDI 名を指定します
メッセージ・セクター	メッセージ・ヘッダー内のフィールド、および、メッセージ・プロパティ内のフィールドを元に、取得するメッセージを選択する場合に設定します
宛先タイプ	MDB が監視する宛先のタイプを指定します

- vi) 接続メソッドの選択では、「このウィザードに必要な情報をすべて入力」を選択し、「OK」をクリックします



図 7-63 接続メソッドの選択

vii) Qmgr の情報を下記のとおり設定します

表 7-48 JMS トピック接続ファクトリーの設定 (キュー・マネージャーの詳細)

項目	説明
キュー・マネージャーまたは キュー共有グループ名	Qmgr 名、またはキュー共有グループ名を設定します。

viii) 接続情報を下記のとおり設定します

表 7-49 JMS トピック接続ファクトリーの設定 (接続詳細)

項目	説明
トランスポート	接続方法を以下から指定します。 ・バインディングとクライアント ・クライアント ・バインディング
サーバー接続チャンネル	Qmgr 上に定義されている、サーバー接続チャンネル名を設定します。
「個別のホスト名値およびポート値の形式でホストおよびポート情報を入力」を選択した場合	
ホスト名	Qmgr が稼動するホスト名を設定します。
ポート	Qmgr への接続に使用されるポート番号を設定します。
「接続名リストの形式でホストおよびポート情報を入力」を選択した場合	
接続名リスト	Qmgr のホスト名とポート番号をリスト形式で指定します。リストは、「host(port),host(port)」のようにカンマ区切りで表現します。ポート番号を省略した場合は、デフォルトの 1414 番ポートが使用されます。

7.4.2.7 MDB を使用する際の構成

MDB の構成には、アクティベーション・スペックとリスナー・ポートのどちらかを使用可能です。どちらを使用するかを選択指針に関しては、以下の Information Center を参照してください。ここでは、リスナー・ポートを利用した構成について説明します

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/index.jsp?topic=/com.ibm.websphere.nd.doc/ae/cmb_aslp.html

リスナー・ポートの構成

リスナー・ポートは MDB を動作させるために必要で、宛先の監視に必要な、接続ファクトリーや宛先の情報を設定します。

- i) 管理コンソールのメニューより、[サーバー]→[すべてのサーバー]→[サーバー名]→[通信:メッセージング:メッセージ・リスナー・サービス]→[リスナー・ポート]を選択します

- ii) 「新規作成」をクリックします

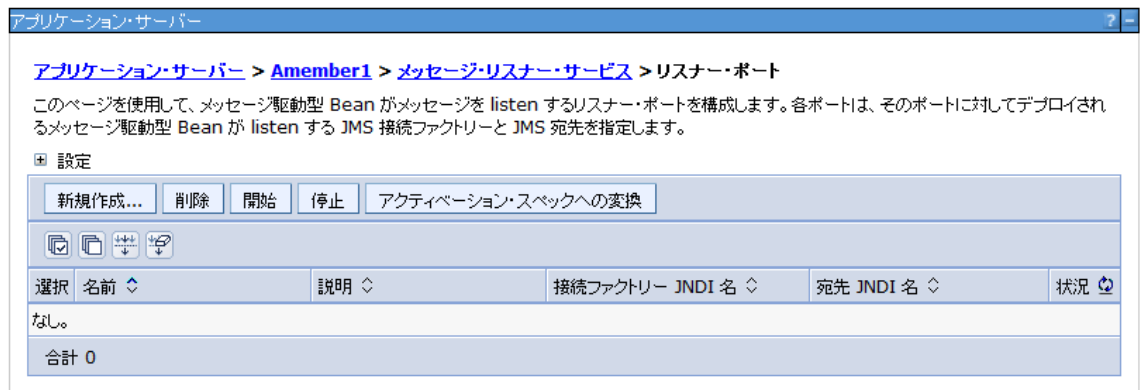


図 7-64 リスナー・ポートの作成

- iii) リスナー・ポートを以下のとおり設定します。

表 7-50 リスナー・ポートの設定

項目	説明
名前	リスナー・ポートの名前です
初期状態	サーバーが最初に開始されるときの、リスナー・ポートの実行状態です。開始済み／停止済みの 2 つが選択できます。MDB によるキュー／トピックの監視を行うためには、開始済みになっている必要があります
接続ファクトリー JNDI 名	JMS 接続ファクトリーの JNDI 名を指定します
宛先 JNDI 名	JMS 宛先の JNDI 名を指定します
最大セッション数	リスナーが使用できる JMS サーバーのセッション数の最大値です
最大再試行数	ここに指定した回数だけ、リスナーはメッセージの配信を試行します。この数だけ再試行を行うと、リスナーは停止します
最大メッセージ数	JMS サーバー・セッションで、リスナーが処理できるメッセージ数の最大値です

また、メッセージ・リスナー・サービスに関する下記のカスタム・プロパティを設定できます。

表 7-51 メッセージ・リスナー・サービスのカスタム・プロパティ

プロパティ名	説明
MQJMS.POOLING.TIMEOUT	プール内の未使用の接続が破棄されるまでのミリ秒数を指定します
MQJMS.POOLING.THRESHOLD	プール内の未使用接続の最大数を指定します
MAX.RECOVERY.RETRIES	リスナー・ポートが停止するまでに障害のリカバリーを試みる最大回数を指定します
RECOVERY.RETRY.INTERVAL	障害のリカバリーを試みる間隔を秒数で指定します

NON.ASF.RECEIVE.TIMEOUT	管理スレッドは存在せず、各スレッドがそれぞれメッセージを受信する NON.ASF モードにて、receive メソッドの引数 (受信待機時間) をミリ秒で指定します
-------------------------	--

MDB を動的クラスター上で稼働させる場合の設定

MDB を実行させるアプリケーション・サーバーとして、動的クラスターのメンバーを使用することが可能です。動的クラスターを使用できるのは、WMQ を利用する MDB の場合のみであり、SIBus を利用した MDB では使用できません。動的クラスターを利用する場合の考慮事項については、以下のリンク先の Infocenter を参照ください。

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/topic/com.ibm.websphere.wve.doc/ae/cwve_elasticity.html

デフォルトでは、アプリケーション配置コントローラーは、オンデマンド・ルーターから生成される情報を使用して、負荷に応じた動的クラスター内のアプリケーション・サーバーを開始や停止のタイミングを決定します。MDB は、オンデマンド・ルーターを経由しない処理のため、デフォルトの設定では、MDB

の負荷に応じた動的クラスター内のアプリケーション・サーバーの開始と停止が実行されません。そのため、MDB を動的クラスター上で稼働させる場合は、アプリケーション配置コントローラーのカスタム・プロパティを設定する必要があります。そのため、アプリケーション配置コントローラーが CPU の値を基に動的クラスター内のアプリケーション・サーバーの起動と停止を実行できるように、以下のカスタム・プロパティを設定します。

カスタム・プロパティの設定は、管理コンソールのメニューより、[動作ポリシー]→[オートノミック・マネージャー]→[アプリケーション配置コントローラー]→[追加プロパティ: カスタム・プロパティ]を選択して新規作成を押下します。名前として APC.predictor、値として CPU を設定します。

図 7-65 MDB を動的クラスター上で稼働させるための設定

7.4.2.8 接続プールの設定

SIBus の場合と同様に、Java EE Connector Architecture (JCA または J2C) の仕様に基づいたコネクション・プーリング機能が利用可能です。設定方法は SIBus と同じですので、「7.4.1.8 接続プールの設定」を参照してください。

7.4.2.9 セッション・プールの設定

JMS クライアントにおいては、接続を作成し、その接続を元にセッションを作成／開始します。接続 (Connection) は、クライアントから JMS プロバイダーへのアクティブな接続を表すオブジェクトであり、この接続から、Session オブジェクトを作成します。

セッション (Session) は、メッセージを送信／受信するためのシングルスレッドのコンテキストであり、プロデューサー、コンシューマーのファクトリーとして機能します。また、一同期点処理の単位となります。WMQ の観点から見ると、JMS セッションが作成されると、MQCONN が発行されることとなります。接続ファクトリーを利用する場合には `javax.jms.Session`、キュー接続ファクトリーを利用する場合は `javax.jms.QueueSession`、トピック接続ファクトリーを利用する場合は `javax.jms.TopicSession` インターフェースを利用します。

このセッションを、未使用になった時点で破棄せずにセッション・プールに保持しておき、次のリクエスト時にこのプールからセッションを再利用させることができます。

管理コンソールの [リソース]→[JMS]→[接続ファクトリー]→[接続ファクトリー名]→[追加プロパティ: セッション・プール]から最大接続数などの項目が設定可能です。

[接続ファクトリー](#) > [SampleFactory](#) > セッション・プール

構成

一般プロパティ

有効範囲

* 接続タイムアウト

* 最大接続数

* 最小接続数

* リープ時間

* 未使用タイムアウト

* 経過時間タイムアウト

バージ・ポリシー

図 7-66 セッション・プールの設定