WebSphere Adapters

バージョン 6 リリース 1

WebSphere Adapter for Flat Files ユーザーズ・ガイド
バージョン 6 リリース 1

WebSphere Adapters

バージョン 6 リリース 1

WebSphere Adapter for Flat Files ユーザーズ・ガイド
バージョン 6 リリース 1

お願い

本書および本書で紹介する製品をご使用になる前に、 175 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 WebSphere Adapter for Flat Files バージョン 6、リリース 1、モディフィケーション 0 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。

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。

IBM 発行のマニュアルに関する情報のページ

`http://www.ibm.com/jp/manuals/`

こちらから、日本語版および英語版のオンライン・ライブラリーをご利用いただけます。また、マニュアルに関するご意見やご感想を、上記ページよりお送りください。今後の参考にさせていただきます。

(URL は、変更になる場合があります)

お客様の環境によっては、資料中の円記号がバックスラッシュと表示されたり、バックスラッシュが円記号と表示されたりする場合があります。

原　典： WebSphere® Adapters
Version 6 Release 1
WebSphere Adapter for Flat Files User Guide
Version 6 Release 1

発　行： 日本アイ・ビー・エム株式会社

担　当： ナショナル・ランゲージ・サポート

第1刷 2008.2

目次

第 1 章 WebSphere Adapter for Flat Files の概要

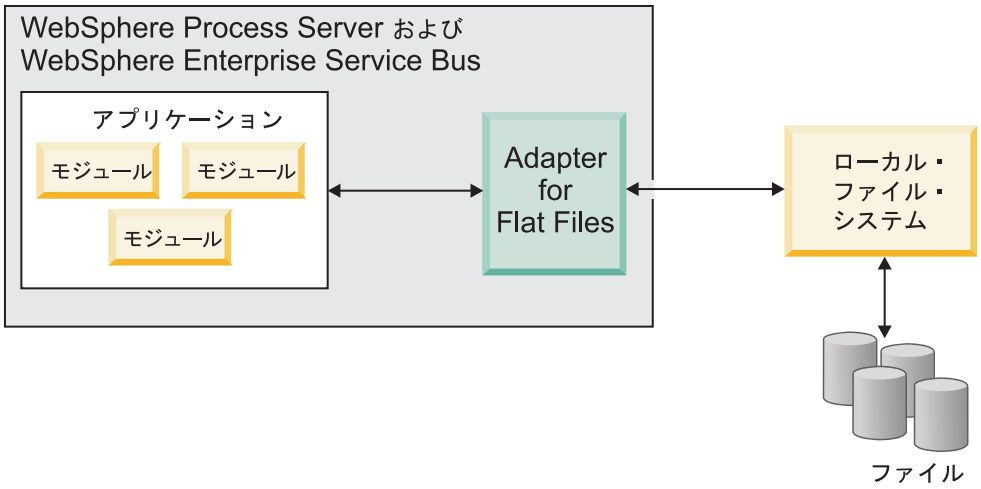
WebSphere Adapter for Flat Files を使用すれば、特別なコーディングを行わなくても、ローカル・ファイル・システムとのデータ交換を含む統合プロセスを作成することができます。

アダプターを使用してローカル・ファイル・システムのファイルからデータを読み取り、そのデータを WebSphere Process Server または WebSphere Enterprise Service Bus で実行中のアプリケーションで使用してから、ローカル・ファイル・システムに送信して戻すことができます。また、アダプターを使用してローカル・ファイル・システムのディレクトリーをポーリングして新しいファイルを探し、それらをアプリケーションに送信して処理することもできます。

アダプターを使用すると、ローカル・ファイル・システムに保管されているすべてのタイプのファイルを読み書きすることができます。実行可能な操作は以下のとおりです。

- 新規ファイルの作成
- 既存のファイルへの付加または上書き
- 特定のファイル内容の検索、ディレクトリー内ファイル名リストの検索、またはファイルの削除
- 特定のファイルが存在するかどうかの確認
- 新規ファイルを探すためのディレクトリーのポーリング、および処理するためのアプリケーションへのファイルの送信

以下の図は、SOA の実装の一部としてアダプターを示したものです。



アダプターの概要

このリリースの新機能

WebSphere Adapter for Flat Files、バージョン 6.1.0 はアダプターを機能拡張しました。また、このリリースでは、一部の機能が非推奨になっています。

これについての更新情報は、WebSphere Adapters 製品サポート Web サイトで入手できます。更新情報または追加情報については、http://www.ibm.com/software/integration/wbiadapters/support/を参照してください。

非推奨の機能とは、サポートされていても推奨されてはおらず、廃止される可能性がある機能です。バージョン 6.1.0 で推奨されない Adapter for Flat Files の旧バージョンの機能のリストについては、37 ページの『非推奨の 機能』 を参照してください。

バージョン 6.1.0 の新機能は次のとおりです。

- エンタープライズ・サービス・ディスカバリー・ウィザードの新規名、使用可能度の向上、および機能強化。ウィザードの名前は、外部サービス・ウィザードに変更になりました。また、機能と使用可能度が改善され、アダプターで使用するサービスの作成が容易になりました。ウィザードは、定義済みデータ・バインディング、データ・ハンドラーおよび関数セレクターを利用して、ファイルとビジネス・オブジェクト間の変換を自動化する場合に役立ちます。
- アダプター・パターン・ウィザードは、アダプターで単純なサービスを素早く簡単に作成する方法を提供します。
- オペレーティング・システム・サポートの拡張。バージョン 6.1.0 でサポートされるオペレーティング・システムについて詳しくは、IBM® Web サイト (http://www.ibm.com/support/docview.wss?uid=swg27006249) にある WebSphere Adapter for Flat Files のハードウェアおよびソフトウェアの要件を参照してください。
- アダプター RAR ファイルは WebSphere Integration Developer 内で使用可能です。つまり、アダプター RAR ファイルを別にインストールする必要はありません。ウィザードは、アダプター・ファイルを自動的にプロジェクトにコピーします。
- アダプターに関する文書は、WebSphere Integration Developer インフォメーション・センターの『アダプターの構成と使用』セクションにあります。
- ノード・レベルまたはスタンドアロンのアダプターのデプロイメントのサポート。
- バージョン 6.0.2 での各ビジネス・オブジェクトが含まれたビジネス・グラフが、オプションになりました。バージョン 6.0.2 で作成されたビジネス・オブジェクトを持つモジュール専用のビジネス・グラフ、または ApplyChanges Outbound 操作を使用するバージョン 6.1.0 の新規モジュール用のビジネス・グラフが必要になります。
- WebSphere Application Server の症状データベースに組み込むことができる First Failure Data Capture (FFDC) 構成のサポートにより、ログに記録されたデータを診断モジュールでカスタマイズできるように情報および推奨アクションを提供します。
- ビジネス・フォールトのサポート。アダプターで、ビジネス例外のビジネス・フォールトを生成するようになりました。これにより、それらのエラー条件に修正アクションを容易に割り当てることができます。
- データ・バインディング・プロパティーの構成サポート。
- Outbound Retrieve 操作でのデータ形式変更のサポート。
- 以下の機能を提供する、Outbound 処理用の追加オプション

  - Create および Append 操作で固有のファイル名を作成
  - Create 操作でファイルの固有のシーケンス番号を生成
  - ファイルの取得後にファイルを削除
  - 取得したファイルを削除前にアーカイブ
- Windows® および UNIX® の改行を区切り文字とする、プラットフォームに依存しないサポート。
- IPv6 アドレスのサポート。
- イベント・ストアのメモリー内表現を使用する Inbound 処理中のイベント・パーシスタンスのサポート。

ハードウェア要件とソフトウェア要件

WebSphere Adapters のハードウェアおよびソフトウェア要件は、以下のロケーションにある IBM Web サイトで文書化されています。

WebSphere Adapters のハードウェアおよびソフトウェア要件: http://www.ibm.com/support/docview.wss?uid=swg27006249

追加情報

以下のリンクでは、アダプターの構成およびデプロイに必要とする場合がある追加情報を参照できます。

- WebSphere Business Integration Adapters および WebSphere Adapters の互換性マトリックスは、アダプターに必要なソフトウェアのサポートされるバージョンを識別します。この文書を表示するには、WebSphere Adapters サポート・ページ (http://www.ibm.com/software/integration/wbiadapters/support/) にアクセスして、「**Planning upgrades**」の下の互換性マトリックスのリンクをクリックしてください。
- WebSphere Adapters のテクニカル・ノートには、製品文書に含まれていない予備手段および追加情報が記述されています。アダプターのテクニカル・ノートを表示するには、Web ページ http://www.ibm.com/support/search.wss?tc=SSMKUK&rs=695&rank=8&dc=DB520+D800+D900+DA900+DA800+DB560&dtm にアクセスして、「**Product category**」 リストから、ご使用のアダプターの名前を選択し、検索アイコンをクリックしてください。

Adapter for Flat Files の技術概要

IBM WebSphere Adapter for Flat Files を使用すると、WebSphere Process Server または WebSphere Enterprise Service Bus で実行されるサービスが、ローカル・ファイル・システムとの間でデータを交換できるようになります。

サービスは、アダプターを使用して、以下の 2 つの方法でローカル・ファイル・システムとデータを交換できます。

- *Outbound 処理* により、WebSphere Process Server または WebSphere Enterprise Service Bus で実行されるサービスは、アダプターを使用してローカル・ファイル・システムにあるファイルに対して操作 (例えば、注文文書の更新) を実行します。

- *Inbound 処理* により、WebSphere Process Server または WebSphere Enterprise Service Bus で実行されるサービスは、アダプターを使用してローカル・ファイル・システムからイベント (例えば、顧客レポートが更新されたことの通知) を受け取ります。

WebSphere Integration Developer で起動する外部サービス・ウィザードを使用して、この処理を行うようにアダプターを構成します。外部サービス・ウィザードを使用して、モジュール を作成します。モジュールは、WebSphere Integration Developer のプロジェクトと、WebSphere Process Server に対するデプロイメントの単位で構成されています。各モジュールには、サービスを構成するコンポーネントと、インポート または エクスポート が含まれています。

- インポート は、SCA モジュールが外部サービス (SCA モジュール外のサービス) に、ローカル・アクセスを行うようにアクセスするポイントです。インポートは、SCA モジュールとサービス・プロバイダーの間の対話を定義します。インポートには、バインディングおよび 1 つ以上のインターフェースがあります。
- エクスポート (エンドポイントとも言う) は、外部にビジネス・サービスを提供する Service Component Architecture (SCA) モジュールの公開インターフェースです。エクスポートには、サービス要求元からのサービス (Web サービスなど) へのアクセス方法を定義するバインディングが含まれます。

モジュールはパッケージ化され、エンタープライズ・アーカイブ (EAR) ファイルとして WebSphere Process Server にデプロイされます。

モジュールとローカル・ファイル・システムの間で交換されるファイルを表すため、アダプターはビジネス・オブジェクトを使用します。ビジネス・オブジェクトは、アダプターで処理されるデータを含む論理データ・コンテナーです。ビジネス・オブジェクトを作成するには、WebSphere Integration Developer で、外部サービス・ウィザードを使用するか、ビジネス・オブジェクト・エディターを使用します。

アダプターは、アダプター固有のデータ・バインディング およびデータ・ハンドラー を使用して、 Inbound および Outbound の処理中に、ある形式から別の形式にデータを変換します。データ・バインディング は基本的に、ビジネス・オブジェクトのフォーマット方法を定義するマップです。データ・バインディングにより、ビジネス・オブジェクト内のフィールドが読み取られ、ファイル内の対応するフィールドが記入されます。使用されるデータ・バインディングは、ファイルの内部形式に応じて異なります。データ・タイプにはそれぞれ同等のデータ・バインディングがあります。この 外部サービス・ウィザード を使用して、データ・バインディングを構成します。

データ・ハンドラー は、ビジネス・オブジェクトとネイティブ形式の間の変換を実行します。ビジネス・オブジェクトを含むデータ・タイプを選択する場合は、変換を行うデータ・ハンドラーを指定する必要があります。データ・ハンドラーは、WebSphere Process Server または WebSphere Enterprise Service Bus によって提供されます。

Outbound 処理

アダプターは、Outbound 処理中に、ローカル・ファイル・システム 内のファイル上で処理を実行するために、ビジネス・オブジェクトの形式でモジュールから要求

を受け取ります。アダプターは要求された操作を実行して、該当する場合は、操作の結果を表すビジネス・オブジェクトをコンポーネントに返します。

以下の図は、WebSphere Adapter for Flat Files の Outbound 処理フローを示しています。

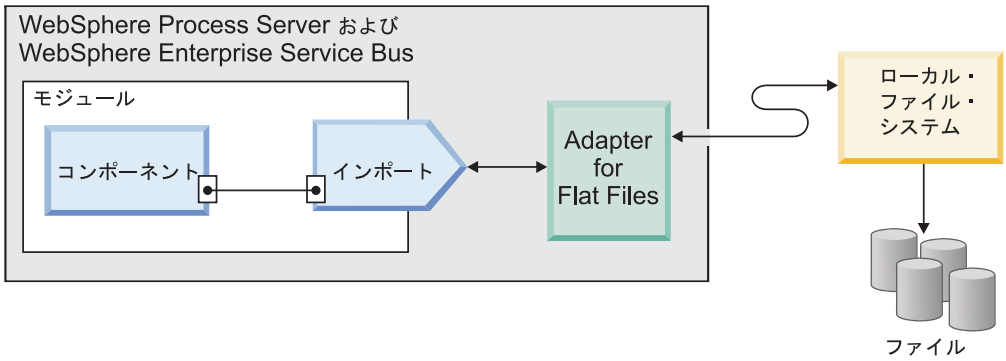


図 1. Outbound 処理

Outbound 操作

操作は、Outbound 処理時にアダプターがローカル・ファイル・システムで実行可能なアクションです。操作名は通常、アダプターが実行するアクションのタイプを示します。

このアダプターは、Outbound 処理時の以下の操作をサポートします。

Append 操作:

Append 操作によって、指定したファイルにコンテンツが追加されます。

Append

外部サービス・ウィザードで「出力が必要」を指定した場合は、ファイル名はビジネス・オブジェクトのコンポーネントに返されます。

CreateFileIfNotExists プロパティーを `true` に設定すると、アダプターは新規ファイルを作成します。GenerateUniqueFile プロパティーを `true` に設定すると、アダプターは固有ファイルを生成し、ファイル名プロパティーの値を無視します。

付加するファイルが存在しないときに、CreateFileIfNotExists プロパティーを `false` に設定すると、アダプターは `RecordNotFoundException` エラーを生成します。

Create 操作:

Create 操作によって、指定した名前のファイルが作成されます。

Create

外部サービス・ウィザードで「出力が必要」を指定した場合は、ファイル名はビジネス・オブジェクトのコンポーネントに返されます。指定した名前のファイルが既に存在している場合、アダプターは `DuplicateRecordException` エラーを生成します。ファイルは作成されません。

GenerateUniqueFile プロパティーを `true` に設定すると、アダプターは固有の名前でファイルを生成し、ファイル名プロパティーの値を無視します。アダプターによって生成される固有ファイルの名前は、ビジネス・オブジェクト名に乱数によるプレフィックスが付き、ファイル拡張子を .tmp にした形式となります。例: `Customer23423.tmp`

FileSequenceLog 管理接続プロパティーを指定すると、アダプターは、要求で指定された出力ファイル名にシーケンス番号を付加します。例えば、要求内に出力ファイル名が Customer.txt の場合は、Customer.*n*.txt という名前のファイルが作成されます。*n* は特定の要求のシーケンス番号で、1 から始まります。出力ファイル名が Order.txt になっている別の要求を受け取ると、Order.txt に対して 1 から始まる新規のシーケンスが生成されます。出力ファイル名に拡張子がない場合は、ファイル名の末尾にシーケンスが付加されます。例えば、要求内の出力ファイル名が Customer の場合は、Customer.*n* という名前のファイルが作成されます。

要求ごとにビジネス・オブジェクトで出力ディレクトリーとファイル名を設定する手間を省くため、管理接続レベルで出力ディレクトリーとファイル名を設定することにより、特定タイプの要求に対するファイル順序付けを生成できます。ファイルの Create 要求を受け取ると、アダプターはファイル・シーケンス・ログを検査し、その名前が既に存在するかどうかを確認します。その名前が存在する場合、アダプターはファイル・シーケンス番号を使用して新規ファイル名を作成します。

注: ビジネス・オブジェクトで指定されるディレクトリー・パスとファイル名は、管理接続プロパティーの値よりも優先されます。

クラスター環境 (複数のシステム上で稼働するアダプターのインスタンスが 1 つ存在する環境) では、FileSequenceLog プロパティーによって指定されるシーケンス・ファイルが、クラスター内のすべてのノードからアクセス可能なマップされたドライブに存在する必要があります。アダプターには、シーケンス・ログ・ファイルに対する書き込み許可が必要です。この許可がない場合は、IOException エラーが返されます。

FileSequenceLog プロパティーが指定されており、GenerateUniqueFile プロパティーが有効になっている場合は、GenerateUniqueFile プロパティーが FileSequenceLog プロパティーより優先されます。

シーケンス・ファイルを手動で削除すると、順序付けが 1 から再開されます。シーケンスは、シーケンス・ファイル内のシーケンス値を変更することによってリセットできます。

シーケンス番号は、アダプターの再始動後まで増分され続けます。

Delete 操作:

Delete 操作によって、指定したファイルが削除されます。

Delete

そのファイルが存在しない場合、アダプターは `RecordNotFoundException` エラーを生成します。

Exists 操作:

Exists 操作によって、指定されたファイルが存在するかどうかが検査されます。

Exists

指定されたファイルが存在する場合は、成功応答がビジネス・オブジェクトの形式でコンポーネントに返されます。ビジネス・オブジェクトには、ファイルが存在する場合には true に、ファイルが存在しない場合には false に設定される 1 つの属性があります。ファイルが存在しない場合、またはディレクトリーが存在しない場合、アダプターは `false` を返します。

List 操作:

List 操作によって、指定されたディレクトリー内のファイル名がリストされます。

List

そのディレクトリーが存在しない場合、アダプターは `RecordNotFoundException` エラーを生成します。

Overwrite 操作:

Overwrite 操作によって、指定されたファイルは要求で指定された内容で上書きされます。

Overwrite

外部サービス・ウィザードで「出力が必要」を指定した場合は、ファイル名はビジネス・オブジェクトのコンポーネントに返されます。ステージング・ディレクトリーが StagingDirectory プロパティーで指定されている場合、上書きされるファイルは出力ディレクトリーからステージング・ディレクトリーにコピーされ、内容はステージング・ディレクトリーにあるファイルで上書きされます。その後、出力ディレクトリーにファイルが戻されます。ステージング・ディレクトリーが指定されていない場合は、出力ディレクトリーにあるファイルで内容が上書きされます。

注: ステージング・ディレクトリーは、Overwrite 操作がファイルを返す前にそのファイル内容が書き込まれる場合にのみ構成できます。Overwrite 操作が出力ストリームを返し、コンポーネントがこのストリームに書き込む場合は、ステージング・ディレクトリーを使用できません。

Input 要求を FlatFileOutputStreamRecord レコードとして受け取ると、アダプターは出力ストリームを返します。

CreateIfFileNotExists プロパティーを `true` に設定すると、アダプターは新規ファイルを作成します。GenerateUniqueFile プロパティーを `true` に設定すると、アダプターは固有ファイルを生成し、Filename プロパティーで指定されている値を無視します。

更新するファイルが存在しないときに、CreateFileIfNotExists プロパティーを false に設定すると、アダプターは `RecordNotFoundException` エラーを生成します。

Retrieve 操作:

Retrieve 操作によって、指定されたファイルのコンテンツが取り出され、ビジネス・オブジェクトの形式で返されます。

Retrieve

ファイルのコンテンツが取り出されて、汎用またはコンテンツ固有のビジネス・オブジェクトの形式で返されます。ファイルのコンテンツは、対話仕様で定義される SplittingFunctionClassName プロパティーおよび SplitCriteria プロパティーに従って分割されます。データ・ハンドラーが構成されている場合、アダプターはコンテンツ固有のビジネス・オブジェクトを返します。それ以外の場合は、汎用ビジネス・オブジェクトを返します。

取り出された後でファイルを削除するように指定するには、対話仕様で DeleteOnRetrieve プロパティーを設定します。削除する前にファイルをアーカイブするように指定するには、ArchiveDirectoryForDeleteOnRetrieve プロパティーを設定します。

Retrieve 要求で指定されたファイルが存在しない場合、アダプターは `RecordNotFoundException` エラーを生成します。

Outbound データ変換

Outbound 処理中、アダプターは、外部サービス・ウィザードでのアダプターの Outbound 処理の構成時に選択したアダプター固有のデータ・バインディングおよびデータ・ハンドラーに基づいて、データ変換を行います。

データ変換を伴う Outbound 処理

Outbound 処理中、アダプターは、アプリケーションで予測されるデータ形式にビジネス・オブジェクトを変換します。この処理は、Outbound 処理用にモジュールを構成した際に選択したアダプター固有のデータ・バインディングおよびデータ・ハンドラーによって制御されます。

9 ページの図 2 は、Outbound 処理中にデータが変換される方法を示したものです。

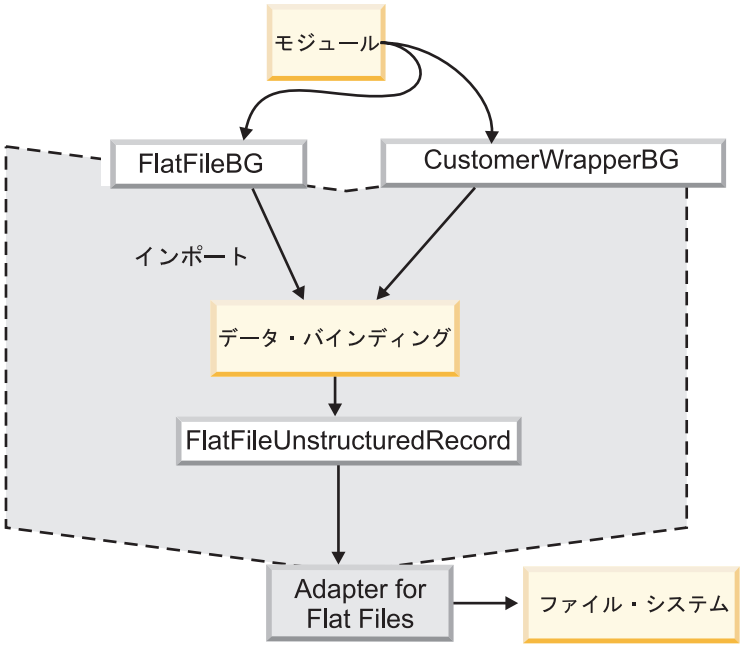


図 2. Outbound 処理中のデータ変換

以下のステップで、データ変換を伴う Outbound 処理について説明します。

1. Retrieve 以外のすべての操作では、アダプターは、入力データ・タイプおよび構成済みデータ・ハンドラーに基づいて、データ変換を行います。入力が汎用タイプ (FlatFile または FlatFileBG) 以外の場合、アダプターはデータを変換します。Retrieve 操作では、アダプターは、データ・バインディングのデータ・ハンドラー・プロパティーが構成済みである場合にのみ、データを変換します。
2. 構成済みのデータ・バインディングが呼び出され、ビジネス・オブジェクトを処理します。
3. データ・バインディングによって、データ・バインディング・プロパティーのデータ・ハンドラー・プロパティーに対して指定された値が検査され、データ・ハンドラー・プロパティーに設定された値に基づいて、コンテンツ固有のデータ・ハンドラーが呼び出されます。
4. アダプターは、ファイルに対して要求された操作を行い、以下のような応答ビジネス・オブジェクトを返す場合があります。
   - Create、Append、および Overwrite 操作では、出力が構成済みである場合、応答ビジネス・オブジェクトにはファイル名が含まれます。
   - List 操作では、応答ビジネス・オブジェクトには指定されたディレクトリー内のファイル・リストが含まれます。
   - Exists 操作では、応答ビジネス・オブジェクトには `true` または `false` の値が含まれます。
   - Retrieve 操作では、取り出されたファイルのコンテンツが汎用またはコンテンツ固有の応答ビジネス・オブジェクトの形式で返されます。
   - Delete 操作では、出力は返されません。

データ変換を伴わない Outbound 処理

Retrieve 以外のすべての操作では、入力データが汎用タイプ (FlatFile または FlatFileBG) の場合、アダプターはデータ変換を伴わない Outbound 処理を行います。 Retrieve 操作では、データ・バインディングのデータ・ハンドラー・プロパティーに値が設定されていない場合、データ変換は実行されません。このタイプの処理中は、UnstructuredContent という特殊なデータ構造を使用してコンテンツが保持されます。

図 3 はデータ変換を伴わない Outbound 処理を示したものです。

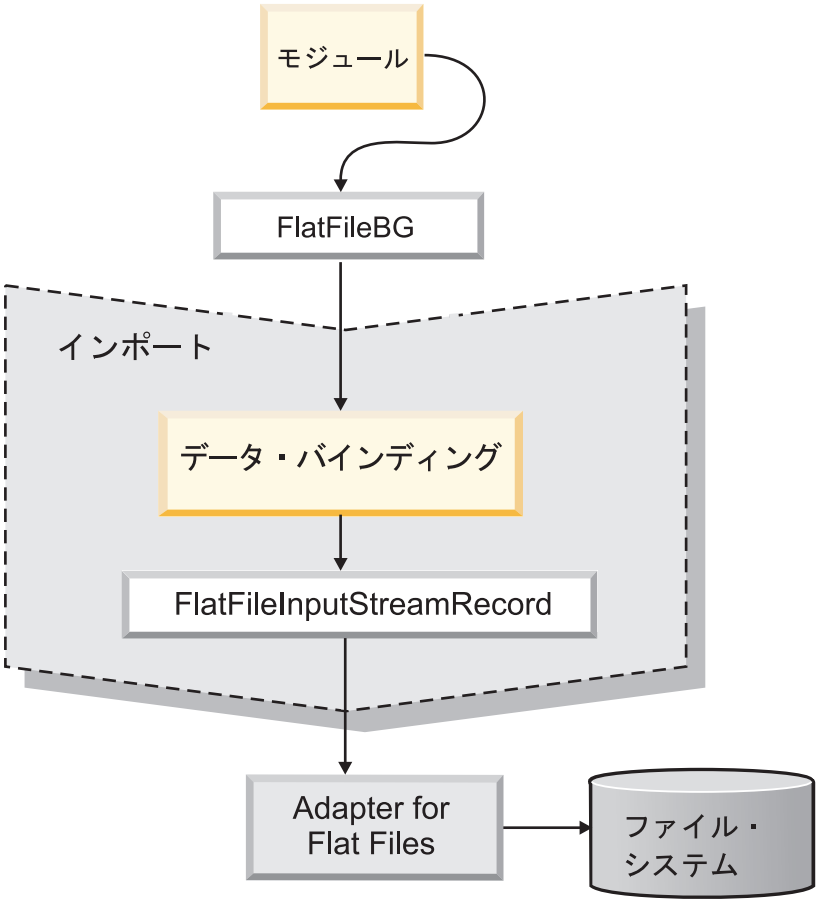


図 3. データ変換を伴わない Outbound 処理

以下のステップで、データ変換を伴わない Outbound 処理について説明します。

1. Retrieve 以外のすべての操作では、アダプターは要求データ・オブジェクトの入力タイプを検査します。入力が汎用タイプ (FlatFile または FlatFileBG) の場合、アダプターは着信オブジェクトに対してデータ変換を行いません。 Retrieve 操作では、アダプターはデータ・ハンドラー・プロパティーを検査します。値を指定しなかった場合、データ変換が行われます。
2. 構成済みのデータ・バインディングが呼び出され、ビジネス・オブジェクトを処理します。

3. Retrieve 操作では、アダプターはデータ・ハンドラー・プロパティーを検査します。データ・ハンドラーに値が設定されていない場合は、アダプターはデータ変換を行いません。
4. アダプターは、ファイルに対して要求された操作を行い、以下のような応答ビジネス・オブジェクトを返す場合があります。
   - Create、Append、および Overwrite 操作では、出力が構成済みである場合、応答ビジネス・オブジェクトにはファイル名が含まれます。
   - List 操作では、応答ビジネス・オブジェクトには指定されたディレクトリー内のファイル・リストが含まれます。
   - Exists 操作では、応答ビジネス・オブジェクトには `true` または `false` の値が含まれます。
   - Retrieve 操作では、取り出されたファイルのコンテンツが汎用またはコンテンツ固有の応答ビジネス・オブジェクトの形式で返されます。
   - Delete 操作では、出力は返されません。

ファイル分割

複数のレコードを含むファイルに対応するため、アダプターではオプションでファイル分割機能が提供されます。この機能を使用する場合、アダプターは大きなファイルを小さいチャンクに分割して、別々に取得します。

ファイルは、内包するコンテンツのタイプに応じて、区切り文字またはサイズで分割できます。

- ビジネス・オブジェクトのコンテンツが明確な構造を持つ場合 (例えば、名前、住所、および市などの要素を持つ場合など) は、ファイルは区切り文字で分割されます。
- ビジネス・オブジェクトに含まれるのがプレーン・テキストまたはバイナリー・ファイルなどの非構造化データの場合、ファイルはサイズによって分割されます。

デフォルトでは、アダプターはファイルをサイズで分割します。

SplitCriteria プロパティーで指定する値によって、使用する方式が決まります。SplitCriteria プロパティーのデフォルト値がゼロの場合、分割は行われません。分割を必要としない場合は、SplitCriteria プロパティーおよび SplittingFunctionClassName プロパティーの値を空のままにしておいてもかまいません。

オプションでカスタム・ファイルのスプリッター・クラスを指定できます。クラス名に SplittingFunctionClassName プロパティーを設定してください。

区切り文字によるファイル分割

ファイル内のビジネス・オブジェクトを区切るのに、コンマ (,)、セミコロン (;)、引用符 (″、’)、中括弧 ({})、またはスラッシュ (/ ¥) などの 1 つ以上の文字 (区切り文字) が使用されるとき、アダプターは区切り文字に基づいて、ファイルを小さいチャンクに分割できます。ファイルのビジネス・オブジェクトを分離する区切り文字は、SplitCriteria プロパティーに定義します。

区切り文字の使用には、以下の規則が適用されます。

- 区切り文字内のすべての改行は、プラットフォーム固有の改行文字で示す。プラットフォーム固有の改行文字を表 1 に示します。

表 1.

プラットフォーム	改行文字
Macintosh	¥r
Microsoft® Windows	¥r¥n
UNIX	¥n

- 複数の区切り文字がある場合、それぞれの区切り文字はセミコロン (;) で分離する必要があります。区切り文字は、指定された順に突き合わせが行われます。セミコロンが区切り文字の一部である場合、`¥;` のようにしてエスケープする必要があります。例えば、区切り文字が `##¥;##` の場合、`##;##` として処理されます。
- 区切り文字の一部であるコンテンツをスキップする場合、直前に 2 つのセミコロン (;;) を指定すると、区切り文字の間のコンテンツがスキップされます。例えば、以下の形式のビジネス・オブジェクトがイベント・ファイルに含まれていて、区切り文字が `##;;$$` の場合、アダプターは、`##$$` を区切り文字と見なし、「`content skipped by the adapter`」の部分をスキップします。

```
Name=Smith
Company=IBM
##content skipped by the adapter$$
```

- 区切り文字には任意の値を使用することができ、制限はありません。区切り文字は、有効なストリング、改行文字 (¥n など)、および区切り文字が複数ある場合はセミコロンの分離文字を組み合わせたものです。区切り文字に改行文字およびセミコロンが含まれていなくてもかまいません。改行文字が使用されるのは、ファイルのコンテンツを分割するときに改行文字が考慮される場合に限られます。有効な区切り文字の例は以下のとおりです。
  - `####;¥n;¥n`
  - `####;$$$$;¥n;####`
  - `%%%%;$$$$$;#####`
  - `¥n;¥n;$$$$`
  - `####¥;####;¥n;$$$$$`
  - `¥n;¥n;¥n`
  - `####;;$$$$`
  - `¥r`
  - `¥r¥n`
  - `$$$$;¥r¥n`
- 区切り文字がファイルの最後に配置された場合、SplitCriteria プロパティーは `END_OF_FILE` を使用して、ファイルの物理的な末尾を決定します。

一般的なシナリオの例と推奨される区切り文字の形式を以下に示します。

表 2.

データ・バインディング	BO コンテンツ	推奨される区切り文字の形式
XML	<?xml version="1.0" encoding="UTF-8" ?> <customer:Customer xsi:type="customer:Customer" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:customer="http://www.ibm.com/xmlns/prod/websphere/ j2ca/flatfile/customer"> <CustomerName>Deepa</CustomerName> <Address>IBM</Address> <City>Bangalore</City> <State>KA</State> </customer:Customer>	</customer:Customer>;¥n

サイズによるファイル分割

SplittingFunctionClassName プロパティーで指定する値によって、ファイルがサイズで分割されるかどうかが決まります。SplittingFunctionClassName プロパティーを com.ibm.j2ca.extension.utils.filesplit.SplitByDelimiter に設定した場合、SplitCriteria プロパティーには最大ファイル・サイズをバイト単位で表す有効な数字が含まれている必要があります。ファイルが SplitCriteria プロパティーで指定した値よりも大きい場合、ファイルはチャンクに分割され、それぞれのチャンクはインポートに別々に送られます。ファイルが SplitCriteria の値よりも小さい場合、ファイル全体がインポートに送られます。

イベント・ファイルがチャンクに分割されると、それぞれのチャンクがビジネス・オブジェクトとなります。すなわち、PollQuantity プロパティーに指定した値と、インポートに送達したビジネス・オブジェクトの数が、異なっていてもかまいません。アダプターは PollQuantity の値に従ってポーリングを行いますが、実際にはファイル内のビジネス・オブジェクトは一度に 1 つずつ処理します。例えば、イベント・ファイルが 3 つのチャンクに分けられた場合、1 つのファイルがポーリングされて 3 つのビジネス・オブジェクトがインポートに送達されます (それぞれのチャンクが 1 つのビジネス・オブジェクトを作成しているため)。

インポートでは、アダプターはチャンク・データを単一ファイルに再組み立てすることはしませんが、WebSphere Process Server でこれらを単一ファイルに再組み立てできるように、チャンクについての情報を提供します。チャンク情報は FlatFileInputStreamRecord レコードの ChunkFileName プロパティーに組み込まれ、バイト単位のチャンク・サイズおよびイベント ID が含まれます。チャンクのイベント ID は、eventFileLocation_/_timestampStr_/_MofN の形式を使用します (M は現在のチャンク番号、N は総チャンク数)。イベント ID の例は、

C:¥flatfile¥eventdir¥eventfile.in_/_2005_01_10_10_17_49_864_/_3of5 のようになります (timestampStr は year_month_day_hour_minutes_seconds_milliseconds の形式になります)。

固有ファイル名の生成

Create 操作中に、永続シーケンス番号をデフォルトのファイル名に追加するか、乱数を使用してファイル名を生成することで、固有ファイル名を生成します。 Append および Overwrite 操作中は、乱数メソッドを使用してください。

Create 操作中に固有ファイル名を生成する方法は、次の 2 とおりです。

1. 永続シーケンス番号をデフォルト・ファイル名に追加する。このメソッドは特にクラスター化された環境の場合にお勧めします。
2. 乱数を使用して、永続的ではない固有ファイル名を生成する。

Append および Overwrite 操作の場合は、乱数メソッドを使用してください。

永続シーケンス番号を使用する固有ファイル名の生成

永続シーケンス番号を使用して固有ファイル名を生成するには、以下を指定します。

- シーケンス・ファイル。シーケンス番号を保管するファイルの完全パスです。
- デフォルト・ターゲット・ファイル名

アダプターは、デフォルトのターゲット・ファイル名にシーケンス番号が追加された構成のファイル名を生成します。

固有ファイル名の生成を制御するプロパティーは、次の 3 つの場所にあります。

- 管理接続ファクトリー・プロパティー (デフォルト・ターゲット・ファイル名およびシーケンス・ファイル・プロパティー)
- 対話仕様プロパティー (デフォルト・ターゲット・ファイル名および固有ファイルの生成プロパティー)
- ラッパー・ビジネス・オブジェクト

ビジネス・オブジェクトのプロパティーは対話仕様のプロパティーよりも優先され、対話仕様のプロパティーは管理接続ファクトリー・プロパティーよりも優先されます。特定のオブジェクトをさまざまな方法で処理する場合を除いて、管理接続ファクトリーでプロパティーを使用して、ファイル名の生成を制御してください。

デフォルト・ファイル名に拡張子が付いている場合、その拡張子の前にシーケンス番号が付加されます。例えば、管理接続ファクトリーのデフォルト・ファイル名が `Customer.txt` の場合、作成される出力ファイル名は、`Customer.1.txt`、`Customer.2.txt` のようになります。シーケンス番号は、ビジネス・オブジェクト・タイプごとに個別に保持されます。

シーケンスは、次の形式でシーケンス・ファイルに保管されます。

```
<dirPath>/Customer.txt = 2
```

ここで `Customer.txt` はデフォルト・ファイル名を表し、 `2` はアダプターが同一ファイルに対する別の Create 要求を受け取った場合に使用されるシーケンス番号を表しています。

乱数を使用する固有ファイル名の生成

乱数を使用して固有ファイル名を生成するには、対話仕様で「固有ファイルの生成 (GenerateUniqueFile)」プロパティーを設定するか、ビジネス・オブジェクトを `true` に設定します。アダプターは、`ffa[`*`RandomNumber`*`].tmp` という形式で固有ファイル名を生成します。ここで、*RandomNumber* はアダプターが生成した乱数を表します。例えば、`ffa23423.tmp` のようになります。

Append および Overwrite 操作では、「ファイルが存在しない場合に新規ファイルを作成する (CreateFileIfNotExists)」対話仕様プロパティーを `true` に設定していて、そのファイルが既に存在する場合、アダプターは新規ファイルを作成します。同じファイル名生成ルールが Create 操作にも適用されます。

Inbound 処理

Adapter for Flat Files は Inbound イベント処理をサポートします。これは、イベント (ファイルの作成や変更など) に指定された間隔で、ローカル・ファイル・システムのポーリングを行います。イベントを検出すると、イベント・データをビジネス・オブジェクトに変換し、これをモジュールに送信して処理します。

以下の図は、WebSphere Adapter for Flat Filesの Inbound 処理フローを示しています。

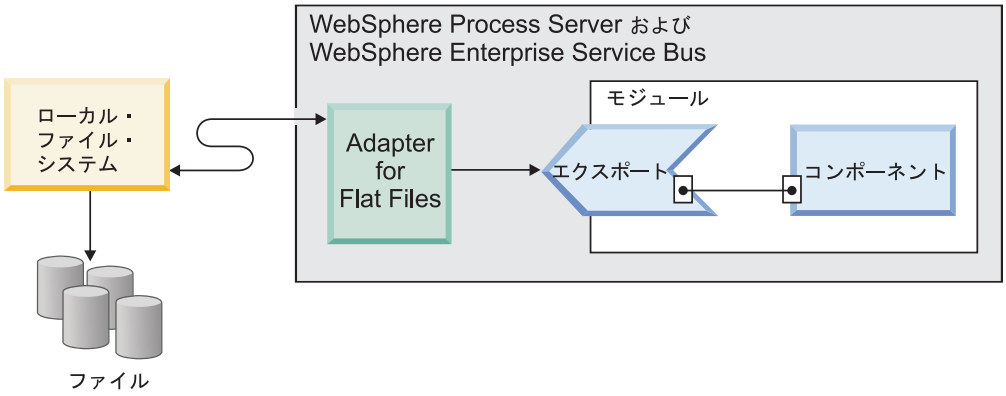


図 4. Inbound 処理

ローカル・ファイル・システムで変更が生じた場合、新規または変更済みファイルであるイベント・ファイルが特定のディレクトリーに作成されます。このディレクトリーをアダプターのイベント・ディレクトリーとして構成します。イベント・ファイルはファイル・システム内の 1 つ以上のイベントを表すことが可能ですが、アダプターへの 1 転送単位のみを形成します。

アダプターは、ファイル・システムのイベント・ディレクトリーを定期的な間隔でポーリングします。この間隔は、PollPeriod プロパティーで設定した値に基づいています。ファイルがイベント・ディレクトリーに到達すると、アダプターはファイルの内容をエクスポートに送信します。ファイル内容は一括で送信されるか、いくつかのビジネス・オブジェクト、あるいはチャンクに分割されます。アダプターは、関数セレクターを使用して、ビジネス・オブジェクトをエクスポートに送信します。関数セレクターは、コンポーネントで起動する操作を選択し、正しいデータ・バインディングを提供します。

Inbound 処理フローは以下のとおりです。

1. イベントは、ファイルの形式でファイル・システム内に生成されます。
2. アダプターが、イベント・ディレクトリーをポーリングします。
3. アダプターが、各イベントにイベント ID を割り当て、そのイベント ID をイベント・ストアに保管します。イベント・ストアは、ポーリング・アダプターがイ

ベント・レコードを処理できるようになるまでイベント・レコードが保存されるパーシスタント・キャッシュです。アダプターを構成する前に、このデータベースを作成する必要があります。データベースのデフォルト名は FFDB です。

4. アダプターは各イベント・ファイルをバイトとして読み取ります。ファイル分割が有効な場合、アダプターは SplittingFunctionClassName および SplitCriteria プロパティーに設定された値に基づいて、イベント・ファイルを解析します。
   - 分割が区切り文字に基づく場合、この機能を実行するクラスと分割基準が提供されます。
   - 分割がファイル・サイズに基づく場合、この機能を実行するクラス名が提供されます。
5. 構成されたデータ・タイプがオブジェクト固有 (CustomerWrapper など) である場合、データ・ハンドラーが DataBinding に構成され、アダプターがそのデータを変換します。
6. 構成されたデータ・タイプが FlatFile または FlatFileBG である場合、アダプターはファイルの内容を FlatFile ビジネス・オブジェクト内のバイト配列として渡し、変換は行われません。

   注: ファイル分割が有効な場合、ビジネス・オブジェクトにはファイル・サイズおよびイベント ID が含まれます。
7. アダプターは、関数セレクターを使用して、ビジネス・オブジェクトをエクスポートに送信します。関数セレクターは、コンポーネントで起動する操作を選択し、正しいデータ・バインディングを提供します。
8. ビジネス・オブジェクトがエクスポートに到達すると、イベントはイベント・ストアから削除されます。アーカイブ機能が有効な場合は、イベントは削除される前にアーカイブ・テーブルに移されます。

イベント・アーカイブ

正しくポーリングされたイベントを追跡するために、外部サービス・ウィザードの ArchiveDirectory アクティベーション・スペック・プロパティーを使用して、ファイル・システムにアーカイブ・ディレクトリーを構成することができます。ファイルは、アクティベーション・スペックの指定に従って `success` 拡張子または `fail` 拡張子が付けられて、アーカイブ・ディレクトリーにコピーされます。

イベント・ファイル・ロック

ファイル・ロックの動作はオペレーティング・システムに応じて異なります。Windows では、アダプターによってイベント・ディレクトリーからポーリングされた任意のファイルが他のアプリケーションで使用されており、イベント・ディレクトリーへのコピー処理中である場合は、アダプターでの処理に使用できません。

ただし、AIX® などの UNIX 環境の場合は、アプリケーションが書き込みを受けているファイルにアクセスできなくなるようなファイル・ロック・メカニズムはありません。別のアプリケーションでイベント・ディレクトリーにコピーされるファイルは、アダプターでの処理に使用できますが、正常な結果は得られません。 Java™ には、ファイルが書き込みを受けているかどうかを確認する、プラットフォームに依存しない方法はありません。

まずイベント・ファイルをステージング・ディレクトリーにコピーしてから、move コマンドを使ってそのファイルをイベント・ディレクトリーに移動することで、この状態を回避できます。 UNIX スクリプトのいくつかのサンプルが、アダプターの一部として提供されています。 CheckIfFileIsOpen.sh という名前のスクリプト・ファイルを、アダプター・インストーラーの Unix-script-file フォルダーから使用できます。

イベント・パーシスタンス

アダプターは突然終了した場合の Inbound 処理のイベント・パーシスタンスをサポートしています。イベント・パーシスタンス (または 1 回の送達) とは、障害時に、イベントがエクスポートに確実に 1 回だけ送達されるようにする手段です。アダプターは、イベント処理中に、データ・ソースにあるイベント・ストアにイベントの状態を永続させます。このデータ・ソースは、イベント・ストアが作成可能になる前に、 WebSphere Process Server を使用してセットアップする必要があります。 WebSphere Process Server で提供されているリカバリー機能を使用するには、アクティベーション・スペックで AssuredOnceDelivery プロパティーを true に設定します。このリカバリー機能はデフォルトではオンになっています。

また、アダプターはイベント・ストアのメモリー内表現を使用して、イベント・パーシスタンスを提供することもできます。この機能を使用すると、JNDI データ・ソースまたは外部イベント・ストアを作成する必要がなく、イベント処理が高速になります。ただし、この機能ではイベント・リカバリーがサポートされていません。サーバーに障害が発生すると、メモリー内のイベント・ストアは失われます。サーバー障害時にイベントが失われることを回避するために、データベース・イベント・ストアを使用するアプローチをお勧めします。

アダプターのメモリー内イベント・パーシスタンス機能を使用するには、AssuredOnceDelivery プロパティーを `false` に設定する必要があります。設定しない場合、アダプターは警告メッセージをログに記録します。

イベント・ストア

イベント・ストアは、ポーリング・アダプターがイベント・レコードを処理できるようになるまでイベント・レコードが保存されるパーシスタント・キャッシュです。アダプターは、Inbound イベントがシステム内を通過するときに、イベント・ストアを使用してその Inbound イベントを追跡します。ファイルが作成、更新、または削除されるたびに、アダプターはイベント・ストア内のイベントの状況を更新します。イベントがエクスポートに送信されるまでは、各イベントの状況は、リカバリーのためにアダプターによって継続的に更新されます。

Inbound モジュール用のイベント・ストアがローカル・ファイル・システム内に存在しないことを検出すると、アダプターはアプリケーションがランタイムにデプロイされるときに自動的にイベント・ストアを作成します。アダプターによって作成された各イベント・ストアは、特定の Inbound モジュールに関連付けられます。アダプターは、同じイベント・ストアを指す複数のアダプター・モジュールをサポートしていません。

ローカル・ファイル・システムをポーリングするときに、アダプターはアクティベーション・スペック・プロパティーで指定した検索条件に一致する各イベントにつ

いて、イベント・ストア内にエントリーを作成します。アダプターは、新しい各エントリーの状況を NEW として記録します。

イベントが正常に送られた場合、イベント・ストア項目は削除されます。失敗したイベントの場合、エントリーはイベント・ストアに残されます。オプションとして、アダプターは、正常にポーリングされたイベント・ファイルをアーカイブ・ディレクトリーにアーカイブできます。

注: 失敗したイベントによって、イベント・ファイル内のデータが正確でなくなる場合があります。例えば、fname という名前のフィールドが fnam と示される場合があります。これを訂正するには、正しいデータを含むイベント・ファイルを再送するしかありません。

アダプターは、イベントの送達を 1 回としています。つまり、各イベントは 1 回のみ配信されます。AssuredOnceDelivery アクティベーション・スペック・プロパティーを True に設定した場合、アダプターは各イベントの XID (トランザクション ID) の値をイベント・ストアに保管します。イベントが処理のために取得されると、以下の操作が実行されます。

1. イベントの XID 値がイベント・ストア内で更新されます。
2. イベントが対応するエクスポートに送達されます。
3. イベントがイベント・ストアから削除されます。

次の図はアダプター用のイベント管理フローを示しています。

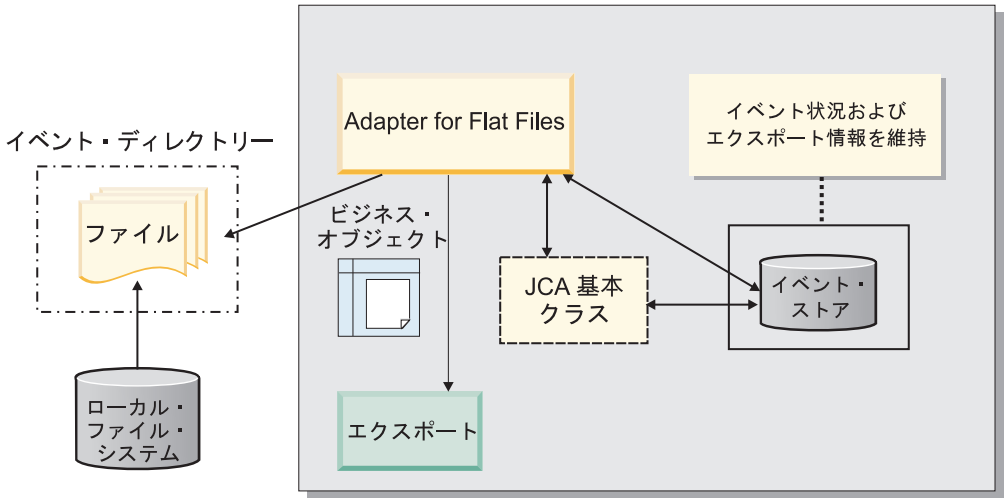


図 5. イベント管理フロー

イベント・ストア構造:

イベント・ストアはイベントを追跡するために、アダプターによって使用されます。次の表に、イベントごとに保管される値を示します。

表 3. イベント・ストア構造

列名	タイプ (長さ)	説明
EVNTID	Varchar (255)	Inbound 処理中のイベントの追跡に使用されます。各イベントには、トラッキングのため、イベント ID が必要です。これはテーブル内での固有 ID である必要があります。
EVNTSTAT	Integer	イベントの状況。アダプターはこの状況を使用して、イベントが新規か処理中かを判断します。 イベント状況の値: **NEW(0)** イベントの処理の準備ができています。 **PROCESSED (1)** アダプターがイベントを正常に処理し送信しました。 **FAILED (-1)** 1 つ以上の問題により、アダプターがこのイベントを処理できませんでした。
XID	Varchar (255)	イベント送達およびイベント・リカバリーが行われたかどうかを確認するために、アダプターによって使用されます。
EVNTDATA	Varchar (255)	失敗したイベントを追跡して、これらがリカバリー中に再処理されないようにするために使用されます。失敗したイベントには「ARCHIVED」のマークが付けられます。

イベント・アーカイブ値:

ディレクトリー内の処理済みイベント・ファイルをアーカイブするようにアダプターを構成すると、それにアクセスして処理済みイベントのリストを入手することができます。ファイル拡張子には、アーカイブされたイベントが正常に実行されたかどうかが反映されます。

指定されたアーカイブ・ディレクトリー内のアーカイブされたイベントは、すべて success、failure、および original のいずれかのファイル拡張子を付けて保管されま

す。 success 拡張子は、イベント処理が成功した場合に使用されます。イベント処理に失敗した場合は、ファイルは failure 拡張子および original の拡張子付きでアーカイブされます。イベント・ファイルに複数のビジネス・オブジェクトがあり、その一部が正常に処理された場合、そのファイルにも success 拡張子が付けられます。

アーカイブ拡張子は、アクティベーション・スペック・プロパティー FailedArchiveExt、OriginalArchiveExt、および SuccessArchiveExt に基づいて構成できます。

以下の表は、アダプターが使用するアーカイブ拡張子をリストしたものです。

表 4. イベント・アーカイブ値

拡張子	定義	フォーマット
SUCCESS	イベント・ファイルはエクスポートに送達されました。	<filename>_<timestamp>.SUCCESS
FAIL	イベント・ファイルはエクスポートに送達されませんでした。	<filename>_<timestamp>.FAIL

関数セレクター

Inbound 処理中に、関数セレクターはサービスで呼び出される適切な操作を返します。外部サービス・ウィザードで Inbound 処理用にアダプターを構成した場合は、関数セレクターを選択してください。アダプターは、`FilenameFunctionSelector` および `EmbeddedNameFunctionSelector` の 2 つの関数セレクターを提供します。

FilenameFunctionSelector

`FilenameFunctionSelector` は、ファイル名にマップする正規表現に基づくオブジェクトの名前解決を提供する、ルール・ベースの関数セレクターです。正規表現は、一定の構文規則に従って、一連のストリングを記述または突き合わせる場合に使用するストリングです。

以下の表に、突き合わせルールの例を示します。このルールは ObjectName フィールドおよび Rule フィールドで構成されます。

表 5. FilenameFunctionSelector の突き合わせルールの例

FileName	ObjectName	Rule
`Customer0001.txt`	`Customer`	`CUST.*TXT`
`2231ORZ93.z21`	`Order`	`[0-9]*OR[A-Z][0-9]{2}.*`
`2231ORZ93.z21`	`Order`	`*OR.*`

2 行目と 3 行目のルールは同じ名前に解決されていますが、2 行目のルールでは、特定の数値や文字のシーケンスを含んでいることがファイル名が一致していると見なされる条件であるため、比較的短い突き合わせとなります。一方、3 行目のルールでは、ファイル名に「OR」という文字が含まれるすべてのオブジェクト名に解決されます。「`.*`」という文字の組み合わせは、任意の文字が任意の回数使用される可能性があることを示しています。

固有の関数名を生成する場合、関数セレクターは、指定されたオブジェクト名の先頭に emit を追加します。例えば、オブジェクト名が Customer の場合、関数セレクターは emitCustomer という関数名を返します。オブジェクト名としては、Customer または Order のようなペイロード・オブジェクト名を指定する必要があります。ラッパーまたはビジネス・グラフ名は指定できません。パススルーのシナリオの場合は、オブジェクト名として FlatFile を使用してください。

それぞれがオブジェクト名とファイル名と突き合わせる正規表現を含む複数のルールを使用して、FilenameFunctionSelector を構成することができます。複数のルールが一致する場合、関数セレクターは最初に一致したルールに基づいてオブジェクト名を戻します。ルールが一致しない場合は、アダプターはエラーを生成します。構成にルールが存在しない場合、関数セレクターは emitFlatFile という関数名を使用します。

正規表現の使用に対して適用されるルールについて詳しくは、https://java.sun.com/j2se/1.4.2/docs/api/java/util/regex/Pattern.html に記載されている Java クラス・パターンの文書を参照してください。

EmbeddedNameFunctionSelector

EmbeddedNameFunctionSelector はコンテンツ固有のビジネス・オブジェクトに使用されますが、この場合、オブジェクト名はイベント・ファイルに埋め込まれています。 EmbeddedNameFunctionSelector は、ラッパーではなく、必要なコンテンツ・データに基づく関数名を戻します。例えば、コンテンツ固有のビジネス・オブジェクトが CustomerWrapperBG の場合、関数セレクターで戻される関数は emitCustomer です。

EmbeddedNameFunctionSelector はデータ・ハンドラーと一緒に構成する必要があります。データ・バインディングは、アダプター固有の WrapperDataBinding でなければならず、関数セレクターと一緒に構成されたものと同じデータ・ハンドラーを使用するように構成する必要があります。

ファイル分割

イベント処理中のメモリー・ロードを削減するために、アダプターはファイル分割機能をオプションでサポートしています。この機能を使用する場合、アダプターは大きなイベント・ファイルを小さいチャンクに分割して、エクスポートに別々に送信します。

アダプターは SplitCriteria プロパティーに指定された値に基づいて、大きなイベント・ファイルを複数のビジネス・オブジェクト (チャンクとも呼ばれる) に分割します。この値は、区切り文字またはチャンク・サイズとして指定できます。各ビジネス・オブジェクトは別々にエクスポートに送達されます。ビジネス・オブジェクトのコンテンツが明確な構造を持つ場合 (例えば、名前、住所、および市などの要素を持つ Customer ビジネス・オブジェクトなどが該当) は、ファイルを区切り文字で分割します。ビジネス・オブジェクトに含まれるのがプレーン・テキストまたはバイナリー・ファイルなどの非構造化データの場合、ファイルはサイズによって分割します。

イベント・ファイルがそのようなチャンクに分割される場合は、それぞれのチャンクがビジネス・オブジェクトを作成します。すなわち、PollQuantity プロパティーに

指定した値と、エクスポートに送達したビジネス・オブジェクトの数が、異なっていてもかまいません。区切り文字に基づいたファイル分割が有効な場合は、PollQuantity アクティベーション・スペック・プロパティーにより、イベント・ストア内に存在する当該イベント・ファイルの数が指定され、SplittingFunctionClassName アクティベーション・スペック・プロパティーでイベント・ファイルの分割に使用されるクラスが設定されます。

アダプターはチャンクに分けたデータの再組み立ては行いません。

SplitCriteria プロパティーで指定する値によって、使用する方式が決まります。SplitCriteria プロパティーのデフォルト値がゼロの場合、分割は行われません。分割を必要としない場合は、SplitCriteria プロパティーおよび SplittingFunctionClassName プロパティーの値を空のままにしておいてもかまいません。

オプションでカスタム・ファイルのスプリッター・クラスを指定できます。クラス名に SplittingFunctionClassName プロパティーを設定してください。

区切り文字によるファイル分割

ファイル内のビジネス・オブジェクトを区切るために、コンマ (,)、セミコロン (;)、引用符 (″、’)、中括弧 ({})、またはスラッシュ (/ ¥) などの 1 つ以上の文字 (区切り文字) が使用されている場合、アダプターは区切り文字に基づいて、ファイルを小さいチャンクに分割できます。それぞれのチャンクは、WebSphere Process Server または WebSphere Enterprise Service Bus に転送されたときにビジネス・オブジェクトを作成するために使用される論理単位です。ファイルのビジネス・オブジェクトを分離する区切り文字は、SplitCriteria プロパティーに定義します。

区切り文字によるファイル分割で使用される PollQuantity 値の働きを示すため、2 つのイベント・ファイルについて考察します。第 1 のイベント・ファイルにはビジネス・オブジェクトが 1 つ、第 2 のイベント・ファイルにはビジネス・オブジェクトが 2 つあります。PollQuantity の値が 2 の場合、最初のビジネス・オブジェクトは第 1 のイベント・ファイルから、次のビジネス・レコードは第 2 のイベント・ファイルから、最初のポーリング周期で送信されます。第 2 のファイルの 2 番目のビジネス・オブジェクトは、次のポーリング周期で送信されます。

区切り文字の使用には、以下の規則が適用されます。

- 区切り文字内のすべての改行は、プラットフォーム固有の改行文字で示す。プラットフォーム固有の改行文字を表 6 に示します。

表 6.

プラットフォーム	改行文字
Macintosh	¥r
Microsoft Windows	¥r¥n
UNIX	¥n

- 複数の区切り文字がある場合、それぞれの区切り文字はセミコロン (;) で分離する必要があります。区切り文字は、指定された順に突き合わせが行われます。セミコロンが区切り文字の一部である場合、`¥;` のようにしてエスケープする必要があります。例えば、区切り文字が `##¥;##` の場合、`##;##` として処理されます。

- 区切り文字の一部であるコンテンツをスキップする場合、直前に 2 つのセミコロン (;;) を指定すると、区切り文字の間のコンテンツがスキップされます。例えば、以下の形式のビジネス・オブジェクトがイベント・ファイルに含まれていて、区切り文字が ##;;$$ の場合、アダプターは、##$$ を区切り文字と見なし、「content skipped by the adapter」の部分をスキップします。

  ```
  Name=Smith
  Company=IBM
  ##content skipped by the adapter$$
  ```

- 区切り文字には任意の値を使用することができ、制限はありません。区切り文字は、有効なストリング、改行文字 (¥n など)、および区切り文字が複数ある場合はセミコロンの分離文字を組み合わせたものです。区切り文字に改行文字およびセミコロンが含まれていなくてもかまいません。改行文字が使用されるのは、ファイルのコンテンツを分割するときに改行文字が考慮される場合に限られます。有効な区切り文字の例は以下のとおりです。
  - ####;¥n;¥n
  - ####;$$$$;¥n;####
  - %%%%;$$$$$;#####
  - ¥n;¥n;$$$$
  - ####¥;####;¥n;$$$$$
  - ¥n;¥n;¥n
  - ####;;$$$$
  - ¥r
  - ¥r¥n
  - $$$$;¥r¥n
- 区切り文字がファイルの最後に配置された場合、SplitCriteria プロパティーは END_OF_FILE を使用して、ファイルの物理的な末尾を決定します。

一般的なシナリオの例と推奨される区切り文字の形式を以下に示します。

表 7.

データ・バインディング	BO コンテンツ	推奨される区切り文字の形式
XML	<?xml version="1.0" encoding="UTF-8" ?> <customer:Customer xsi:type="customer:Customer" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:customer="http://www.ibm.com/xmlns/prod/websphere/ j2ca/flatfile/customer"> <CustomerName>Deepa</CustomerName> <Address>IBM</Address> <City>Bangalore</City> <State>KA</State> </customer:Customer>	</customer:Customer>;¥n

サイズによるファイル分割

SplittingFunctionClassName プロパティーで指定する値によって、ファイルがサイズで分割されるかどうかが決まります。SplittingFunctionClassName プロパティーを com.ibm.j2ca.extension.utils.filesplit.SplitByDelimiter に設定した場合、

SplitCriteria プロパティーには最大ファイル・サイズをバイト単位で表す有効な数字が含まれている必要があります。イベント・ファイルが SplitCriteria プロパティーで指定した値よりも大きい場合、ファイルはチャンクに分割され、それぞれのチャンクはエクスポートに別々に送られます。イベント・ファイルが SplitCriteria の値よりも小さい場合、イベント・ファイル全体がエクスポートに送られます。

イベント・ファイルがチャンクに分割されると、それぞれのチャンクがビジネス・オブジェクトとなります。すなわち、PollQuantity プロパティーに指定した値と、エクスポートに送達したビジネス・オブジェクトの数が、異なっていてもかまいません。アダプターは PollQuantity の値に従ってポーリングを行いますが、実際にはファイル内のビジネス・オブジェクトは一度に 1 つずつ処理します。例えば、イベント・ファイルが 3 つのチャンクに分けられた場合、1 つのファイルがポーリングされて 3 つのビジネス・オブジェクトがエクスポートに送達されます (それぞれのチャンクが 1 つのビジネス・オブジェクトを作成しているため)。

エクスポートでは、アダプターはチャンク・データを単一ファイルに再組み立てすることはしませんが、WebSphere Process Server でこれらを単一ファイルに再組み立てできるように、チャンクについての情報を提供します。チャンク情報は FlatFileInputStreamRecord レコードの ChunkFileName プロパティーに組み込まれ、バイト単位のチャンク・サイズおよびイベント ID が含まれます。チャンクのイベント ID は、`eventFileLocation_/_timestampStr_/_MofN` の形式を使用します (M は現在のチャンク番号、N は総チャンク数)。イベント ID の例は、

`C:¥flatfile¥eventdir¥eventfile.in_/_2005_01_10_10_17_49_864_/_3of5` のようになります (timestampStr は `year_month_day_hour_minutes_seconds_milliseconds` の形式になります)。

Inbound データ変換

Inbound 処理中、アダプターは、外部サービス・ウィザードでのモジュールの構成時に選択したアダプター固有のデータ・バインディングおよびデータ・ハンドラーに基づいて、データ変換を行います。

データ変換を伴う Inbound 処理

Inbound 処理中のデータ変換処理は、モジュール構成時に選択したアダプター固有のデータ・バインディングおよびデータ・ハンドラーによって制御されます。以下のステップで、データ変換を伴う Inbound 処理について説明します。

1. それぞれのイベントは、SplitCriteria プロパティーに設定された値に基づいてイベント・ファイルから取り出されます。そのコンテンツはレコードで設定され、データ・バインディングに送信されます。
2. アダプターは、Inbound 操作の予測されるデータ・タイプを確認します。データが汎用タイプ (FlatFile または FlatFileBG) 以外である場合、アダプターはデータ・バインディング内のデータ・ハンドラー・プロパティーを確認します。
3. データ・ハンドラーが設定されている場合は、アダプターはデータを変換します。データ・バインディングによって、データ・ハンドラーが呼び出され、コンテンツ固有のビジネス・オブジェクトが返されます。

4. FlatFileInputStreamRecord レコードはファウンデーション・クラスに送信され、そこから EmbeddedNameFunctionSelector 関数セレクターに送信されます。この関数セレクターは、アダプターによって生成されたイベントと適切なエクスポート関数名をマップします。
5. アダプターは、このコンテンツ固有のビジネス・オブジェクトを、関数セレクターによって返されたメソッドを呼び出して、エンドポイントに渡します。

データ変換を伴わない Inbound 処理

コンテンツに対するデータ変換が不要な場合 (例えば、text¥xml コンテンツを text¥xml コンテンツとして保持する必要がある場合など)、イベント・データはビジネス・オブジェクトに変換されませんが、非構造のコンテンツとしてパススルーされます。

以下のステップで、データ変換を伴わない Inbound 処理について説明します。

1. それぞれのイベントは、SplitCriteria プロパティーに設定された値に基づいてイベント・ファイルから取り出されます。そのコンテンツはレコードで設定され、データ・バインディングに送信されます。
2. データ・バインディングは、イベントの予測されるタイプを検査します。汎用タイプ (FlatFile または FlatFileBG) の場合、アダプターはデータを変換しません。
3. データ・バインディングによって、UnstructuredContent レコードでコンテンツが設定され、アダプターに戻されます。
4. アダプターは、このビジネス・オブジェクトを、関数セレクターによって返されたメソッドを呼び出して、エンドポイントに渡します。

ビジネス・オブジェクト

ビジネス・オブジェクトは、アダプターで処理されるデータを表す論理データ・コンテナーです。データは、ビジネス・エンティティー (送り状または従業員レコードなど) または非構造化テキスト (E メールの本文またはワープロ文書) のいずれかを表すことができます。アダプターはビジネス・オブジェクトを使用してローカル・ファイル・システムにデータを送信するか、またはデータを取得します。

アダプターがビジネス・オブジェクトを使用する方法

Outbound 処理時に、アダプターは以下を行います。

1. ローカル・ファイル・システム内のファイル上で処理を行うために、要求を表すビジネス・オブジェクトをモジュールから受け取ります。
2. 必要に応じて、ローカル・ファイル・システムで認識できる形式にビジネス・オブジェクトを変換します。
3. 要求された操作を実行します。
4. 該当する場合は、操作の結果を表すビジネス・オブジェクトをモジュールに返します。

Inbound 処理時に、アダプターは以下を行います。

1. ローカル・ファイル・システム上のイベント・ディレクトリーからファイルを取得します。

2. 必要に応じて、データを必要な形式に変換し、データからビジネス・オブジェクトを作成します。
3. ビジネス・オブジェクトをエクスポートに送信します。

ビジネス・オブジェクトの作成方法

ビジネス・オブジェクトは、外部サービス・ウィザードまたはビジネス・オブジェクト・エディターのいずれかを使用して作成できます。これらはどちらも WebSphere Integration Developer から起動できます。外部サービス・ウィザードを使用する場合、ウィザードはファイル・システム内のファイルを調べて、それらを表すビジネス・オブジェクトを生成します。また、アダプターに必要な他の成果物も生成します。

ビジネス・オブジェクト・エディターを使用する場合、ビジネス・オブジェクトは手動で作成します。ビジネス・オブジェクトを作成した後、ビジネス・オブジェクト・エディターを使用してビジネス・オブジェクトの階層を定義することができます。

外部サービス・ウィザードを実行すると、Adapter for Flat Filesは、コンテンツ固有および汎用という 2 つのタイプのビジネス・オブジェクトを生成します。アダプターは以下の汎用ビジネス・オブジェクト XSD ファイルを生成します。

- FlatFile.xsd
- FlatFileBG.xsd
- UnstructuredContent.xsd
- FileContent.xsd

コンテンツ固有のビジネス・オブジェクトの例として、Customer があります。Customer を選択した場合は、汎用 XSD ファイルのほかに、次のようなコンテンツ固有の XSD ファイルが生成されます。

- Customer.xsd
- CustomerWrapper.xsd
- CustomerWrapperBG.xsd

注: この例では、ビジネス・グラフ CustomerWrapperBG.xsd が生成されます。ビジネス・グラフの生成はオプションです。

アダプターの構成時に、オプションでビジネス・グラフの生成を選択することができます。バージョン 6.0.2では、各トップレベルのビジネス・オブジェクトがビジネス・グラフに含まれています。このビジネス・グラフには、アプリケーションがバージョン 6.0.2で実行する操作に関する追加情報を指定するために使用できる動詞が組み込まれています。バージョン 6.1.0 では、ビジネス・グラフはオプションです。バージョン 6.1.0 より前のバージョンの WebSphere Integration Developer で作成されたモジュールにビジネス・オブジェクトを追加する場合にのみ必要になります。ビジネス・グラフが存在する場合は処理されますが、動詞は無視されます。

外部サービス・ウィザード

アダプターが WebSphere Process Server または WebSphere Enterprise Service Bus にデプロイされる前に、外部サービス・ウィザードを使用してアダプターを構成し

ます。ウィザードは、ローカル・ファイル・システム上のファイルを調べ、(指定した検索条件に基づいて) サービスを構築し、ビジネス・オブジェクトおよびインターフェースを生成します。

外部サービス・ウィザードは、ビジネス・オブジェクトの青写真を提供します。これによって目的の成果物を選択でき、デプロイ可能なサービス・オブジェクトおよび記述を生成できます。メタデータのツリー構造からメタオブジェクト・ノードを選択することにより、EIS またはデータベース・エンティティーのビジネス・オブジェクトを生成することができます。メタデータは、ビジネス・グラフとビジネス・オブジェクトで構成されるサービス・データ・オブジェクトに変換されます。

次の図は、外部サービス・ウィザードのフローを示しています。完了すると、アダプター・プロジェクトのすべての情報を収容する EAR ファイルが作成されます。次にこの EAR ファイルは、アプリケーション・サーバーにデプロイされます。

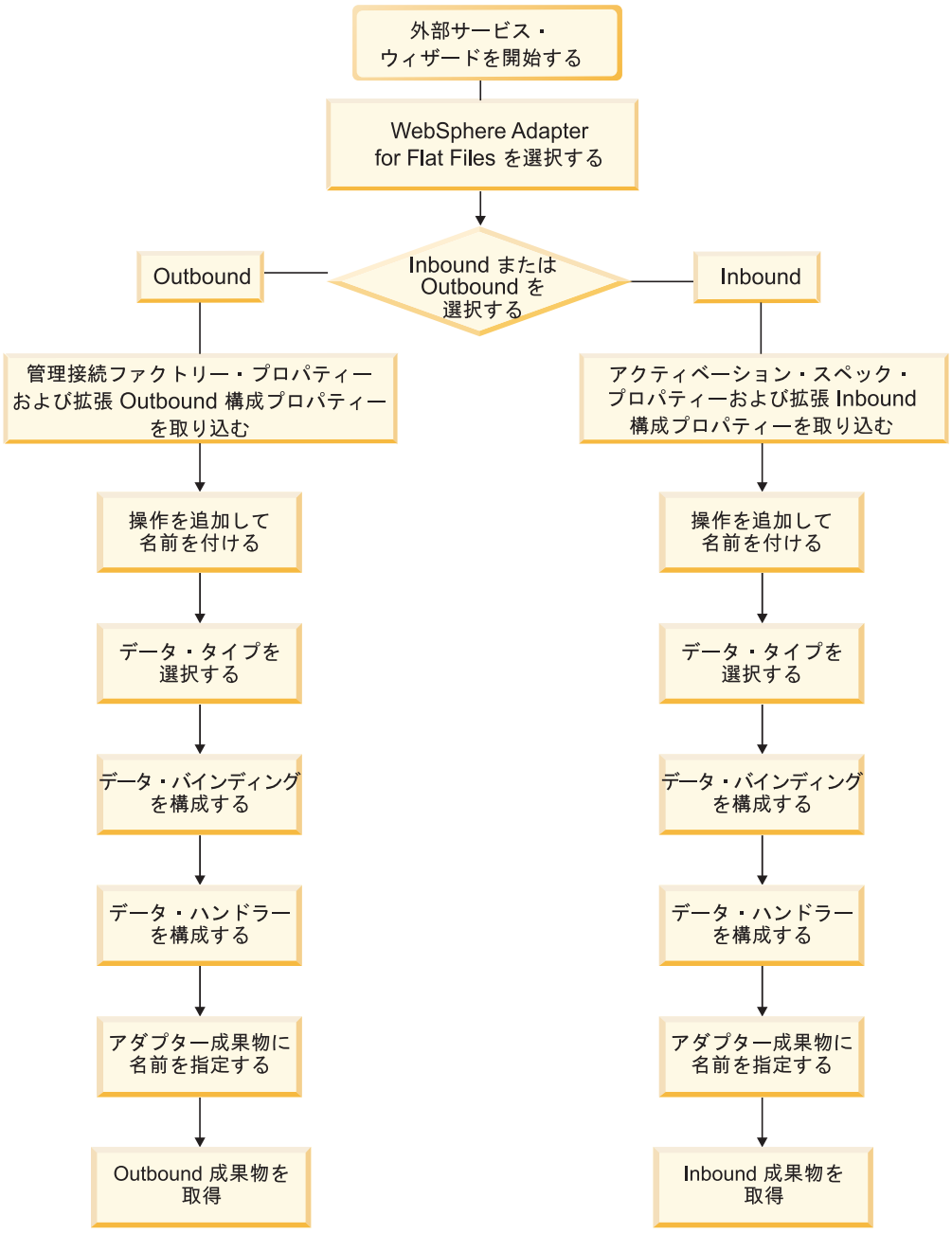


図 6. 基本的な外部サービス・ディスカバリー・ウィザードのフロー

標準規格の準拠

この製品は、アクセシビリティー標準やインターネット・プロトコル標準といった、いくつかの行政標準および業界標準に準拠しています。

アクセシビリティー

IBM は、年齢や能力を問わず、すべての人が便利に使用できる製品の提供に努めています。WebSphere Adapters は、完全にアクセス可能で、米国リハビリテーション

法第 508 条に準拠しています。アクセシビリティー機能を使用すると、運動障害または視覚障害など身体に障害を持つユーザーがソフトウェア・プロダクトを快適に操作できるようになります。これらの機能は、WebSphere Adapters のインストール機能や管理機能に組み込まれています。

管理

ランタイム管理コンソールは、エンタープライズ・アプリケーションのデプロイメントおよび管理用の基本インターフェースです。コンソールは、標準の Web ブラウザー内に表示されます。Microsoft Internet Explorer や Netscape Browser などのアクセス可能な Web ブラウザーを使用すると、次のことが可能になります。

- スクリーン・リーダー・ソフトウェアとデジタル・スピーチ・シンセサイザーを使用して、画面上に表示されている内容を聞く
- IBM ViaVoice® などの音声認識ソフトウェアを使用したデータの入力とユーザー・インターフェースへのナビゲート
- マウスの代わりにキーボードを使用して機能を操作する

プロダクト機能は、提供されるグラフィカル・インターフェースではなく、標準テキスト・エディターおよびスクリプト・インターフェースまたはコマンド行インターフェースにより、構成および使用できます。

場合によっては、特定の製品機能についての文書に、その機能のアクセシビリティーについての追加情報が記載されています。

外部サービス・ウィザード

外部サービス・ウィザードは、モジュールを作成するのに使用する主コンポーネントです。WebSphere Integration Developer を通して使用可能な Eclipse プラグインとして実装されるこのウィザードは、完全にアクセス可能です。

キーボード・ナビゲーション

この製品では、標準の Microsoft Windows ナビゲーション・キーを使用します。

IBM とアクセシビリティー

IBM のアクセシビリティーに対する取り組みについては、*IBM Accessibility Center* の Web サイト (http://www.ibm.com/able/) を参照してください。

インターネット・プロトコル バージョン 6 (IPv6)

WebSphere Process Server および WebSphere Enterprise Service Bus は、インターネット・プロトコル バージョン 6 (IPv6) の互換性について、WebSphere Application Server に依存しています。

IBM WebSphere Application Server、バージョン 6.1.0 以降では、のピュア インターネット・プロトコル バージョン 6.0 (IPv6) がサポートされます。

WebSphere Application Server におけるこの互換性について詳しくは、http://www.ibm.com/software/webservers/appserv/was/library/で、IPv6 サポートを参照してください。

IPv6 について詳しくは、http://www.ipv6.org を参照してください。

第 2 章 アダプター実装の計画

WebSphere Adapter for Flat Files を実装するには、Inbound および Outbound 処理の計画を立て、セキュリティーおよびパフォーマンス要件を考慮する必要があります。また、WebSphere Adapter for Flat Files を旧バージョンからマイグレーションする場合は、任意のマイグレーション作業を実行してください。

始める前に

アダプターのセットアップと使用を開始する前に、ビジネス・インテグレーションの概念、使用する統合開発ツールおよびランタイム環境の機能と要件、およびソリューションを構築して使用する環境について、完全に理解しておく必要があります。

WebSphere Adapter for Flat Files を構成して使用するには、以下に示す概念、ツール、およびタスクを理解し、経験しておく必要があります。

- 構築するソリューションの業務要件。
- Service Component Architecture (SCA) プログラミング・モデルなどのビジネス・インテグレーションの概念およびモデル。
- ソリューションを構築するために使用する統合開発ツールによって提供される機能。これらのツールの使用によるモジュールの作成方法、コンポーネントのテスト方法、その他の統合作業の実行方法を理解しておく必要があります。
- 統合ソリューションに使用するランタイム環境の機能および要件。ホスト・サーバーを構成および管理する方法と、管理コンソールを使用して、プロパティー定義の設定および変更、接続の構成、およびイベントの管理を行う方法を知っておく必要があります。

セキュリティー

Adapter for Flat Files は、 WebSphere Process Server を開始するユーザーの許可に依存します。

アダプターのユーザーは、アダプターがアクセス、読み取り、または変更を試行するディレクトリーおよびファイルに対して、十分なアクセス権限を持つ必要があります。

デプロイメント・オプション

デプロイされるアプリケーションの一部としてアダプターを組み込むか、スタンドアロン RAR ファイルをデプロイするかを選択することができます。

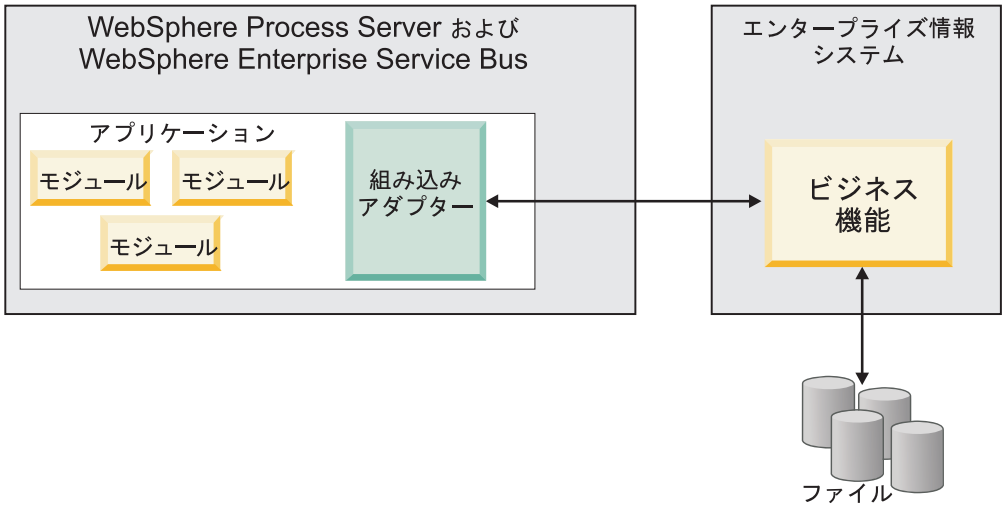
デプロイメント・オプションを以下に説明します。

- **単一アプリケーションが使用するモジュールで (With module for use by single application)**。アダプター・ファイルをモジュール内に組み込むと、モジュールをすべてのアプリケーション・サーバーにデプロイすることができます。アダプタ

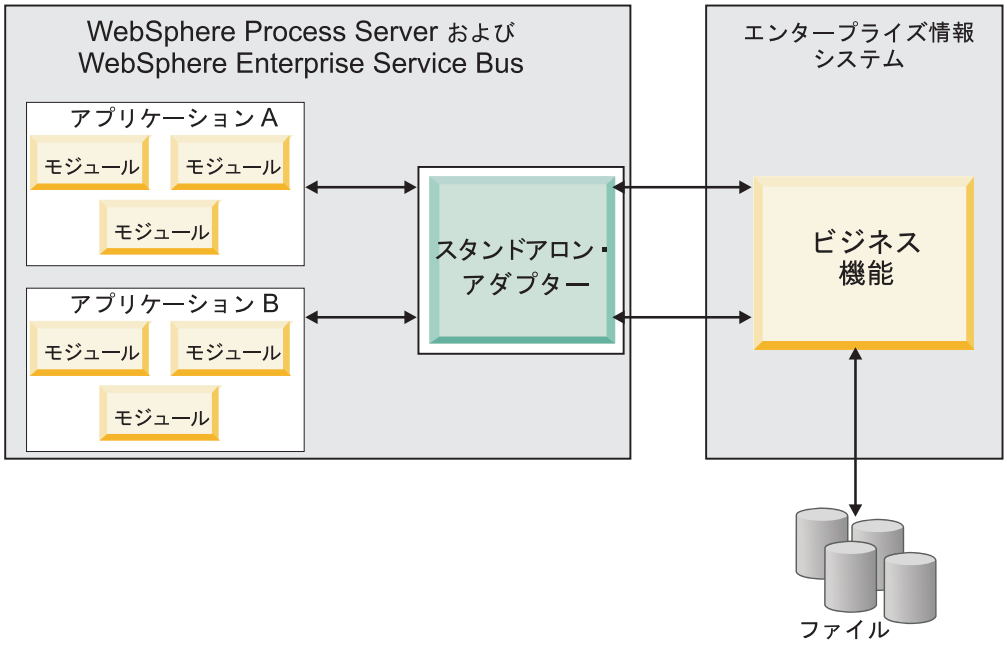
ーを使用する単一モジュールがある場合、または複数のモジュールで複数の異なるバージョンのアダプターを実行する必要がある場合は、組み込み済みアダプターを使用します。組み込み済みアダプターを使用すると、アダプター・バージョンの変更にともない他のモジュールの安定性を損なうことなく、単一モジュールのアダプターをアップグレードすることができます。

- **複数アプリケーションが使用するサーバー上 (On server for use by multiple applications)**。モジュール内にアダプター・ファイルを組み込んでいない場合は、このモジュールの実行先アプリケーション・サーバー上にスタンドアロン・アダプターとして、これらのアダプター・ファイルをインストールする必要があります。複数のモジュールで同じバージョンのアダプターを使用可能であり、そのアダプターを中央で管理する場合は、スタンドアロン・アダプターを使用します。スタンドアロン・アダプターを使用すると、複数のモジュールで単一のアダプター・インスタンスを実行することで必要となるリソースが削減されます。

エンタープライズ・アーカイブ (EAR) ファイル内には、組み込みアダプターがバンドルされています。この組み込みアダプターは、同梱のデプロイ済みアプリケーションでのみ使用することができます。



スタンドアロン・アダプターを表すのは、スタンドアロンのリソース・アダプター・アーカイブ (RAR) ファイルです。これは、デプロイすると、サーバー・インスタンス内のすべてのデプロイ済みアプリケーションで使用できるようになります。



WebSphere Integration Developer を使用してアプリケーション用のプロジェクトを作成するときには、アダプターのパッケージ方法 [(EAR) ファイルでのバンドルか、スタンドアロンの (RAR) ファイルの形式か] を選択することができます。この選択に応じて、ランタイム環境におけるアダプターの使用方法や、管理コンソールにおけるアダプターのプロパティーの表示方法が異なります。

アダプターをアプリケーションに組み込むか、スタンドアロン・モジュールとしてデプロイするかは、アダプターの管理方法に応じて選択します。アダプターを 1 コピー保持して、アダプターのアップグレード時に複数のアプリケーションを中断させなくても済むようにしたい場合は、アダプターをスタンドアロン・モジュールとしてデプロイした方が便利です。

複数のバージョンを稼働させる計画があるため、アダプターのアップグレード時に起こる可能性のある中断により配慮する場合は、アダプターをアプリケーションに組み込むことになります。アダプターをアプリケーションに組み込む場合、アダプターのバージョンをアプリケーションのバージョンに関連付けて、単一のモジュールとして管理することができます。

アダプターのアプリケーションへの組み込みに関する考慮事項

アダプターをアプリケーションに組み込む計画がある場合は、以下の点を考慮してください。

- 組み込みアダプターには、クラス・ローダーの独立性があります。

  クラス・ローダーは、アプリケーションのパッケージ化、およびランタイム環境にデプロイされたパッケージ済みアプリケーションの動作に影響を与えます。*クラス・ローダーの独立性* とは、アダプターが、他のアプリケーションまたはモジュールからクラスをロードできないということを意味します。クラス・ローダーの独立性により、異なるアプリケーション内の類似する名前を持つ 2 つのクラスは互いに干渉しなくなります。

- アダプターが組み込まれた各アプリケーションを、別々に管理する必要があります。

スタンドアロン・アダプターを使用する際の考慮事項

スタンドアロン・アダプターを使用する場合は、以下の点を考慮してください。

- スタンドアロン・アダプターには、クラス・ローダーの独立性がありません。

  スタンドアロン・アダプターにはクラス・ローダーの独立性がないため、指定された任意の Java 成果物の 1 つのバージョンのみが実行され、その成果物のバージョンおよびシーケンスは確定されません。例えば、スタンドアロン・アダプターを使用する場合、*1* つの リソース・アダプター・バージョン、 *1* つの アダプター・ファウンデーション・クラス (AFC) バージョン、 *1* つの サード・パーティーの JAR バージョンのみがあります。スタンドアロン・アダプターとしてデプロイされたアダプターはすべて、単一の AFC バージョンを共有し、1 つのアダプターのすべてのインスタンスは同じコードのバージョンを共有します。1 つのサード・パーティー・ライブラリーを使用するアダプター・インスタンスはすべて、そのライブラリーを共有しなければなりません。
- これらの共有成果物のいずれかを更新する場合、その成果物を使用するすべてのアプリケーションが影響を受けることになります。

  例えば、サーバー・バージョン X で動作しているアダプターを使用しているときに、クライアント・アプリケーションのバージョンをバージョン Y に更新する場合、元のアプリケーションの処理が停止することがあります。
- AFC には前のバージョンとの互換性がありますが、スタンドアロン方式でデプロイされるすべての RAR ファイル内には、最新の AFC バージョンが必要です。

  スタンドアロン・アダプターのクラスパス内に、任意の JAR ファイル・コピーが複数ある場合、使用される JAR ファイルはランダムになります。このため、すべてのファイルを最新バージョンにしておく必要があります。

クラスター環境での WebSphere Adapters

モジュールをクラスター化されたサーバー環境にデプロイすることで、アダプターのパフォーマンスおよび可用性を向上させることができます。スタンドアロン・アダプター、または組み込みアダプターのどちらを使用してモジュールをデプロイする場合も、モジュールは、クラスター内のすべてのサーバー内に複製されます。

WebSphere Process Server、WebSphere Application Server Network Deployment、および WebSphere Extended Deployment では、クラスター化された環境がサポートされます。クラスターとは、ワークロードの平衡を取り、高可用性とスケーラビリティーを提供するために、一緒に管理されるサーバー・グループのことです。サーバー・クラスターをセットアップするときには、デプロイメント・マネージャー・プロファイルを作成してください。デプロイメント・マネージャーのサブコンポーネントである HAManager により、アダプター・インスタンスを活動状態にするよう JCA (Java EE Connector Architecture) コンテナーに通知されます。JCA コンテナーにより、アダプター・インスタンスのランタイム環境が提供されます。「クラスター環境の作成について詳しくは、リンク http://publib.boulder.ibm.com/infocenter/

wasinfo/v6r1/index.jsp?topic=/com.ibm.websphere.nd.doc/info/ae/ae/trun_wlm_cluster_v61.html を参照してください。

必要に応じて、 WebSphere Extended Deployment を使用して、クラスター環境内のアダプター・インスタンスのパフォーマンスを向上させることができます。WebSphere Extended Deployment は、 WebSphere Application Server Network Deployment で使用される静的作業負荷マネージャーの代わりに、動的作業負荷マネージャーを使用して、 WebSphere Application Server Network Deployment の機能を拡張します。動的作業負荷マネージャーは、要求による負荷の平衡化を動的に行うことによって、クラスター内のアダプター・インスタンスのパフォーマンスを最適化できます。これは、負荷の変動に応じて、アプリケーション・サーバー・インスタンスを自動的に停止したり始動したりできることを意味します。これにより、能力や構成が異なる複数のマシンが負荷の変動に一様に対処できるようになります。WebSphere Extended Deployment の利点について詳しくは、リンク http://publib.boulder.ibm.com/infocenter/wxdinfo/v6r1/index.jsp を参照してください。

クラスター化された環境では、アダプター・インスタンスにて、Inbound 処理および Outbound 処理の両方を処理することができます。

制約事項: Inbound および Outbound 通信中、WebSphere Adapter for Flat Files は WebSphere Process Server クラスター・バックアップ・ノードとクラスターのプライマリー・ノードが異なるオペレーティング・システムにインストールされている場合、これらのノード間でプーリングを切り替えることができません。例えば、アダプターがプライマリー Windows ノードでプーリングを開始する場合、バックアップ UNIX ノードに切り替えることはできず、進行中のイベントのディレクトリー保管に使用される Windows パスを処理することはできません。

Inbound 処理の高可用性

Inbound 処理は、ローカル・ファイル・システムのデータを更新した結果、起動するイベントに基づいています。WebSphere Adapter for Flat Files は、イベント・テーブルをポーリングすることで更新を検出するよう構成されます。その後、アダプターはイベントをそのエンドポイントにパブリッシュします。

重要: クラスター化された環境では、イベント・ディレクトリーはファイル共用システムに存在しなければならず、いずれのクラスター・マシンに対してもローカルであってはなりません。

モジュールをクラスターにデプロイすると、JCA (Java EE Connector Architecture) コンテナーにより、enableHASupport リソース・アダプター・プロパティーが検査されます。enableHASupport プロパティーの値が真である場合 (デフォルトの設定)、すべてのアダプター・インスタンスはポリシー N のうちの 1 つを持つ HAManager に登録されます。 このポリシーは、アダプター・インスタンスのうちの 1 つのみがイベントのポーリングを開始することを意味します。 クラスター内のその他のアダプター・インスタンスが開始していても、それらのインスタンスは、アクティブなアダプター・インスタンスがイベントの処理を完了するまで、アクティブ・イベントに関して休止のままとなります。ポーリング・スレッドが開始しているサーバーが何らかの理由でシャットダウンした場合は、バックアップ・サーバーのいずれかで稼働しているアダプター・インスタンスが活動状態になります。

重要: enableHASupport プロパティーの設定は、変更しないでください。

Outbound 処理の高可用性

クラスター化された環境では、Outbound 処理要求の実行に、複数のアダプター・インスタンスが使用可能です。そのため、ご使用の環境に Outbound 要求のためにWebSphere Adapter for Flat Filesと対話するアプリケーションが複数ある場合、モジュールをクラスター化された環境にデプロイすることで、パフォーマンスが向上することがあります。クラスター化された環境では、複数の Outbound 要求が同じレコードを処理しようとしない限り、複数の Outbound 要求を同時に処理することができます。

複数の Outbound 要求が同じレコード (顧客の住所など) を処理しようとした場合、その要求は WebSphereApplication Server Network Deployment のワークロード管理機能によって、受信順に使用可能なアダプター・インスタンスの間で分配されます。このため、クラスター化された環境では、この種の Outbound 要求は、単一サーバー環境内と同じように処理されます。つまり、1 つのアダプター・インスタンスが一度に処理するのは、1 つの Outbound 要求のみです。ワークロード管理について詳しくは、リンク http://publib.boulder.ibm.com/infocenter/wasinfo/v6r1/index.jsp?topic=/com.ibm.websphere.nd.doc/info/ae/ae/trun_wlm.html を参照してください。

バージョン 6.1.0 へのマイグレーション

WebSphere Adapter for Flat Files のバージョン 6.1 へマイグレーションすると、アダプターが前のバージョンから自動的にアップグレードされます。さらに、前のバージョンのアダプターが組み込まれたアプリケーションをマイグレーションして、バージョン 6.1 にあるフィーチャーや機能を使用できるようにすることができます。

マイグレーションに関する考慮事項

WebSphere Adapter for Flat Files バージョン 6.1.0 には、既存のアプリケーションに影響を与える可能性のある更新が含まれています。 WebSphere Adapter for Flat Files を使用するアプリケーションをマイグレーションする前に、後のセクションに記載されている情報を考慮してください。

旧バージョンとの互換性

WebSphere Adapter for Flat Files バージョン 6.1.0 は、アダプターの バージョン 6.0.2 と完全に互換性があり、カスタム・ビジネス・オブジェクト (XSD ファイル) およびデータ・バインディングを操作できます。

WebSphere Adapter for Flat Files バージョン 6.1 は バージョン 6.02 と完全に互換性があるため、WebSphere Adapter for Flat Files バージョン 6.0.2 を使用していたすべてのアプリケーションは、バージョン 6.1 にアップグレードする際に、変更せずにそのまま実行することができます。ただし、アプリケーションで、バージョン 6.1 のアダプターで提供されるフィーチャーと機能を使用したい場合は、マイグレーション・ウィザードを実行してください。

マイグレーション・ウィザードでは、バージョン 602 のアダプターをバージョン 6.1 に置き換え (アップグレードし)、*バージョン 6.1 のフィーチャーと機能をアプリケーションで使用*できるようにします。

注: マイグレーション・ウィザードでは、バージョン 6.1 のアダプターで動作するマッパーやメディエーターなどの緩和コードを新規作成したり、既存の緩和コードを変更したりすることはありません。任意のアプリケーションにバージョン 6.0.2 以前のアダプターが組み込まれており、バージョン 6.1.0 へのアップグレードを行っている場合、6.1 のフィーチャーと機能を利用したいのであれば、これらのアプリケーションに変更を加えなければならない可能性があります。

成果物に、単一モジュール内の*バージョン管理* に関する矛盾がある場合、このモジュール全体にその旨を示すマークが付けられ、マイグレーションの際に選択できなくなります。バージョンに矛盾がある場合、プロジェクトが破損している可能性があるため、その矛盾の内容はワークスペース・ログに記録されます。

アップグレードのみを行うか、アップグレードとマイグレーションの両方を行うことを決定

マイグレーション・ウィザードのデフォルト処理では、アダプターのアップグレードを行い、アプリケーションでバージョン 6.1 のアダプターのフィーチャーと機能を使用できるように、アプリケーション成果物をマイグレーションします。コネクター・プロジェクトを選択してコネクターをアップグレードする設定を行った場合は、ウィザードが自動的にマイグレーション対象の関連付けられた成果物を選択します。

アダプターをバージョン 6.0.2 からバージョン 6.1 にアップグレードするが、 アダプター成果物をマイグレーションしない場合は、マイグレーション・ウィザードの該当するページでアダプター成果物の選択を解除することでそのように設定できます。

アダプター・成果物を選択しないでマイグレーション・ウィザードを実行すると、アダプターがインストールされてアップグレードされますが、成果物はマイグレーションされません。また、アプリケーションで、バージョン 6.1 のアダプターで提供されるフィーチャーや機能を利用できません。

テスト環境におけるマイグレーション・ウィザードの最初の実行

アダプター・マイグレーションを行うには、WebSphere Adapter for Flat Files バージョン 6.1 を使用するアプリケーションの変更が必要になる場合があるため、必ず最初に開発環境でマイグレーションを行って、アプリケーションをテストしてから、そのアプリケーションを実稼働環境にデプロイしてください。

マイグレーション・ウィザードは開発環境と完全に統合されています。

非推奨の機能

バージョン 6.1.0 の非推奨となっている機能を理解してから、アプリケーションに必要な変更を行ってください。

非推奨の機能とは、サポートされていても推奨されてはおらず、廃止される可能性がある機能です。バージョン 6.1.0 で推奨されない、旧バージョンの WebSphere Adapter for Flat Filesの機能は以下のとおりです。

- アクティベーション・スペック:
  - アーカイビング・プロセス
  - イベント・コンテンツ・タイプ (EventContentType)
  - デフォルト・オブジェクト名 (DefaultObjectName)
- 対話仕様:
  - デフォルト・オブジェクト名 (DefaultObjectName)
- ラッパー・プロパティー:
  - 取り出されたコンテンツのタイプ (RetrieveContentType)
  - デフォルト・オブジェクト名 (DefaultObjectName)

マイグレーションの実行

アダプター・マイグレーション・ウィザードを使用して、プロジェクトまたは EAR ファイルを バージョン 6.1.0 にマイグレーションできます。ツールが終了するとマイグレーションは完了し、プロジェクトの処理またはモジュールのデプロイを行うことができます。

始める前に

「*マイグレーションに関する考慮事項* 」に記載されている情報を確認します。

このタスクを実行する理由および時期

WebSphere Integration Developer でマイグレーションを実行するには、以下のステップを完了してください。

注: マイグレーションの完了後、モジュールは、前のバージョンの WebSphere Process Server、WebSphere Enterprise Service Bus、または WebSphere Integration Developer との互換性を持たなくなります。

注: 以下のステップでは、WebSphere Integration Developer の J2EE パースペクティブにあるコネクター・プロジェクト・コンテキスト・メニューから、アダプター・マイグレーション・ウィザードを実行する方法を説明します。

注: 次のいずれかの方法で、マイグレーションを行うこともできます。

- J2EE パースペクティブのプロジェクトを右クリックし、「**マイグレーション**」 → 「**プロジェクトのマイグレーション (Migrate project)**」と選択します。
- 「問題」ビューから、マイグレーション固有のメッセージを右クリックし、「**クイック・フィックス (Quick Fix)**」を選択して問題を訂正します。

このタスクの手順

1. 既存のプロジェクト用の PI (プロジェクト交換) ファイルまたはデプロイ済みアプリケーション用の EAR (エンタープライズ・アーカイブ) ファイルをワークスペースにインポートします。
2. J2EE パースペクティブへ変更します。

3. モジュールを右クリックして、「**マイグレーション**」 → 「**コネクター・プロジェクトの更新**」と選択します。
4. ウェルカム・ページにあるタスクおよび警告を確認してから、「**次へ**」を選択します。
5. 「プロジェクトの選択」ウィンドウで、「**次へ**」を選択します。

   デフォルトでは、ウィザードは、コネクター・プロジェクトとすべての従属プロジェクトをマイグレーションします。プロジェクトに従属プロジェクトがあり、現時点ではそれらの 1 つ以上をマイグレーションしたくない場合、「**従属アダプター・プロジェクト (Dependent adapter project)**」リストのチェック・ボックスのチェック・マークを外します。ウィザードを再実行して、後で従属プロジェクトをマイグレーションすることができます。既にマイグレーション済みのプロジェクト、現行バージョンのプロジェクト、およびエラーを含むプロジェクトは、マイグレーションで選択不可であるため、選択されません。
6. 「アダプターのマイグレーション」ウィンドウで、オプションで、マイグレーションの変更内容を確認します。ただし、選択項目は変更しないでください。「**終了**」をクリックします。
7. 「問題」ビューを調べて、ストリング CWPAD で始まるマイグレーション・ウィザードからのメッセージを探します。
8. EAR ファイルをマイグレーションする場合、オプションで、マイグレーション済みのアダプターおよび成果物を持つ新規の EAR ファイルを作成し、WebSphere Process Server または WebSphere Enterprise Service Bus にデプロイします。 EAR ファイルのエクスポートおよびデプロイについて詳しくは、本書の該当するトピックを参照してください。

結果

プロジェクトまたは EAR ファイルは、バージョン 6.1.0 にマイグレーションされます。アダプター・マイグレーション・ウィザードを終了した後で、外部サービス・ウィザードを実行する必要はありません。

バージョン 6.0.2 プロジェクトをマイグレーションせずに更新する

アダプター・プロジェクト成果物をマイグレーションしないように選択した状態で、バージョン 6.0.2 から バージョン 6.1.0 へアダプターをアップグレードすることができます。

このタスクを実行する理由および時期

アダプター・ウィザード WebSphere Integration Developer バージョン 6.1.0 を使用する前に、バージョン 6.1.0 で変更されたアダプターの内部名、バージョン 6.0.2 プロジェクトの成果物は、新規名を使用するように更新する必要があります。バージョン 6.0.2 プロジェクトを更新するには、マイグレーション・ウィザードを使用します。次に、WebSphere Integration Developerのクイック・フィックス機能を使用して、プロジェクト成果物のアダプター名を変更します。

このタスクの手順

1. プロジェクト交換 (PI) ファイルをワークスペースへインポートします。

2. J2EE パースペクティブで、プロジェクト名を右クリックし、「**マイグレーション**」 → 「**コネクター・プロジェクトの更新**」をクリックします。 アダプター・マイグレーション・ウィザードが開きます。
3. ウェルカム・ページで、「**次へ**」をクリックします。
4. 「プロジェクトの選択」ウィンドウで、従属成果物プロジェクトを選択解除して、「**終了**」をクリックします。
5. 「クイック・フィックス (Quick Fix)」ウィンドウで、修正「**参照されたアダプターの名前変更 (Rename the referenced adapter)**」が選択されていることを確認して、「**OK**」をクリックします。
6. エラーがまだ表示される場合、「**プロジェクト**」 → 「**クリーン**」とクリックし、先ほど更新したプロジェクトを選択して、「**OK**」をクリックします。

結果

プロジェクトは、WebSphere Adapter for Flat Files バージョン 6.1.0 で使用することができるようになりました。

第 3 章 サンプルとチュートリアル

WebSphere Integration Developer のオンラインのサンプル/チュートリアル・ギャラリーには、WebSphere Adapters を使用する際に役立つサンプルとチュートリアルが含まれています。

以下のようにしてオンラインのサンプル/チュートリアル・ギャラリーにアクセスできます。

- WebSphere Integration Developer を始動したときに開くウェルカム・ページから。WebSphere Adapter for Flat Files のサンプルとチュートリアルを表示するには、「**取得 (Retrieve)**」 をクリックします。次に、表示されたカテゴリーを参照して選択します。
- Web 上のロケーション http://publib.boulder.ibm.com/bpcsamp/index.html から。

第 4 章 デプロイメントのためのモジュールの構成

アダプターを WebSphere Process Server または WebSphere Enterprise Service Bus 上にデプロイできるように構成するには、WebSphere Integration Developer を使用して、アダプターをデプロイするときに EAR ファイルとしてエクスポートされるモジュールを作成します。その後、作成するビジネス・オブジェクトと、作成を行うシステムを指定します。これらのステップを完了すると、外部サービスが正常に作成されます。

モジュールの構成のためのロードマップ

ランタイム環境で WebSphere Adapter for Flat Files を使用できるようにするには、まずモジュールを構成する必要があります。このタスクの概要を理解すれば、タスクを達成するのに必要な手順を実行できるようになります。

WebSphere Adapter for Flat Filesのモジュールを構成するには、WebSphere Integration Developerを使用します。以下の図は、構成作業の流れを示しています。また、図の後に示す手順で、この作業の概要を説明します。これらの各ステップの実行方法の詳細については、このロードマップの後に記載するトピックを参照してください。

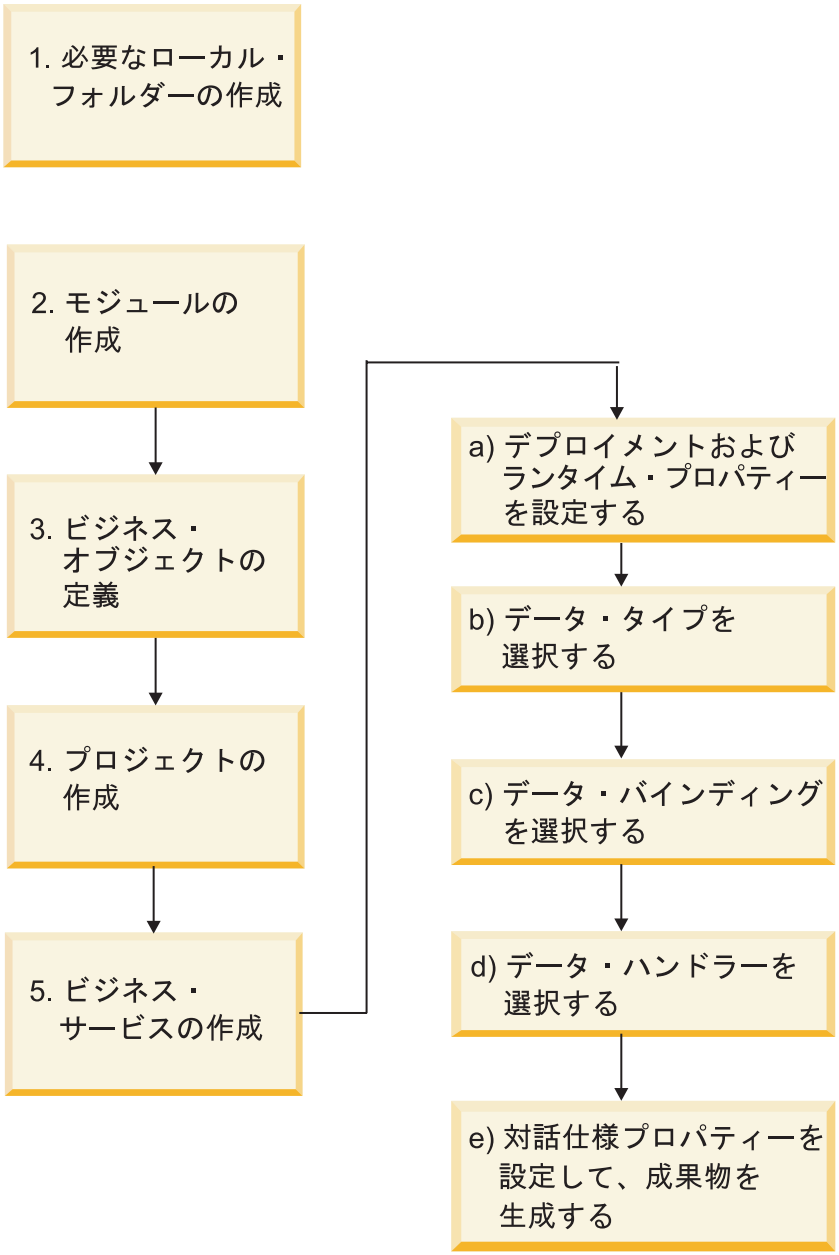


図 7. モジュールの構成のためのロードマップ

モジュールの構成

この作業は、次の概略的なステップから成ります。

注: これらのステップでは、データ変換が必要なユーザー定義のビジネス・オブジェクトを使用していることを前提としています。データ変換が不要な汎用ビジネス・オブジェクトを使用している場合は、以下のステップの一部は無視されます。例えば、データ・バインディングとデータ・ハンドラーを選択する必要はありません。

1. WebSphere Integration Developer 内にモジュールを作成します。次に、モジュールにビジネス・オブジェクトを作成します。
2. プロジェクトで使用されるビジネス・オブジェクトを定義します。

3. プロジェクトを作成します。これは、WebSphere Integration Developer で外部サービス・ウィザードを使用していてアダプターに関連付けられたファイルを編成するために使用します。
4. WebSphere Integration Developer から外部サービス・ウィザードを実行してビジネス・サービスを作成した後に、以下の手順を実行します。
   a. 以下のデプロイメントおよびランタイム・プロパティーを指定します。
      - 接続プロパティー
      - セキュリティー・プロパティー
      - デプロイメント・オプション
      - 関数セレクター - Inbound のみ
   b. データ・タイプを選択し、そのデータ・タイプに関連付けられる操作を指定します。操作ごとに、以下を指定します。
      - 操作の種類。例えば、Create、Append、Exists です。
      - 操作をパススルーまたはユーザー定義に指定します。
   c. データ・バインディングを選択します。各データ・タイプには、ビジネス・オブジェクトのフィールドを読み取ったり、ファイルの対応するフィールドを設定したりするために使用する同等のデータ・バインディングがあります。
   d. ビジネス・オブジェクトとネイティブ形式の間の変換を実行するデータ・ハンドラーを選択します。
   e. 対話仕様プロパティー値を指定して、成果物を生成します。外部サービス・ウィザードを実行した結果生成される出力は、ビジネス・インテグレーション・モジュールに保存されます。ここには、ビジネス・オブジェクト (1 つまたは複数)、およびインポートまたはエクスポート・ファイルが格納されます。

必須のローカル・フォルダーの作成

Inbound モジュールまたは Outbound モジュールを作成する前に、イベントおよび出力用のフォルダーをローカル・ファイル・システムに作成する必要があります。またオプションで、ステージングおよびアーカイブ用のフォルダーを作成することができます。

Inbound モジュールまたは Outbound モジュールを作成する前に、イベント・ディレクトリーおよび出力ディレクトリーを、外部サービス・ウィザードの「サービス構成プロパティー」画面で指定する必要があります。また、ステージング・ディレクトリーおよびアーカイブ・ディレクトリーも作成することができますが、これらは必須ではありません。

- イベント・ディレクトリーは Inbound 処理のイベントを保管します。アダプターは、一定の間隔でこのフォルダーをポーリングし、イベントが検出されると、それをビジネス・オブジェクトの形式でサーバーに送信します。
- 出力ディレクトリーは、Outbound 処理中の Create、Append、および Overwrite 操作の最終出力ファイルを書き込むために、アダプターが使用します。

- ステージング・ディレクトリーは、アダプターが書き込みの競合を回避するために、Create および Overwrite 操作中に初期出力ファイルを書き込む一時ディレクトリーです。出力ファイルはその後名前変更され、出力ディレクトリーにコピーされます。
- アーカイブ・ディレクトリーは、アダプターが処理済みイベント・ファイルを保管するディレクトリーです。

モジュールの作成

WebSphere Integration Developer 内に、モジュールを作成します。このモジュールにより、プロジェクトで使用されるビジネス・オブジェクトを定義できます。

このタスクを実行する理由および時期

外部サービス・ウィザードを起動し、この手順に従って新規モジュールを作成してください。

このタスクの手順

1. WebSphere Integration Developer が現在実行されていない場合は、開始します。
   a. **「スタート」 → 「プログラム」 → 「IBM WebSphere」 → 「Integration Developer V6.1.0」 → 「WebSphere Integration Developer V6.1.0」** をクリックします。
   b. ワークスペースを指定するようにプロンプトが出された場合は、デフォルト値を受け入れるか、または別のワークスペースを選択します。

      ワークスペースとは、WebSphere Integration Developer がプロジェクトを保管するディレクトリーのことです。
   c. オプション: WebSphere Integration Developer ウィンドウが表示されたら、**「ビジネス・インテグレーション・パースペクティブに移動 (Go to the Business Integration perspective)」**をクリックします。
2. WebSphere Integration Developer ウィンドウの「ビジネス・インテグレーション」セクション内を右クリックします。**「新規」 → 「モジュール」**をクリックします。

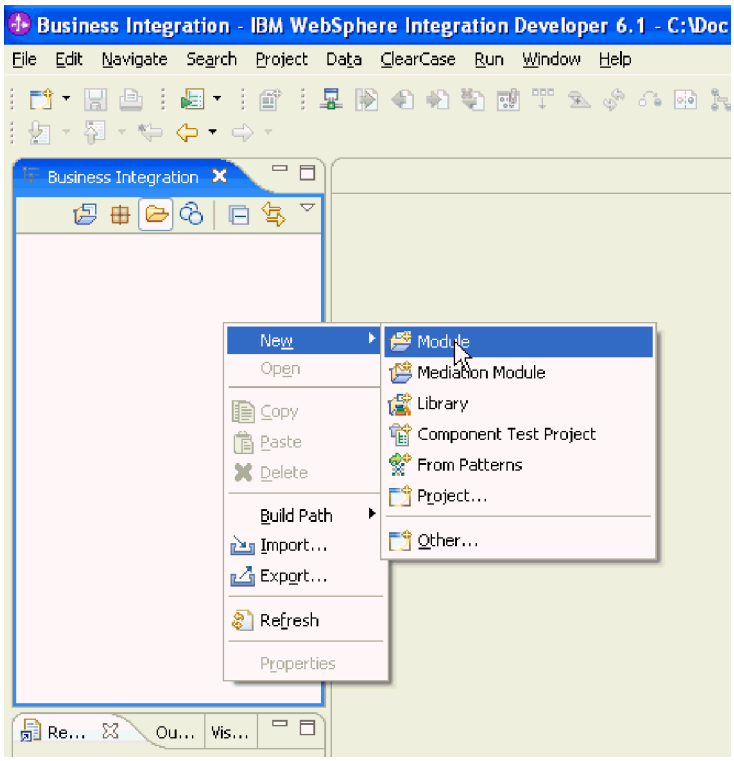


図 8. ウィンドウの「ビジネス・インテグレーション」セクション

3. 「新規モジュール」ウィンドウで、「**モジュール名**」に新しいモジュールの名前を入力します。その他のオプション (「**デフォルト・ロケーションの使用 (Use default location)**」および「**モジュール・アセンブリー・ダイアグラムを開く (Open module assembly diagram)**」) はチェック・マークを付けたままにします。

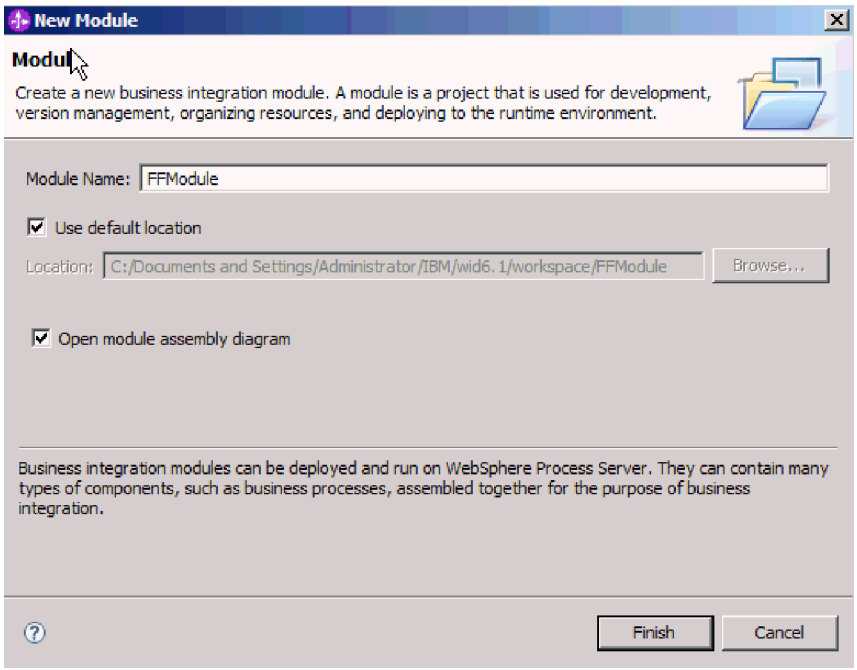


図 9. 「新規モジュール」ウィンドウ

4. **「終了」**をクリックします。

結果

「ビジネス・インテグレーション」ウィンドウに新しいモジュールがリストされます。

次のタスク

プロジェクトを作成します。これは、アダプターに関連付けられたファイルを編成するために使用されます。

ビジネス・オブジェクトの定義

次のトピックで作成するプロジェクトで使用されるビジネス・オブジェクトを WebSphere Integration Developer で事前定義します。

このタスクを実行する理由および時期

ビジネス・オブジェクト・エディターを使用して、新規ビジネス・オブジェクトの事前定義を行うには、以下のステップを実行します。

このタスクの手順

1. WebSphere Integration Developer ウィンドウの「ビジネス・インテグレーション」セクション内にある新しいモジュールを展開します。
2. **「データ・タイプ」**フォルダーを右クリックして、**「新規」 >「ビジネス・オブジェクト」**を選択します。

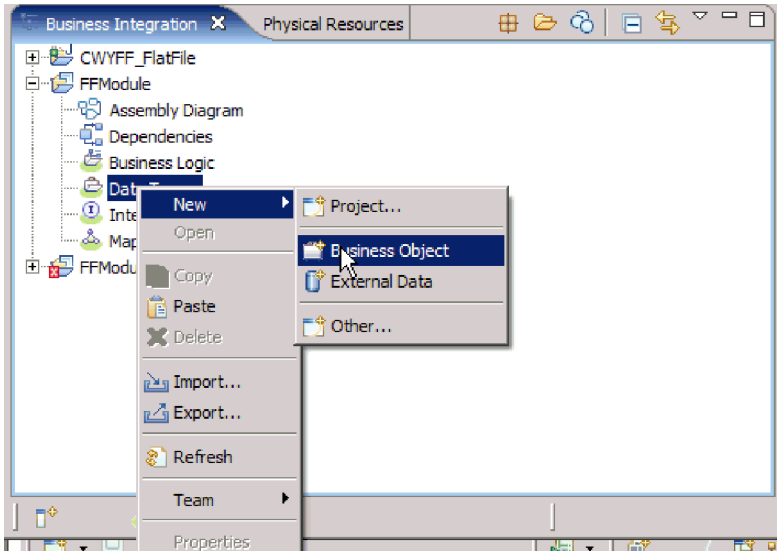


図 10. 新規ビジネス・オブジェクトの選択ビュー

3. 「ビジネス・オブジェクト」ウィンドウで、「**名前**」に新しい名前を入力します。

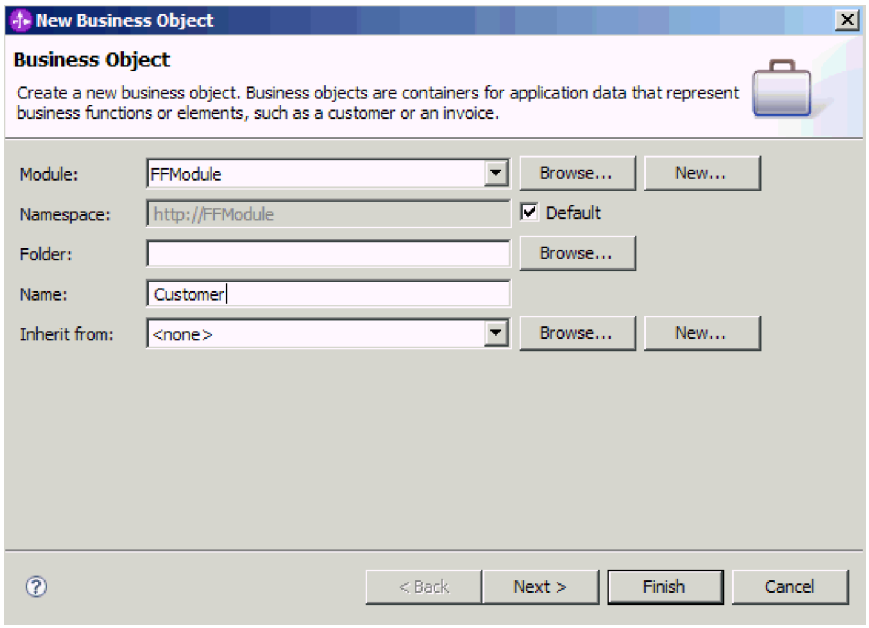


図 11. 「ビジネス・オブジェクト」ウィンドウ

4. 「**終了**」をクリックします。 「**データ・タイプ**」フォルダーに新しいビジネス・オブジェクトが追加されます。
5. 「**ビジネス・オブジェクトにフィールドを追加 (Add a field to a business object)**」アイコンをクリックして、ビジネス・オブジェクトに必要なフィールドを追加します。

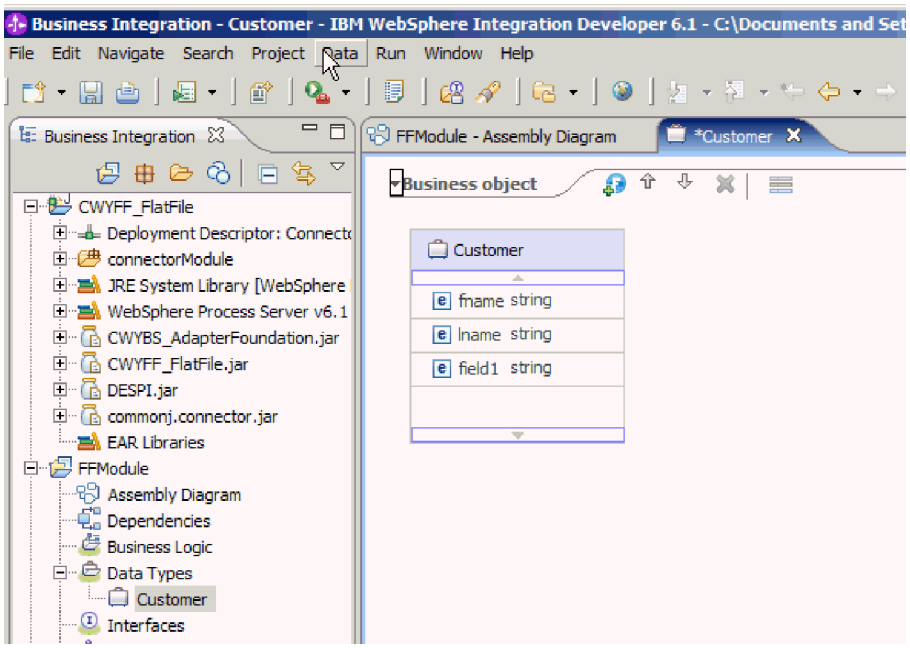


図 12. 「ビジネス・オブジェクト・フィールドの追加 (Add Business object fields)」アイコンを追加します。

6. 「保管」アイコンをクリックします。
7. 作成する各ビジネス・オブジェクトについて、上記の手順を繰り返します。

結果

新しいビジネス・オブジェクトが定義されます。

次のタスク

プロジェクトを作成します。これは、アダプターに関連付けられたファイルを編成するために使用されます。

アダプター・パターン・ウィザードを使用した単純なサービスの作成

アダプター・パターンは、アダプターで単純なサービスを素早く簡単に作成する方法を提供します。

始める前に

RetrieveAFileModule という名前のモジュールと、Customer という名前のビジネス・オブジェクトが既に作成されています。

このタスクを実行する理由および時期

以下のアダプター・パターンが、Adapter for Flat Filesで使用可能です。

表 8.

アダプター・パターン	説明
Inbound Flat File パターン	Flat File Inbound パターンは、ローカル・ファイル・システム上の特定のディレクトリーにあるファイルを取り出すサービスを作成します。ファイルの形式が XML 以外の場合は、ファイル内容の形式をビジネス・オブジェクトに変換するデータ・ハンドラーを指定できます。ファイル内容に処理対象のデータ構造のコピーが複数含まれている場合は、その内容を分割することができます。
Outbound Flat File パターン	Flat File Outbound パターンは、ローカル・ファイル・システム上の特定のディレクトリーにあるファイルにデータを保管するサービスを作成します。必要な出力形式が XML 以外の場合は、ビジネス・オブジェクトをファイル内容の形式に変換するデータ・ハンドラーを指定できます。

ここでは、処理対象のファイル・システムからファイルを受け取る Flat File Inbound サービスの作成例を示します。この例の完全なサービスでは、ファイルを読み取り、区切り文字に基づいて内容を個別のファイルに分割します。

以下のステップを実行して、アダプター・パターン・ウィザードでサービスを作成します。

このタスクの手順

1. WebSphere Integration Developer ウィンドウの「**ビジネス・インテグレーション**」セクション内にある「**RetrieveAFileModule**」を右クリックし、「**新規**」 → 「**パターンから**」を選択します。「パターンから新規作成」ウィンドウが開きます。
2. 「**ローカル・ファイルから読み取る Inbound Flat File サービスを作成 (Create an inbound Flat File service to read from a local file)**」を選択して、「**次へ**」をクリックします。

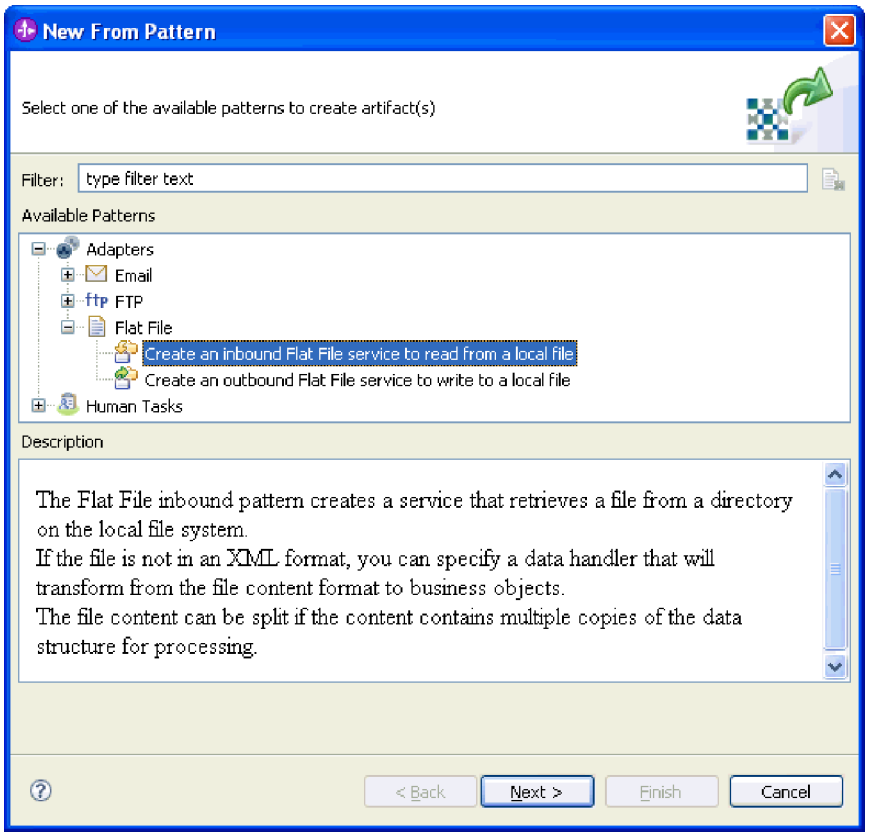


図 13. 「パターンから新規作成」ウィンドウ

3. 「新規 Inbound Flat File サービス (New Inbound Flat File Service)」ウィンドウで、名前を「`FlatFileInboundInterface`」などの有効な名前に変更してから「**次へ**」をクリックします。

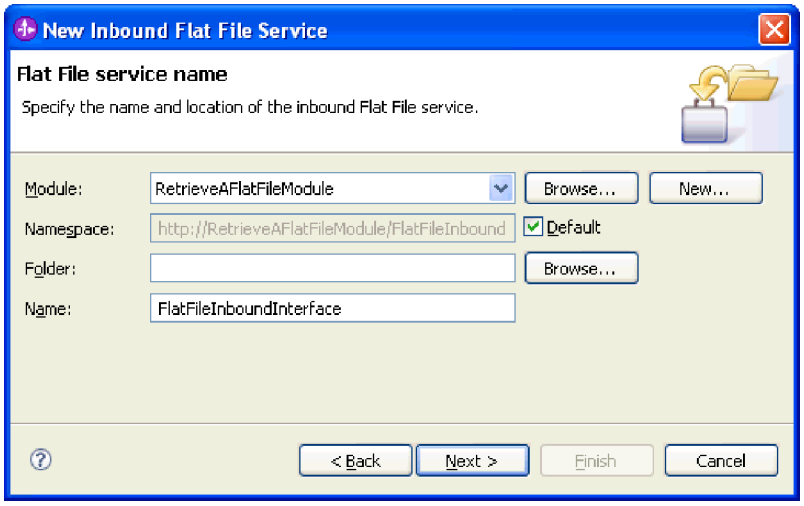


図 14. 「Flat File サービス名 (Flat File service name)」ウィンドウ

4. 「ビジネス・オブジェクトおよびディレクトリー (Business object and directory)」ウィンドウで、「**参照**」をクリックして、「**Customer**」ビジネス・オブジェクトにナビゲートします。

5. 入力ファイルを配置したディレクトリー (この場合は、「FFInboundEvents」ディレクトリー) を指定して、「**次へ**」をクリックします。

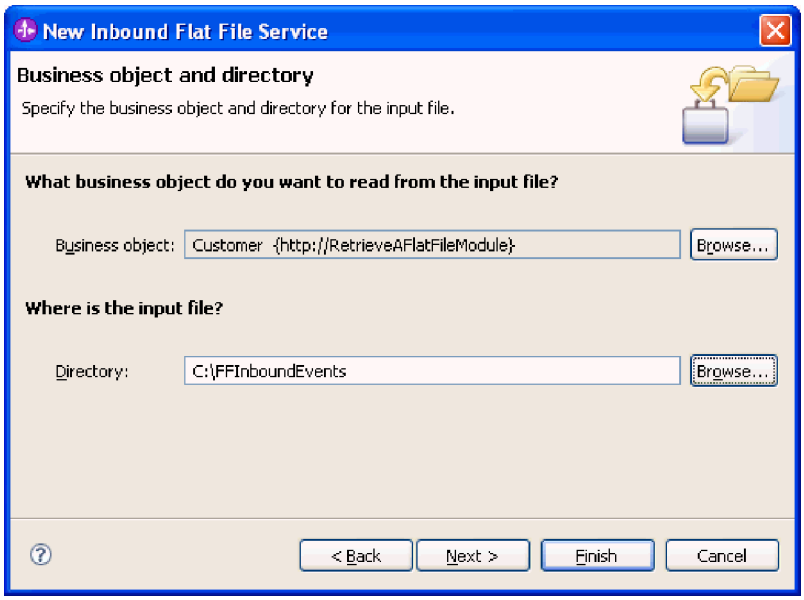


図 15. 「ビジネス・オブジェクトおよびディレクトリー (Business object and directory」ウィンドウ

6. 「入力ファイル・フォーマットとファイル内容の分割オプション」ウィンドウで、デフォルトの XML 入力ファイル形式を受け入れるか、「**その他**」を選択して、ネイティブ形式からビジネス・オブジェクト形式にデータを変換するデータ・ハンドラーを指定します。
7. 「**区切り文字でファイル内容を分割する**」を選択し、使用する区切り文字 (この場合は、「####;¥r¥n」) を入力します。「**次へ**」をクリックします。

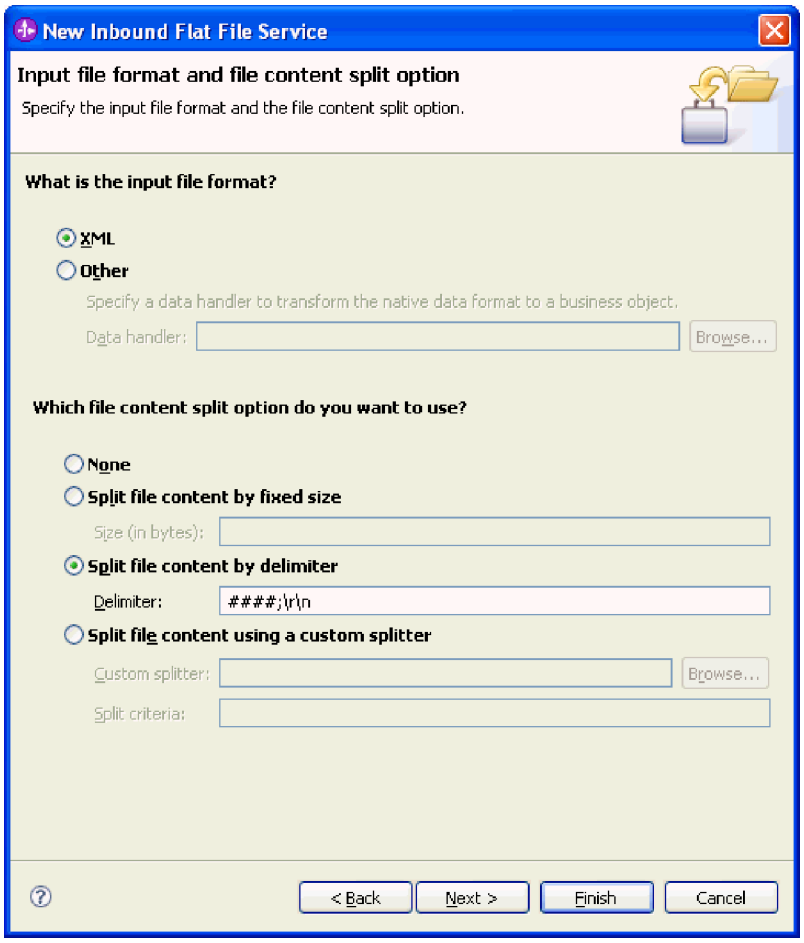


図 16. 「入力ファイル・フォーマットとファイル内容の分割オプション」ウィンドウ

8. 「アーカイブ・ディレクトリーおよびラッパー・ビジネス・オブジェクト」ウィンドウで、「**ローカル・アーカイブ・ディレクトリー**」 (この場合は、「`FFInboundArchive`」) を指定します。アダプター固有の情報を組み込む場合は、「**入力ファイル情報を追加するためにラッパー・ビジネス・オブジェクトを使用する**」を選択します。「**終了**」をクリックします。

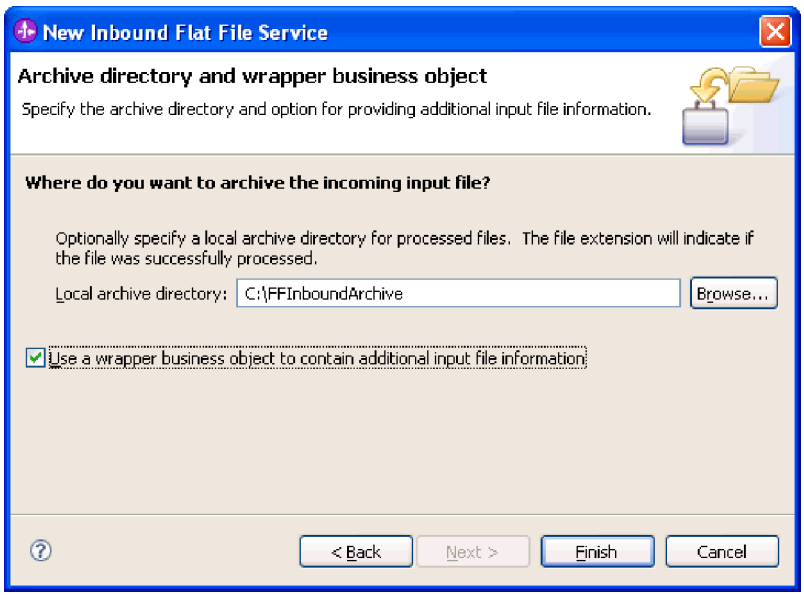


図 17. アーカイブ・ディレクトリーおよびラッパー・ビジネス・オブジェクト」ウィンドウ

結果

以下の成果物を含む Inbound サービスが作成されます。

表 9.

成果物	名前	説明
エクスポート	FlatFileInboundInterface	エクスポートは、モジュールを外部に公開します。ここでは、WebSphere Adapter for Flat Files が対象になります。
ビジネス・オブジェクト	Customer、CustomerWrapper	Customer ビジネス・オブジェクトには、名前、住所、都道府県などのお客様データを入力するフィールドが含まれます。CustomerWrapper ビジネス・オブジェクトには、アダプター固有の情報を入力する追加フィールドが含まれます。
インターフェース	FlatFileInboundInterface	このインターフェースには、起動可能な操作が含まれます。
操作	emitCustomerInput	emitCustomerInput は、インターフェースにある唯一の操作です。

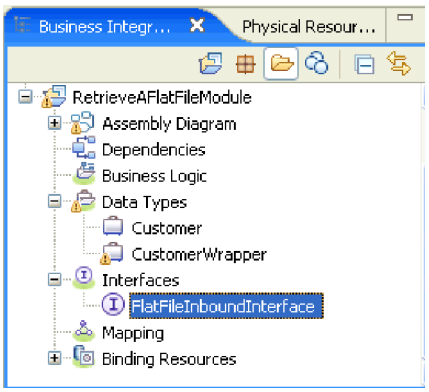


*図 18. 新規成果物を表示する WebSphere Integration Developer ウィンドウの「**ビジネス・インテグレーション**」セクション*

プロジェクトの作成

モジュールの作成およびデプロイを行うには、WebSphere Integration Developer の外部サービス・ウィザードを開始します。このウィザードは、モジュールに関連付けられたファイルを編成するために使用するプロジェクトを作成します。

このタスクを実行する理由および時期

外部サービス・ウィザードを開始して、 WebSphere Integration Developer のアダプター用のプロジェクトを作成します。既存プロジェクトがある場合は、ウィザードで新規作成する代わりにそれを選択することができます。

外部サービス・ウィザードを開始し、プロジェクトを作成するには、次の手順を実行します。

このタスクの手順

1. WebSphere Integration Developer が現在実行されていない場合は、開始します。
   a. 「**スタート**」 → 「**プログラム**」 → 「**IBM Software Development Platform**」 → 「**IBM WebSphere Integration Developer 6.1**」 → 「**WebSphere Integration Developer V6.1**」をクリックします。
   b. ワークスペースを指定するようにプロンプトが出された場合は、デフォルト値を受け入れるか、別のワークスペースを選択します。

      ワークスペースとは、WebSphere Integration Developer がプロジェクトを保管するディレクトリーのことです。
   c. WebSphere Integration Developer ウィンドウが表示されたら、「**ビジネス・インテグレーション・パースペクティブに移動 (Go to the Business Integration perspective)**」をクリックします。
2. 外部サービス・ウィザードを開始するには、「**ファイル**」 → 「**新規**」 → 「**外部サービス (External Service)**」をクリックします。
3. 「新規の外部サービス (New external service)」ウィンドウで、「**アダプター (Adapters)**」が選択済みであることを確認して、「**次へ**」をクリックします。

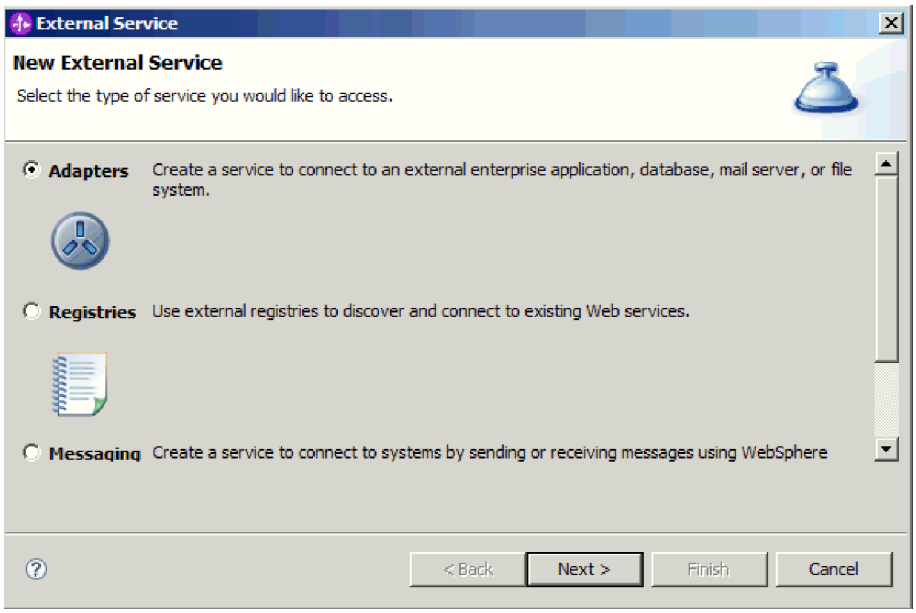


図 19. 「新規の外部サービス (New external service)」ウィンドウ

4. 「エンタープライズ・サービス・リソース・アダプターを選択 (Select an Enterprise Service Resource Adapter)」ウィンドウから、プロジェクトを作成または既存プロジェクトを選択します。
   - プロジェクトを作成するには、以下の手順を実行します。
     a. 「**IBM WebSphere Adapter for Flat Files**」を選択し、「**次へ**」をクリックします。
     b. 「コネクター・インポート」ウィンドウで、プロジェクトの別名を指定し (「**CWYFF_FlatFile**」 以外の名前を使用するため)、サーバーを選択して (例えば、「**WebSphere Process Server v6.1**」)、「**次へ**」をクリックします。
   - 既存のプロジェクトを選択するには、以下の手順を実行します。
     a. 「**IBM WebSphere Adapter for Flat Files**」を展開します。
     b. プロジェクトを選択します。

        例えば、CWYFF_FlatFiles という名前の既存プロジェクトがある場合、以下の図に示すように、「**IBM WebSphere Adapter for Flat Files**」を展開し、「**CWYFF_FlatFile**」を選択できます。

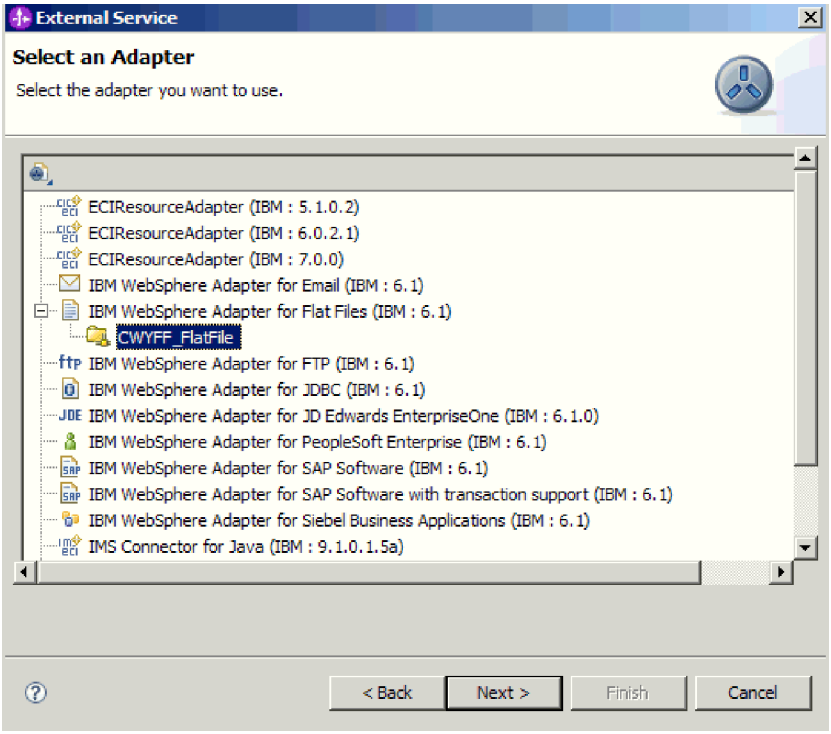


図 20. 「エンタープライズ・サービス・リソース・アダプターの選択」ウィンドウ

c. 「**次へ**」をクリックします。

結果

新規プロジェクトが作成され、「ビジネス・インテグレーション」ウィンドウにリストされます。

次のタスク

Outbound 処理のモジュールの構成

アダプターを Outbound 処理に使用するようにモジュールを構成するには、WebSphere Integration Developer 内で外部サービス・ウィザードを使用して、ビジネス・サービスを作成し、データ変換処理を指定して、ビジネス・オブジェクト定義および関連する成果物を生成します。

デプロイメントおよびランタイム・プロパティーの設定

WebSphere Integration Developer の外部サービス・ウィザードを使用して、モジュールを ローカル・ファイル・システム との Outbound 通信と Inbound 通信のいずれに使用するかを選択します。次に、管理接続ファクトリー・プロパティーを構成します。管理接続ファクトリー・プロパティーはビジネス・オブジェクトに格納され、モジュールとローカル・ファイル・システムを接続するためにアダプターが必要とする情報を保持します。

始める前に

このセクションでプロパティーを設定するには、その前にアダプター・モジュールを作成しておく必要があります。これは、WebSphere Integration Developer ではアダプター・プロジェクトの下に表示されます。アダプター・プロジェクトの作成について詳しくは、本書の該当するトピックを参照してください。

このタスクを実行する理由および時期

接続プロパティーを設定するには、以下の手順に従います。このトピックに記載されているプロパティーについて詳しくは、本書の管理接続ファクトリー・プロパティーに関する参照トピックを参照してください。

このタスクの手順

1. 「処理指示 (Processing Direction)」ウィンドウで「**Outbound**」を選択し、「**次へ**」をクリックします。

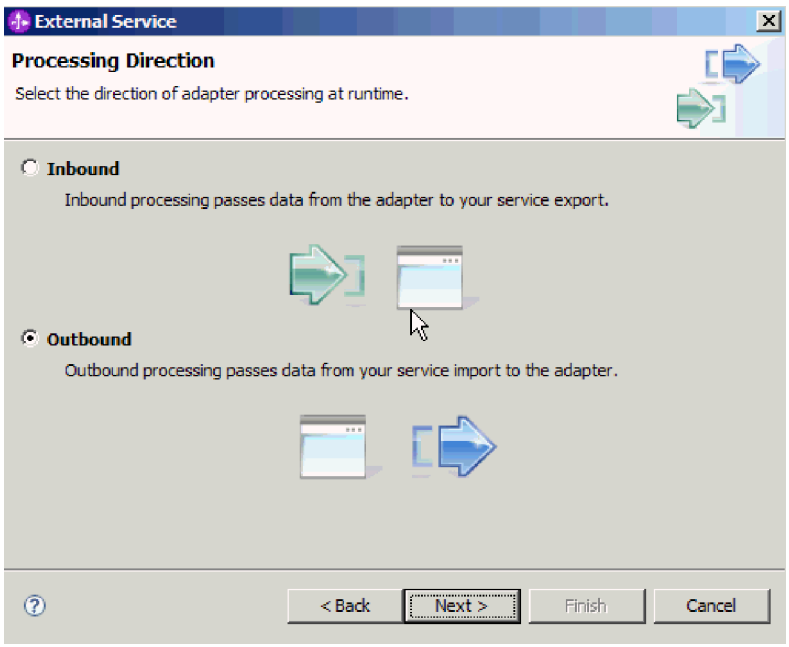


図 21. 外部サービス・ウィザードでの Inbound または Outbound 処理の選択

2. 「サービス構成プロパティー」ウィンドウの「コネクター・プロジェクトのデプロイ」フィールドで、「**単一アプリケーションが使用するモジュールで (With module for use by single application)**」を選択します。
3. モジュールの接続プロパティーを定義します。このウィンドウに表示されるプロパティーについて詳しくは、管理接続ファクトリー・プロパティーに関する参照トピックを参照してください。

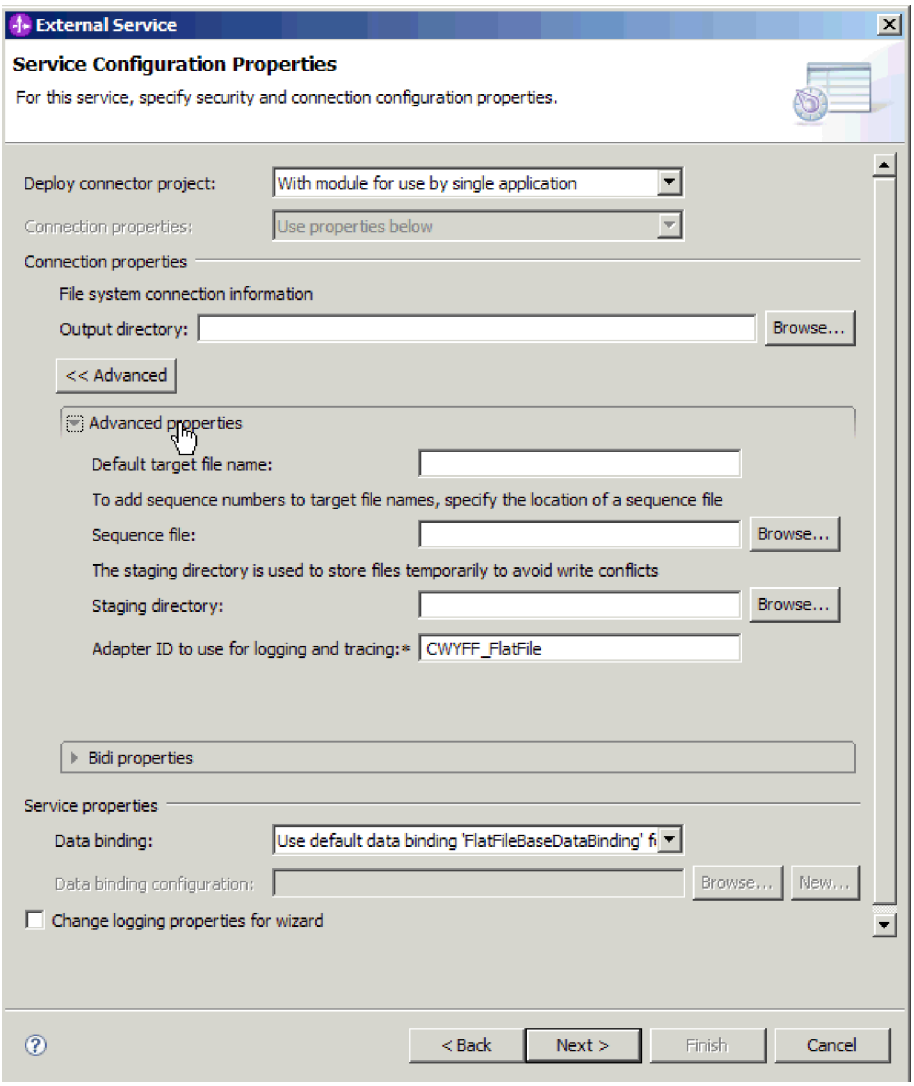


図 22. 接続プロパティーの設定

4. オプション: 「**ロギングおよびトレースで使用するアダプター ID**」を変更するには、新しい値を入力します。 このプロパティーについて詳しくは、『リソース・アダプター・プロパティー』の参照トピックを参照してください。
5. オプション: ログ・ファイルの出力先を指定したり、このモジュールのロギング・レベルを定義したりする場合は、「**ウィザードのロギング・プロパティーを変更します。**」チェック・ボックスを選択します。ロギング・レベルについて詳しくは、『トラブルシューティングおよびサポート』トピックの『ロギング・プロパティーの構成』の項を参照してください。
6. 「**次へ**」をクリックします。

結果

アダプターにより接続プロパティーが保存されます。

次のタスク

モジュールのデータ・タイプと、選択したデータ・タイプに関連付ける操作の名前を指定します。

操作およびデータ・タイプの選択

外部サービス・ウィザードを使用して、ローカル・ファイル・システムの機能にアクセスするために使用される Outbound 操作と、その操作で使用されるデータ・タイプを選択します。サポートされる操作は、
Create、Append、Overwrite、Delete、Exists、List、および Retrieve です。外部サービス・ウィザードでは、3 種類 (汎用 FlatFile ビジネス・オブジェクト、ビジネス・グラフ付きの汎用 FlatFile ビジネス・オブジェクト、およびユーザー定義タイプ) の中からデータ・タイプを選択できます。各データ・タイプは、ビジネス・オブジェクト構造に対応しています。

始める前に

以下の手順を実行する前に、ローカル・ファイル・システム との接続のために、アダプターの接続プロパティーを指定しておく必要があります。

このタスクを実行する理由および時期

Outbound 操作およびその操作で使用されるデータ・タイプを選択するには、以下の手順を実行します。

このタスクの手順

1. 「操作」ウィンドウで、「**追加**」をクリックします。

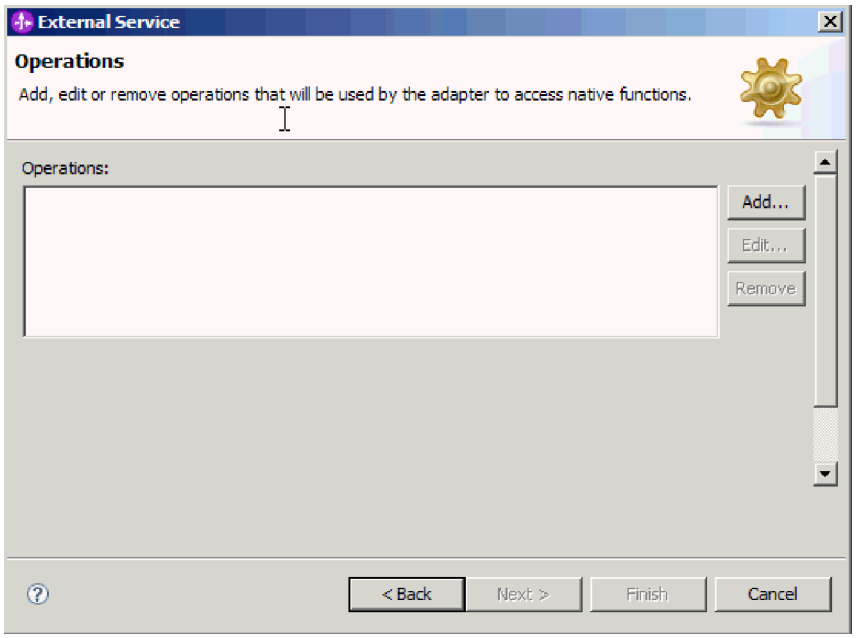


図 23. 操作の追加

2. 「操作の追加 (Add Operation)」ウィンドウで、「操作の種類」の横にあるドロップダウン・リストを開いて操作を選択します。この例では、「Create」を選択します。

3. 「操作の追加 (Add Operations)」ウィンドウで、データ・タイプを選択し、「**次へ**」をクリックします。この例では、ユーザー定義タイプを選択します。

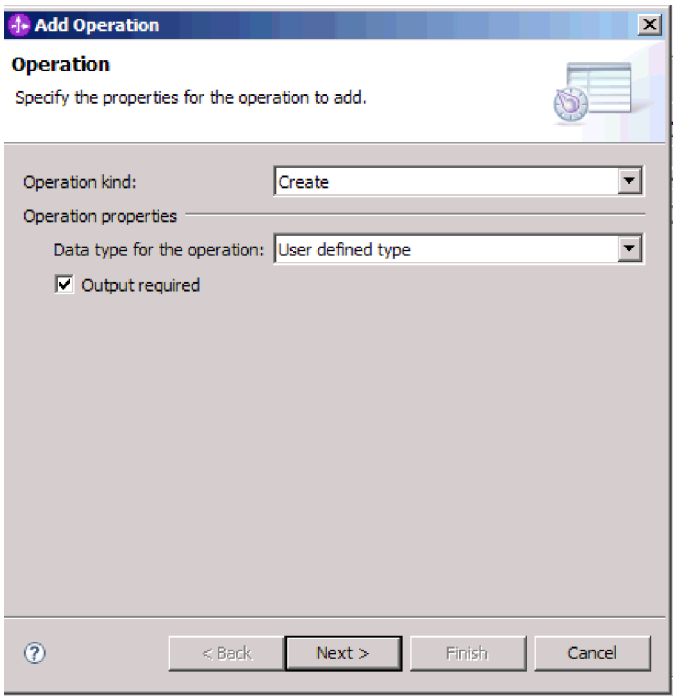


図 24. 操作のデータ・タイプの選択

Delete、Retrieve、Exists、および List 操作の場合には、入力として汎用データ・タイプ (汎用 FlatFile ビジネス・オブジェクト、またはビジネス・グラフ付きの汎用 FlatFile ビジネス・オブジェクト) のみがサポートされます。これらの操作のいずれかでユーザー定義タイプを選択する場合は、そのタイプをサポートするユーザー定義データ・バインディングを指定する必要があります。

Create、Append、および Overwrite 操作の場合は、ユーザー定義タイプ、汎用 FlatFile ビジネス・オブジェクト、およびビジネス・グラフ付きの汎用 FlatFile ビジネス・オブジェクトから選択します。データ・タイプについて詳しくは、ビジネス・オブジェクト構造について説明している本書のトピックを参照してください。

4. オプション: Create、Append および Overwrite 操作では、「**出力が必要**」チェック・ボックスを選択すれば、ファイル名を戻すことができます。固有ファイル名を生成する場合、またはファイルの順序付けを有効にしている場合は、これを選択してください。詳しくは、GenerateUniqueFile および FileSequenceLog の対話仕様プロパティーを参照してください。 Exists、List および Retrieve 操作の場合は、出力が必要です。「**出力が必要**」チェック・ボックスにチェック・マークを付けて有効にします。 Delete 操作の場合は、出力は必要ありません。「**出力が必要**」チェック・ボックスからチェック・マークを外して無効にします。

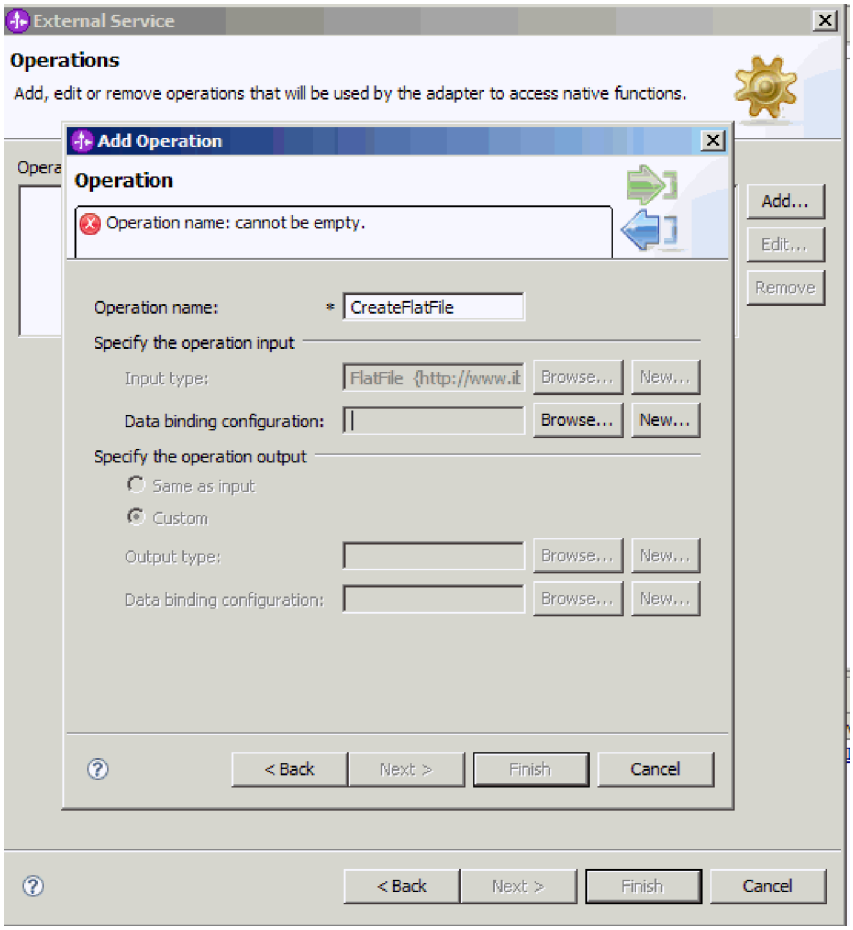


図 25. 操作の命名と入力データ・タイプの指定

5. 「**次へ**」をクリックします。
6. 「操作の追加 (Add Operation)」画面で、「**操作名 (Operation name)**」を入力します。 操作には分かりやすい名前を付けてください。アダプターが実行可能な操作のタイプについて詳しくは、本書のサポートされる操作に関するトピックを参照してください。

   注: 名前にスペースを含めることはできません。

   デフォルトでは、出力のデータ・タイプが CreateResponse または CreateResponseBG に設定されています。
7. 入力タイプを選択します。「**参照**」をクリックし、以前に作成したビジネス・オブジェクトを選択します。汎用データ・タイプ (汎用 FlatFile ビジネス・オブジェクト、またはビジネス・グラフ付きの汎用 FlatFile ビジネス・オブジェクト) を指定した場合、入力タイプはデフォルトで FlatFile または FlatFileBG に設定されます。

結果

モジュールのデータ・タイプが定義され、そのデータ・タイプに関連した操作に名前が付けられます。

次のタスク

モジュールで使用するデータ・バインディングを追加して構成します。

データ・バインディングの構成

各データ・タイプには、ビジネス・オブジェクトのフィールドを読み取ったり、対応するフィールドを設定したりするために使用するデータ・バインディングが対応しています。外部サービス・ウィザードで、モジュールにデータ・バインディングを追加し、追加したデータ・バインディングを、使用するデータ・タイプに合うように構成します。このようにして、アダプターはファイル内のフィールドに、ビジネス・オブジェクト内で受け取った情報を取り込む方法を識別します。

始める前に

操作とその操作で使用されるデータ・タイプを選択しておく必要があります。

このタスクを実行する理由および時期

モジュール用のデータ・バインディングを追加し、構成するには、以下の手順を実行します。

注: データ・バインディングは、WebSphere Integration Developer を使用して外部サービス・ウィザードを実行する前に構成できます。構成を行うには、WebSphere Integration Developer で「**新規**」 → 「**リソース構成 (Resource configuration)**」を選択し、本書で説明されているデータ・バインディング画面を完了してください。

このタスクの手順

1. 「操作の追加 (Add Operation)」ウィンドウの、「Input 操作のデータ・バインディング構成 (operation input Data binding configuration)」フィールドで「**新規**」を選択します。この操作は、初めてデータ・バインディングを設定するときに実行します。あとで同じデータ・バインディング構成を使用するには、「**参照**」をクリックし、その構成を選択します。
2. データ・バインディングの「**名前**」 (この例では DBConfg を使用) を入力し、「**次へ**」をクリックします。

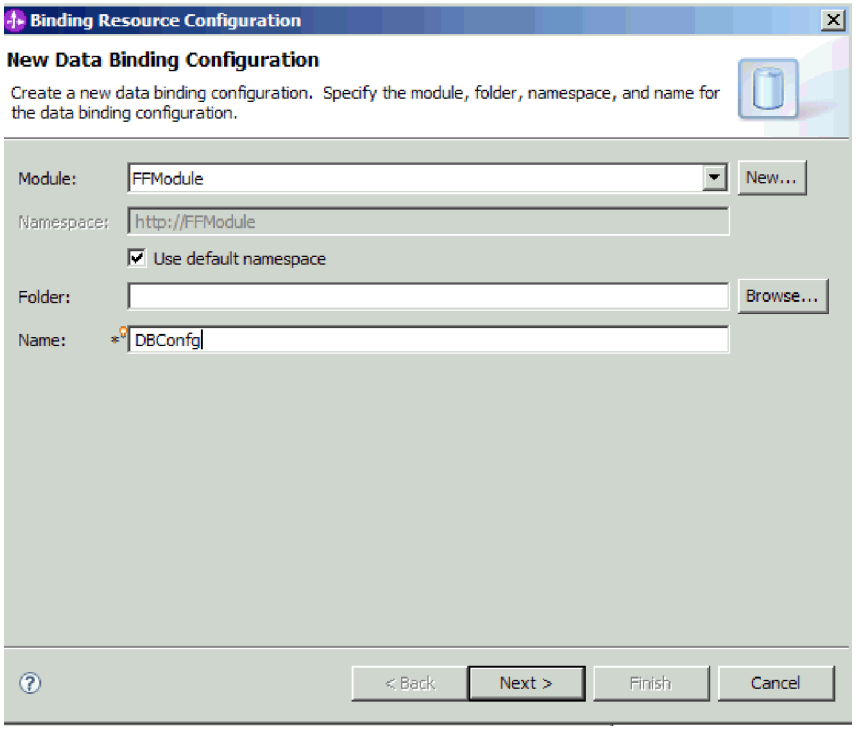


図 26. データ・バインディングの名前の指定

3. 「**次へ**」をクリックします。

結果

データ・バインディングがモジュールで使用できるように構成されます。

次のタスク

データ・ハンドラー構成を選択します。

データ・ハンドラーの構成

データ・ハンドラーは、ビジネス・オブジェクトとネイティブ形式の間の変換を実行します。

始める前に

モジュールのデータ・ハンドラーを指定する前に、データ・バインディングを作成しておく必要があります。また、WebSphere Integration Developer Business Object Editor を使用して、ビジネス・オブジェクトを事前に定義しておく必要があります。ここでウィザードを停止してビジネス・オブジェクトを作成する場合は、ウィザードのステップを最初から開始する必要があります。

注: データ・ハンドラーは、WebSphere Integration Developer を使用して外部サービス・ウィザードを実行する前に構成できます。構成を行うには、WebSphere Integration Developer で「**新規**」 → 「**リソース構成 (Resource configuration)**」を選択し、本書で説明されているデータ・ハンドラー画面を完了してください。

このタスクを実行する理由および時期

ビジネス・オブジェクトのデータ・ハンドラーを指定するには、以下の手順を実行します。

このタスクの手順

1. 「操作の追加 (Add Operation)」ウィンドウで、 「**新規**」をクリックし、データ・ハンドラー構成の名前 (この例では DataBindingConfiguration を使用) を入力します。この操作は、初めてデータ・ハンドラーを設定するときに実行します。あとで同じデータ・ハンドラーを使用するには、「**参照**」をクリックし、そのハンドラーを選択します。

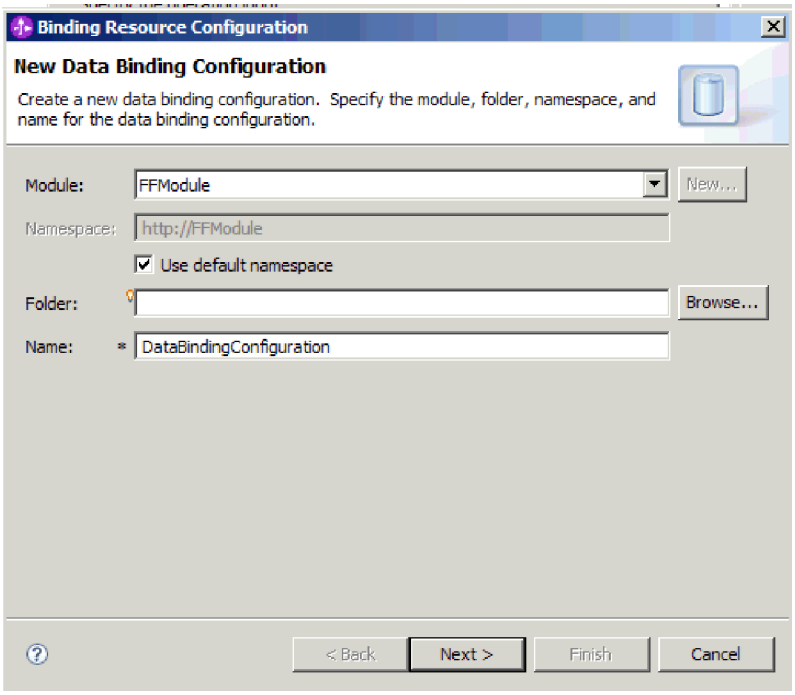


図 27. データ・ハンドラー構成の名前の指定

2. 「**次へ**」をクリックします。
3. 「データ・バインディング・プロパティー」ウィンドウで、バインディング・タイプ・プロパティーの横にあるドロップダウン・リストをクリックします。2 つの選択項目「DataBinding」および「DataHandler」が表示されます。旧バージョンのアダプター用に作成されたデータ・バインディングを使用する場合は、「DataBinding」を選択します。新しいデータ・ハンドラーを構成する場合は、「DataHandler」を選択します。「新規」をクリックして新しいデータ・ハンドラー構成を作成します。
4. 「**新規**」をクリックします。
5. 「新規データ・ハンドラー構成 (New Data Handler Configuration)」ウィンドウで、データ・ハンドラー構成のモジュール、ネーム・スペース、フォルダーおよび名前を指定します。

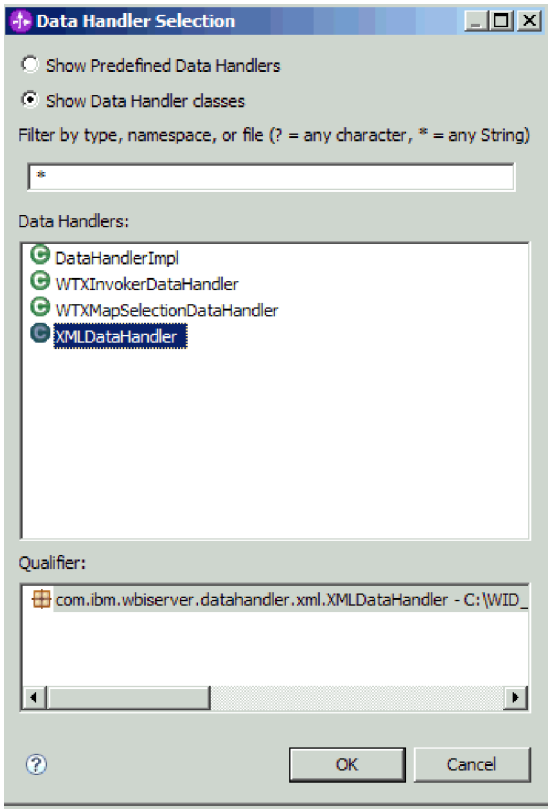


図 28. 新規データ・ハンドラー構成の作成

6. 「**次へ**」をクリックします。
7. データ・ハンドラーのクラス名を選択します。「構成タイプの選択 (Select a Configuration Type)」ウィンドウで、「**参照**」をクリックして、データ・ハンドラーのクラス名を選択します。「**データ・ハンドラー・クラスの表示 (Show data handler classes)**」ラジオ・ボタンを選択します。選択可能なデータ・ハンドラー・クラスのリストが表示されます。データ・ハンドラー・クラスを選択します (この例では XMLDataHandler を使用)。「**OK**」をクリックします。
8. 「**次へ**」をクリックします。
9. 「プロパティーの指定 (Specify properties)」ウィンドウで、エンコード方式を指定します。デフォルトは UTF-8 です。

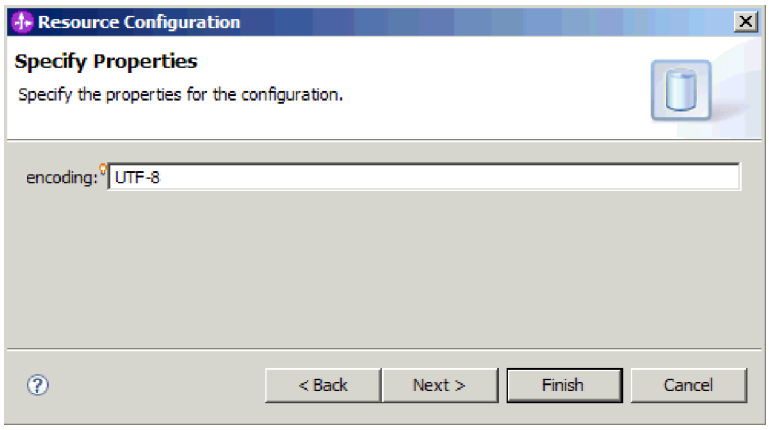


図 29. データ・ハンドラー構成のエンコード方式の指定

10. 「**終了**」をクリックします。
11. Output 操作でのデータ・バインディング構成を選択します。「操作の追加 (Add operation」ウィンドウの、「Output 操作のデータ・バインディング構成 (output Data binding configuration)」フィールドで、「**参照**」をクリックします。アダプターでは 1 つのデータ・バインディングのみを使用でき、そのデータ・バインディングは Input 操作の DataBinding タイプの設定時に構成済みであるため、 Output 操作の DataBinding タイプでも同じデータ・バインディング・タイプ (DBConfg) を選択します。

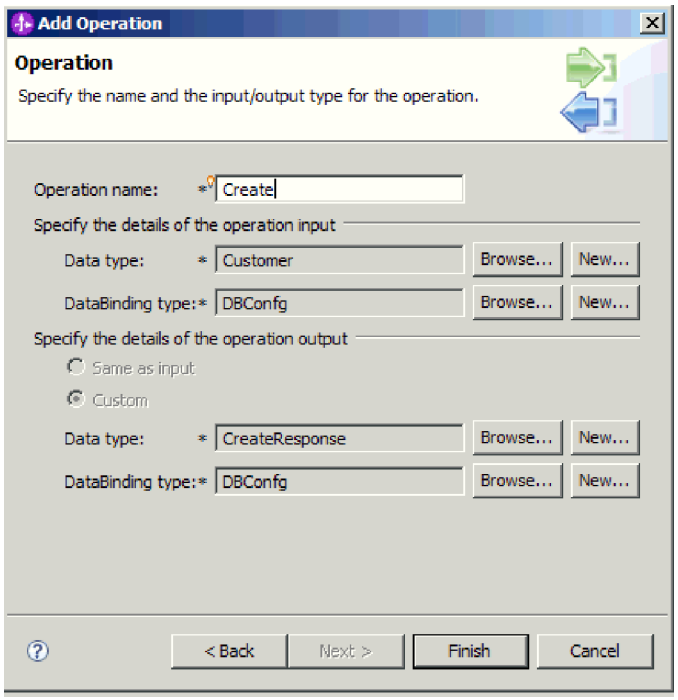


図 30. Output 操作でのデータ・バインディング構成の選択

12. 「**終了**」をクリックします。次の画面に、追加された Create 操作が対話仕様プロパティーとともに表示されます。

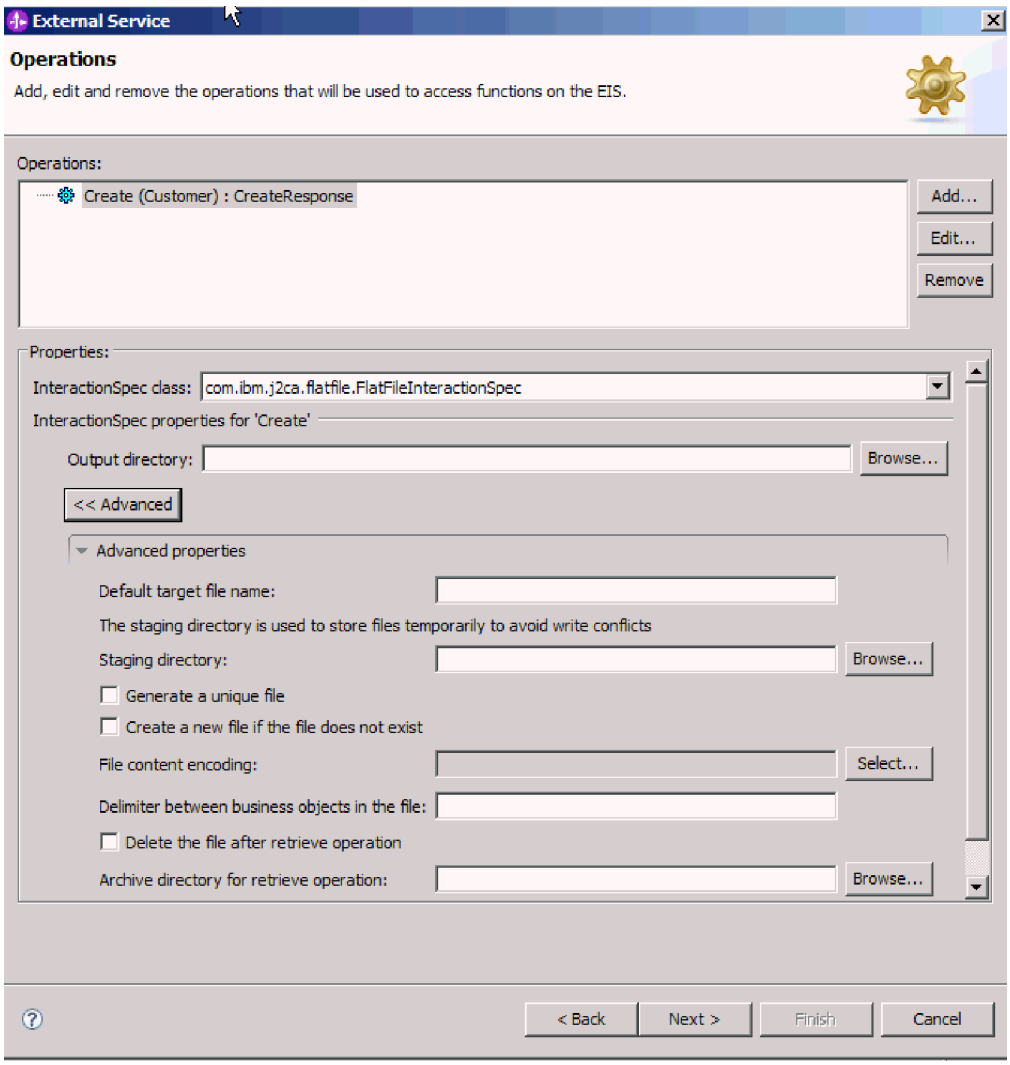


図 31. Create 操作と InteractionSpec プロパティー

13. 「**終了**」をクリックします。

結果

データ・ハンドラーが作成されます。

次のタスク

モジュールの対話仕様プロパティーを指定し、成果物を生成します。

対話プロパティーの設定およびサービスの生成

対話プロパティーはオプションです。設定すると、指定した値が、外部サービス・ウィザードによって生成されるすべての親ビジネス・オブジェクトのデフォルトとして表示されます。アダプターは、モジュールの成果物を作成するときにインポート・ファイルを生成します。インポート・ファイルには、トップレベル・ビジネス・オブジェクトの操作が含まれます。

始める前に

対話仕様プロパティーを設定してモジュールの成果物を生成するには、事前にデータ・バインディングを構成し、ビジネス・オブジェクトを選択しておく必要があります。

このタスクを実行する理由および時期

対話仕様プロパティーを設定してモジュールの成果物を生成するには、以下の手順を実行します。対話仕様プロパティーについて詳しくは、本書の該当する参照トピックを参照してください。

このタスクの手順

1. オプション: 対話仕様プロパティーを設定するには、以下のステップを実行します。
   a. 「操作」ウィンドウで、「**拡張**」をクリックします。
   b. デフォルトとして設定するすべてのフィールドに値を入力します。
   c. 「**次へ**」をクリックします。

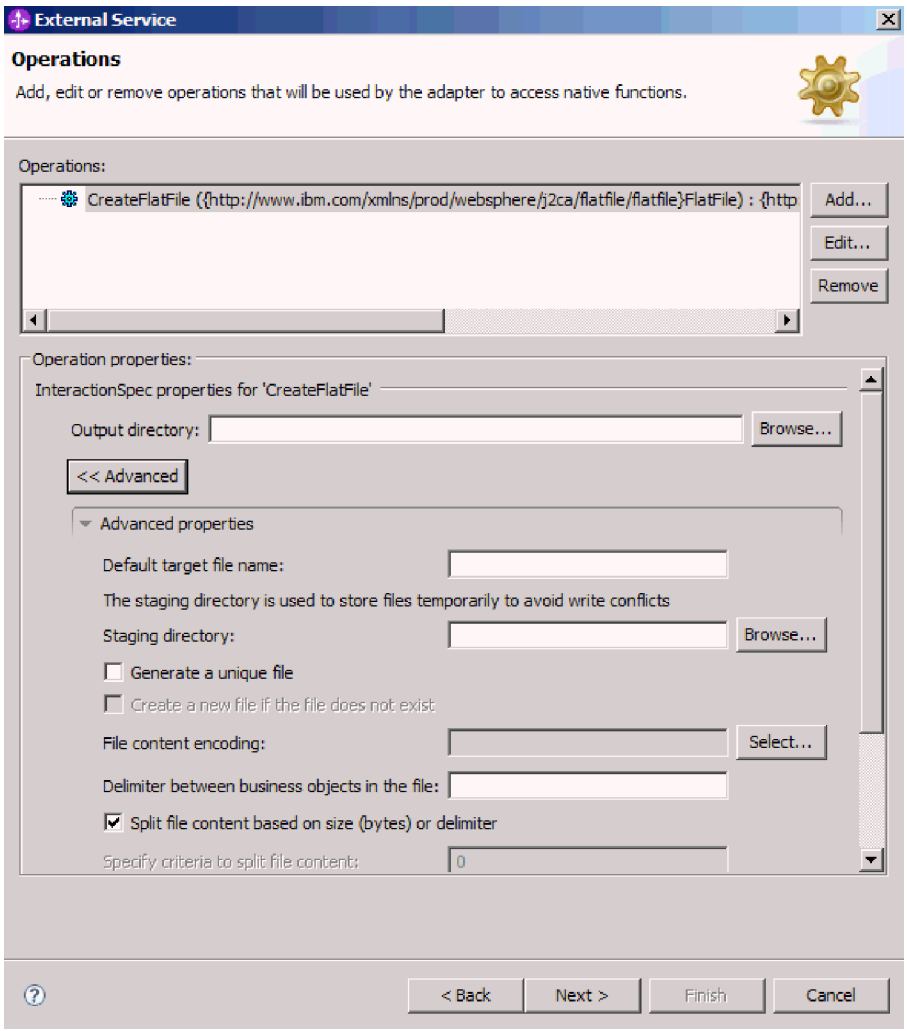


図 32. 対話仕様プロパティーの設定

2. 「操作」ウィンドウで、「**次へ**」をクリックします。「サービスの生成 (Generate Service)」画面で、インターフェースの名前を指定します。この名前は、WebSphere Integration Developer アセンブリー・ダイアグラムに表示されます。

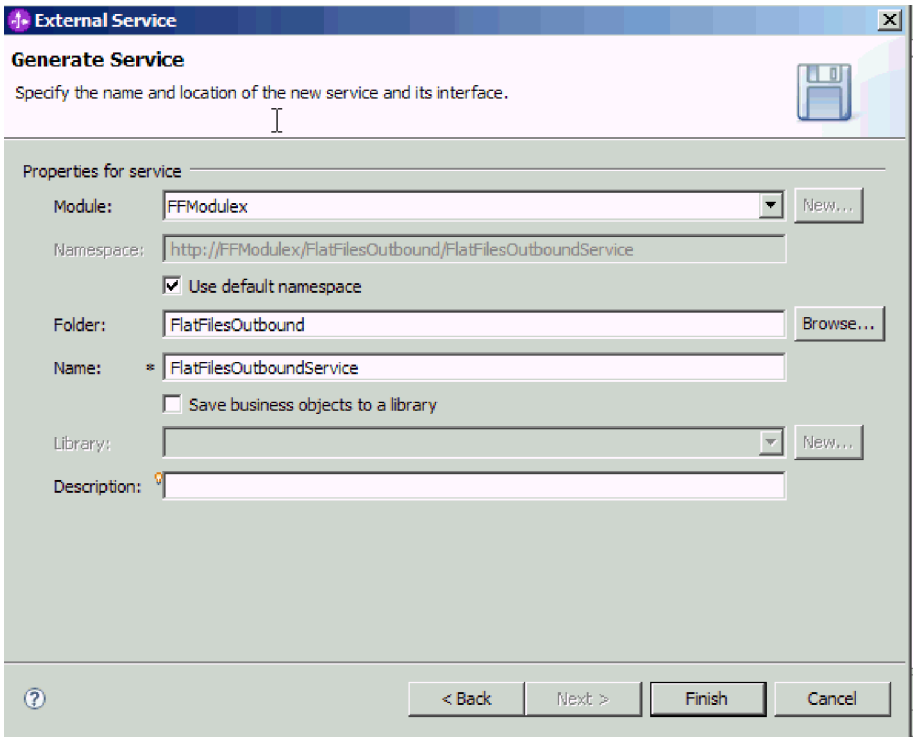


図 33. サービスの名前指定

3. 「**終了**」をクリックします。

結果

WebSphere Integration Developer がサービスおよびインポートを生成します。作成された Outbound 操作の成果物は、WebSphere Integration Developer Project Explorer 内のモジュールの下に表示されます。

次のタスク

モジュールをデプロイします。

Inbound 処理のモジュールの構成

アダプターを Inbound 処理に使用するようにモジュールを構成するには、WebSphere Integration Developer 内で外部サービス・ウィザードを使用して、ビジネス・サービスを作成し、データ変換処理を指定して、ビジネス・オブジェクト定義および関連する成果物を生成します。

デプロイメントおよびランタイム・プロパティーの設定

WebSphere Integration Developer の外部サービス・ウィザードを使用して、モジュールを ローカル・ファイル・システム との Outbound 通信と Inbound 通信のいずれ

に使用するかを選択します。次に、アクティベーション・スペック・プロパティーを構成します。アクティベーション・スペック・プロパティーは、エクスポート用の Inbound イベント処理の構成情報を保持しています。

始める前に

このセクションでプロパティーを設定するには、その前にアダプター・モジュールを作成しておく必要があります。これは、WebSphere Integration Developer ではアダプター・プロジェクトの下に表示されます。アダプター・プロジェクトの作成について詳しくは、本書の該当するトピックを参照してください。

このタスクを実行する理由および時期

アクティベーション・スペック・プロパティーを設定するには、以下の手順に従います。このトピックに記載されているプロパティーについて詳しくは、本書のアクティベーション・スペック・プロパティーに関する参照トピックを参照してください。

このタスクの手順

1. 「処理指示 (Processing Direction)」ウィンドウで「**Inbound**」を選択し、「**次へ**」をクリックします。

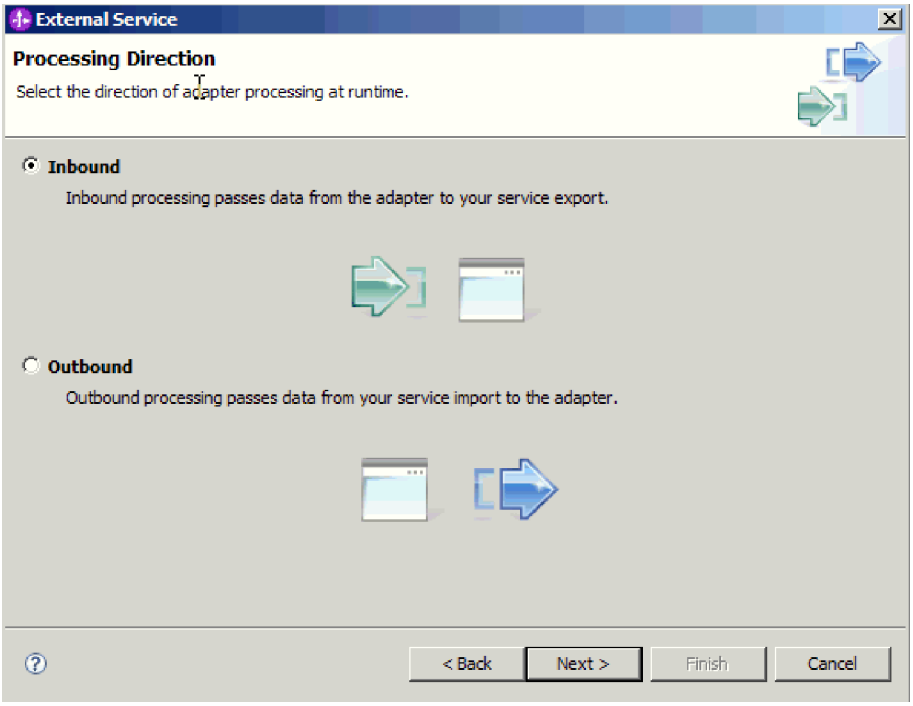


図 34. 外部サービス・ウィザードでの Inbound または Outbound の選択

2. 「サービス構成プロパティー」ウィンドウの「コネクター・プロジェクトのデプロイ」フィールドで、「**単一アプリケーションが使用するモジュールで (With module for use by single application)**」を選択します。
3. モジュールのアクティベーション・スペック・プロパティーを定義します。このウィンドウに表示されるプロパティーについて詳しくは、アクティベーション・

スペック・プロパティーに関する参照トピックをご覧ください。

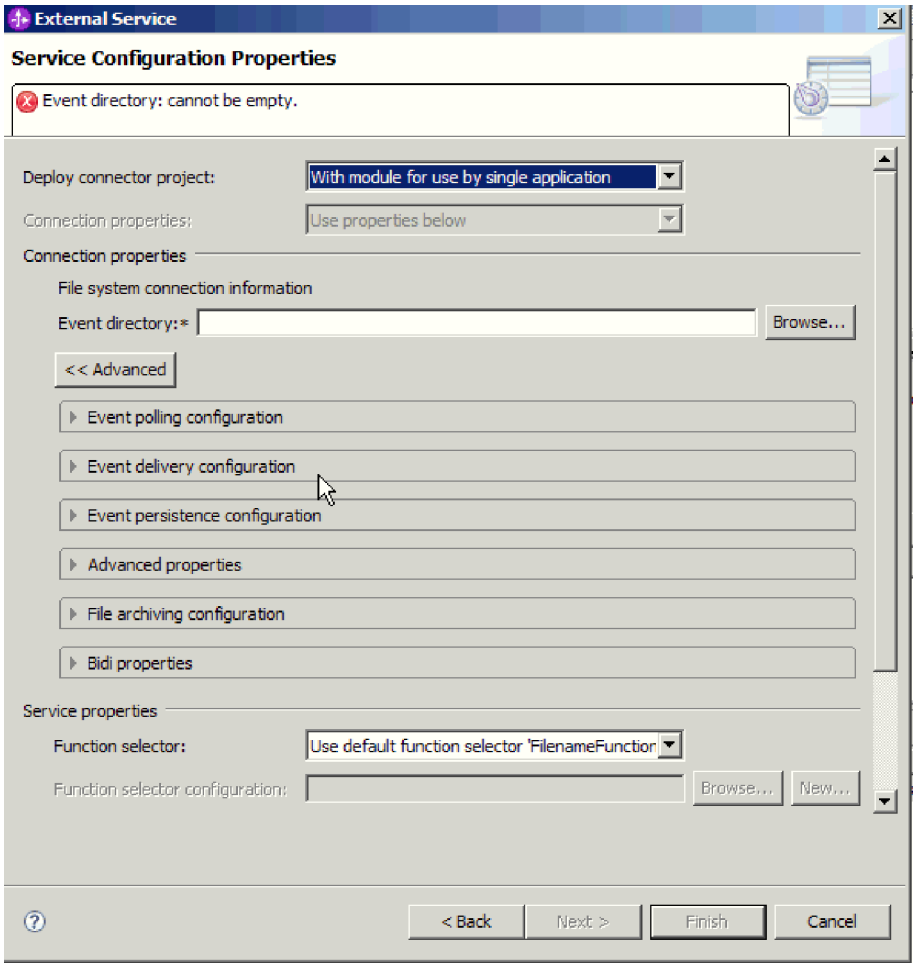


図 35. 接続プロパティーの設定

4. 「**イベント・ディレクトリー**」プロパティーでは、イベント・ファイルの保管先ローカル・ファイル・システムのディレクトリーを指定します。
5. オプション: 「**ロギングおよびトレースで使用するアダプター ID**」を変更するには、新しい値を入力します。 このプロパティーについて詳しくは、『リソース・アダプター・プロパティー』の参照トピックを参照してください。
6. オプション: ログ・ファイルの出力先を指定したり、このモジュールのロギング・レベルを定義したりする場合は、「**ウィザードのロギング・プロパティーを変更します。**」チェック・ボックスを選択します。ロギング・レベルについて詳しくは、『トラブルシューティングおよびサポート』トピックの『ロギング・プロパティーの構成』の項を参照してください。
7. 「**関数セレクター**」フィールドでは、既存の関数セレクター構成を使用するか、新しい構成を作成するかを選択します。関数セレクターによって、サービスの正しい操作に対して着信メッセージまたは要求が割り当てられます。
   a. 既存の関数セレクター構成を使用するには、「**参照**」をクリックして関数セレクターのリストを表示します。使用可能な関数セレクターの説明については、外部サービス・ウィザードの、接続プロパティーのトピックを参照してください。

b. 新しい関数セレクター構成を作成するには「**新規**」をクリックします。「**新規関数セレクター構成**」ウィンドウで、関数セレクター構成の**モジュール**、**フォルダー**、および**名前**を指定します。「**次へ**」をクリックします。

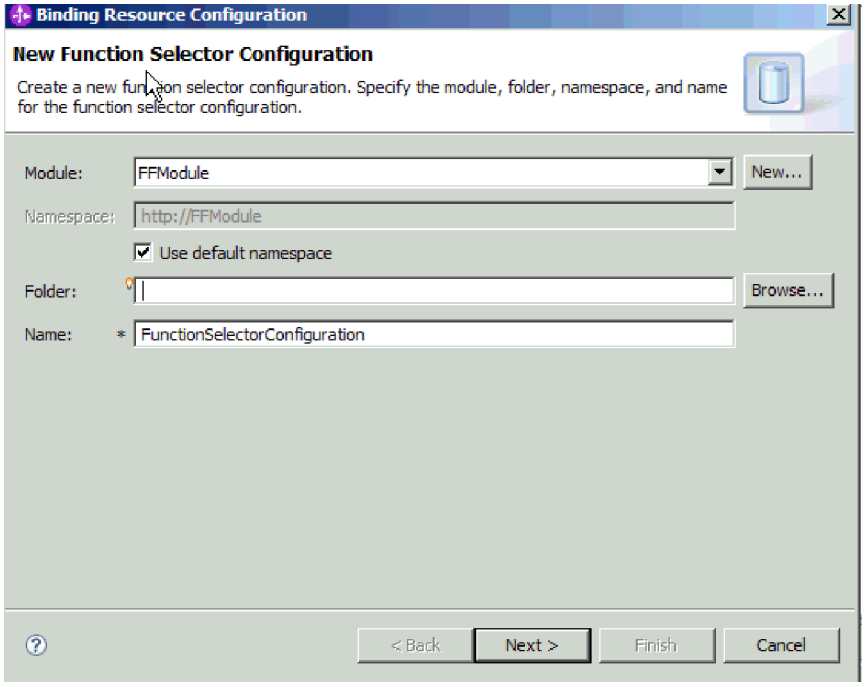


図 36. 新規関数セレクター構成の作成

注: EIS 関数名は 外部サービス・ウィザードでは使用できません。アダプター (基底クラス) で生成されるデフォルト以外の値を指定する場合、アセンブリー・エディターを使用してその値を編集することができます。

8. 「**終了**」をクリックします。

結果

アダプターによりアクティベーション・スペック・プロパティーが保存されます。

次のタスク

モジュールのデータ・タイプと、選択したデータ・タイプに関連付ける操作の名前を指定します。

操作およびデータ・タイプの選択

データ・タイプを選択し、データ・タイプに関連付けられる操作に名前を付けるには、外部サービス・ウィザードを使用します。外部サービス・ウィザードでは、3 種類 (汎用 FlatFile ビジネス・オブジェクト、ビジネス・グラフ付きの汎用 FlatFile ビジネス・オブジェクト、およびユーザー定義タイプ) の中からデータ・タイプを選択できます。各データ・タイプは、ビジネス・オブジェクト構造に対応しています。

始める前に

以下の手順を実行する前に、ローカル・ファイル・システム との接続のために、アダプターの接続プロパティーを指定しておく必要があります。

このタスクを実行する理由および時期

データ・タイプを選択し、それに関連付けられる操作に名前を付けるには、以下の手順を実行します。

このタスクの手順

1. 「操作」ウィンドウで、「**追加**」をクリックします。

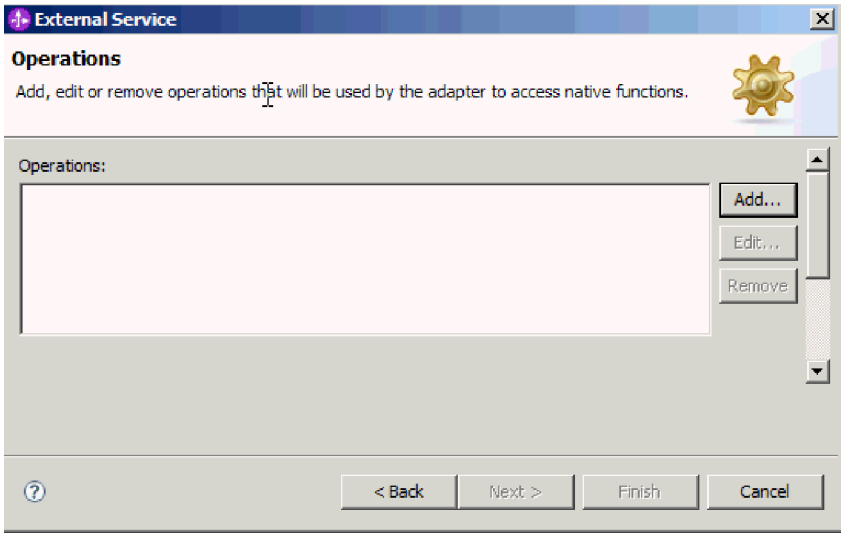


図 37. 操作の追加

2. 「操作の追加 (Add Operations)」ウィンドウで、データ・タイプを選択します。3 つのデータ・タイプ (汎用 FlatFile ビジネス・オブジェクト、ビジネス・グラフ付きの汎用 FlatFile ビジネス・オブジェクト、およびユーザー定義タイプ) から選択できます。各データ・タイプと、どのタイプのビジネス・オブジェクトの生成に各データ・タイプが使用されるかについて詳しくは、本書のビジネス・オブジェクトの構造に関するセクションを参照してください。この例では、「汎用 FlatFile ビジネス・オブジェクト」を選択します。
3. 「**次へ**」をクリックします。「操作」ウィンドウに、「**emitFlatFile**」という操作名が表示されます。 Inbound 処理で使用できる操作は、出力操作のみです。

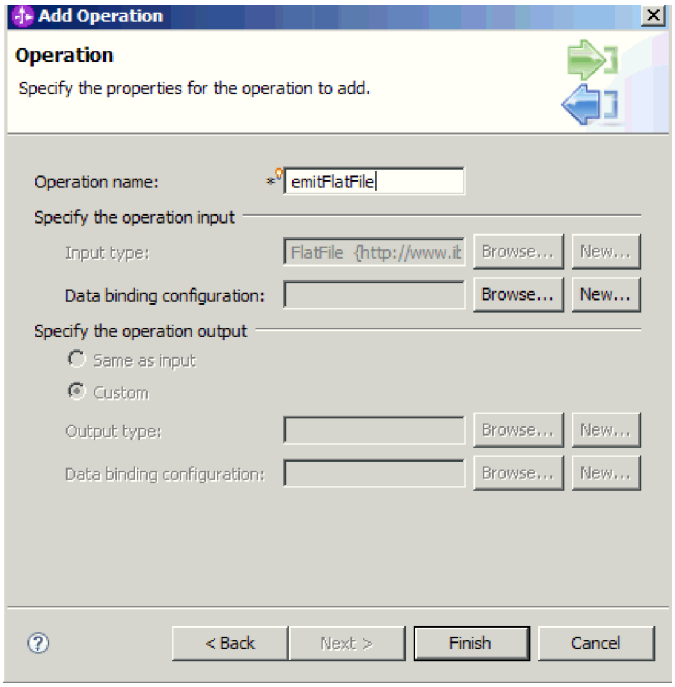


図 38. 操作の追加

結果

モジュールのデータ・タイプが定義され、そのデータ・タイプに関連した操作に名前が付けられます。

次のタスク

モジュールで使用するデータ・バインディングを追加して構成します。

データ・バインディングの構成

各データ・タイプには、ビジネス・オブジェクトのフィールドを読み取ったり、対応するフィールドを設定したりするために使用するデータ・バインディングが対応しています。外部サービス・ウィザードで、モジュールにデータ・バインディングを追加し、追加したデータ・バインディングを、使用するデータ・タイプに合うように構成します。このようにして、アダプターはファイル内のフィールドに、ビジネス・オブジェクト内で受け取った情報を取り込む方法を識別します。

始める前に

データ・タイプを選択し、そのデータ・タイプに関連付ける操作名を選択しておく必要があります。

このタスクを実行する理由および時期

モジュール用のデータ・バインディングを追加し、構成するには、以下の手順を実行します。

注: データ・バインディングは、WebSphere Integration Developer を使用して外部サービス・ウィザードを実行する前に構成できます。構成を行うには、WebSphere

Integration Developer で「**新規**」 → 「**リソース構成 (Resource configuration)**」を選択し、本書で説明されているデータ・バインディング画面を完了してください。

このタスクの手順

1. 「操作」ウィンドウの、「Input 操作のデータ・バインディング構成 (operation input Data binding configuration)」フィールドで「**新規**」を選択します。この操作は、初めてデータ・バインディングを設定するときに実行します。あとで同じデータ・バインディング構成を使用するには、「**参照**」をクリックし、その構成を選択します。
2. オプション: 「新規データ・バインディング構成」 画面には、「**モジュール**」にデフォルトでこのウィザードで既に入力したモジュール名が示されています。このモジュールがデータ・バインディングを作成するモジュールでない場合は、「**新規**」を選択して新しいモジュールを作成します。
3. オプション: 成果物用に新しいフォルダーを選択する場合は、「**参照**」をクリックして、新しいフォルダーの格納場所を選択します。 新しいフォルダーの場所を参照しなかった場合、成果物はモジュールのルート・ディレクトリーに作成されます。
4. データ・バインディング構成の「**名前**」 (この例では DataBindingConfiguration を使用) を入力します。「**次へ**」をクリックします。

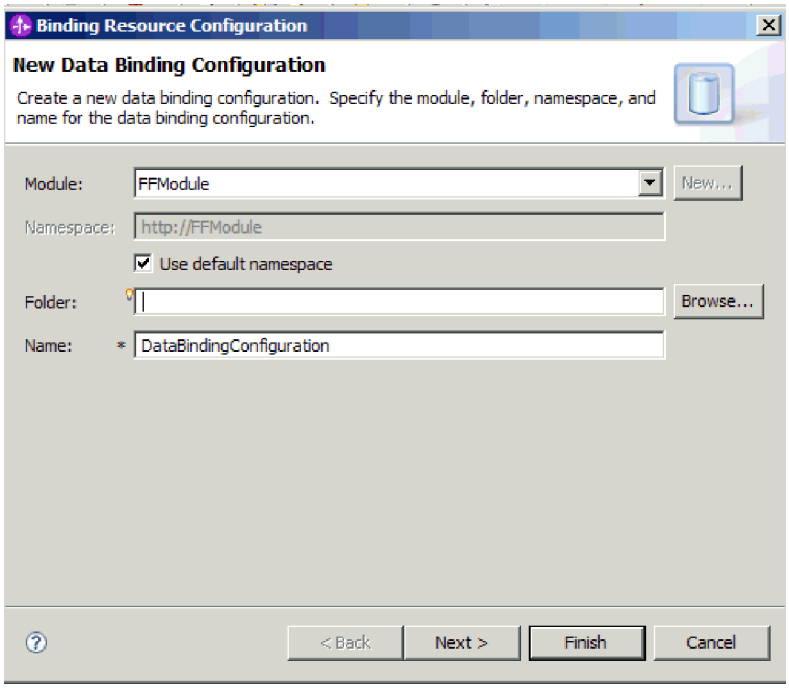


図 39. データ・バインディング構成の名前指定

5. 「**次へ**」をクリックします。

結果

データ・バインディングがモジュールで使用できるように構成されます。

次のタスク

データ・ハンドラー構成を選択します。

データ・ハンドラーの構成

データ・ハンドラーは、ビジネス・オブジェクトとネイティブ形式の間の変換を実行します。

始める前に

モジュールのデータ・ハンドラーを指定する前に、データ・バインディングを作成しておく必要があります。また、WebSphere Integration Developer Business Object Editor を使用して、ビジネス・オブジェクトを事前に定義しておく必要があります。ここでウィザードを停止してビジネス・オブジェクトを作成する場合は、ウィザードのステップを最初から開始する必要があります。

注: データ・ハンドラーは、WebSphere Integration Developer を使用して外部サービス・ウィザードを実行する前に構成できます。構成を行うには、WebSphere Integration Developer で「**新規**」 → 「**リソース構成 (Resource configuration)**」を選択し、本書で説明されているデータ・ハンドラー画面を完了してください。

このタスクを実行する理由および時期

ビジネス・オブジェクトのデータ・ハンドラーを指定するには、以下の手順を実行します。

このタスクの手順

1. 「データ・バインディング・プロパティー」ウィンドウで、「**新規**」をクリックし、データ・ハンドラー構成の名前 (この例では DHConfig を使用) を入力します。初めて「**新規**」をクリックするときには、データ・ハンドラーを設定します。あとでこのデータ・ハンドラーを使用するときには、「**参照**」をクリックします。

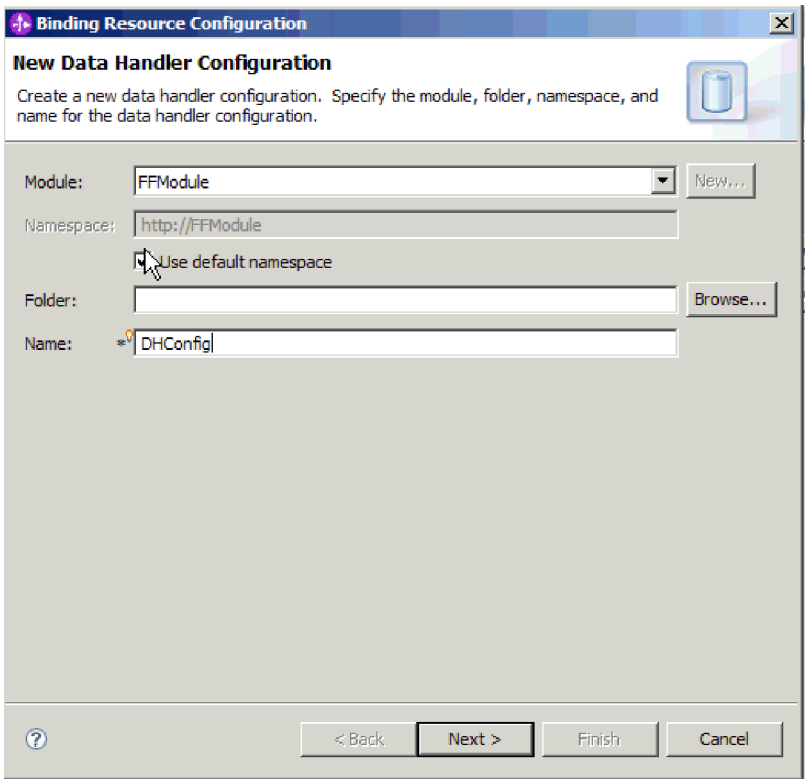


図 40. データ・ハンドラー構成の名前の指定

2. 「**次へ**」をクリックします。
3. データ・ハンドラーのクラス名を選択します。「構成タイプの選択 (Select a Configuration Type)」ウィンドウで、「**参照**」をクリックして、データ・ハンドラーのクラス名を選択します。「**データ・ハンドラー・クラスの表示 (Show data handler classes)**」ラジオ・ボタンを選択します。選択可能なデータ・ハンドラー・クラスのリストが表示されます。データ・ハンドラー・クラスを選択します (この例では XMLDataHandler を使用)。「**OK**」をクリックします。

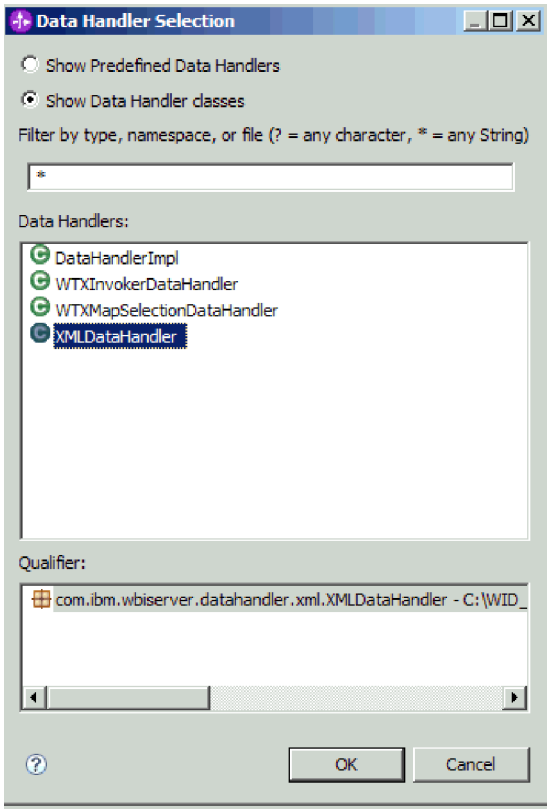


図 41. データ・ハンドラー・クラスの選択

4. 「**次へ**」をクリックします。
5. 「プロパティーの指定 (Specify properties)」ウィンドウで、エンコード方式を指定します (この例では UTF-8 を使用)。

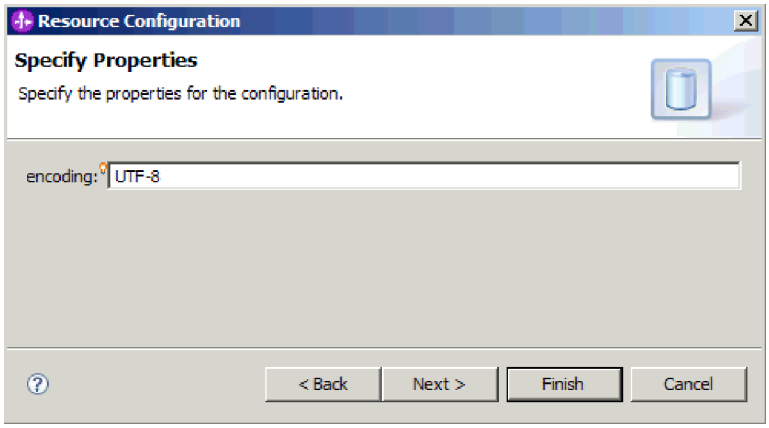


図 42. データ・ハンドラー構成のエンコード方式の指定

6. 「**終了**」をクリックします。次の画面に、追加された Inbound 操作が対話仕様プロパティーとともに表示されます。

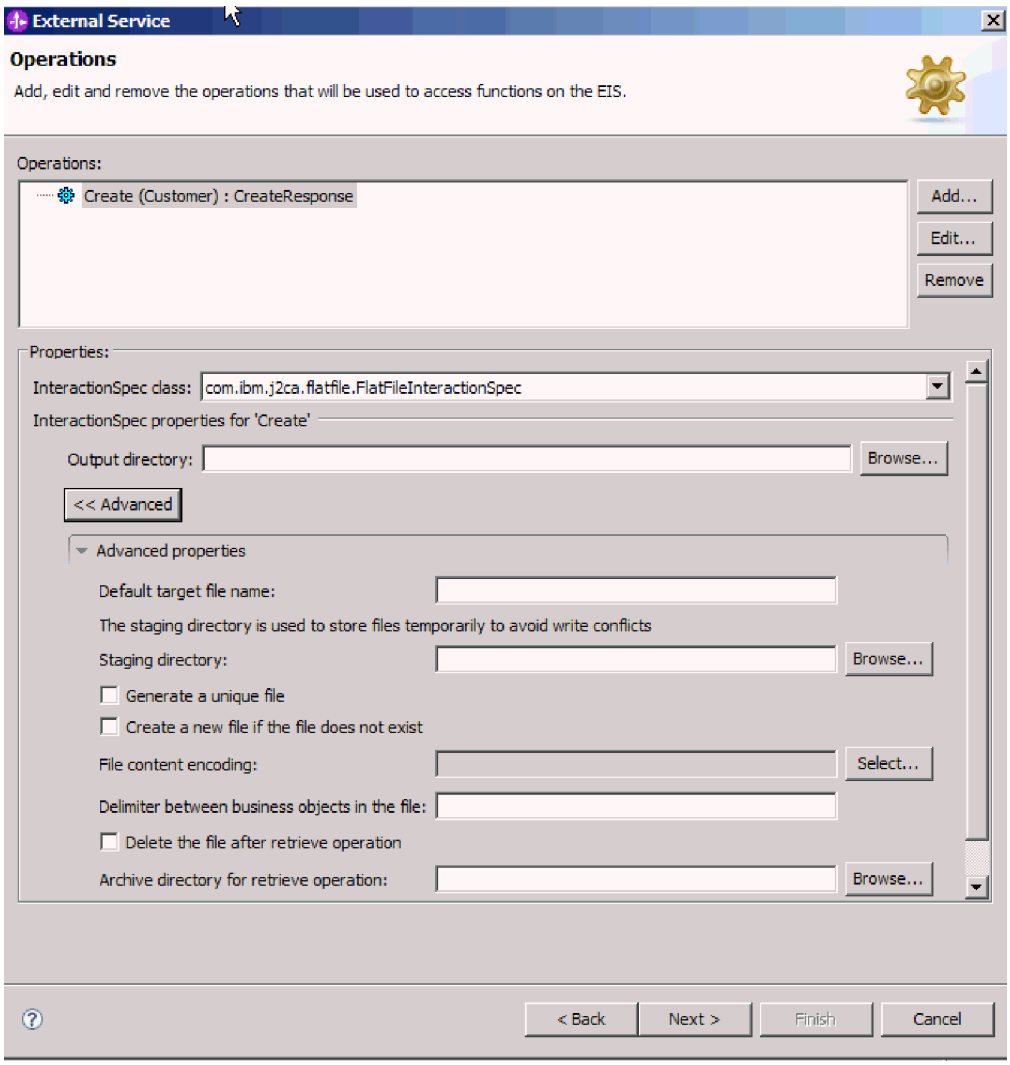


図 43. Inbound 操作と InteractionSpec プロパティー

7. 「**終了**」をクリックします。

結果

データ・ハンドラーが作成されます。

次のタスク

モジュールの対話仕様プロパティーを指定し、成果物を生成します。

デプロイメント・プロパティーの設定およびサービスの生成

外部サービス・ウィザードを使用して、アクティベーション・スペック・プロパティーを設定し、モジュールで使用する成果物を生成します。成果物は、外部サービスの一部として作成される、ビジネス・オブジェクト、WSDL ファイル、およびインポート・ファイルとエクスポート・ファイルです。アダプターは、モジュールの成果物を作成するときにエクスポート・ファイルを生成します。エクスポート・ファイルには、最上位レベルのビジネス・オブジェクトの操作が含まれます。

始める前に

アクティベーション・スペック・プロパティーを設定してモジュールの成果物を生成するには、事前にデータ・バインディングを構成し、ビジネス・オブジェクトを選択しておく必要があります。

このタスクを実行する理由および時期

アクティベーション・スペック・プロパティーを設定して成果物を生成するには、以下の手順を実行します。アクティベーション・スペック・プロパティーについて詳しくは、本書の該当する参照トピックを参照してください。

このタスクの手順

1. アクティベーション・スペック・プロパティーを設定して成果物を生成するには、以下のステップを完了します。
   a. 「サービス構成プロパティー」ウィンドウで、「**拡張**」をクリックします。
   b. デフォルトとして設定するすべてのフィールドに値を入力します。
   c. 「**次へ**」をクリックします。
2. 「操作」ウィンドウで、「**次へ**」をクリックします。「サービスの生成 (Generate Service)」画面で、インターフェースの名前を指定します。この名前は、WebSphere Integration Developer アセンブリー・ダイアグラムに表示されます。

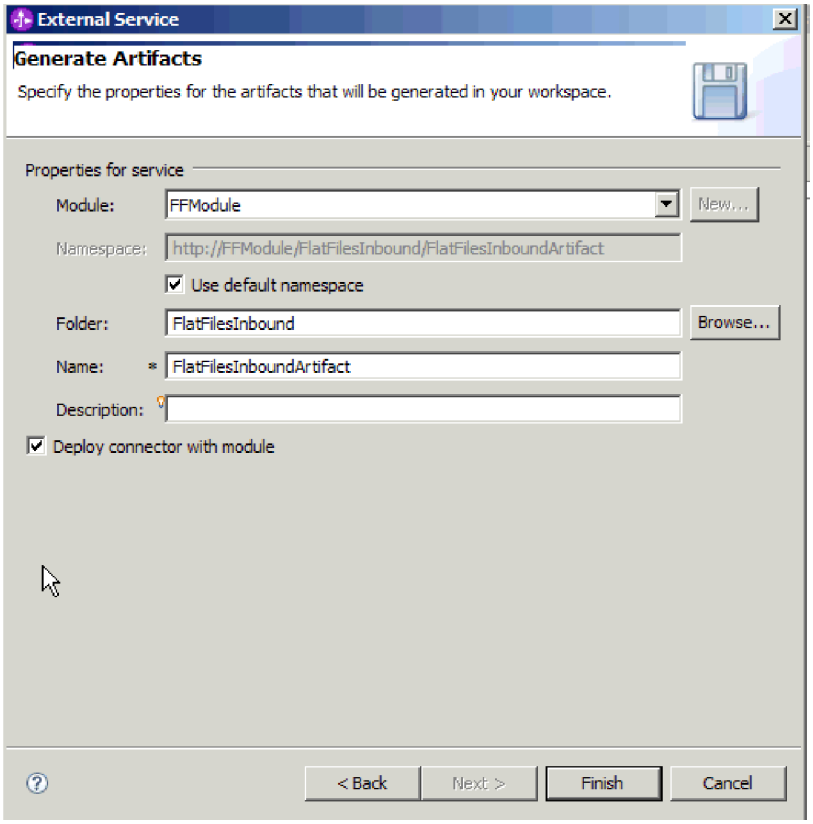


図 44. 成果物への命名

3. 「**終了**」をクリックします。

結果

WebSphere Integration Developer が成果物およびインポートを生成します。作成された Inbound 操作の成果物は、WebSphere Integration Developer Project Explorer 内のモジュールの下に表示されます。

次のタスク

モジュールをデプロイします。

第 5 章 アセンブリー・エディターの使用による対話仕様プロパティーの変更

サービスを生成した後、アダプター・モジュールの対話仕様プロパティーを変更するには、WebSphere Integration Developer のアセンブリー・エディターを使用します。

始める前に

外部サービス・ウィザードを使用して、アダプターのサービスが生成済みである必要があります。

このタスクを実行する理由および時期

アダプターのサービスを生成した後、対話仕様プロパティーを変更したい場合があります。対話仕様プロパティー (オプション) は、特定のビジネス・オブジェクトの特定の操作に対し、メソッド・レベルで設定されます。指定した値は、外部サービス・ウィザードによって生成されるすべての親ビジネス・オブジェクトのデフォルトとして表示されます。 EAR ファイルをエクスポートする前に、これらのプロパティーを変更することができます。アプリケーションのデプロイ後に、これらのプロパティーを変更することはできません。

対話仕様プロパティーを変更するには、次の手順を使用してください。

このタスクの手順

1. WebSphere Integration Developerの Business Integration パースペクティブから、モジュール名を展開します。
2. 「**アセンブリー・ダイアグラム (Assembly Diagram)**」を展開して、インターフェースをダブルクリックします。
3. アセンブリー・エディターのインターフェースをクリックします。 (それ以上クリックしなければ、モジュール・プロパティーが表示されます。)
4. 「**プロパティー**」タブをクリックします。 (図のインターフェースを右クリックして、「**プロパティーに表示 (Show in Properties)**」をクリックすることもできます。)
5. 「**バインディング**」の下で、**メソッド・バインディング**」をクリックします。インターフェースのメソッドが、ビジネス・オブジェクトと操作の組み合わせにごとに 1 つ表示されます。
6. 対話仕様プロパティーを変更するメソッドを選択します。
7. 「**拡張**」をクリックし、「**汎用 (Generic)**」タブでプロパティーを変更します。対話仕様プロパティーを変更する各メソッドについて、この手順を繰り返します。

結果

アダプター・モジュールに関連付けられた対話仕様プロパティーが変更されます。

次のタスク

モジュールをデプロイします。

第 6 章 モジュールのデプロイ

モジュールをデプロイし、モジュールとアダプターを構成しているファイルを実動またはテスト用の稼働環境に置きます。 WebSphere Integration Developer では、統合テスト環境は、インストール中に選択したテスト環境プロファイルによって、WebSphere Process Server、または WebSphere Enterprise Service Bus、あるいはこれら両方のランタイム・サポートが可能です。

デプロイメント環境

モジュールおよびアダプターをデプロイできるテスト環境および実稼働環境があります。

WebSphere Integration Developer では、テスト環境の 1 つ以上のサーバーにモジュールをデプロイできます。これは、通常ビジネス・インテグレーション・モジュールを実行およびテストするための最も一般的な方法です。しかし、管理コンソール・ツールまたはコマンド行ツールを使用して、サーバー・デプロイメント用のモジュールを EAR ファイルとして WebSphere Process Server または WebSphere Enterprise Service Bus にエクスポートすることもできます。

テストするモジュールのデプロイ

WebSphere Integration Developer では、組み込みアダプターを含むモジュールをテスト環境にデプロイし、サーバー構成の編集、サーバーの始動と停止、およびモジュール・コードでのエラーのテストなどのタスクを実行できるサーバー・ツールで作業を行うことができます。テストは、一般にコンポーネントのインターフェース操作で実行され、これにより、コンポーネントが正しく実装されているかどうか、および参照が正しく関連付けられているかどうかを判別できます。

Inbound 処理のテスト用ターゲット・コンポーネントの生成および配線

テスト環境に Inbound 処理用のアダプターを含むモジュールをデプロイする前に、まずターゲット・コンポーネントを生成し、配線する必要があります。このターゲット・コンポーネントは、アダプターによるイベントの送信の*宛先* として機能します。

始める前に

外部サービス・ウィザードを使用して、エクスポート・モジュールが生成済みでなければなりません。

このタスクを実行する理由および時期

Inbound 処理のテスト用ターゲット・コンポーネントの生成および配線は、テスト環境のみにおいて必要です。実稼働環境においてアダプターをデプロイする際には必要ありません。

ターゲット・コンポーネントはイベントを受け取ります。 WebSphere Integration Developer のアセンブリー・エディターを使用して、ターゲット・コンポーネントへのエクスポート (2 つのコンポーネントを接続) を配線します。アダプターは、配線を使用して、(ターゲット・コンポーネントへのエクスポートから) イベント・データを受け渡します。

このタスクの手順

1. ターゲット・コンポーネントの作成
   a. WebSphere Integration Developer の Business Integration パースペクティブから、「**アセンブリー・ダイアグラム (Assembly Diagram)**」を展開し、エクスポート・コンポーネントをダブルクリックします。 デフォルト値を変更しなかった場合、エクスポート・コンポーネントの名前は、ご使用のアダプターの名前 + **InboundInterface** です。

      インターフェースは、呼び出すことのできる操作および受け渡されるデータ (入力引数、戻り値、例外など) を指定します。 **InboundInterface** は、Inbound 処理をサポートするのにアダプターが必要とする操作を含み、外部サービス・ウィザードを実行する際に作成されます。
   b. 「**コンポーネント**」を展開し、「**型なしコンポーネント**」を選択して、コンポーネントをアセンブリー・ダイアグラムにドラッグすることで、新規コンポーネントを作成します。

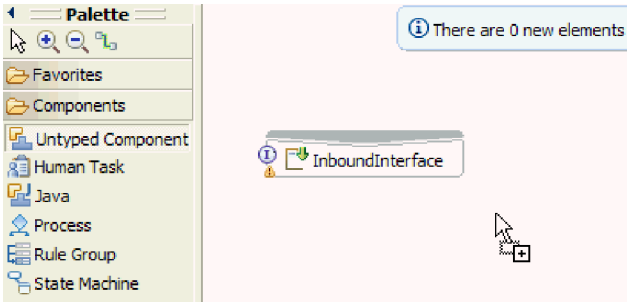


図 45. アセンブリー・ダイアグラムへのコンポーネントの追加

      カーソルが配置アイコンに変わります。
   c. アセンブリー・ダイアグラムに表示させるにはコンポーネントをクリックします。
2. コンポーネントを配線します。
   a. エクスポート・コンポーネントをクリックして新規コンポーネントにドラッグします。 これにより、以下の図に示されるように、エクスポート・コンポーネントから新規コンポーネントへ線が引かれます。

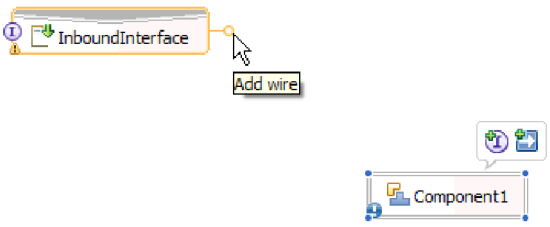


図 46. 「線 (wire)」アイコンの選択

b. アセンブリー・ダイアグラムを保管します。 「**ファイル**」 → 「**保管**」をクリックします。

3. 生成、新規コンポーネントの実装の

a. 新規コンポーネントを右クリックして、「**実装の生成 (Generate implementation)**」 → 「**Java**」を選択します。

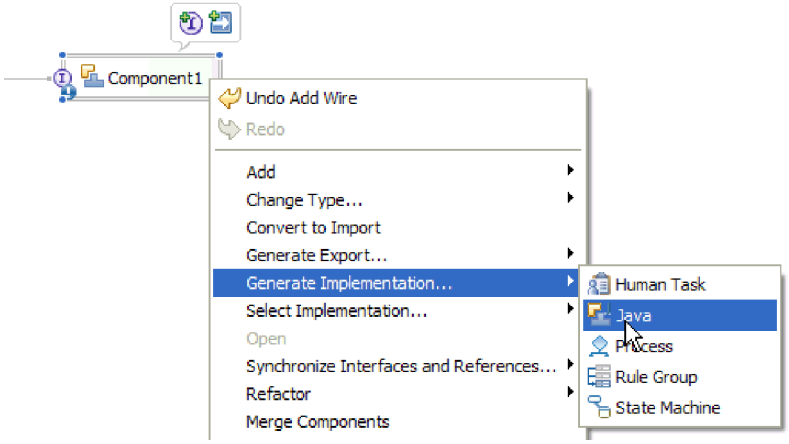


図 47. Java 実装の生成

b. 「**(デフォルト・パッケージ)**」を選択して、「**OK**」をクリックします。 これにより、インバウンド・モジュールのエンドポイントが作成されます。

Java 実装は、別のタブに表示されます。

c. **オプション:** エンドポイント・メソッドのそれぞれのエンドポイントで受信されたデータ・オブジェクトを出力するため、print ステートメントを追加します。

d. 「**ファイル**」 → 「**保管**」をクリックして変更を保管します。

次のタスク

テストするモジュールのデプロイを続行します。

サーバーへのモジュールの追加

WebSphere Integration Developer では、テスト環境の 1 つ以上のサーバーにモジュールを追加できます。

始める前に

テスト中のモジュールが、アダプターを使用して Inbound 処理を実行する場合、アダプターによるイベントの送信先であるターゲット・コンポーネント を生成して配線する必要があります。

このタスクを実行する理由および時期

モジュールと、モジュールによるアダプターの使用をテストするために、サーバーへモジュールを追加する必要があります。

このタスクの手順

1. *条件:* **「サーバー」**ビューにサーバーがない場合、以下のステップを実行して、新規サーバーを追加および定義します。
   a. **「サーバー」**ビューにカーソルを置き、右クリックして**「新規」** → **「サーバー」**と選択します。
   b. 「新規サーバーの定義 (Define a New Server)」ウィンドウから、サーバー・タイプを選択します。
   c. サーバーの設定を構成します。
   d. **「終了」**をクリックすると、サーバーが公開されます。
2. モジュールをサーバーに追加します。
   a. 「サーバー」ビューに切り替えます。 WebSphere Integration Developer で、**「ウィンドウ (Windows)」** → **「ビューの表示 (Show View)」** → **「サーバー」**と選択します。
   a. サーバーを始動します。 WebSphere Integration Developer の画面の右下のペインの「サーバー」タブで、「サーバー」を右クリックして、**「始動 (Start)」**を選択します。
3. サーバー状況が*「始動済み」*である場合は、「サーバー」を右クリックし、**「プロジェクトの追加および除去」**を選択します。
4. 「プロジェクトの追加および除去」画面でプロジェクトを選択し、**「追加」**をクリックします。 プロジェクトは、**「使用可能プロジェクト」**リストから**「構成プロジェクト」**リストに移動します。
5. **「終了」**をクリックします。 これにより、モジュールがサーバーにデプロイされます。

   モジュールがサーバーに追加されている間に、右下のペインの「コンソール」タブに、ログが表示されます。

次のタスク

モジュールとアダプターの機能をテストしてください。

テスト・クライアントを使用した Outbound 処理用モジュールのテスト

WebSphere Integration Developer 統合テスト・クライアントを使用して、アセンブルされた、Outbound 処理用モジュールおよびアダプターをテストします。

始める前に

最初にモジュールをサーバーに追加する必要があります。

このタスクを実行する理由および時期

モジュールのテストは、一般にコンポーネントのインターフェース操作で実行され、これにより、コンポーネントが正しく実装されているかどうか、および参照が正しく関連付けられているかどうかを判別できます。

このタスクの手順

1. テストするモジュールを選択して右クリックし、「**テスト**」 → 「**モジュールのテスト (Test Module)**」を選択します。
2. テスト・クライアントを使用したモジュールのテストについて詳しくは、WebSphere Integration Developer インフォメーション・センターの「モジュールおよびコンポーネントのテスト *(Testing modules and components)*」トピックを参照してください。

次のタスク

モジュールとアダプターのテスト結果が満足できるものであれば、モジュールとアダプターを実稼働環境にデプロイできます。

実動用のモジュールのデプロイ

外部サービス・ウィザードで作成したモジュールの、実稼働環境のWebSphere Process Server または WebSphere Enterprise Service Bus へのデプロイは、2 つのステップからなるプロセスです。最初に、WebSphere Integration Developer 内にモジュールをエンタープライズ・アーカイブ (EAR) ファイルの形でエクスポートします。次に、WebSphere Process Server 管理コンソールを使用して、EAR ファイルをデプロイします。

RAR ファイルのインストール (スタンドアロン・アダプターを使用するモジュールの場合のみ)

アダプターをモジュールに組み込まないが、サーバー・インスタンスのデプロイされたすべてのアプリケーションで使用可能にすることを選択する場合は、RAR ファイルの形式でアダプターをアプリケーション・サーバーにインストールする必要があります。 RAR ファイルは、Java 2 Connector (J2C) アーキテクチャーのリソース・アダプターをパッケージ化するのに使用される Java アーカイブ (JAR) ファイルです。

始める前に

外部サービス・ウィザードの「サービス生成およびデプロイメント構成 (Service Generation and Deployment Configuration)」ウィンドウで、「**コネクター・プロジェクトのデプロイ**」を「**サーバー上 (複数のアダプターによる使用) (On server for use by multiple adapters)**」に設定しておく必要があります。

このタスクを実行する理由および時期

RAR ファイルの形式でアダプターをインストールすると、アダプターは、サーバー・ランタイムに実行されているすべての J2EE アプリケーション・コンポーネントで使用可能になります。

このタスクの手順

1. 管理コンソールを開始します。
2. 「**リソース**」 → 「**リソース・アダプター**」 → 「**リソース・アダプター**」とクリックします。
3. 「リソース・アダプター」ページから、「**RAR のインストール**」をクリックします。

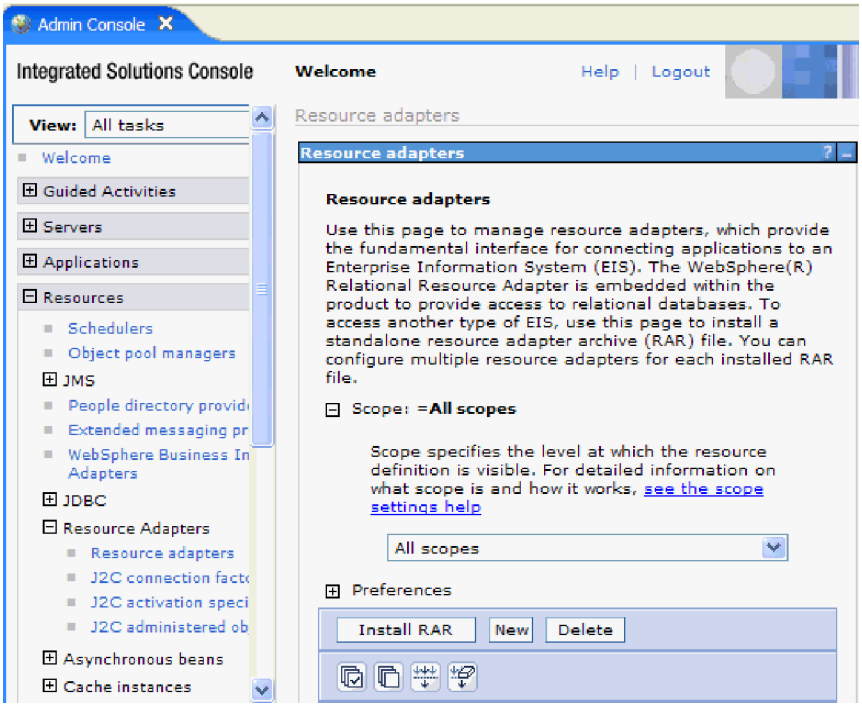


図 48. 「リソース・アダプター」ページの「RAR のインストール」ボタン

4. 「RAR ファイルのインストール」ページから、「**参照**」をクリックして、ご使用のアダプターの RAR ファイルにナビゲートします。

   RAR ファイルは、通常、パス *WID_installation_directory*/ResourceAdapters/*adapter_name*/deploy/*adapter*.rar にインストールされます。
5. 「**次へ**」をクリックします。
6. 「リソース・アダプター (Resource adapters)」ページから、オプションで、アダプターの名前を変更し、説明を追加します。
7. 「**OK**」をクリックします。
8. ページの上部にある「**メッセージ**」ボックスの「**保管**」をクリックします。

次のタスク

次のステップでは、サーバーにデプロイ可能なモジュールを EAR ファイルとしてエクスポートします。

EAR ファイルとしてのモジュールのエクスポート

WebSphere Integration Developer を使用して、モジュールを EAR ファイルとしてエクスポートします。EAR ファイルを作成することによって、モジュールのすべての内容を WebSphere Process Server または WebSphere Enterprise Service Bus に容易にデプロイできる形式で取り込みます。

始める前に

モジュールを EAR ファイルとしてエクスポートするには、事前にサービスと通信するためのモジュールを作成しておく必要があります。このモジュールを、WebSphere Integration Developer ビジネス・インテグレーション・パースペクティブ内に表示する必要があります。

このタスクを実行する理由および時期

モジュールを EAR ファイルとしてエクスポートするには、以下の手順を実行してください。

このタスクの手順

1. モジュールを右クリックして、「**エクスポート**」を選択します。
2. 「選択」ウィンドウで、「**J2EE**」を展開します。
3. 「**EAR ファイル**」を選択して、「**次へ**」をクリックします。

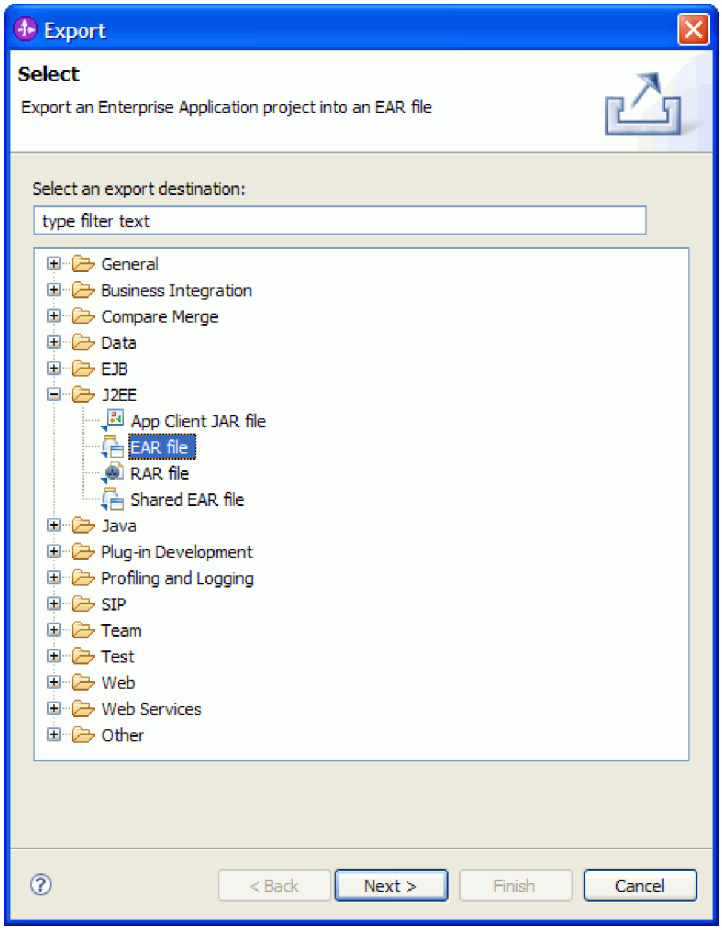


*図 49. 「選択」ウィンドウからの「**EAR ファイル**」の選択*

4. オプション: 正しい EAR アプリケーションを選択します。 EAR アプリケーションは、モジュールにちなんで名前が付けられますが、名前の末尾に「App」が追加されます。
5. EAR ファイルを格納するローカル・ファイル・システム上で、「**参照**」を選択してフォルダーを参照します。
6. オプションで、ソース・ファイルをエクスポートする場合は、「**ソース・ファイルのエクスポート (Export source files)**」を選択します。 このオプションは、EAR ファイルに加えてソース・ファイルをエクスポートする場合に提供されます。ソース・ファイルには、Java コンポーネント、データ・マップなどに関連付けられたファイルが含まれています。
7. 既存のファイルを上書きする場合は、「**既存のファイルを上書き (Overwrite an existing file)**」をクリックします。
8. 「**終了**」をクリックします。

結果

モジュールの内容が EAR ファイルとしてエクスポートされます。

次のタスク

このモジュールを管理コンソールにインストールします。これにより、モジュールが WebSphere Process Server にデプロイされます。

EAR ファイルのインストール

EAR ファイルのインストールは、デプロイメント・プロセスの最後のステップです。 EAR ファイルをサーバーにインストールして実行すると、EAR ファイルの一部として組み込まれているアダプターが、インストール済みアプリケーションの一部として稼働します。

始める前に

WebSphere Process Server 上にモジュールをインストールする前に、モジュールを EAR ファイルとしてエクスポートしておく必要があります。

このタスクを実行する理由および時期

EAR ファイルをインストールするには、次の手順を実行します。アダプター・モジュール・アプリケーションのクラスター化については、 http://www.ibm.com/software/webservers/appserv/was/library/ を参照してください。

このタスクの手順

1. サーバー・インスタンスを右クリックし、「**管理コンソールの実行**」を選択して、WebSphere Process Server 管理コンソールを開きます。
2. 管理コンソール・ウィンドウで、「**アプリケーション (Applications)**」 → 「**新規アプリケーションのインストール (Install New Applications)**」をクリックします。

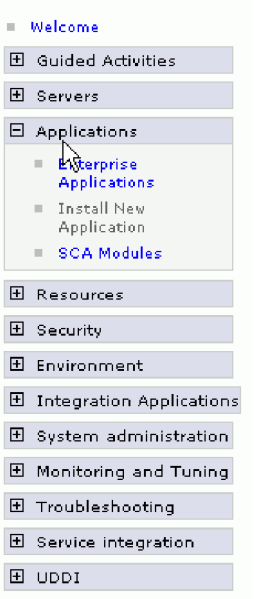


図 50. 「アプリケーション・インストールの準備」ウィンドウ

3. 「**参照**」をクリックして、EAR ファイルを位置指定し、「**次へ**」をクリックします。 EAR ファイル名は、モジュール名の後に「App」が付いたものです。

4. オプション: クラスター化された環境にデプロイする場合は、以下の手順を実行します。
   a. 「**ステップ 2: サーバーにモジュールをマップ**」ウィンドウで、モジュールを選択します。
   b. サーバー・クラスターの名前を選択します。
   c. 「**適用**」をクリックします。
5. 「**次へ**」をクリックして、「要約」を開きます。すべての設定が正しいことを確認して、「**終了**」をクリックします。
6. オプション: 認証別名を使用している場合は、以下の手順を実行します。
   a. 「**セキュリティー**」を展開して、「**ビジネス・インテグレーションの認証別名 (Business Integration Authentication Aliases)**」を選択します。
   b. 構成する認証別名を選択します。 認証別名の構成を変更するための管理者権限またはオペレーター権限を持っている必要があります。
   c. オプション: 「**ユーザー名**」を入力します (まだ入力されていない場合)。
   d. 「**パスワード**」を入力します (まだ入力されていない場合)。
   e. 「**確認パスワード (Confirm Password)**」フィールドに再度パスワードを入力します (まだ入力されていない場合)。
   f. 「**OK**」をクリックします。

結果

この時点で、プロジェクトがデプロイメントされ、「エンタープライズ・アプリケーション」ウィンドウが表示されます。

次のタスク

いずれかのプロパティーを設定または再設定する場合、あるいは、アダプター・プロジェクトのアプリケーションをクラスター化したい場合は、トラブルシューティング・ツールを構成する前に、管理コンソールを使用して対応する変更を行ってください。

第 7 章 アダプター・モジュールの管理

アダプターをスタンドアロンのデプロイメントで稼働している場合は、アダプター・モジュールの開始、停止、モニター、およびトラブルシューティングには、サーバーの管理コンソールを使用します。組み込みアダプターを使用しているアプリケーションでは、アプリケーションの開始時または停止時にアダプター・モジュールが開始または停止します。

組み込みアダプターの構成プロパティーの変更

アダプターをモジュールの一部としてデプロイした後に構成プロパティーを変更するには、ランタイム環境の管理コンソールを使用します。 リソース・アダプター・プロパティー (一般的なアダプター操作に使用される)、管理接続ファクトリー・プロパティー (Outbound 処理に使用される)、およびアクティベーション・スペック・プロパティー (Inbound 処理に使用される) を更新することができます。

組み込みアダプターのリソース・アダプター・プロパティーの設定

アダプターをモジュールの一部としてデプロイした後に、このアダプターのリソース・アダプター・プロパティーを設定するには、管理コンソールを使用します。構成するプロパティーの名前を選択してから、その値を変更または設定します。

始める前に

ご使用のアダプター・モジュールは、WebSphere Process Server または WebSphere Enterprise Service Bus にデプロイする必要があります。

このタスクを実行する理由および時期

カスタム・プロパティーは、すべての WebSphere アダプターが共用するデフォルトの構成プロパティーです。

管理コンソールを使用してプロパティーを構成するには、次の手順を使用してください。

このタスクの手順

1. 管理コンソールを開始します。
2. 「**アプリケーション**」の下で、「**エンタープライズ・アプリケーション**」を選択します。
3. 「**エンタープライズ・アプリケーション**」 リストから、プロパティーを変更するアダプター・モジュールの名前をクリックします。
4. 「**モジュール**」の下で、「**モジュールの管理**」をクリックします。

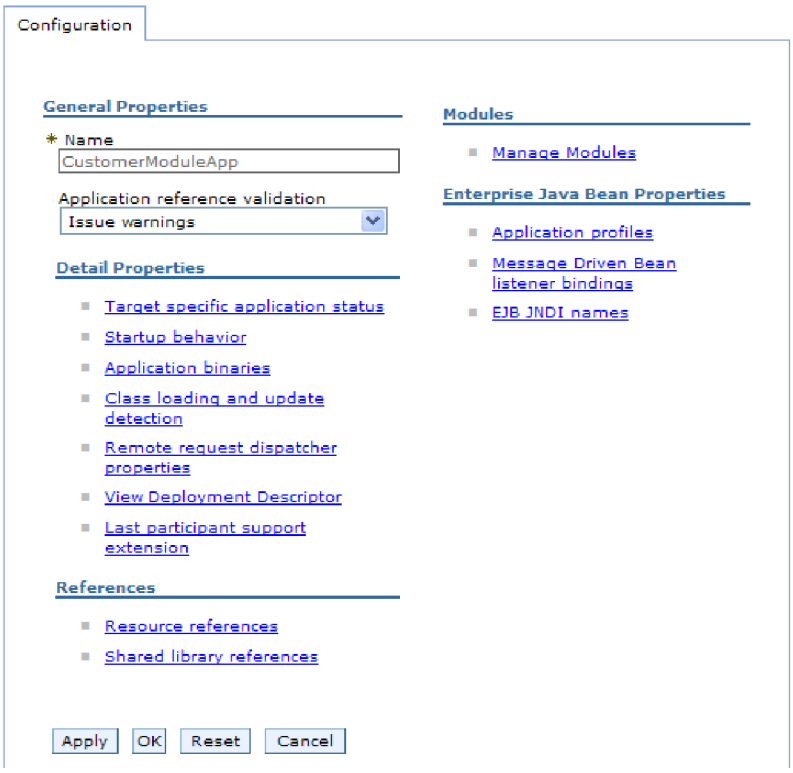


図 51. 「構成」タブでの「モジュールの管理」の選択

5. 「**IBM WebSphere Adapter for Flat Files**」をクリックします。
6. 「**追加プロパティー**」リストから、「**リソース・アダプター**」をクリックします。
7. 次のページで、「**追加プロパティー**」リストから、「**カスタム・プロパティー**」をクリックします。
8. 変更するプロパティーごとに、次の手順を実行します。

   注: ここで示すプロパティーについて詳しくは、142 ページの『リソース・アダプター・プロパティー』を参照してください。

   a. プロパティーの名前をクリックします。
   b. 「**値**」フィールドの内容を変更するか、フィールドが空の場合は値を入力します。

      例えば、「**logNumberOfFiles**」をクリックした場合、以下のページが表示されます。

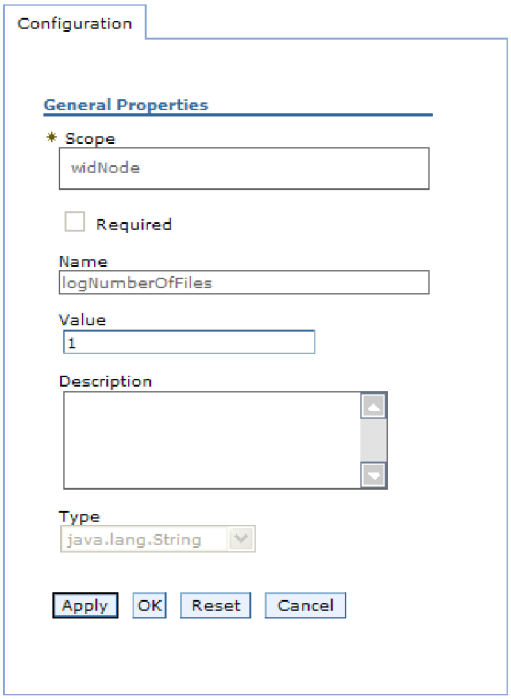


図 52. logNumberOfFiles プロパティーの「構成」タブ

「**値**」 フィールドの数値を変更し、プロパティーの説明を追加することができます。

c. 「**OK**」をクリックします。

9. ウィンドウの上部にある「**メッセージ**」ボックスの「**保管**」リンクをクリックします。

結果

アダプター・モジュールに関連付けられたリソース・アダプター・プロパティーが変更されます。

組み込みアダプターの管理 (J2C) 接続ファクトリー・プロパティーの設定

アダプターをモジュールの一部としてデプロイした後に、このアダプターの管理接続ファクトリー・プロパティーを設定するには、管理コンソールを使用します。構成するプロパティーの名前を選択してから、その値を変更または設定します。

始める前に

ご使用のアダプター・モジュールは、WebSphere Process Server または WebSphere Enterprise Service Bus にデプロイする必要があります。

このタスクを実行する理由および時期

管理接続ファクトリー・プロパティーは、ターゲット・ローカル・ファイル・システムのインスタンスを構成する場合に使用します。

注: 管理コンソール内では、このプロパティーを「J2C 接続ファクトリー・プロパティー」と呼びます。

管理コンソールを使用してプロパティーを構成するには、次の手順を使用してください。

このタスクの手順

1. 管理コンソールを開始します。
2. 「**アプリケーション**」の下で、「**エンタープライズ・アプリケーション**」を選択します。
3. 「**エンタープライズ・アプリケーション**」 リストから、プロパティーを変更するアダプター・モジュールの名前をクリックします。
4. 「**モジュール**」の下で、「**モジュールの管理**」をクリックします。

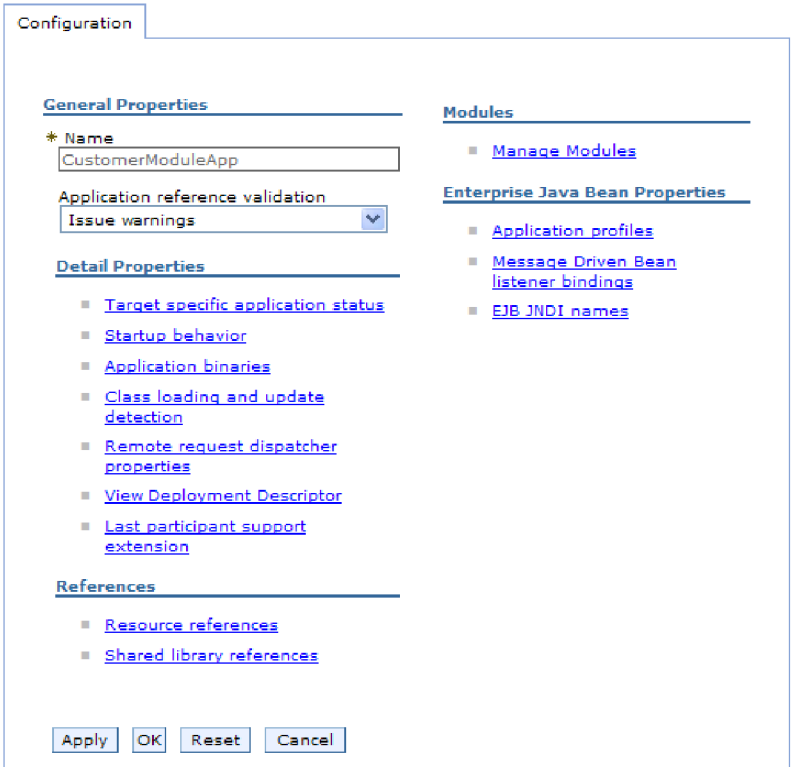


図 53. 「構成」タブでの「モジュールの管理」の選択

5. 「**IBM WebSphere Adapter for Flat Files**」をクリックします。
6. 「**追加プロパティー**」リストから、「**リソース・アダプター**」をクリックします。
7. 次のページで、「**追加プロパティー**」リストから、「**J2C 接続ファクトリー**」をクリックします。
8. アダプター・モジュールと関連付けられた接続ファクトリーの名前をクリックします。
9. 「**追加プロパティー**」リストから、「**カスタム・プロパティー**」をクリックします。

カスタム・プロパティーは、Adapter for Flat Files に特有の J2C 接続ファクトリー・プロパティーです。接続プールおよび拡張接続ファクトリー・プロパティーは、ユーザーが独自にアダプターを作成する場合に構成するプロパティーです。

10. 変更するプロパティーごとに、次の手順を実行します。

    注: ここで示すプロパティーについて詳しくは、 139 ページの『管理接続ファクトリー・プロパティー』を参照してください。

    a. プロパティーの名前をクリックします。
    b. 「**値**」フィールドの内容を変更するか、フィールドが空の場合は値を入力します。
    c. 「**OK**」をクリックします。
11. ウィンドウの上部にある「**メッセージ**」ボックスの「**保管**」リンクをクリックします。

結果

アダプター・モジュールに関連付けられた管理接続ファクトリー・プロパティーが変更されます。

組み込みアダプターのアクティベーション・スペック・プロパティーの設定

アダプターをモジュールの一部としてデプロイした後に、このアダプターのアクティベーション・スペック・プロパティーを設定するには、管理コンソールを使用します。構成するメッセージ・エンドポイント・プロパティーの名前を選択してから、その値を変更または設定します。

始める前に

ご使用のアダプター・モジュールは、WebSphere Process Server または WebSphere Enterprise Service Bus にデプロイする必要があります。

このタスクを実行する理由および時期

アクティベーション・スペック・プロパティーは、エンドポイントを Inbound 処理用に構成する場合に使用します。

管理コンソールを使用してプロパティーを構成するには、次の手順を使用してください。

このタスクの手順

1. 管理コンソールを開始します。
2. 「**アプリケーション**」の下で、「**エンタープライズ・アプリケーション**」を選択します。
3. 「**エンタープライズ・アプリケーション**」 リストから、プロパティーを変更するアダプター・モジュールの名前をクリックします。
4. 「**モジュール**」の下で、「**モジュールの管理**」をクリックします。

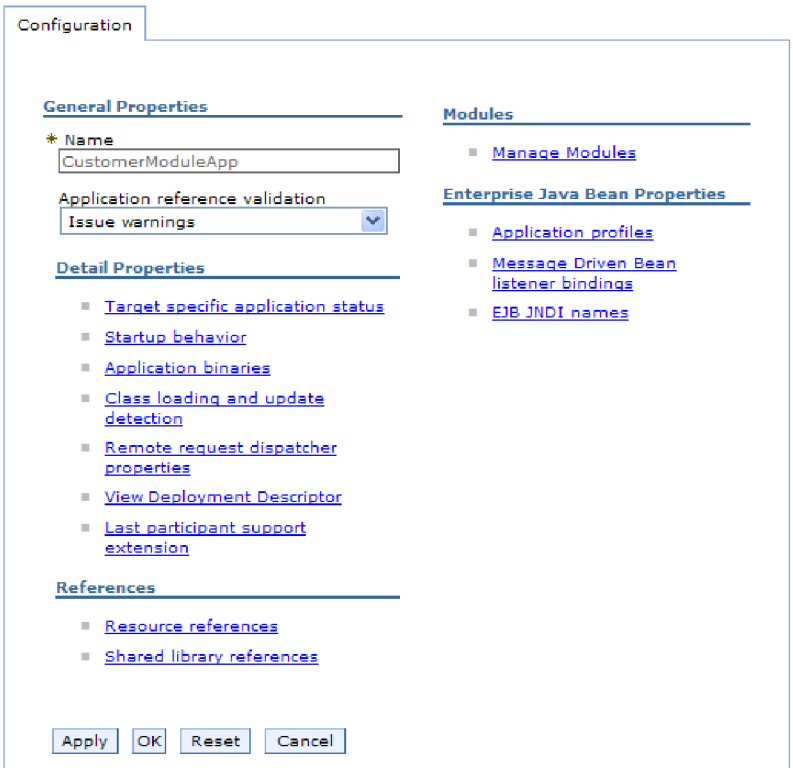


図 54. 「構成」タブでの「モジュールの管理」の選択

5. 「**IBM WebSphere Adapter for Flat Files**」をクリックします。
6. 「**追加プロパティー**」リストから、「**リソース・アダプター**」をクリックします。
7. 次のページで、「**追加プロパティー**」リストから、「**J2C アクティベーション・スペック**」をクリックします。
8. アダプター・モジュールと関連付けられたアクティベーション・スペックの名前をクリックします。
9. 「**追加プロパティー**」リストから、「**J2C アクティベーション・スペックのカスタム・プロパティー (J2C activation specification custom properties)**」をクリックします。
10. 変更するプロパティーごとに、次の手順を実行します。

    注: ここで示すプロパティーについて詳しくは、 154 ページの『アクティベーション・スペック・プロパティー』を参照してください。

    a. プロパティーの名前をクリックします。
    b. 「**値**」フィールドの内容を変更するか、フィールドが空の場合は値を入力します。
    c. 「**OK**」をクリックします。
11. ウィンドウの上部にある「**メッセージ**」ボックスの「**保管**」リンクをクリックします。

結果

アダプター・モジュールに関連付けられたアクティベーション・スペック・プロパティーが変更されます。

スタンドアロン・アダプターの構成プロパティーの変更

スタンドアロン・アダプターをインストールした後に構成プロパティーを設定するには、ランタイム環境の管理コンソールを使用します。アダプターに関する一般情報を提供してから、リソース・アダプター・プロパティー (一般的なアダプター操作に使用される) を設定します。アダプターが、Outbound 操作に使用される場合は、接続ファクトリーを作成し、その接続ファクトリーのプロパティーを設定します。 アダプターが、Inbound 操作に使用される場合は、アクティベーション・スペックを作成し、そのアクティベーション・スペックのプロパティーを設定します。

スタンドアロン・アダプターのリソース・アダプター・プロパティーの設定

スタンドアロン・アダプターを WebSphere Process Server または WebSphere Enterprise Service Bus にインストールした後に、このアダプターのリソース・アダプター・プロパティーを設定するには、管理コンソールを使用します。構成するプロパティーの名前を選択してから、その値を変更または設定します。

始める前に

ご使用のアダプターは、WebSphere Process Server または WebSphere Enterprise Service Bus にインストールする必要があります。

このタスクを実行する理由および時期

カスタム・プロパティーは、すべての WebSphere アダプターが共用するデフォルトの構成プロパティーです。

管理コンソールを使用してプロパティーを構成するには、次の手順を使用してください。

このタスクの手順

1. 管理コンソールを開始します。
2. 「**リソース**」 → 「**リソース・アダプター**」 → 「**リソース・アダプター**」とクリックします。
3. 「リソース・アダプター」ページから、「**IBM WebSphere Adapter for Flat Files**」をクリックします。
4. 「**追加プロパティー**」リストから、「**カスタム・プロパティー**」をクリックします。
5. 変更するプロパティーごとに、次の手順を実行します。

   注: ここで示すプロパティーについて詳しくは、 142 ページの『リソース・アダプター・プロパティー』を参照してください。

   a. プロパティーの名前をクリックします。
   b. 「**値**」フィールドの内容を変更するか、フィールドが空の場合は値を入力します。

例えば、「**logNumberOfFiles**」をクリックした場合、以下のページが表示されます。

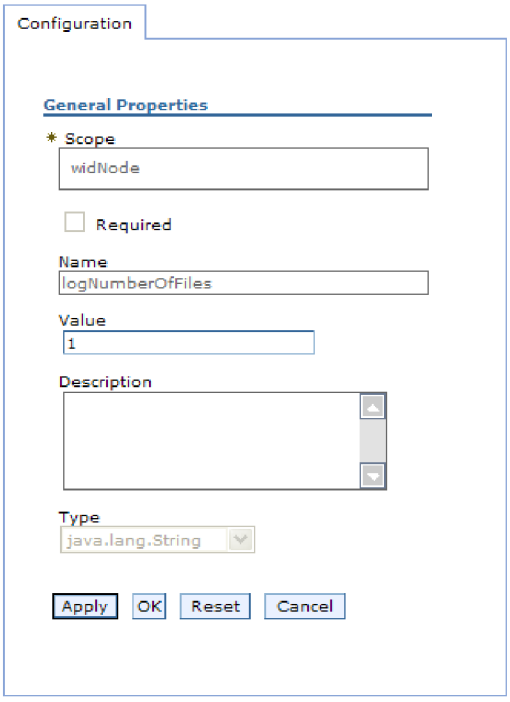


図 55. logNumberOfFiles プロパティーの「構成」タブ

「**値**」 フィールドの数値を変更し、プロパティーの説明を追加することができます。

c. 「**OK**」をクリックします。

6. ページの上部にある「**メッセージ**」ボックスの「**保管**」をクリックします。

結果

アダプターに関連付けられたリソース・アダプター・プロパティーが変更されます。

スタンドアロン・アダプターの管理 (J2C) 接続ファクトリー・プロパティーの設定

スタンドアロン・アダプターを WebSphere Process Server または WebSphere Enterprise Service Bus にインストールした後に、このアダプターの管理接続ファクトリー・プロパティーを設定するには、管理コンソールを使用します。構成するプロパティーの名前を選択してから、その値を変更または設定します。

始める前に

ご使用のアダプターは、WebSphere Process Server または WebSphere Enterprise Service Bus にインストールする必要があります。

このタスクを実行する理由および時期

管理接続ファクトリー・プロパティーは、ターゲット・ローカル・ファイル・システムのインスタンスを構成する場合に使用します。

注: 管理コンソール内では、このプロパティーを「J2C 接続ファクトリー・プロパティー」と呼びます。

管理コンソールを使用してプロパティーを構成するには、次の手順を使用してください。

このタスクの手順

1. 管理コンソールを開始します。
2. 「**リソース**」 → 「**リソース・アダプター**」 → 「**リソース・アダプター**」とクリックします。
3. 「リソース・アダプター」ページから、「**IBM WebSphere Adapter for Flat Files**」をクリックします。
4. 「**追加プロパティー**」リストから、「**J2C 接続ファクトリー**」をクリックします。
5. 既存の接続ファクトリーを使用する場合は、手順6にスキップします。

   注: 外部サービス・ウィザードを使用してアダプター・モジュールを構成したときに、「**事前定義された接続プロパティーを使用する**」を選択した場合は、接続ファクトリーを作成する必要はありません。

   接続ファクトリーを作成する場合は、以下の手順を実行します。

   a. 「**新規**」をクリックします。
   b. 「**構成**」タブの「**一般プロパティー (General Properties)**」セクションで、接続ファクトリーの名前を入力します。例えば、`AdapterCF` と入力します。
   c. 「**JNDI 名**」に値を入力します。 例えば、 `com/eis/AdapterCF` と入力します。
   d. 「**コンポーネント管理認証別名**」リストから認証別名を選択します。
   e. 「**OK**」をクリックします。
   f. ページの上部にある「**メッセージ**」ボックスの「**保管**」をクリックします。

   新規に作成された接続ファクトリーが表示されます。

図 56. 接続ファクトリーのリスト

6. 接続ファクトリーのリストから、使用したい接続ファクトリーをクリックします。

7. 「**追加プロパティー**」リストから、「**カスタム・プロパティー**」をクリックします。

   カスタム・プロパティーは、Adapter for Flat Files に特有の J2C 接続ファクトリー・プロパティーです。接続プールおよび拡張接続ファクトリー・プロパティーは、ユーザーが独自にアダプターを作成する場合に構成するプロパティーです。
8. 変更するプロパティーごとに、次の手順を実行します。

   注: ここで示すプロパティーについて詳しくは、139 ページの『管理接続ファクトリー・プロパティー』を参照してください。
   a. プロパティーの名前をクリックします。
   b. 「**値**」フィールドの内容を変更するか、フィールドが空の場合は値を入力します。
   c. 「**OK**」をクリックします。
9. プロパティーの設定が完了したら、「**適用**」をクリックします。
10. ウィンドウの上部にある「**メッセージ**」ボックスの「**保管**」をクリックします。

結果

アダプターに関連付けられた管理接続ファクトリー・プロパティーが設定されます。

スタンドアロン・アダプターのアクティベーション・スペック・プロパティーの設定

スタンドアロン・アダプターを WebSphere Process Server または WebSphere Enterprise Service Bus にインストールした後に、このアダプターのアクティベーション・スペック・プロパティーを設定するには、管理コンソールを使用します。構成するメッセージ・エンドポイント・プロパティーの名前を選択してから、その値を変更または設定します。

始める前に

ご使用のアダプターは、WebSphere Process Server または WebSphere Enterprise Service Bus にインストールする必要があります。

このタスクを実行する理由および時期

アクティベーション・スペック・プロパティーは、エンドポイントを Inbound 処理用に構成する場合に使用します。

管理コンソールを使用してプロパティーを構成するには、次の手順を使用してください。

このタスクの手順

1. 管理コンソールを開始します。
2. 「**リソース**」 → 「**リソース・アダプター**」 → 「**リソース・アダプター**」とクリックします。

3. 「リソース・アダプター」ページから、「**IBM WebSphere Adapter for Flat Files**」をクリックします。
4. 「**追加プロパティー**」リストから、「**J2C アクティベーション・スペック**」をクリックします。
5. 既存のアクティベーション・スペックを使用する場合は、手順6にスキップします。

   注: 外部サービス・ウィザードを使用してアダプター・モジュールを構成したときに、「**事前定義された接続プロパティーを使用する**」を選択した場合は、アクティベーション・スペックを作成する必要はありません。

   アクティベーション・スペックを作成する場合は、以下の手順を実行します。
   a. 「**新規**」をクリックします。
   b. 「**構成**」タブの「**一般プロパティー (General Properties)**」セクションで、アクティベーション・スペックの名前を入力します。例えば、`AdapterAS` と入力します。
   c. 「**JNDI 名**」に値を入力します。 例えば、 `com/eis/AdapterAS` と入力します。
   d. 「**認証別名**」リストから認証別名を選択します。
   e. メッセージ・リスナー・タイプを選択します。
   f. 「**OK**」をクリックします。
   g. ページの上部にある「**メッセージ**」ボックスの「**保管**」をクリックします。

      新規に作成されたアクティベーション・スペックが表示されます。
6. アクティベーション・スペックのリストから、使用したいスペックをクリックします。
7. 「追加プロパティー」リストから、「**J2C アクティベーション・スペックのカスタム・プロパティー (J2C activation specification custom properties)**」をクリックします。
8. 設定するプロパティーごとに、以下の手順を実行します。

   注: ここで示すプロパティーについて詳しくは、 154 ページの『アクティベーション・スペック・プロパティー』を参照してください。
   a. プロパティーの名前をクリックします。
   b. 「**値**」フィールドの内容を変更するか、フィールドが空の場合は値を入力します。
   c. 「**OK**」をクリックします。
9. プロパティーの設定が完了したら、「**適用**」をクリックします。
10. ページの上部にある「**メッセージ**」ボックスの「**保管**」をクリックします。

結果

アダプターに関連付けられたアクティベーション・スペック・プロパティーが設定されます。

アダプターを使用するアプリケーションの開始

アダプターを使用するアプリケーションを開始するには、サーバーの管理コンソールを使用します。デフォルトでは、アプリケーションは、サーバーの始動時に自動的に開始します。

このタスクを実行する理由および時期

アプリケーションを開始するには、アプリケーションが組み込みアダプターを使用している場合でもスタンドアロン・アダプターを使用している場合でも、この手順を使用します。組み込みアダプターを使用するアプリケーションでは、アプリケーションの開始時にアダプターが開始します。スタンドアロン・アダプターを使用するアプリケーションでは、アプリケーション・サーバーの始動時にアダプターが開始します。

このタスクの手順

1. 管理コンソールで、「**アプリケーション**」 → 「**エンタープライズ・アプリケーション**」をクリックします。

   注: 管理コンソールには、「Integrated Solutions Console」というラベルが付けられます。
2. 開始したいアプリケーションのチェック・ボックスを選択します。アプリケーション名は、インストールした EAR ファイル名から .EAR ファイル 拡張子を除いたものです。
3. 「**開始 (Start)**」をクリックします。

結果

アプリケーションの状況が「開始済み (Started)」に変化し、アプリケーションが開始したことを示すメッセージが管理コンソールの上部に表示されます。

アダプターを使用するアプリケーションの停止

アダプターを使用するアプリケーションを停止するには、サーバーの管理コンソールを使用します。デフォルトでは、アプリケーションは、サーバーの停止時に自動的に停止します。

このタスクを実行する理由および時期

アプリケーションを停止するには、アプリケーションが組み込みアダプターを使用している場合でもスタンドアロン・アダプターを使用している場合でも、この手順を使用します。組み込みアダプターを使用するアプリケーションでは、アプリケーションの停止時にアダプターが停止します。スタンドアロン・アダプターを使用するアプリケーションでは、アプリケーション・サーバーの停止時にアダプターが停止します。

このタスクの手順

1. 管理コンソールで、「**アプリケーション**」 → 「**エンタープライズ・アプリケーション**」をクリックします。

注: 管理コンソールには、「Integrated Solutions Console」というラベルが付けられます。

2. 停止したいアプリケーションのチェック・ボックスを選択します。アプリケーション名は、インストールした EAR ファイル名から .EAR ファイル 拡張子を除いたものです。
3. 「**停止 (Stop)**」をクリックします。

結果

アプリケーションの状況が「停止中 (Stopped)」に変化し、アプリケーションが停止したことを示すメッセージが管理コンソールの上部に表示されます。

Performance Monitoring Infrastructure を使用したパフォーマンスのモニター

Performance Monitoring Infrastructure (PMI) は、管理コンソールの機能の 1 つで、これを使用すると、実稼働環境内で Adapter for Flat Files を含む、コンポーネントのパフォーマンスを動的にモニターすることができます。PMI は、サーバー内のさまざまなコンポーネントから、平均応答時間や要求の総数などのアダプターのパフォーマンス・データを収集して、そのデータをツリー構造に編成します。このデータは、Tivoli® Performance Viewer (WebSphere Process Server の管理コンソールに組み込まれているグラフィカル・モニター・ツール) を通して表示することができます。

このタスクを実行する理由および時期

PMI により、以下の時点のデータを収集することによって、アダプターのパフォーマンスをモニターすることができます。

- Outbound 処理時。Outbound 要求をモニターします。
- Inbound イベントの取り出し時。イベント・テーブルからのイベントの取り出しをモニターします。
- Inbound イベントの送達時。エンドポイント (1 つまたは複数の) へのイベントの送達をモニターします。

使用するアダプター用に PMI を使用可能に設定し、構成するためには、まず、トレース機能の詳細レベルを設定し、パフォーマンス・データの収集元となるいくつかのイベントを実行する必要があります。

アダプター環境の全体的なパフォーマンスをモニターし、向上させるために、PMI がどう役立つかについての詳細な説明は、WebSphere Application Server の Web サイト http://www.ibm.com/software/webservers/appserv/was/library/ で PMI を検索してください。

Performance Monitoring Infrastructure の構成

Performance Monitoring Infrastructure (PMI) を、アダプターのパフォーマンス・データ (平均応答時間や要求の総数など) を収集するように構成することができます。アダプター用に PMI を構成した後、 Tivoli Performance Viewer を使用してアダプターのパフォーマンスをモニターすることができます。

始める前に

使用するアダプター用に PMI を構成するためには、まず、トレース機能の詳細レベルを設定し、パフォーマンス・データの収集元となるいくつかのイベントを実行する必要があります。

1. トレース機能を使用可能にしてイベント・データを受け取るためには、トレース・レベルを fine、finer、finest、または all のいずれかに設定する必要があります。*=info の後に、コロンとストリングを追加します。例えば、次のように入力します。

   ```
   *=info: WBILocationMonitor.CEI.ResourceAdapter.
   *=finest: WBILocationMonitor.LOG.ResourceAdapter.*=finest:
   ```

   トレース・レベルの設定方法については、113 ページの『Common Event Infrastructure (CEI) によるトレースの使用可能化』を参照してください。
2. 1 つ以上の Outbound 要求 または Inbound イベントを生成して、構成可能なパフォーマンス・データを生成します。

このタスクの手順

1. アダプターに対して PMI を使用可能にします。
   a. 管理コンソールで、「**モニターおよびチューニング**」を展開してから、「**Performance Monitoring Infrastructure (PMI)**」を選択します。
   b. サーバーのリストから、ご使用のサーバーの名前をクリックします。
   c. 「構成」タブを選択してから、「**Performance Monitoring (PMI) を使用可能にする (Enable Performance Monitoring (PMI))**」チェック・ボックスを選択します。
   d. 「**カスタム**」を選択して、選択的に統計を使用可能または使用不可に設定します。

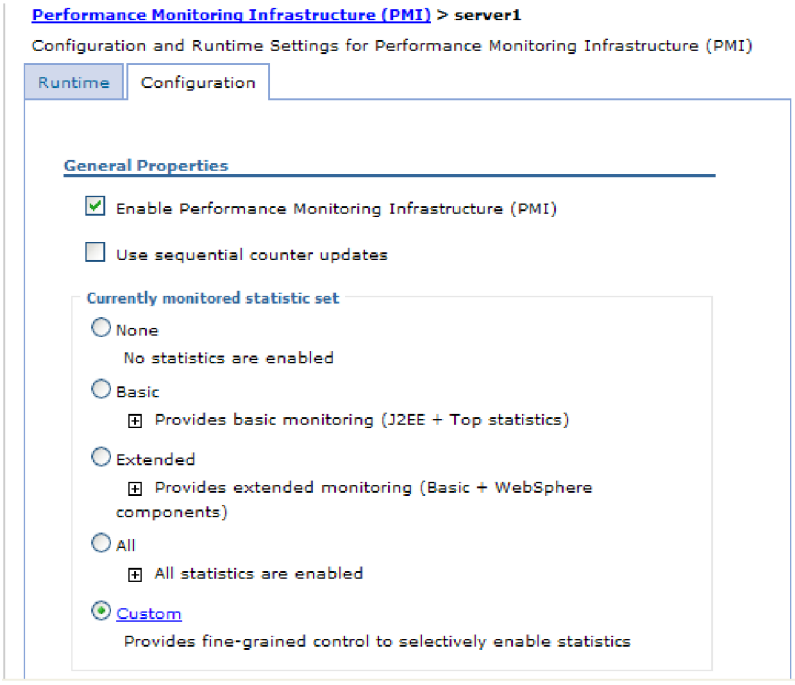


図 57. Performance Monitoring Infrastructure の使用可能化

e. 「**適用**」または「**OK**」をクリックします。

f. 「**保管**」をクリックします。 これで、PMI が使用可能になりました。

2. アダプター用に PMI を構成します。

a. 管理コンソールで、「**モニターおよびチューニング**」を展開してから、「**Performance Monitoring Infrastructure (PMI)**」を選択します。

b. サーバーのリストから、ご使用のサーバーの名前をクリックします。

c. 「**カスタム**」を選択します。

d. 「**ランタイム**」タブを選択します。 以下の図は、「ランタイム」タブを示しています。

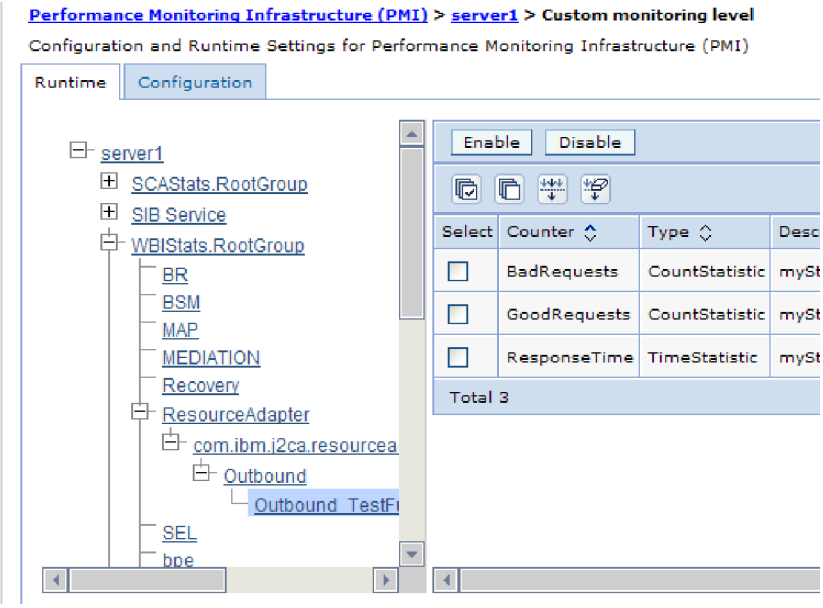


図 58. PMI の構成に使用される「ランタイム」タブ

e. 「**WBIStats.RootGroup**」をクリックします。 これは、ルート・グループで収集されるデータ用の PMI サブモジュールです。この例では、ルート・グループに WBIStats という名前を使用しています。

f. 「**ResourceAdapter**」をクリックします。 これは、JCA アダプターについて収集されるデータ用のサブモジュールです。

g. アダプターの名前をクリックして、モニターするプロセスを選択します。

h. 右側のペインで、収集する統計のチェック・ボックスを選択してから、「**使用可能**」をクリックします。

結果

PMI がアダプター用に構成されます。

次のタスク

これで、アダプターのパフォーマンス統計を表示することができるようになりました。

パフォーマンスに関する統計の表示

アダプターのパフォーマンス・データは、グラフィカル・モニター・ツール Tivoli Performance Viewer を使用して表示することができます。Tivoli Performance Viewer は、WebSphere Process Server の管理コンソールに組み込まれています。

始める前に

アダプター用の Performance Monitoring Infrastructure の構成。

このタスクの手順

1. 管理コンソールで、「**モニターおよびチューニング** 」を展開し、「**Performance Viewer**」を展開した後、「**現行アクティビティー**」を選択します。

2. サーバーのリストで、ご使用のサーバー名をクリックします。
3. サーバー名の下で、「**パフォーマンス・モジュール**」を展開します。
4. 「**WBIStatsRootGroup**」をクリックします。
5. 「**ResourceAdapter**」およびアダプター・モジュールの名前をクリックします。
6. 複数のプロセスがある場合は、統計を表示させるプロセスのチェック・ボックスを選択します。

結果

右側のパネルに統計が表示されます。「**グラフの表示**」をクリックして、データのグラフを表示するか、または「**表の表示**」をクリックして、統計を表形式で表示することができます。以下の図では、アダプターのパフォーマンス統計をグラフの形で表示しています。

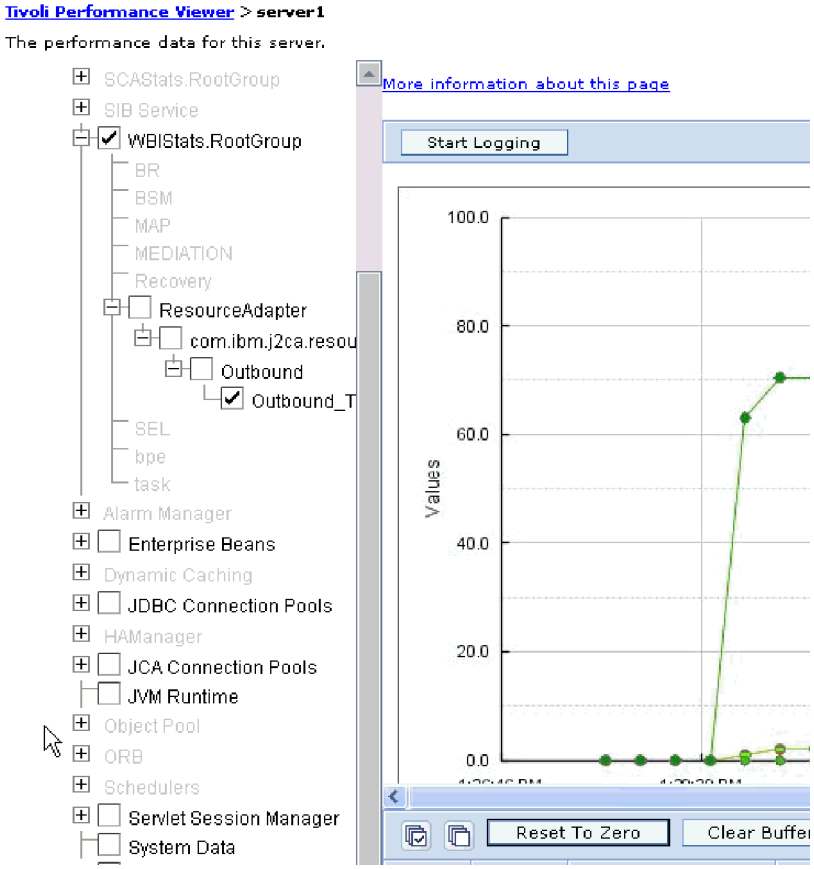


図 59. グラフ表示によるアダプターのパフォーマンス統計

Common Event Infrastructure (CEI) によるトレースの使用可能化

アダプターは、サーバーに組み込まれたコンポーネントである Common Event Infrastructure を使用して、ポーリング周期の開始と停止などの重要なビジネス・イベントに関するデータについて報告を行います。イベント・データは、構成設定に応じて、データベースまたはトレース・ログ・ファイルに書き込むことができます。

このタスクの手順

1. 管理コンソールで「**トラブルシューティング**」をクリックします。
2. 「**ログおよびトレース (Logs and Trace)**」をクリックします。
3. サーバーのリストで、ご使用のサーバー名をクリックします。
4. 「**ログ詳細レベルの変更**」 ボックスで、アダプターがイベント・データを書き込む CEI データベース (例えば、WBIEventMonitor.CEI.ResourceAdapter.*)、またはトレース・ログ・ファイル (例えば、WBIEventMonitor.LOG.ResourceAdapter.*) の名前をクリックします。
5. アダプターがデータベースまたはトレース・ログ・ファイルに書き込むビジネス・イベントの詳細レベルを選択し、(オプションで) メッセージとトレースに関連付けられた詳細の細分度を調整します。
   - 「**ロギングなし**」。イベント・ロギングをオフにします。
   - 「**メッセージのみ (Messages Only)**」。 アダプターは、イベントを報告します。
   - 「**すべてのメッセージとトレース (All Messages and Traces)**」。 アダプターは、イベントについての詳細を報告します。
   - 「**メッセージ・レベルとトレース・レベル (Message and Trace Levels)**。イベントに関連付けられたビジネス・オブジェクト・ペイロードについてアダプターが報告する詳細度の制御の設定。詳細レベルを調整する場合は、以下のいずれかを選択してください。

     「**詳細 - 中**」。 アダプターはイベントを報告しますが、ビジネス・オブジェクト・ペイロードは報告しません。

     「**詳細 - 高**」。アダプターはイベントとビジネス・オブジェクト・ペイロードの説明を報告します。

     「**詳細 - 必須**」。アダプターはイベントとすべてのビジネス・オブジェクト・ペイロードを報告します。
6. 「**OK**」をクリックします。

結果

イベント・ロギングは使用可能です。管理コンソール内の Common Base Event ブラウザーを使用して、トレース・ログ・ファイルの CEI 項目を表示させることができます。

トラブルシューティングおよびサポート

共通のトラブルシューティング手法とセルフ・ヘルプ情報は、問題を迅速に識別して解決するのに役立ちます。

ロギングおよびトレースの構成

要件に合うようロギングおよびトレースを構成します。アダプターのロギングを使用可能にし、イベント処理の状況を制御します。アダプターのログ・ファイル名およびトレース・ファイル名を変更し、他のログ・ファイルおよびトレース・ファイルから区別します。

ロギング・プロパティーの構成

ログを使用可能にし、ログの出力プロパティー (ログのロケーション、詳細レベル、出力形式など) を設定するには、管理コンソールを使用します。

このタスクを実行する理由および時期

モニター・イベントをアダプターでログに記録するには、モニターするサービス・コンポーネント・イベント・ポイント、各イベントに必要な詳細レベル、およびイベントをログにパブリッシュする際に使用する出力形式を指定する必要があります。管理コンソールを使用して、以下のタスクを実行します。

- 特定のイベント・ログを使用可能または使用不可にする。
- ログの詳細レベルを指定する。
- ログ・ファイルの保管場所と保持数を指定する。
- ログの出力形式を指定する。

  ログ・アナライザーの出力形式を設定した場合は、ログ・アナライザー・ツール (プロセス・サーバーに同梱されるアプリケーション) を使用して、トレース出力を開くことができます。これは、2 つの異なるサーバー・プロセスからのトレースを相関しようとする場合に便利です。なぜなら、これにより、ログ・アナライザーのマージ機能が使用できるからです。

プロセス・サーバー (サービス・コンポーネントとイベント・ポイントを含む) のモニターの詳細については、ご使用のプロセス・サーバーの資料を参照してください。

ログ構成は、静的または動的に変更できます。静的構成は、アプリケーション・サーバーを始動または再始動したときに有効になります。動的構成 (実行時構成) の変更は、直ちに適用されます。

ログを作成すると、そのログの詳細レベルは構成データから設定されます。特定のログ名に対して構成データが提供されていない場合、そのログのレベルは、親ログから取得されます。親ログに構成データが存在しない場合は、更にその親ログを検査するという動作を繰り返し、非ヌル・レベルの値を持つログが見つかるまで、ツリーをさかのぼっていきます。ログのレベルを変更すると、その変更は子ログに伝搬されます。この変更は、必要に応じて、更にその子ログに再帰的に伝搬されます。

ロギングを使用可能にし、ログの出力プロパティーを設定するには、以下の手順を実行します。

このタスクの手順

1. 管理コンソールのナビゲーション・ペインで、「**サーバー**」 → 「**アプリケーション・サーバー**」をクリックします。
2. 操作するサーバーの名前をクリックします。
3. 「**トラブルシューティング**」の下で「**ログおよびトレース (Logs and trace)**」をクリックします。
4. 「**ログの詳細レベルの変更 (Change Log Detail Levels)**」をクリックします。
5. いつ変更を有効にするのかを指定します。

- 構成を静的に変更する場合は、「**構成**」タブをクリックします。
- 構成を動的に変更する場合は、「**実行時 (Runtime)**」タブをクリックします。

6. 変更したいロギング・レベルのパッケージの名前をクリックします。
   WebSphere Adapters のパッケージ名は、**com.ibm.j2ca** で始まります。
   - アダプターの基本コンポーネントの場合は、「**com.ibm.j2ca.base**」を選択します。
   - アダプターの基本コンポーネント、およびデプロイされたすべてのアダプターの場合は、「**com.ibm.j2ca.base.***」を選択します。
   - Adapter for Flat Files の場合のみ、**com.ibm.j2ca.flatfile** パッケージを選択します。
7. ロギング・レベルを選択します。

ロギング・レベル	説明
致命的	タスクを続行できないか、コンポーネントが機能しません。
重大	タスクは続行できませんが、コンポーネントはまだ機能します。このロギング・レベルには、差し迫った致命的エラーを示す状況、つまりリソースが枯渇寸前であることを強く示す状況も含まれます。
警告	潜在的なエラーが発生したか、重大なエラーが差し迫っています。このロギング・レベルには、進行性の障害 (リソース・リークの可能性など) を示す状況も含まれます。
監査	サーバーの状態またはリソースに影響を及ぼす重大なイベントが発生しました。
情報	タスクが実行中です。このロギング・レベルには、タスクの全体的な進行状況を示す一般情報が含まれます。
構成	構成の状況が報告されるか、構成の変更が発生しました。
詳細	サブタスクが実行中です。このロギング・レベルには、サブタスクの進行状況を詳しく示す一般情報が含まれます。

8. 「**適用**」をクリックします。
9. 「**OK**」をクリックします。
10. 静的な構成変更を有効にするには、プロセス・サーバーを停止し、再始動します。

結果

これ以降、ログ項目には、選択したアダプター・コンポーネントについての指定したレベルの情報が格納されます。

ログ・ファイル名およびトレース・ファイル名の変更

アダプター・ログおよびトレース情報を他のプロセスとは分離して保持するには、管理コンソールを使用してファイル名を変更します。デフォルトでは、プロセス・サーバー上にあるすべてのプロセスおよびアプリケーションのログ情報およびトレース情報は、それぞれ SystemOut.log ファイルおよび trace.log ファイルに書き込まれます。

始める前に

アダプター・モジュールをアプリケーション・サーバーにデプロイした後は、ログ・ファイル名およびトレース・ファイル名はいつでも変更できます。

このタスクを実行する理由および時期

ログ・ファイルおよびトレース・ファイルは、静的または動的に変更できます。アプリケーション・サーバーを始動または再始動すると、静的変更が反映されます。動的変更またはランタイム構成変更は、即座に適用されます。

ログ・ファイルおよびトレース・ファイルは、*install_root*/profiles/*profile_name*/logs/*server_name* フォルダーにあります。

ログ・ファイル名およびトレース・ファイル名を設定または変更するには、次の手順を実行します。

このタスクの手順

1. 管理コンソールのナビゲーション・ペインで、「**アプリケーション**」>「**エンタープライズ・アプリケーション**」を選択します。
2. 「エンタープライズ・アプリケーション」リストから、アダプター・アプリケーションの名前をクリックします。これは、アダプターの EAR ファイルの名前から .ear ファイル拡張子を除いたものです。 例えば、EAR ファイルの名前が Accounting_OutboundApp.ear である場合は、**Accounting_OutboundApp** をクリックします。
3. 「構成」タブの「モジュール」リストから、「**モジュールの管理**」をクリックします。
4. モジュールのリストで、IBM WebSphere Adapter for Flat Files をクリックします。
5. 「構成」タブの「追加プロパティー」の下で、「**リソース・アダプター**」をクリックします。
6. 「構成」タブの「追加プロパティー」の下で、「**カスタム・プロパティー**」をクリックします。
7. 「カスタム・プロパティー」テーブル内で、ファイル名を変更します。
   a. 「**logFilename**」をクリックして、ログ・ファイルの名前を変更します。あるいは、「**traceFilename**」をクリックして、トレース・ファイルの名前を変更します。
   b. 「構成」タブで、「**値**」フィールドに新しい名前を入力します。 デフォルトでは、ログ・ファイルの名前は SystemOut.log、トレース・ファイルの名前は trace.log になります。
   c. 「**適用**」または「**OK**」をクリックします。 変更内容がローカル・マシン上に保存されます。
   d. 変更内容をサーバー上のマスター構成に保存するには、次のいずれかの手順を実行します。
      - **静的変更**: サーバーを停止してから再始動します。この方法では、変更を行うことは可能ですが、サーバーを停止してから始動するまで、行った変更は有効になりません。

- **動的変更**: 「カスタム・プロパティー」テーブルの上にあるメッセージ・ボックス内にある「**保管**」リンクをクリックします。プロンプトが出されたら、再度「**保管**」をクリックします。この方法では、行った変更をすぐに有効にすることができます。

First Failure Data Capture (FFDC) のサポート

アダプターは、First Failure Data Capture (FFDC) をサポートします。これにより、WebSphere Process Server または WebSphere Enterprise Service Bus のランタイム中に発生する障害および重大なソフトウェアの問題が永続的に記録されます。

FFDC 機能は、バックグラウンドで実行され、ランタイムに発生するイベントおよびエラーを収集します。この機能により、互いに失敗を関連付ける手段が提供され、ソフトウェアは障害の結果をその原因に結び付けることができ、障害の根本原因を迅速に突き止めることが容易になります。収集されたデータを使用して、アダプターのランタイム中に発生する例外処理を識別することができます。

問題が発生した場合、アダプターは例外メッセージおよびコンテキスト・データを *install_root*/profiles/*profile*/logs/ffdc ディレクトリーにあるログ・ファイルに書き込みます。

First Failure Data Capture (FFDC) について詳しくは、WebSphere Process Server または WebSphere Enterprise Service Bus の文書を参照してください。

ビジネス・フォールト

アダプターは、Outbound サービス記述またはインポートで予測および宣言される例外であるビジネス・フォールトに対応します。ビジネス・フォールトは、ビジネス・ルールや制約に違反した場合にビジネス・プロセス中で生じるため、予測可能です。

WebSphere Process Server および WebSphere Enterprise Service Bus は、他のタイプのフォールトに対応しますが、アダプターはビジネス・フォールト (この文書中では単にフォールト と呼びます) のみを生成します。すべての例外がフォールトになるわけではありません。フォールトは、アクション可能なエラー、つまり、アプリケーションの終了を必要としないリカバリー・アクションが可能なエラーに対して生成されます。例えば、アダプターは、必要なデータが含まれていない Outbound 処理のビジネス・オブジェクトを受け取ったときか、Outbound 処理中に特定のエラーが発生した場合に、フォールトを生成します。

フォールト・ビジネス・オブジェクト

外部サービス・ウィザードで、アダプターが生成できるフォールトごとにビジネス・オブジェクトを作成します。さらに、ウィザードでは WBIFault スーパーセット・オブジェクトを作成します。このオブジェクトには、 119 ページの図 60 に示される message、errorCode、および primarySetKey の属性など、すべてのフォールトに共通する情報が含まれます。

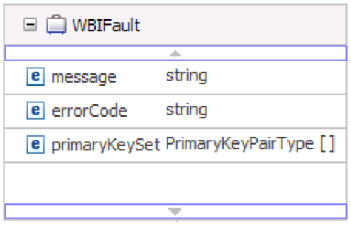


図 60. WBIFault ビジネス・オブジェクトの構造

一部のフォールトには matchCount 属性が含まれ、エラーに関する追加情報を提供します。それ以外の場合は、WBIFault に、フォールトの処理に必要なすべての情報が含まれます。

ウィザードで、以下のフォールト・ビジネス・オブジェクトを作成します。

- DuplicateRecordFault

  このフォールトは、指定したディレクトリーにファイルが既に存在する場合の Outbound Create 操作時に生成されます。

- RecordNotFoundFault

  このフォールトは、指定したディレクトリーにファイルが存在しない場合の Append、Delete、Overwrite、および Retrieve 操作時に生成されます。

- MissingDataFault

  Outbound 操作に受け渡されたビジネス・オブジェクトに、必要なすべての属性がない場合は、アダプターによりこのフォールトがスローされます。

  例えば、指定したファイルの内容が NULL であるか、ファイル名またはディレクトリー・パスが空である場合に、アダプターはこのフォールトをスローします。

- MultipleMatchingRecordsFault

  Retrieve 操作の処理時に、照会で指定されたキーのレコードが複数返された場合に、アダプターによりこのフォールトがスローされます。このフォールトのビジネス・オブジェクトには 1 つのプロパティー matchCount があり、このプロパティーは一致した数が含まれるストリングです。

フォールト処理のモジュールの構成

ビジネス・フォールトに対応するようにモジュールを構成するには、外部サービス・ウィザードを使用してモジュールを構成しておく必要があります。

フォールト処理を有効にするには、.import ファイルおよび WSDL ファイルをモジュールに合わせて変更する必要があります。バインディング・レベルまたはメソッド・レベルでフォールトを構成することができます。バインディング・レベルで変更を行った場合は、変更内容がインポートのすべてのメソッドに適用されます。メソッド・バインディング・レベルで変更を行った場合は、メソッドごとに異なるフォールトを構成することができます。

表 10 に、各フォールト名と、各フォールトのフォールト・バインディングを示します。モジュールを構成する際に、このフォールト名とフォールト・バインディング・クラスを使用してください。

表 10. 各フォールトのフォールト名とフォールト・バインディング・クラス

フォールト名	関連するフォールト・バインディング・クラス
DUPLICATE_RECORD	com.ibm.j2ca.extension.emd.runtime.WBIFaultDataBindingImpl
MULTIPLE_MATCHING_RECORDS	com.ibm.j2ca.extension.emd.runtime.MatchingFaultDataBinding
MISSING_DATA	com.ibm.j2ca.extension.emd.runtime.WBIFaultDataBindingImpl
RECORD_NOT_FOUND	com.ibm.j2ca.extension.emd.runtime.WBIFaultDataBindingImpl

1. .import ファイルを編集して、バインディング・レベルまたはメソッド・レベルでフォールトを構成します。
   - バインディング・レベルでフォールトを構成するには、以下を実行します。
     a. バインディング・セクションで、faultSelector 属性とフォールト・セレクターの名前を追加します。フォールト・セレクター名は com.ibm.j2ca.extension.emd.runtime.WBIFaultSelectorImpl です。
     b. 使用可能にするフォールトごとに、<faultBinding> エレメントを追加します。このエレメントについて、表 10 からフォールト名とフォールト・データ・バインディング・クラス名を指定します。

        以下の .import ファイルは、すべてのメソッドに構成される DUPLICATE_RECORD フォールトおよび RECORD_NOT_FOUND フォールトを示しています。**太字**は、フォールト処理を有効にするために加えられた変更を表します。

```
<esbBinding xsi:type="eis:EISImportBinding"
   dataBindingType="com.ibm.j2ca.flatfile.emd.runtime.FlatFileBaseDataBinding"
   faultSelector="com.ibm.j2ca.extension.emd.runtime.WBIFaultSelectorImpl">
    <resourceAdapter
       name="FFOutApp.IBM WebSphere Adapter for Flat Files"
       type="com.ibm.j2ca.flatfile.FlatFileResourceAdapter">
      <properties/>
    </resourceAdapter>
    <faultBinding
       fault="DUPLICATE_RECORD"
       faultBindingType="com.ibm.j2ca.extension.emd.runtime.MatchingFaultDataBinding"/>
    <faultBinding
       fault="RECORD_NOT_FOUND"
       faultBindingType="com.ibm.j2ca.extension.emd.runtime.WBIFaultDataBindingImpl"/>
```

   - メソッド・レベルでフォールトを構成するには、以下を実行します。
     a. フォールトに関連付けるメソッドのメソッド・バインディング・セクションで、フォールト・セレクターの名前を追加します。フォールト・セレクターの値は com.ibm.j2ca.extension.emd.runtime.WBIFaultSelectorImpl です。
     b. メソッド・バインディング・セクションでフォールト・バインディング・エレメントを追加します。 表 10 のフォールト名と対応するフォールト・データ・バインディング・クラス名を使用してください。

        以下の .import ファイルは、createCUSTOMER メソッドに構成される DUPLICATE_RECORD フォールトおよび RECORD_NOT_FOUND フォールトを示しています。**太字**は、フォールト処理を有効にするために加えられた変更を表します。

```
<methodBinding
   inDataBindingType="com.ibm.xmlns.prod.wbi.j2ca.flatfile.customerbg.CustomerBGDataBin
ding"
   method="createCUSTOMER"
   outDataBindingType="com.ibm.xmlns.prod.wbi.j2ca.flatfile.customerbg.CustomerBGDataBind
ing"
   faultSelector="com.ibm.j2ca.extension.emd.runtime.WBIFaultSelectorImpl">
     <interaction>
       <properties>
         <functionName>Create</functionName>
       </properties>
     </interaction>
     <faultBinding
         fault="DUPLICATE_RECORD"
         faultBindingType="com.ibm.j2ca.extension.emd.runtime.WBIFaultDataBindingImpl"/>
     <faultBinding
         fault="RECORD_NOT_FOUND"
         faultBindingType="com.ibm.j2ca.extension.emd.runtime.WBIFaultDataBindingImpl"/>
</methodBinding>
```

2. フォールトのターゲット Namespace を判別します。使用可能にするフォールトごとに、Namespace を以下のように判別します。
   a. テキスト・エディターでフォールト・スキーマ (XSD ファイル) を開きます。
   b. ターゲットの Namespace を特定します。ターゲット Namespace は、以下のフォールト・スキーマ部分に**太字**で示されています。

```
<?xml version="1.0" encoding="UTF-8" ?>
<schema xmlns="http://www.w3.org/2001/XMLSchema"
   targetNamespace="http://com/ibm/j2ca/fault/afcfault"
   xmlns:basefault="http://com/ibm/j2ca/fault">
<import namespace="http://com/ibm/j2ca/fault" schemaLocation="WBIFault.xsd"/>

. . .
```

   フォールトは、すべて同一のターゲット Namespace を持つことも、異なるターゲット Namespace を持つことも可能です。

3. WSDL ファイルを編集して、サービスのフォールトを宣言します。これらの変更部分が強調表示された WSDL ファイルの例が、リストの末尾にあります。
   a. <definitions> エレメントで、フォールト・スキーマ・ファイルから取得した情報を用いて、フォールト Namespace ごとに Namespace を追加します。すべてのフォールト・スキーマが同じ targetNamespace を持つ場合は、別名を 1 つだけ追加します。一方、さまざまな targetNamespace を持つ場合は、個々の Namespace ごとに別名を追加してください。
   b. <xsd:import> エレメントを作成して、使用可能にするフォールトごとにスキーマをインポートします。
   c. フォールト・タイプごとに import ステートメントを宣言します。ステップ 3a で定義されている正しい別名を使用して、 `type=`*`alias`*`:`*`faultBOName`*`.xsd` の複合タイプを解決していることを確認してください。
   d. フォールト・タイプごとにメッセージ・タグを宣言します。
   e. フォールト宣言を、フォールトを処理するメソッドごとに追加します。

   以下の WSDL ファイルでは、DUPLICATE_RECORD フォールトおよび RECORD_NOT_FOUND フォールトが定義されています。**太字**は、フォールト処理を有効にするために加えられた変更を表します。

ステップ 3a (121 ページ)

ステップ 3b (121 ページ)

ステップ 3c (121 ページ)

ステップ 3d (121 ページ)

```
<definitions
 xmlns="http://schemas.xmlsoap.org/wsdl/"
 xmlns:CustomerBG="http://www.ibm.com/xmlns/prod/wbi/j2ca/flatfile/customerbg"
 xmlns:intf="http://FFOut/FlatFileOutboundInterface"
 xmlns:xsd="http://www.w3.org/2001/XMLSchema"
 xmlns:fault="http://com/ibm/j2ca/fault/afcfault"
  name="FlatFileOutboundInterface.wsdl"
  targetNamespace="http://FFOut/FlatFileOutboundInterface">
  <types>
    <xsd:schema
     xmlns:tns="http://FFOut/FlatFileOutboundInterface"
     xmlns:xsd1="http://www.ibm.com/xmlns/prod/wbi/j2ca/flatfile/customerbg"
     elementFormDefault="qualified"
     targetNamespace="http://FFOut/FlatFileOutboundInterface"
     xmlns:xsd="http://www.w3.org/2001/XMLSchema">
    <xsd:import
     namespace="http://www.ibm.com/xmlns/prod/wbi/j2ca/flatfile/customerbg"
     schemaLocation="CustomerBG.xsd"/>
   <xsd:import namespace="http://com/ibm/j2ca/fault/afcfault"
        schemaLocation=" DuplicateRecordFault.xsd"/>
    <xsd:import namespace="http://com/ibm/j2ca/fault/afcfault"
        schemaLocation="RecordNotFoundFault.xsd"/>

. . .
    <xsd:element name="duplicateRecordFaultX">
        <xsd:complexType>
         <xsd:sequence>
           <xsd:element name="duplicateRecordFaultElement"
                        type="fault:DuplicateRecordFault"/>
          </xsd:sequence>
        </xsd:complexType>
      </xsd:element>

      <xsd:element name="recordNotFoundFaultX">
        <xsd:complexType>
         <xsd:sequence>
            <xsd:element name="recordNotFoundFaultElement"
                        type="fault:RecordNotFoundFault"/>
          </xsd:sequence>
        </xsd:complexType>
      </xsd:element>
    </xsd:schema>
  </types>

  . . .
<message name="duplicateRecordFault">
  <part element="intf:duplicateRecordFaultX"
        name="duplicateRecordFaultPart"/>
</message>
<message name="recordNotFoundFault">
  <part element="intf:recordNotFoundFaultX"
        name="recordNotFoundFaultPart"/>
</message>
<portType name="FlatFileOutboundInterface">

  . . .

<operation name="createCUSTOMER">
<input message="intf:createCUSTOMERRequest"
       name="createCUSTOMERRequest"/>
<output message="intf:createCUSTOMERResponse"
        name="createCUSTOMERResponse"/>
```

ステップ 3e (121 ページ)

```
<fault message="intf:duplicateRecordFault"
          name="duplicateRecordFaultFault" />
<fault message="intf:recordNotFoundFault"
       name="recordNotFoundFaultFault" />
</operation>
</portType>
</definitions>
```

XAResourceNotAvailableException

com.ibm.ws.Transaction.XAResourceNotAvailableException 例外の報告がプロセス・サーバーのログに繰り返し含まれているときは、トランザクション・ログを除去し、問題を訂正してください。

症状:

アダプターが始動すると、プロセス・サーバーのログ・ファイルに以下の例外が繰り返し記録されます。

com.ibm.ws.Transaction.XAResourceNotAvailableException

問題:

プロセス・サーバーがリソースのトランザクションをコミットまたはロールバックしている間に、そのリソースが除去されました。アダプターは、始動するとトランザクションのリカバリーを試みますが、リソースが除去されているため、それができません。

解決策:

この問題を訂正するには、以下の手順を実行します。

1. プロセス・サーバーを停止します。
2. そのトランザクションを含むトランザクション・ログ・ファイルを削除します。例外トレース内の情報を使用して、トランザクションを識別します。これにより、サーバーは、それらのトランザクションのリカバリーを試みないようになります。

   注: テスト環境または開発環境では、通常はトランザクション・ログをすべて削除できます。 WebSphere Integration Developer では、トランザクション・ログ・ディレクトリー *server_install_directory*¥profiles¥*profile_name*¥tranlog に含まれるファイルとサブディレクトリーを削除します。

   実稼働環境では、処理する必要のないイベントを表すトランザクションのみを削除します。これを行う方法の一つは、アダプターを再インストールし、使用した元のイベント・データベースをそのアダプターに参照させ、不要なトランザクションのみを削除することです。もう一つの方法は、以下のディレクトリー内の log1 ファイルまたは log2 ファイルからトランザクションを削除することです。

   server_install_directory¥profiles¥*profile_name*¥tranlog¥*node_name*¥wps¥*server_name*¥transaction¥tranlog
3. プロセス・サーバーを始動します。

org.xml.sax.SAXParseException

アダプターが XML データ・ハンドラーで構成されているときに、内容が指定されたビジネス・オブジェクト形式ではない場合は、org.xml.sax.SAXParseException 例外が生成されます。問題を訂正するには、ファイル内容がビジネス・オブジェクト構造と一致していることを確認してください。ファイルに複数のビジネス・オブジェクトが含まれている場合、区切り文字が正しく指定されていることを確認してください。

症状:

アダプターが XML データ・ハンドラーで構成されているとき、以下の例外がスローされます。

org.xml.sax.SAXParseException: 内容が後続のセクションで許可されていません (org.xml.sax.SAXParseException: Content is not allowed in trailing section)

問題:

ファイルの内容が、指定されたビジネス・オブジェクト形式ではありません。

解決策:

この問題を訂正するには、以下の手順を実行します。

1. ファイル内容がビジネス・オブジェクト構造と一致していることを確認してください。
2. 内容ファイルに複数のビジネス・オブジェクトが含まれている場合、区切り文字が正しく指定されていることを確認してください。

セルフ・ヘルプ・リソース

IBM ソフトウェア・サポートのリソースは、最新のサポート情報やテクニカル文書を入手したり、サポート・ツールやフィックスをダウンロードしたり、WebSphere Adapters の問題を回避したりするために使用することができます。また、セルフ・ヘルプ・リソースは、アダプターに関連する問題を診断したり、 IBM ソフトウェア・サポートへの連絡方法を指定するのにも役立ちます。

サポート Web サイト

WebSphere Adapters ソフトウェアのサポート Web サイト (http://www.ibm.com/software/integration/wbiadapters/support/) では、WebSphere Adapters の学習、使用、およびトラブルシューティングに役立つ多数のリソースへのリンクを提供しています。以下の種類のリソースがあります。

- フラッシュ (製品に関する警告)
- 製品のインフォメーション・センター、マニュアル、IBM Redbooks®、およびホワイト・ペーパーを含む技術情報
- 教育関連のオファリング
- テクニカル・ノート

推奨される修正

適用すべき、推奨される修正のリストは、ロケーション http://www.ibm.com/support/docview.wss?fdoc=aimadp&rs=695&uid=swg27010397 で入手できます。

テクニカル・ノート

テクニカル・ノートは、Adapter for Flat Files に関する最新の資料を提供します。以下のトピックがあります。

- 問題とそれに対する現在使用可能な解決策
- よくある質問に対する答え
- アダプターのインストール、構成、使用法、トラブルシューティングに関する手引きとなる情報
- *IBM* ソフトウェア・サポート・ハンドブック

WebSphere Adapters のテクニカル・ノートのリストが必要な場合は、以下のアドレスにアクセスしてください。

http://www.ibm.com/support/search.wss?tc=SSMKUK&rs=695&rank=8&dc=DB520+D800+D900+DA900+DA800+DB560&dtm

IBM Support Assistant のプラグイン

Adapter for Flat Files では、IBM Support Assistant のプラグインを提供します。これは、無料の保守容易性ローカル・ソフトウェア・ワークベンチです。 IBM Support Assistant のインストールおよび使用に関する情報ついては、以下のアドレスから参照してください。

http://www.ibm.com/software/support/isa/

第 8 章 参照情報

ユーザーの作業をサポートするための参照情報として、外部サービス・ウィザードによって生成されるビジネス・オブジェクトに関する詳細情報や、アダプター・プロパティーに関する情報 (双方向変換をサポートするアダプター・プロパティーなど) を提供しています。また、アダプターのメッセージや関連製品情報についても示しています。

ビジネス・オブジェクト情報

ビジネス・オブジェクトの目的は、ビジネス・オブジェクト定義ファイル内のアプリケーション固有情報と、ビジネス・オブジェクトの名前の両方を調べることによって判断できます。アプリケーション固有情報は、ローカル・ファイル・システムで実行可能な操作を示します。名前には通常、実行される操作とビジネス・オブジェクトの構造が反映されます。

ビジネス・オブジェクトの構造

Adapter for Flat Filesは、外部サービス時にビジネス・オブジェクトを定義および生成します。ビジネス・オブジェクト構造は、基本 XML スキーマとしてモデル化された、汎用 WebSphere Business Integration ビジネス・オブジェクト構造を基にしています。

汎用 FlatFileBG オブジェクト

エンタープライズ・メタデータ・ディスカバリー中には、2 つのタイプのビジネス・オブジェクト (コンテンツ固有および汎用) が生成されます。

汎用 FlatFileBG ビジネス・オブジェクトは、汎用 XSD ファイル (UnstructuredContent など) で使用されます。FlatFileBG ビジネス・オブジェクトは、FlatFile ビジネス・オブジェクトを子として含むラッパー・ビジネス・オブジェクトです。以下の図にこの関係を示します。

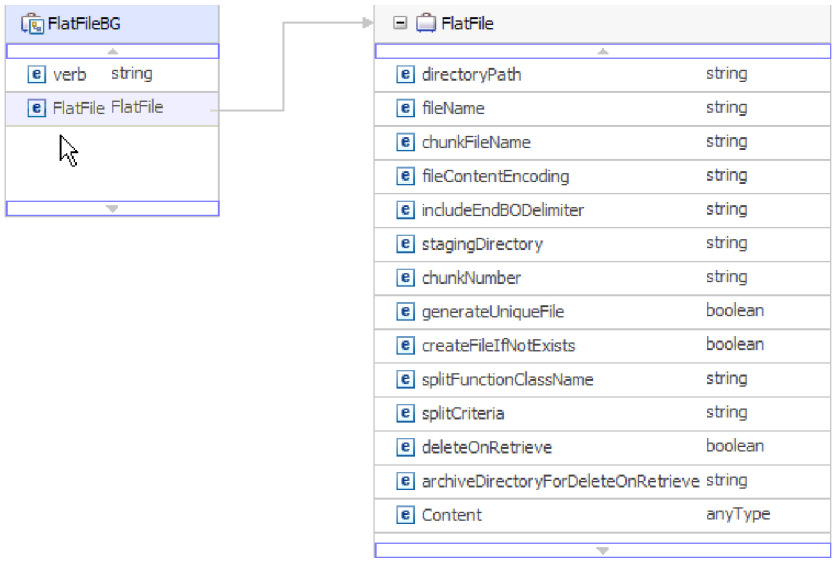


図 61. 汎用 FlatFileBG ビジネス・オブジェクト構造

CustomerWrapperBG オブジェクト

この例では、CustomerWrapperBG がコンテンツ固有の XSD ファイルを表しています。CustomerWrapperBG は、CustomerWrapper ビジネス・オブジェクトを子として含むラッパー・ビジネス・オブジェクトです。以下の図にこの関係を示します。

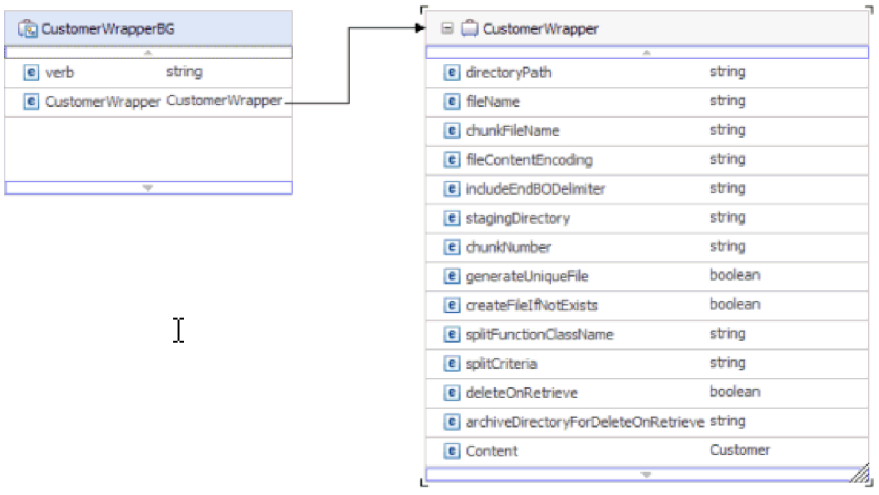


図 62. CustomerWrapperBG ビジネス・オブジェクト構造

Append 操作応答ビジネス・オブジェクト

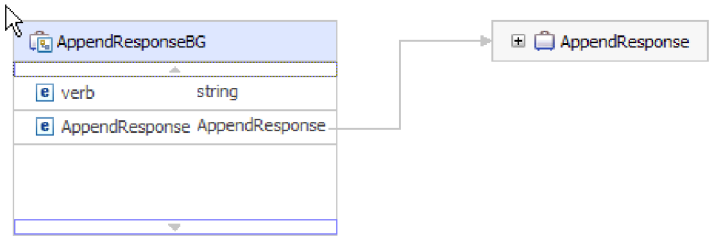


図 63. Append 操作応答ビジネス・オブジェクト構造

Create 操作応答ビジネス・オブジェクト

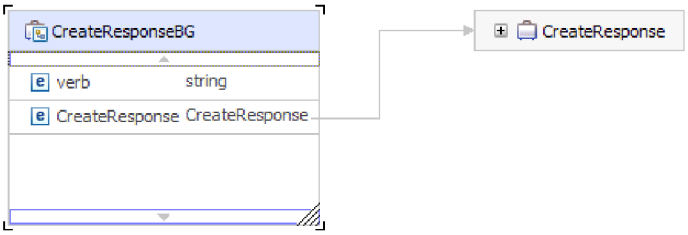


図 64. Create 操作応答ビジネス・オブジェクト構造

Exists 操作応答ビジネス・オブジェクト

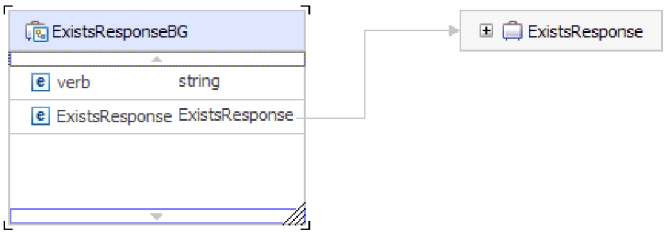


図 65. Exists 操作応答ビジネス・オブジェクト構造

List 操作応答ビジネス・オブジェクト

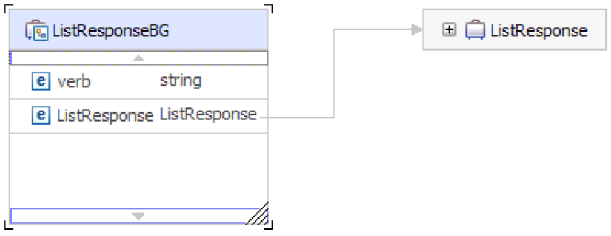


図 66. List 操作応答ビジネス・オブジェクト構造

Overwrite 操作応答ビジネス・オブジェクト

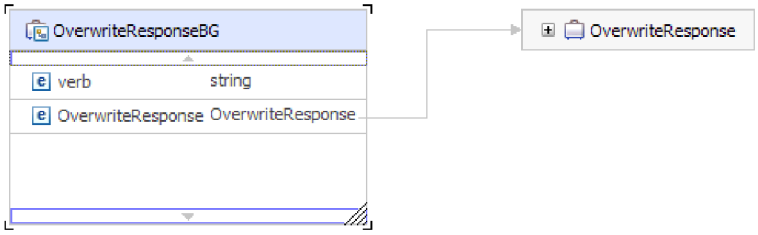


図 67. Overwrite 操作応答ビジネス・オブジェクト構造

Retrieve 操作応答ビジネス・オブジェクト

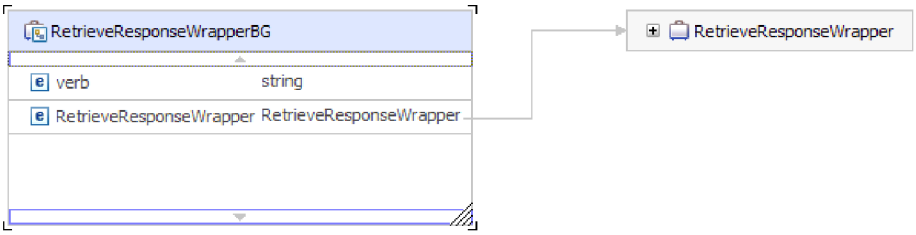


図 68. Retrieve 操作応答ビジネス・オブジェクト構造

属性プロパティー

ビジネス・オブジェクト・アーキテクチャーは、属性に適用されるさまざまなプロパティーを定義します。このセクションでは、アダプターがこれらのプロパティーを解釈する方法について説明します。

以下の表に、これらのプロパティーの説明を示します。

表 11. 属性プロパティー

属性プロパティー	説明
カーディナリティー	子ビジネス・オブジェクトまたは子ビジネス・オブジェクトの配列を表す各ビジネス・オブジェクト属性は、値 1 または複数 (n) のカーディナリティーを持ちます。単一の基数を持つフラット・ビジネス・オブジェクトだけがサポートされる。
キーおよび外部キー	これらの属性はアダプターで使用されません。
名前	属性の固有の名前を示します。
必須	この属性はアダプターで使用されません。

表 11. 属性プロパティー (続き)

属性プロパティー	説明
タイプ	属性タイプは、単純または複合のいずれかです。 単純型は、Boolean、String、LongText、Integer、Float、Double および Byte[ ] です。 標準的な複合型は、別のビジネス・オブジェクト型です。

命名規則

外部サービス・ウィザードは、ビジネス・オブジェクトを生成する場合、ビジネス・オブジェクトの作成に使用するローカル・ファイル・システムにあるオブジェクト名に基づいてビジネス・オブジェクトの名前を指定します。

外部サービス・ウィザード でビジネス・オブジェクトの名前が提供される場合、そのオブジェクト名は大/小文字混合に変換されます。これは、スペースや下線などのあらゆる分離文字が除去され、各単語の先頭文字が大文字にされることを意味します。例えば、外部サービス・ウィザードが CUSTOMER_ADDRESS という名前のローカル・ファイル・システム・オブジェクトを使用してビジネス・オブジェクトを生成する場合、CustomerAddress という名前のビジネス・オブジェクトを生成します。

生成したビジネス・オブジェクト名で、ビジネス・オブジェクトの構造を示すことができます。ただし、ビジネス・オブジェクト名には、アダプターにとって意味のある値はありません。これは、ビジネス・オブジェクト名を変更しても、ビジネス・オブジェクトの動作は変わらないことを意味します。

重要: ビジネス・オブジェクトを名前変更する場合、 WebSphere Integration Developer のリファクタリング機能を使用して、すべてのビジネス・オブジェクト依存関係を確実に更新してください。リファクタリングを使用したビジネス・オブジェクトの名前変更についての説明は、以下のリンクを参照してください。
http://publib.boulder.ibm.com/infocenter/dmndhelp/v6rxmx/topic/com.ibm.wbit.help.refactor.doc/topics/trenameboatt.html

以下の表に、外部サービス・ウィザードが Adapter for Flat Filesのビジネス・オブジェクトを生成するときに使用する命名規則の説明を示します。

表 12. 命名規則

エレメント	命名規則	例
ビジネス・グラフの名前	親ビジネス・オブジェクトを含むビジネス・グラフには、内包するビジネス・オブジェクトに従って名前が付けられ、その後に BG ストリングが続きます。ビジネス・グラフが使用できるのは、ラッパーがある場合に限ります。 CustomerWrapperBG は、CustomerWrapper ビジネス・オブジェクトを子として含むラッパー・ビジネス・オブジェクトです。	CustomerWrapperBG

注: ビジネス・グラフの生成はオプションであり、WebSphere Process Server の場合のみサポートされています。

カスタム・ファイル分割

分割論理を含むカスタム・クラスを実装することができます。アダプターは、クラス用の Java インターフェースを提供します。インターフェースの詳細を以下に示します。

```
public interface SplittingFunctionalityInterface extends Iterator{
  public int getTotalBOs(String filename) throws SplittingException;
public void setBODetails(String filename, int currentPosition, int totalBOs,
  boolean includeEndBODelimiter) throws SplittingException;
   public void setSplitCriteria(String splitCriteria);
   public void setEncoding(String encoding);
   public void setLogUtils(LogUtils logUtils);
public boolean isSplitBySize()
}
```

- `public int getTotalBOs(String filename) throws SplittingException`

  このメソッドは、`filename` に指定されるイベント・ファイルに存在するビジネス・オブジェクトの総数を戻します。

- `public void setSplitCriteria(String splitCriteria)`

  このメソッドは、イベント・ファイル内のビジネス・オブジェクトの数に基づく `splitCriteria` を使用します。各ビジネス・オブジェクトは、`next()` 呼び出しの間に戻されます。

- `public void setLogUtils(LogUtils logUtils)`

  このメソッドは、`LogUtils` オブジェクトを設定する際に使用されます。このオブジェクトは、ユーザーがファイルへのトレースの書き込みやメッセージの記録に使用できるクラスです。

- `public void setEncoding(String encoding)`

  このメソッドは、イベント・ファイル内容のエンコード方式を設定する場合に使用します。このエンコード方式は、ファイル内容を読み取る際に使用します。`SplitCriteria` の場合にもこのエンコード方式を使用します。

- `public void setBODetails(String filename, int currentPosition, int totalBOs, boolean includeEndBODelimiter) throws SplittingException`

  このメソッドは、現行のビジネス・オブジェクト数を設定する場合に使用します。このメソッドにより、`next()` 呼び出しが行われた場合は常に、`currentPosition` に設定したビジネス・オブジェクト数が戻されます。このメソッドでは、`includeEndBODelimiter` パラメーターも使用します。このパラメーターを、`true` に設定すると、ビジネス・オブジェクトの内容の最後に `SplitCriteria` が組み込まれます。このメソッドは、すべての `next()` 呼び出しの前に呼び出す必要があります。これにより、`next()` メソッドは、このメソッドに設定したビジネス・オブジェクトにビジネス・オブジェクトの内容を戻します。

- イテレーターには、`hasNext()`、`next` および `remove()` の 3 つのメソッドがあり、これらも実装する必要があります。 `next()` メソッドは、`setBODetails()` に設定したビジネス・オブジェクトの位置にビジネス・オブジェクトの内容を (byte[] として) 戻します。ビジネス・オブジェクトの位置を設定しないと、正常に実行されません。 `hasNext()` メソッドは、`setBODetails()` にビジネス・オブジェクトの位置が設定されているかどうかを示します。 `hasNext()` を呼び出す前に、`setBODetails()` メソッドを呼び出す必要があります。 `remove()` メソッドは、イベント・パーシスタンス・テーブルから削除中であるビジネス・オブジェクト・エントリーごとに呼び出されます。このメソッド内のイベント・ファイルは削除しないでください。使用中であるリソースのクリーンアップのみを行ってください。

- `public boolean isSplitBySize()`

  このメソッドは、イベント・ファイルの構文解析が、サイズまたは区切り文字のどちらに基づいて行われるかを示します。

Outbound 構成プロパティー

WebSphere Adapter for Flat Files には、オブジェクトやサービスを生成したり作成したりするときに、外部サービス・ウィザードを使用して設定する、いくつかの種類の Outbound 接続構成プロパティーがあります。 WebSphere Integration Developer または WebSphere Process Server 管理コンソールを使用してモジュールを WebSphere Process Server にデプロイした後に、リソース・アダプターおよび管理接続ファクトリーのプロパティーを変更することができますが、外部サービス・ウィザードの接続プロパティーは、デプロイメント後に変更することはできません。

プロパティーの詳細についてのガイド

WebSphere Adapter for Flat Files の構成に使用されるプロパティーは、リソース・アダプター・プロパティー、管理接続ファクトリー・プロパティーなどの各構成プロパティー・トピックに含まれている表で詳しく説明されています。これらの表を使用するのに役立つように、表示される可能性のある各行についての情報が以下で説明されます。

以下の表は、構成プロパティーについての表に表示される可能性のある各行の意味を説明しています。

行	説明
必須	アダプターが機能するためには、必須フィールド (プロパティー) に値が必要です。場合によっては、外部サービス・ウィザードによって必須プロパティーのデフォルト値が指定されることもあります。 外部サービス・ウィザードの必須フィールドからデフォルト値を除去しても、デフォルト値は変更されません。必須フィールドに値が全く含まれていない場合、外部サービス・ウィザードは、割り当てられたデフォルト値を使用してフィールドを処理し、そのデフォルト値も管理コンソールに表示されます。 指定可能な値は、**Yes** および **No** です。 場合によっては、別のプロパティーで特定の値が指定された場合にのみ、必須のプロパティーになることもあります。この場合には、表にこの従属関係が示されます。以下に例を示します。 • Yes。EventQueryType プロパティーが Dynamic に設定された場合 • Yes。Oracle データベースの場合
使用可能な値	そのプロパティーで選択可能な値をリストし、説明を示します。
デフォルト	外部サービス・ウィザードによって設定される事前定義値。そのプロパティーが必須の場合は、デフォルト値を受け入れるか、または別の値を指定する必要があります。プロパティーにデフォルト値が設定されていない場合、表には「デフォルト値なし」と表示されます。 `None` は許容されるデフォルト値であり、デフォルト値が存在しないという意味ではありません。
計測単位	プロパティーの計測方法 (キロバイトや秒など) を指定します。
プロパティー・タイプ	プロパティー・タイプについて説明します。有効なプロパティー・タイプは以下のとおりです。 • Boolean • String • Integer
使用法	プロパティーに適用される使用条件または制約事項について説明しています。例えば、制約事項は次のように説明されます。 WebSphere Web Application Server バージョン 6.40 以前では、パスワードに以下の制限があります。 • 大文字である必要があります • 長さが 8 文字である必要があります WebSphere Web Application Server バージョン 6.40 よりも後のバージョンでは、パスワードの制限が以下のように変更されました。 • 大文字小文字を区別しません • 長さが 40 文字まで可能です この項には、このプロパティーに影響を与えたり、このプロパティーによる影響を受けたりする他のプロパティーのリストと、その条件関係の性質についての説明が示されます。

行	説明
例	次のようなサンプル・プロパティー値が示されます。 「言語を JA (日本語) に設定する場合、コード・ページ番号は 8000 に設定されます。」
グローバル化	国際化される場合、プロパティーには各国語サポートがあるので、自国の言語に設定できます。 有効値は **Yes** および **No** です。
BIDI 対応	プロパティーが双方向 (bidi) 処理でサポートされているかどうかを示します。双方向処理は、1 つのファイルに右から左 (ヘブライ語やアラビア語など) と左から右 (URL やファイル・パスなど) の両方の意味構造を含むデータを処理するタスクに関係します。 有効値は **Yes** および **No** です。

ウィザードの接続プロパティー

接続プロパティーはサービス記述を作成して、組み込みの成果物を保存するために使用されます。これらのプロパティーは、外部サービス・ウィザードで構成されます。

以下の表に、外部サービス・ウィザードの接続プロパティーのリストを示します。これらの構成は、外部サービス・ウィザードを使用してのみ行うことができ、デプロイメント後には変更できません。各プロパティーの完全な説明は、表に続くセクションで説明します。後続セクションのプロパティー詳細表の見方について詳しくは、133 ページの『プロパティーの詳細についてのガイド』を参照してください。

表 13. 外部サービス・ウィザードの接続プロパティー

ウィザードのプロパティー名	説明
『BiDi フォーマット・ストリング』	コンテンツ・データの BiDi フォーマット・ストリング
136 ページの『データ・バインディング』	すべての操作で使用されるデータ・バインディングを指定するか、操作別にデータ・バインディングを選択することを指定します。
136 ページの『関数セレクター』	Inbound 処理中に使用される関数セレクター構成の名前。
137 ページの『ログ・ファイルの出力ロケーション』	外部サービス・ウィザードで生成されるログ・ファイルの絶対パス名
137 ページの『ロギング・レベル』	アダプターで使用されるロギングのレベル
138 ページの『ネーム・スペース』	生成されるビジネス・オブジェクトのネーム・スペース
138 ページの『操作名』	外部サービス・ウィザードで定義される操作
138 ページの『処理指示』	Inbound または Outbound の処理指示

BiDi フォーマット・ストリング

コンテンツ・データの BiDi フォーマット・ストリング。

表 14. BiDi フォーマット・ストリング

必須	いいえ

表 14. BiDi フォーマット・ストリング (続き)

デフォルト	なし
プロパティー・タイプ	String

データ・バインディング

すべての操作で使用されるデータ・バインディングを指定するか、操作別にデータ・バインディングを選択することを指定します。

表 15. データ・バインディングの詳細

必須	いいえ
デフォルト	すべての操作にデフォルトのデータ・バインディング FlatFileBaseDataBinding を使用する
使用法	このプロパティーの値は以下のように設定できます。 • すべての操作にデフォルトのデータ・バインディング FlatFileBaseDataBinding を使用する • すべての操作に 1 つのデータ・バインディング構成を使用 • 操作ごとにデータ・バインディングを指定
グローバル化	いいえ
BIDI 対応	いいえ

関数セレクター

Inbound 処理中に使用される関数セレクター構成の名前。

表 16. 関数セレクターの詳細

必須	はい
デフォルト	`FilenameFunctionSelector`
プロパティー・タイプ	String

表 16. 関数セレクターの詳細 (続き)

使用法	関数セレクターはサービスで呼び出される適切な操作を返します。アダプターは、FilenameFunctionSelector および EmbeddedNameFunctionSelector の 2 つの関数セレクターを提供します。 • FilenameFunctionSelector は、ファイル名の正規表現をオブジェクト名と突き合わせる、ルール・ベースの関数セレクターです。 FilenameFunctionSelector は汎用 FlatFile ビジネス・オブジェクトに使用されますが、この場合、オブジェクト名はイベント・ファイルから決定できません。 FilenameFunctionSelector は、*N* 行からなる 2 列のテーブルとしてプロパティーで表されます。 .txt 拡張子を持つイベント・ファイルの場合、対応するオブジェクト名は FlatFile になり、関数セレクターで生成されるエンドポイント・メソッド名は emitFlatFile になります。操作を追加した後、この同じ名前を EISFunctionName プロパティーに設定する必要があります。 それぞれがオブジェクト名とファイル名と突き合わせる正規表現を含む複数のルールを使用して、FilenameFunctionSelector を構成することができます。複数のルールが一致する場合、関数セレクターは最初に一致したルールに基づいてオブジェクト名を戻します。 • EmbeddedNameFunctionSelector はコンテンツ固有のビジネス・オブジェクトに使用されますが、この場合、オブジェクト名はイベント・ファイルに埋め込まれています。EmbeddedNameFunctionSelector は、ラッパーではなく、必要なコンテンツ・データに基づく関数名を戻します。例えば、コンテンツ固有のビジネス・オブジェクトが CustomerWrapperBG の場合、関数セレクターで戻される関数は emitCustomer です。 EmbeddedNameFunctionSelector はデータ・ハンドラーと一緒に構成する必要があります。データ・バインディングは、アダプター固有の WrapperDataBinding でなければならず、関数セレクターと一緒に構成されたものと同じデータ・ハンドラーを使用するように構成する必要があります。
グローバル化	はい
BIDI 対応	いいえ

ログ・ファイルの出力ロケーション

外部サービス・ウィザードで生成されるログ・ファイルの絶対パス名。

表 17. 「ログ・ファイルの出力ロケーション」の詳細

必須	いいえ
デフォルト	¥.metadata ¥FlatFileMetadataDiscoveryImpl.log
プロパティー・タイプ	String
使用法	
グローバル化	いいえ
BIDI 対応	いいえ

ロギング・レベル

アダプターで使用されるロギングのレベル。

表 18. ロギング・レベルの詳細

必須	いいえ
使用可能な値	重大 警告 監査 情報 構成 詳細
デフォルト	重大
プロパティー・タイプ	値リスト
グローバル化	いいえ
BIDI 対応	いいえ

ネーム・スペース

生成されるビジネス・オブジェクトのネーム・スペース。

表 19. 「ネーム・スペース」の詳細

必須	はい
デフォルト	http://www.ibm.com/xmlns/prod/websphere/j2ca/flatfile
プロパティー・タイプ	String
グローバル化	はい
BIDI 対応	いいえ

操作名

このモジュールに定義された操作に指定する名前。

表 20. 操作名の詳細

必須	いいえ
デフォルト	ServiceType プロパティーを `Outbound` に設定すると、操作 Create、Append、Retrieve、Delete、List、Overwrite、および Exists がリストされます。
プロパティー・タイプ	String
グローバル化	いいえ
BIDI 対応	いいえ

処理指示

`Inbound` または `Outbound` の処理指示

表 21. 処理指示の詳細

必須	はい
使用可能な値	`Outbound` `Inbound`

表 21. 処理指示の詳細 (続き)

デフォルト	Outbound
プロパティー・タイプ	String
グローバル化	いいえ
BIDI 対応	いいえ

管理接続ファクトリー・プロパティー

管理接続ファクトリー・プロパティーでは、アダプターがローカル・ファイル・システムとの Outbound 通信の実行時に必要とする情報を指定します。

以下の表は、Outbound 通信用の管理接続ファクトリー・プロパティーをリストしています。外部サービス・ウィザードを使用してアクティベーション・スペック・プロパティーを設定します。これらのプロパティーは、WebSphere Integration Developer アセンブリー・エディターを使用して変更することも、WebSphere Process Server 管理コンソールによるデプロイ後に変更することもできます。

各プロパティーの詳細については、表の後のセクションで説明します。後続セクションのプロパティー詳細表の見方について詳しくは、 133 ページの『プロパティーの詳細についてのガイド』を参照してください。

注: 外部サービス・ウィザードは、これらのプロパティーを管理接続ファクトリー・プロパティーとして参照し、WebSphere Process Server 管理コンソールは、(j2C) 接続ファクトリー・プロパティーとして参照します。

表 22. 管理接続ファクトリー・プロパティー

プロパティー名		説明
ウィザード内	**管理コンソール内**	
『デフォルト・ターゲット・ファイル名』	OutputFileName	出力ディレクトリーに作成されるファイルの名前
140 ページの『出力ディレクトリー』	OutputDirectory	Outbound 操作時にアダプターがファイルを作成するディレクトリーの絶対パス名
140 ページの『シーケンス・ファイル』	FileSequenceLog	Outbound Create 操作時にシーケンスが保管されるファイルの絶対パス名
141 ページの『ステージング・ディレクトリー』	StagingDirectory	Outbound 処理時に、アダプターが Create および Overwrite 操作の初期出力ファイルを書き込む一時ディレクトリーの絶対パス名

デフォルト・ターゲット・ファイル名

出力ディレクトリーに作成されるファイルの名前。

表 23. 「デフォルト・ターゲット・ファイル名」の詳細

必須	いいえ
デフォルト	なし
プロパティー・タイプ	String

表 23. 「デフォルト・ターゲット・ファイル名」の詳細 (続き)

使用法	OutputFileName の値がレコード・オブジェクトで指定された場合、この値はオーバーライドされます。
国際化されるかどうか	はい
BIDI 対応	はい

出力ディレクトリー

Outbound 操作時にアダプターがファイルを作成するディレクトリーの絶対パス名。

表 24. 「出力ディレクトリー」の詳細

必須	いいえ
デフォルト	なし
プロパティー・タイプ	String
使用法	出力ディレクトリーは、Create、Append、および Overwrite 操作についての最終出力ファイルを書き込むために、アダプターが使用します。
国際化されるかどうか	はい
BIDI 対応	はい

シーケンス・ファイル

このプロパティーにより、Outbound Create 操作時にシーケンスが保管されるファイルの絶対パス名を指定します。

表 25. 「シーケンス・ファイル」の詳細

必須	いいえ
デフォルト	なし
プロパティー・タイプ	String

表 25. 「シーケンス・ファイル」の詳細 (続き)

使用法	アダプターは、Create 要求を受け取るとファイル・シーケンス・ログを検査し、その名前のファイルが既に存在するかどうかを確認します。その名前のファイルが存在する場合、アダプターはファイル・シーケンス番号を使用して新規ファイル名を作成します。例えば、要求内の出力ファイル名が Customer.txt の場合、アダプターは Customer.*n*.txt (*n* はシーケンス番号) という名前のファイルを作成します。出力ファイル名に拡張子がない場合は、ファイル名の後にシーケンスが付加されます (例えば、Customer*n*)。 **注:** すべてのシーケンスは 1 から始まります。 このプロパティーが指定されていないときに、既存の名前でファイルを作成する要求を受け取ると、アダプターは `DuplicateRecordException` エラーを生成します。 特定タイプの要求のファイル順序付けを生成するには、管理接続レベルで出力ディレクトリーとファイル名を設定して、要求のたびにビジネス・オブジェクトに対してそれらを繰り返し設定する手間を省きます。 **注:** ビジネス・オブジェクトで指定されるディレクトリー・パスとファイル名は、管理接続レベルで指定される値よりも優先されます。 アダプターをクラスター環境で使用する場合は、FileSequenceLog プロパティーで指定されたシーケンス・ファイルが、すべてのクラスターからアクセス可能なマップされたドライブに存在することを確認しください。アダプターには、シーケンス・ログ・ファイルに対する書き込み許可が必要です。この許可がない場合は、IOException エラーが返されます。 FileSequenceLog プロパティーが指定されており、GenerateUniqueFile プロパティーが有効になっている場合は、GenerateUniqueFile プロパティーが FileSequenceLog プロパティーより優先されます。 シーケンス番号は、アダプターの再始動後まで増分され続けます。 ファイル・シーケンスは、シーケンス・ファイル内のシーケンス値を変更することによってリセットできます。
国際化されるかどうか	はい
BIDI 対応	はい

ステージング・ディレクトリー

Outbound 処理時の書き込み競合を防ぐため、アダプターが Create および Overwrite 操作の初期出力ファイルを一時的に保管するディレクトリーの絶対パス名。

表 26. 「ステージング・ディレクトリー」の詳細

必須	いいえ
デフォルト	なし
プロパティー・タイプ	String
使用法	このプロパティーを指定すると、出力ファイルは最初にステージング・ディレクトリーに書き込まれ、次いで名前変更されて出力ディレクトリーに書き込まれます。
国際化されるかどうか	はい
BIDI 対応	はい

リソース・アダプター・プロパティー

リソース・アダプター・プロパティーでは、ビジネス・オブジェクトのネーム・スペースの指定など、アダプターの一般的な操作を制御します。リソース・アダプターのプロパティーは、アダプターの構成時に外部サービス・ウィザードを使用して設定します。アダプターをデプロイしたあとは、管理コンソールを使用してこれらのプロパティーを変更します。

以下に示すロギングおよびトレースのプロパティーは、バージョン 6.1.0 では必要なくなりました。ただし、旧バージョンとの互換性を維持するため、管理コンソールで表示することは可能です。

- LogFileMaxSize
- LogFileName
- LogNumberOfFiles
- TraceFileMaxSize
- TraceFileName
- TraceNumberOfFiles

以下の表に、リソース・アダプター・プロパティーとその目的のリストを示します。各プロパティーの完全な説明は、表に続くセクションで説明します。後続セクションのプロパティー詳細表の見方について詳しくは、133 ページの『プロパティーの詳細についてのガイド』を参照してください。

表 27. Adapter for Flat Files 用のリソース・アダプター・プロパティー

名前		
ウィザード内	**管理コンソール内**	**説明**
アダプター ID	AdapterID	CEI イベントおよび PMI イベントのアダプター・インスタンスをロギングおよびトレースを基準にして識別します。
(なし)	HA サポートの使用可能化	このプロパティーを変更しないでください。
(なし)	ログ・ファイル最大サイズ	旧バージョンとの互換性を維持するためにサポートされています。
(なし)	LogFilename	旧バージョンとの互換性を維持するためにサポートされています。
(なし)	ログ・ファイル数	旧バージョンとの互換性を維持するためにサポートされています。
(なし)	トレース・ファイル最大サイズ	旧バージョンとの互換性を維持するためにサポートされています。
(なし)	トレース・ファイル名	旧バージョンとの互換性を維持するためにサポートされています。
(なし)	トレース・ファイル数	旧バージョンとの互換性を維持するためにサポートされています。

ロギングおよびトレースで使用するアダプター ID (AdapterID)

このプロパティーは、アダプターの特定のデプロイメント (インスタンス) を識別する場合に使用します。

表 28. 「ロギングおよびトレースで使用するアダプター ID」の詳細

必須	はい
デフォルト	CWYFF_FlatFile
プロパティー・タイプ	String
使用法	このプロパティーは、PMI イベントのアダプター・インスタンスを識別する場合に使用します。アダプターのインスタンスを複数デプロイする場合は、このプロパティーをアダプターのインスタンスごとに固有な値に設定します。 Inbound 処理の場合、このプロパティーはリソース・アダプター・プロパティーから取り出します。Outbound 処理の場合は、管理接続ファクトリー・プロパティーから取り出します。
グローバル化	はい
BIDI 対応	いいえ

高可用性サポートを使用可能にする (enableHASupport)

このプロパティーを変更しないでください。これは `true` に設定する必要があります。

対話仕様プロパティー

対話仕様プロパティーには、アダプターがファイル・システムとのインターフェースとなるときに使用する Outbound 接続プロパティーが含まれています。外部サービス・ウィザード を使用して、これらのプロパティーを構成します。アプリケーションがデプロイされた後に対話仕様プロパティーを変更するには、WebSphere Integration Developerでアセンブリー・エディターを使用します。

対話仕様プロパティーでは、操作での対話方法を制御します。対話仕様プロパティーは、アダプターを構成するときに外部サービス・ウィザードを使用して設定します。通常は、これらのプロパティーを変更する必要はありません。ただし、Outbound 操作の一部のプロパティーは、ユーザーが変更可能です。アプリケーションがデプロイされた後にこれらのプロパティーを変更するには、WebSphere Integration Developer でアセンブリー・エディターを使用します。プロパティーは、インポートのメソッド・バインディング内にあります。

以下の表に、対話仕様プロパティーのリストを示します。各プロパティーの完全な説明は、表に続くセクションで説明します。後続セクションのプロパティー詳細表の見方について詳しくは、133 ページの『プロパティーの詳細についてのガイド』を参照してください。

表 29. 対話仕様プロパティー

プロパティー名		
ウィザード内	**管理コンソール内**	**説明**
144 ページの『retrieve 操作用のアーカイブ・ディレクトリー』	ArchiveDirectoryForDeleteOnRetrieve	DeleteOnRetrieve プロパティーが `true` に設定されている場合に、取り出されたファイルが削除されるまでの間、それらを保管しておくディレクトリー
144 ページの『ファイルが存在しない場合に新規ファイルを作成する』	CreateFileIfNotExists	このプロパティーを `true` に設定すると、ファイルが存在しない場合に、アダプターは Append および Overwrite 操作時に新規ファイルを作成します。

表 29. 対話仕様プロパティー (続き)

プロパティー名		
ウィザード内	**管理コンソール内**	**説明**
145 ページの『デフォルト・ターゲット・ファイル名』	OutputFileName	作成または変更される出力ファイルの名前
145 ページの『retrieve 操作後のファイルの削除』	DeleteOnRetrieve	このプロパティーを true に設定すると、Retrieve 操作時に、ファイル内容が取り出された後にファイルがファイル・システムから削除されます
145 ページの『ファイル内のビジネス・オブジェクト間の区切り文字』	IncludeEndBODelimiter	ファイル内容は、この値と一緒に付加されます。
146 ページの『ファイル内容のエンコード』	FileContentEncoding	ファイルの書き込みで使用されるエンコード
146 ページの『固有ファイルの生成』	GenerateUniqueFile	Create、Append、および Overwrite 操作時に、アダプターが固有ファイルを作成するように指定します。
146 ページの『出力ディレクトリー』	OutputDirectory	アダプターが出力ファイルを書き込む、ローカル・ファイル・システム上のディレクトリーの絶対パス名。
147 ページの『ファイル内容を分割するための基準の指定』	SplitCriteria	取り出されたファイルのビジネス・オブジェクトを分離する区切り文字、または取り出されたファイルを分割するときのチャンク・サイズを指定します。
148 ページの『関数クラス名の分割』	SplittingFunctionClassName	Outbound Retrieve 操作時に、取り出されるファイルの分割方法 (区切り文字ごと、またはサイズごと) を指定します。
148 ページの『ステージング・ディレクトリー』	StagingDirectory	アダプターが Create および Overwrite 操作時に初期出力ファイルを保管する一時ディレクトリー

retrieve 操作用のアーカイブ・ディレクトリー

DeleteOnRetrieve プロパティーが true に設定されている場合に、取り出されたファイルが削除されるまでの間、それらを保管しておくディレクトリー。

表 30. 「retrieve 操作用のアーカイブ・ディレクトリー」の詳細

必須	いいえ
デフォルト	なし
プロパティー・タイプ	String
グローバル化	はい
BIDI 対応	はい

ファイルが存在しない場合に新規ファイルを作成する

このプロパティーを true に設定すると、ファイルが存在しない場合に、アダプターは Append および Overwrite 操作時に新規ファイルを作成します。

表 31. 「ファイルが存在しない場合に新規ファイルを作成する」の詳細

必須	いいえ

表 31. 「ファイルが存在しない場合に新規ファイルを作成する」の詳細 (続き)

使用可能な値	`True` `False`
デフォルト	`False`
プロパティー・タイプ	Boolean
使用法	このプロパティーが `false` に設定されており、ファイルが存在しない場合、アダプターは `RecordNotFoundException` エラーを生成します。
グローバル化	いいえ
BIDI 対応	いいえ

デフォルト・ターゲット・ファイル名

作成または変更される出力ファイルの名前。

表 32. 「デフォルト・ターゲット・ファイル名」の詳細

必須	List 操作を除くすべての Outbound 操作で必須
デフォルト	なし
プロパティー・タイプ	String
グローバル化	はい
BIDI 対応	はい

retrieve 操作後のファイルの削除

Retrieve 操作時に、このプロパティーが `true` に設定されると、ファイル内容が取り出された後にファイル・システムからファイルが削除されます。

表 33. 「retrieve 操作後のファイルの削除」の詳細

必須	いいえ
使用可能な値	`True` `False`
デフォルト	`False`
プロパティー・タイプ	Boolean
使用法	削除される前にファイルをアーカイブするには、ArchiveDirectoryForDeleteOnRetrieve プロパティーでディレクトリーを指定します。
グローバル化	いいえ
BIDI 対応	いいえ

ファイル内のビジネス・オブジェクト間の区切り文字

ファイル内容は、この値と一緒に付加されます。

表 34. ファイル内のビジネス・オブジェクト間の区切り文字の詳細

必須	いいえ

表 34. ファイル内のビジネス・オブジェクト間の区切り文字の詳細 (続き)

デフォルト	なし
プロパティー・タイプ	String
使用法	このプロパティーは、Outbound の Create、Append、および Overwrite 操作時に使用します。
グローバル化	はい
BIDI 対応	はい

ファイル内容のエンコード

ファイルの書き込みで使用されるエンコード。

表 35. 「ファイル内容のエンコード」の詳細

必須	いいえ
使用可能な値	すべての Java 対応エンコード・セット (UTF-8 など)。
デフォルト	なし
プロパティー・タイプ	String
使用法	このプロパティーを指定しない場合、アダプターはオペレーティング・システムの地域設定を使用します。
グローバル化	いいえ
BIDI 対応	いいえ

固有ファイルの生成

Create、Append、および Overwrite 操作時に、アダプターが固有ファイルを作成するように指定します。

表 36. 「固有ファイルの生成」の詳細

必須	いいえ
使用可能な値	`True` `False`
デフォルト	`False`
プロパティー・タイプ	Boolean
使用法	このプロパティーを `True` に設定すると、アダプターは Create 操作時に固有ファイルを作成し、ファイル名プロパティーに設定されている値をすべて無視します。このプロパティーを `True` に設定し、CreateFileIfNotExists プロパティーを `True` に設定すると、アダプターは Append および Overwrite 操作時に固有ファイルを作成します。
グローバル化	はい
BIDI 対応	いいえ

出力ディレクトリー

アダプターが出力ファイルを書き込む、ローカル・ファイル・システム上のディレクトリーの絶対パス名。

表 37. 「出力ディレクトリー」の詳細

必須	いいえ
デフォルト	なし
プロパティー・タイプ	String
使用法	このプロパティーを指定しない場合、アダプターは、要求時に OutputFileName プロパティーによって指定されたディレクトリーに出力ファイルを書き込みます。
グローバル化	はい
BIDI 対応	はい

ファイル内容を分割するための基準の指定

このプロパティーにより、取り出されたファイルのビジネス・オブジェクトを分離する区切り文字、または取り出されたファイルを分割するときのチャンク・サイズを指定します。

表 38. 「ファイル内容を分割するための基準の指定」の詳細

必須	いいえ
使用可能な値	区切り文字または有効な数値
デフォルト	`0`
プロパティー・タイプ	String
使用法	このプロパティーにより、取り出されたファイルのビジネス・オブジェクトを分離する区切り文字、または取り出されたファイルを分割するときのチャンク・サイズを指定します。このプロパティーの値は、SplittingFunctionClassName プロパティーに設定された値によって決まります。 • SplittingFunctionClassName プロパティーが `com.ibm.j2ca.utils.filesplit.SplitByDelimiter` に設定されている場合は、SplitCriteria プロパティーに、取り出されたファイル内のビジネス・オブジェクトを分離する区切り文字が含まれている必要があります。 • SplittingFunctionClassName プロパティーが `com.ibm.j2ca.utils.filesplit.SplitBySize` に設定されている場合は、SplitCriteria プロパティーに、バイト単位のサイズを表す有効な数値が含まれている必要があります。取り出されたファイルのサイズがこの値よりも大きい場合、イベント・ファイルはこの値のチャンクに分割され、そのチャンクの数が送られます。ファイルのサイズがこの値よりも小さい場合、イベント・ファイル全体が送られます。 SplitCriteria プロパティーを 0 に設定すると、チャンク化が無効になります。 SplitCriteria プロパティーには、イベント・ファイルの場合と同じ改行文字の値が含まれている必要があります。例えば、イベント・ファイルが Macintosh システムで作成された場合、改行文字は ¥r であり、SplitCriteria プロパティーには ¥r が含まれている必要があります。プラットフォーム固有の改行文字は以下のとおりです。 Macintosh - **¥r** Microsoft Windows - **¥r¥n** UNIX - **¥n** SplitCriteria プロパティーに複数の区切り文字がある場合は、それぞれの区切り文字をセミコロン (;) で分離する必要があります。セミコロン (;) 自体も区切り文字の一部になっている場合は、`¥;` のようにしてセミコロン (;) をエスケープしてください。例えば、区切り文字が `##¥;##` の場合は、`##;##` と見なされます。

表 38. 「ファイル内容を分割するための基準の指定」の詳細 (続き)

グローバル化	はい
BIDI 対応	はい

関数クラス名の分割

このプロパティーにより、Outbound Retrieve 操作時に取り出されるファイルの分割方法 (区切り文字ごと、またはサイズごと) を指定します。

表 39. 「関数クラス名の分割」の詳細

必須	いいえ
使用可能な値	com.ibm.j2ca.utils.filesplit.SplitByDelimiter – ファイルは、イベント・ファイル内のビジネス・オブジェクトを分離する区切り文字に基づいて分割されます。 com.ibm.j2ca.utils.filesplit.SplitBySize – ファイルは、イベント・ファイルのサイズに基づいて分割されます。
デフォルト	com.ibm.j2ca.utils.filesplit.SplitBySize
プロパティー・タイプ	String
使用法	区切り文字またはファイル・サイズは、SplitCriteria プロパティーで設定します。
グローバル化	いいえ
BIDI 対応	いいえ

ステージング・ディレクトリー

書き込み競合を防ぐため、アダプターが Create および Overwrite 操作時に初期出力ファイルを保管する一時ディレクトリー。

表 40. 「ステージング・ディレクトリー」の詳細

必須	いいえ
デフォルト	なし
プロパティー・タイプ	String
使用法	ステージング・ディレクトリーを指定すると、出力ディレクトリーからステージング・ディレクトリーに操作対象のファイルがコピーされます。ステージング・ディレクトリーのファイルに対して操作が実行されたあと、そのファイルは名前変更され、出力ディレクトリーにコピーされます。
グローバル化	はい
BIDI 対応	はい

Inbound 構成プロパティー

WebSphere Adapter for Flat Files には、オブジェクトやサービスを生成したり作成したりするときに、外部サービス・ウィザードを使用して設定する、いくつかの種類の Inbound 接続構成プロパティーがあります。 WebSphere Integration Developer または WebSphere Process Server 管理コンソールを使用してモジュールをデプロイ

した後に、リソース・アダプターおよびアクティベーション・スペックのプロパティーを変更することができますが、外部サービス・ウィザードの接続プロパティーは、デプロイメント後に変更することはできません。

プロパティーの詳細についてのガイド

WebSphere Adapter for Flat Files の構成に使用されるプロパティーは、リソース・アダプター・プロパティー、管理接続ファクトリー・プロパティーなどの各構成プロパティー・トピックに含まれている表で詳しく説明されています。これらの表を使用するのに役立つように、表示される可能性のある各行についての情報が以下で説明されます。

以下の表は、構成プロパティーについての表に表示される可能性のある各行の意味を説明しています。

行	説明
必須	アダプターが機能するためには、必須フィールド (プロパティー) に値が必要です。場合によっては、外部サービス・ウィザードによって必須プロパティーのデフォルト値が指定されることもあります。 外部サービス・ウィザードの必須フィールドからデフォルト値を除去しても、デフォルト値は変更されません。必須フィールドに値が全く含まれていない場合、外部サービス・ウィザードは、割り当てられたデフォルト値を使用してフィールドを処理し、そのデフォルト値も管理コンソールに表示されます。 指定可能な値は、**Yes** および **No** です。 場合によっては、別のプロパティーで特定の値が指定された場合にのみ、必須のプロパティーになることもあります。この場合には、表にこの従属関係が示されます。以下に例を示します。 • Yes。EventQueryType プロパティーが Dynamic に設定された場合 • Yes。Oracle データベースの場合
使用可能な値	そのプロパティーで選択可能な値をリストし、説明を示します。
デフォルト	外部サービス・ウィザードによって設定される事前定義値。そのプロパティーが必須の場合は、デフォルト値を受け入れるか、または別の値を指定する必要があります。プロパティーにデフォルト値が設定されていない場合、表には「デフォルト値なし」と表示されます。 None は許容されるデフォルト値であり、デフォルト値が存在しないという意味ではありません。
計測単位	プロパティーの計測方法 (キロバイトや秒など) を指定します。
プロパティー・タイプ	プロパティー・タイプについて説明します。有効なプロパティー・タイプは以下のとおりです。 • Boolean • String • Integer

行	説明
使用法	プロパティーに適用される使用条件または制約事項について説明しています。例えば、制約事項は次のように説明されます。 WebSphere Web Application Server バージョン 6.40 以前では、パスワードに以下の制限があります。 • 大文字である必要があります • 長さが 8 文字である必要があります WebSphere Web Application Server バージョン 6.40 よりも後のバージョンでは、パスワードの制限が以下のように変更されました。 • 大文字小文字を区別しません • 長さが 40 文字まで可能です この項には、このプロパティーに影響を与えたり、このプロパティーによる影響を受けたりする他のプロパティーのリストと、その条件関係の性質についての説明が示されます。
例	次のようなサンプル・プロパティー値が示されます。 「言語を JA (日本語) に設定する場合、コード・ページ番号は 8000 に設定されます。」
グローバル化	国際化される場合、プロパティーには各国語サポートがあるので、自国の言語に設定できます。 有効値は **Yes** および **No** です。
BIDI 対応	プロパティーが双方向 (bidi) 処理でサポートされているかどうかを示します。双方向処理は、1 つのファイルに右から左 (ヘブライ語やアラビア語など) と左から右 (URL やファイル・パスなど) の両方の意味構造を含むデータを処理するタスクに関係します。 有効値は **Yes** および **No** です。

ウィザードの接続プロパティー

接続プロパティーはサービス記述を作成して、組み込みの成果物を保存するために使用されます。これらのプロパティーは、外部サービス・ウィザードで構成されます。

以下の表に、外部サービス・ウィザードの接続プロパティーのリストを示します。これらの構成は、外部サービス・ウィザードを使用してのみ行うことができ、デプロイメント後には変更できません。各プロパティーの完全な説明は、表に続くセクションで説明します。後続セクションのプロパティー詳細表の見方について詳しくは、133 ページの『プロパティーの詳細についてのガイド』を参照してください。

表 41. 外部サービス・ウィザードの接続プロパティー

ウィザードのプロパティー名	説明
151 ページの『BiDi フォーマット・ストリング』	コンテンツ・データの BiDi フォーマット・ストリング
151 ページの『データ・バインディング』	すべての操作で使用されるデータ・バインディングを指定するか、操作別にデータ・バインディングを選択することを指定します。

表 41. 外部サービス・ウィザードの接続プロパティー (続き)

ウィザードのプロパティー名	説明
『関数セレクター』	Inbound 処理中に使用される関数セレクター構成の名前。
152 ページの『ログ・ファイルの出力ロケーション』	外部サービス・ウィザードで生成されるログ・ファイルの絶対パス名
152 ページの『ロギング・レベル』	アダプターで使用されるロギングのレベル
153 ページの『ネーム・スペース』	生成されるビジネス・オブジェクトのネーム・スペース
153 ページの『操作名』	外部サービス・ウィザードで定義される操作
153 ページの『処理指示』	Inbound または Outbound の処理指示

BiDi フォーマット・ストリング

コンテンツ・データの BiDi フォーマット・ストリング。

表 42. BiDi フォーマット・ストリング

必須	いいえ
デフォルト	なし
プロパティー・タイプ	String

データ・バインディング

すべての操作で使用されるデータ・バインディングを指定するか、操作別にデータ・バインディングを選択することを指定します。

表 43. データ・バインディングの詳細

必須	いいえ
デフォルト	すべての操作にデフォルトのデータ・バインディング FlatFileBaseDataBinding を使用する
使用法	このプロパティーの値は以下のように設定できます。 • すべての操作にデフォルトのデータ・バインディング FlatFileBaseDataBinding を使用する • すべての操作に 1 つのデータ・バインディング構成を使用 • 操作ごとにデータ・バインディングを指定
グローバル化	いいえ
BIDI 対応	いいえ

関数セレクター

Inbound 処理中に使用される関数セレクター構成の名前。

表 44. 関数セレクターの詳細

必須	はい
デフォルト	FilenameFunctionSelector
プロパティー・タイプ	String

表 44. 関数セレクターの詳細 (続き)

使用法	関数セレクターはサービスで呼び出される適切な操作を返します。アダプターは、FilenameFunctionSelector および EmbeddedNameFunctionSelector の 2 つの関数セレクターを提供します。 • FilenameFunctionSelector は、ファイル名の正規表現をオブジェクト名と突き合わせる、ルール・ベースの関数セレクターです。 FilenameFunctionSelector は汎用 FlatFile ビジネス・オブジェクトに使用されますが、この場合、オブジェクト名はイベント・ファイルから決定できません。 FilenameFunctionSelector は、*N* 行からなる 2 列のテーブルとしてプロパティーで表されます。 .txt 拡張子を持つイベント・ファイルの場合、対応するオブジェクト名は FlatFile になり、関数セレクターで生成されるエンドポイント・メソッド名は emitFlatFile になります。操作を追加した後、この同じ名前を EISFunctionName プロパティーに設定する必要があります。 それぞれがオブジェクト名とファイル名と突き合わせる正規表現を含む複数のルールを使用して、FilenameFunctionSelector を構成することができます。複数のルールが一致する場合、関数セレクターは最初に一致したルールに基づいてオブジェクト名を戻します。 • EmbeddedNameFunctionSelector はコンテンツ固有のビジネス・オブジェクトに使用されますが、この場合、オブジェクト名はイベント・ファイルに埋め込まれています。EmbeddedNameFunctionSelector は、ラッパーではなく、必要なコンテンツ・データに基づく関数名を戻します。例えば、コンテンツ固有のビジネス・オブジェクトが CustomerWrapperBG の場合、関数セレクターで戻される関数は emitCustomer です。 EmbeddedNameFunctionSelector はデータ・ハンドラーと一緒に構成する必要があります。データ・バインディングは、アダプター固有の WrapperDataBinding でなければならず、関数セレクターと一緒に構成されたものと同じデータ・ハンドラーを使用するように構成する必要があります。
グローバル化	はい
BIDI 対応	いいえ

ログ・ファイルの出力ロケーション

外部サービス・ウィザードで生成されるログ・ファイルの絶対パス名。

表 45. 「ログ・ファイルの出力ロケーション」の詳細

必須	いいえ
デフォルト	¥.metadata ¥FlatFileMetadataDiscoveryImpl.log
プロパティー・タイプ	String
使用法	
グローバル化	いいえ
BIDI 対応	いいえ

ロギング・レベル

アダプターで使用されるロギングのレベル。

表 46. ロギング・レベルの詳細

必須	いいえ
使用可能な値	重大 警告 監査 情報 構成 詳細
デフォルト	重大
プロパティー・タイプ	値リスト
グローバル化	いいえ
BIDI 対応	いいえ

ネーム・スペース

生成されるビジネス・オブジェクトのネーム・スペース。

表 47. 「ネーム・スペース」の詳細

必須	はい
デフォルト	http://www.ibm.com/xmlns/prod/websphere/j2ca/flatfile
プロパティー・タイプ	String
グローバル化	はい
BIDI 対応	いいえ

操作名

このモジュールに定義された操作に指定する名前。

表 48. 操作名の詳細

必須	いいえ
デフォルト	ServiceType プロパティーを `Outbound` に設定すると、操作 Create、Append、Retrieve、Delete、List、Overwrite、および Exists がリストされます。
プロパティー・タイプ	String
グローバル化	いいえ
BIDI 対応	いいえ

処理指示

`Inbound` または `Outbound` の処理指示

表 49. 処理指示の詳細

必須	はい
使用可能な値	`Outbound` `Inbound`

表 49. 処理指示の詳細 (続き)

デフォルト	Outbound
プロパティー・タイプ	String
グローバル化	いいえ
BIDI 対応	いいえ

アクティベーション・スペック・プロパティー

アクティベーション・スペック・プロパティーは、エクスポート用の Inbound イベント処理の構成情報を保持しています。アクティベーション・スペック・プロパティーは、外部サービス・ウィザードまたは管理コンソールのいずれかを使用して設定します。

以下に示すアクティベーション・スペック・プロパティーはバージョン 6.1.0 では必要なくなりましたが、旧バージョンとの互換性のためにサポートされています。

- アーカイビング・プロセス (ArchivingProcessed)
- デフォルト・オブジェクト名 (DefaultObjectName)
- イベント・コンテンツ・タイプ (EventContentType)

以下の表は、Inbound 通信のアクティベーション・スペック・プロパティーを示します。外部サービス・ウィザードを使用してアクティベーション・スペック・プロパティーを設定します。これらのプロパティーは、WebSphere Integration Developer アセンブリー・エディターを使用して変更することも、WebSphere Process Server 管理コンソールによるデプロイ後に変更することもできます。

各プロパティーの詳細については、表の後のセクションで説明します。後続セクションのプロパティー詳細表の見方について詳しくは、 133 ページの『プロパティーの詳細についてのガイド』を参照してください。

表 50. アクティベーション・スペック・プロパティー

プロパティー名		
ウィザード内	**管理コンソール内**	**説明**
156 ページの『アーカイブ・ディレクトリー』	ArchiveDirectory	アダプターが処理済みのイベント・ファイルをアーカイブするディレクトリー。
(なし)	ArchivingProcessed	旧バージョンとの互換性を維持するためにサポートされています。
156 ページの『自動作成イベント・テーブル』	EP_Create Table	イベント・パーシスタンス・テーブルを自動的に作成するか手動で作成するかを決定する。
(なし)	DefaultObjectName	旧バージョンとの互換性を維持するためにサポートされています。
送達タイプ	DeliveryType	イベントがアダプターによってエクスポートに配信される順序を指定します。
イベントを一度のみ送達する	AssuredOnceDelivery	アダプターにより、1 回のイベント送達を確保する機能が提供されるかどうかを指定します。
157 ページの『データベース・スキーマ名』	EP_SchemaName	イベント・パーシスタンス処理によって使用されるデータベースのスキーマ名。

表 50. アクティベーション・スペック・プロパティー (続き)

プロパティー名		説明
ウィザード内	管理コンソール内	
将来のタイム・スタンプを持つイベントを処理しない	FilterFutureEvents	アダプターが各イベントのタイム・スタンプをシステム時刻と比較することによって、将来のイベントをフィルターで除去するかどうかを指定します。
(なし)	EventContentType	旧バージョンとの互換性を維持するためにサポートされています。
158 ページの『イベント・ディレクトリー』	EventDirectory	イベント・ファイルが保管されるディレクトリー。
159 ページの『イベント・リカバリー DataSource (JNDI) 名』	EP_DataSource_JNDIName	JDBC データベース接続を取得するためにイベント・パーシスタンス処理で使用されるデータ・ソースの JNDI 名。データ・ソースを WebSphere Process Server に作成する必要があります。
159 ページの『イベント・リカバリー・テーブル名』	EP_TableName	イベント・パーシスタンス処理にアダプターが使用するテーブルの名前。
処理するイベント・タイプ	EventTypeFilter	どのイベントをアダプターが配信するかをアダプターに示す、区切り文字で区切られているイベント・タイプのリスト。
160 ページの『アーカイブ用の障害ファイル拡張子』	FailedArchiveExtension	処理に失敗したビジネス・オブジェクトを入力イベント・ファイルにアーカイブするのに使用するファイル拡張子。
160 ページの『ファイル内容のエンコード』	FileContentEncoding	アダプターによって読み取られるファイルのエンコード方式。
160 ページの『アーカイブ用のファイル拡張子』	OriginalArchiveExtension	オリジナル・イベント・ファイルをアーカイブするのに使用するファイル拡張子。
161 ページの『ファイル内容にビジネス・オブジェクト区切り文字を組み込む』	IncludeEndBO Delimiter	SplitCriteria プロパティーで指定された区切り文字の値が、さらに処理を行うためにビジネス・オブジェクトの内容と一緒に送信されるかどうかを指定する。
ポーリング期間の間隔	ポーリング間隔	ポーリング期間中にアダプターが待機する時間の長さ
システム接続を再試行する回数	RetryLimit	エラーが発生したあと、アダプターが Inbound 接続の再確立を試行する回数。
162 ページの『内容ではなくファイル名およびディレクトリーのみの受け渡し』	FilePassByReference	アダプターがファイルの内容をエクスポートに送達するかどうかを指定する。
163 ページの『イベント DataSource に接続するのに使用されるパスワード』	EP_Password	データ・ソースから JDBC データベース接続を取得するためにイベント・パーシスタンス処理で使用されるパスワード。
ポーリング数量	ポーリング数量	各ポーリング期間中にアダプターがエクスポートに配信するイベント数
163 ページの『ソート順でのファイルの取得』	SortEventFiles	ポーリングされたイベント・ファイルのソート順。
164 ページの『パターンを使用したファイルの取得』	EventFileMask	イベント・ファイル用のファイル・フィルター。
接続が失敗した場合の再試行間隔	RetryInterval	Inbound 操作時のエラー後、新規接続を確立しようとする試行間にアダプターが待機する時間の長さ

表 50. アクティベーション・スペック・プロパティー (続き)

プロパティー名		
ウィザード内	**管理コンソール内**	**説明**
164 ページの『ファイル内容を分割するための基準の指定』	SplitCriteria	イベント・ファイル内でビジネス・オブジェクトを分離する区切り文字か、またはイベント・ファイルの最大サイズ。「分割機能クラス名 (Splitting Function Class Name)」に設定した値によってどちらか決まります。
165 ページの『関数クラス名の分割』	SplittingFunctionClassName	イベント・ファイルの分割方法 (区切り文字またはサイズ) を指定する。
165 ページの『ポーリング時にエラーが検出された場合はアダプターを停止する (StopPollingOnError)』	StopPollingOnError	ポーリング時にアダプターがエラーを検出した場合、アダプターがイベントのポーリングを停止するかどうかを指定します。
166 ページの『アーカイブ用の成功ファイル拡張子』	SuccessArchiveExtension	処理に成功したビジネス・オブジェクトをアーカイブするのに使用するファイル拡張子。
166 ページの『イベント DataSource に接続するのに使用されるユーザー名』	EP_UserName	データ・ソースから JDBC データベース接続を取得するためにイベント・パーシスタンス処理で使用されるユーザー名。

アーカイブ・ディレクトリー

このプロパティーでは、アダプターが処理済みのイベント・ファイルをアーカイブするディレクトリーを指定します。

表 51. 「アーカイブ・ディレクトリー」の詳細

必須	いいえ
デフォルト	なし
プロパティー・タイプ	String
グローバル化	はい
BIDI 対応	はい

自動作成イベント・テーブル

このプロパティーでは、イベント・パーシスタンス・テーブルを自動的に作成するか手動で作成するかを決定します。

表 52. 「自動作成イベント・テーブル」の詳細

必須	いいえ
使用可能な値	`True` `False`
デフォルト	`True`
プロパティー・タイプ	Boolean
使用法	この値を `True` に設定すると、アダプターはイベント・パーシスタンス・テーブルを作成します。この値を `False` に設定した場合、アダプターはテーブルを作成せず、ユーザーが手動でテーブルを作成する必要があります。推奨される設定値は `True` です。

表 52. 「自動作成イベント・テーブル」の詳細 (続き)

グローバル化	いいえ

データベース・スキーマ名

このプロパティーでは、イベント・パーシスタンス処理によって使用されるデータベースのスキーマ名を指定します。

表 53. 「データベース・スキーマ名」の詳細

必須	いいえ
デフォルト	なし
プロパティー・タイプ	String
グローバル化	はい
BIDI 対応	はい

送達タイプ (DeliveryType)

このプロパティーでは、イベントがアダプターによってエクスポートに配信される順序を指定します。

表 54. 「送達タイプ」の詳細

必須	いいえ
使用可能な値	ORDERED UNORDERED
デフォルト	ORDERED
プロパティー・タイプ	String
使用法	以下の値がサポートされています。 • ORDERED: アダプターは、一度に 1 つのイベントをエクスポートに配信します。 • UNORDERED: アダプターは、一度にすべてのイベントをエクスポートに配信します。
グローバル化	いいえ
BIDI 対応	いいえ

将来のタイム・スタンプを持つイベントを処理しない (FilterFutureEvents)

このプロパティーでは、アダプターが各イベントのタイム・スタンプをシステム時刻と比較することによって、将来のイベントをフィルターで除去するかどうかを指定します。

表 55. 「将来のタイム・スタンプを持つイベントを処理しない」の詳細

必須	はい
使用可能な値	True False
デフォルト	False

表 55. 「将来のタイム・スタンプを持つイベントを処理しない」の詳細 (続き)

プロパティー・タイプ	Boolean
使用法	`True` に設定すると、アダプターは各イベントの時刻をシステム時刻と比較します。イベント時刻がシステム時刻より後の時刻である場合、そのイベントは配信されません。 `False` に設定すると、アダプターはすべてのイベントを配信します。
グローバル化	いいえ
BIDI 対応	いいえ

イベントを一度のみ送達する (AssuredOnceDelivery)

このプロパティーでは、Inbound イベントに対して、「イベントを一度のみ送達する」の機能を提供するかどうかを指定します。

表 56. 「イベントを一度のみ送達する」の詳細

必須	はい
使用可能な値	`True` `False`
デフォルト	`True`
プロパティー・タイプ	Boolean
使用法	このプロパティーを `True` に設定すると、アダプターにより、1 回のイベント送達を確保する機能が提供されます。つまり、各イベントは 1 回のみ配信されます。値を `False` にすると、1 回のイベント送達を確保する機能は提供されませんが、パフォーマンスは向上します。 このプロパティーを `True` に設定すると、アダプターにより、トランザクション (XID) 情報のイベント・ストアへの保管が試行されます。このプロパティーを `False` に設定した場合は、アダプターではこの情報の保管は行われません。 このプロパティーは、エクスポート・コンポーネントがトランザクションの対象である場合にのみ使用されます。そうでない場合は、このプロパティーの値に関係なく、トランザクションを使用することはできません。
グローバル化	いいえ
BIDI 対応	いいえ

イベント・ディレクトリー

このプロパティーでは、イベント・ファイルが保管されるローカル・ファイル・システムのディレクトリーを指定します。

表 57. 「イベント・ディレクトリー」の詳細

必須	はい
デフォルト	なし
プロパティー・タイプ	String
グローバル化	はい
BIDI 対応	はい

イベント・リカバリー DataSource (JNDI) 名

このプロパティーでは、JDBC データベース接続を取得するためにイベント・パーシスタンス処理で使用されるデータ・ソースの JNDI 名を指定します。

表 58. 「イベント・リカバリー DataSource (JNDI) 名」の詳細

必須	いいえ
デフォルト	なし
プロパティー・タイプ	String
使用法	データ・ソースを WebSphere Process Server に作成する必要があります。データベースを使用せずにイベント・ポーリングを有効化するには、この値を空のままにします。
グローバル化	はい
BIDI 対応	はい

イベント・リカバリー・テーブル名

このプロパティーでは、イベント・パーシスタンス処理にアダプターが使用するテーブルの名前を指定します。

表 59. 「イベント・リカバリー・テーブル名」の詳細

必須	いいえ
デフォルト	なし
プロパティー・タイプ	String
使用法	複数のアクティベーション・スペック・インスタンスが使用されるとき、この値はアクティベーション・スペック・インスタンスごとに固有である必要があります。
グローバル化	はい
BIDI 対応	はい

処理するイベント・タイプ (EventTypeFilter)

このプロパティーには、どのイベントをアダプターが配信するかをアダプターに示す、区切り文字で区切られているイベント・タイプのリストが入っています。

表 60. 「処理するイベント・タイプ」の詳細

必須	いいえ
使用可能な値	ビジネス・オブジェクト・タイプのコンマ (,) 区切りのリスト
デフォルト	`null`
プロパティー・タイプ	String
使用法	イベントは、ビジネス・オブジェクト・タイプ別にフィルタリングされます。このプロパティーを設定すると、アダプターは、リスト内に存在するイベントのみを配信するようになります。値が `null` の場合は、フィルターが適用されず、すべてのイベントがエクスポートに配信されることを示しています。

表 60. 「処理するイベント・タイプ」の詳細 (続き)

例	Customer ビジネス・オブジェクトおよび Order ビジネス・オブジェクトに関連するイベントのみを受信するには、次の値を指定します。 `Customer,Order`
グローバル化	いいえ
BIDI 対応	いいえ

アーカイブ用の障害ファイル拡張子

このプロパティーでは、処理に失敗したビジネス・オブジェクトを入力イベント・ファイルにアーカイブするのに使用するファイル拡張子を指定します。

表 61. 「アーカイブ用の障害ファイル拡張子」の詳細

必須	いいえ
デフォルト	`fail`
プロパティー・タイプ	String
グローバル化	はい
BIDI 対応	はい

ファイル内容のエンコード

このプロパティーでは、アダプターによって読み取られるファイルのエンコード方式を指定します。

表 62. 「ファイル内容のエンコード」の詳細

必須	いいえ
デフォルト	なし
プロパティー・タイプ	String
使用法	Java でサポートされているエンコード・セット (UTF-8 など) を指定できます。 FileContentEncoding プロパティーを指定しない場合、アダプターはデフォルトのシステム・エンコード方式を使用します。
グローバル化	いいえ
BIDI 対応	いいえ

アーカイブ用のファイル拡張子

このプロパティーでは、オリジナル・イベント・ファイルをアーカイブするのに使用するファイル拡張子を指定します。

表 63. 「アーカイブ用のファイル拡張子」の詳細

必須	いいえ
デフォルト	`original`
プロパティー・タイプ	String

表 63. 「アーカイブ用のファイル拡張子」の詳細 (続き)

使用法	このプロパティーでは、ビジネス・オブジェクトの処理が失敗した場合に備えて、イベント・ファイル全体を参照用に保持します。
グローバル化	はい
BIDI 対応	はい

ファイル内容にビジネス・オブジェクト区切り文字を組み込む

このプロパティーでは、さらに処理を行うために SplitCriteria プロパティーで指定された区切り文字の値がビジネス・オブジェクトの内容と一緒に送信されるかどうかを指定します。

表 64. 「ファイル内容にビジネス・オブジェクト区切り文字を組み込む」の詳細

必須	いいえ
使用可能な値	`True` `False`
デフォルト	`False`
プロパティー・タイプ	Boolean
使用法	このプロパティーを `true` に設定すると、さらに処理を行うために SplitCriteria プロパティーで指定された区切り文字の値がビジネス・オブジェクトの内容と一緒に送信される。このプロパティーは、イベント・ファイルの分割が区切り文字に基づく場合、つまり、SplittingFunctionClassName プロパティーが `com.ibm.j2ca.extension.utils.filesplit.SplitByDelimiter` に設定されている場合のみ有効です。 **注:** このプロパティーは、コンテンツの終了 BO 区切り文字を処理できるカスタム・データ・バインディングと一緒に使用する必要があります。これを XMLDataHandler と一緒に使用すると、データ・バインディング・レベルで障害が発生します。
グローバル化	いいえ
BIDI 対応	いいえ

ポーリング期間の間隔 (PollPeriod)

このプロパティーでは、ポーリング期間中にアダプターが待機する時間の長さを指定します。

表 65. 「ポーリング期間の間隔」の詳細

必須	はい
使用可能な値	`0` 以上の整数。
デフォルト	`2000`
計測単位	ミリ秒
プロパティー・タイプ	Integer
使用法	ポーリング期間は一定の割合で確立されます。つまり、ポーリング周期の実行が何らかの理由で遅延すると (例えば、前のポーリング周期が完了するまでに予想より時間がかかった場合)、その遅延によって失った時間を取り戻すために次のポーリング周期がすぐに開始されます。
グローバル化	いいえ

表 65. 「ポーリング期間の間隔」の詳細 (続き)

BIDI 対応	いいえ

ポーリング期間内の最大イベント数 (PollQuantity)

このプロパティーでは、各ポーリング期間中にアダプターがエクスポートに配信するイベント数を指定します。

表 66. 「ポーリング期間内の最大イベント数」の詳細

必須	はい
デフォルト	10
プロパティー・タイプ	Integer
使用法	値は 0 より大きくする必要があります。この値を大きくすると、ポーリング期間ごとに処理されるイベントの数が増加し、アダプターのパフォーマンス効率が低下する場合があります。この値を小さくすると、ポーリング期間ごとに処理されるイベントの数が減少し、アダプターのパフォーマンスが若干向上することがあります。
グローバル化	いいえ
BIDI 対応	いいえ

システム接続を再試行する回数 (RetryLimit)

このプロパティーでは、アダプターが Inbound 接続の再確立を試行する回数を指定します。

表 67. 「システム接続を再試行する回数」の詳細

必須	いいえ
使用可能な値	正整数
デフォルト	0
プロパティー・タイプ	Integer
使用法	正の値のみが有効です。 このプロパティーでは、アダプターが Inbound 接続に関連したエラーを検出した場合に、アダプターが接続を再開しようとする回数を指定します。値を 0 にすると、再試行回数は無限になります。
グローバル化	はい
BIDI 対応	いいえ

内容ではなくファイル名およびディレクトリーのみの受け渡し

表 68. 「内容ではなくファイル名およびディレクトリーのみの受け渡し」の詳細

必須	いいえ
使用可能な値	True False
デフォルト	False

表 68. 「内容ではなくファイル名およびディレクトリーのみの受け渡し」の詳細 (続き)

プロパティー・タイプ	Boolean
使用法	このプロパティーを True に設定した場合、アダプターはディレクトリー名とファイル名を送信しますが、ファイルの内容は読み込みません。イベント・ファイルにはタイム・スタンプが付加され、アーカイブ・ディレクトリーにアーカイブされます。例えば、a.txt がイベント・ファイルである場合、このファイルは a.txt.yyyy_MM_dd_HH_mm_ss_SSS としてアーカイブ・ディレクトリーにアーカイブされます。 **注:** このプロパティーは、コンテンツが設定されていない場合にランタイムで失敗しないカスタム・データ・バインディングと一緒に使用できますが、パススルー・シナリオでも使用できます。このプロパティーを XMLDataHandler と一緒に使用すると、データ・バインディング・レベルで障害が発生します。これは、XMLDataHandler が、ファイル名とディレクトリーのパスに加えてコンテンツがあることを予期するためです。
グローバル化	いいえ

イベント DataSource に接続するのに使用されるパスワード

このプロパティーでは、データ・ソースからの JDBC データベース接続を取得するためにイベント・パーシスタンス処理で使用されるパスワードを指定します。

表 69. 「イベント *DataSource* に接続するのに使用されるパスワード」の詳細

必須	いいえ
デフォルト	なし
プロパティー・タイプ	String
グローバル化	はい
BIDI 対応	はい

ソート順でのファイルの取得

このプロパティーでは、ポーリングされたイベント・ファイルのソート順を指定します。

表 70. 「ソート順でのファイルの取得」の詳細

必須	いいえ
使用可能な値	File name - ファイル名で昇順にソートする Time stamp- 最終変更日時のタイム・スタンプで昇順にソートする No sort- ソートなし
デフォルト	No sort
プロパティー・タイプ	String
使用法	グローバリゼーションをサポートする目的で、ファイル名のソート機能はシステム・ロケールに応じて提供されます。ICU4J パッケージを使用してロケールおよびロケールに対応した規則を追跡します。
グローバル化	いいえ
BIDI 対応	いいえ

パターンを使用したファイルの取得

このプロパティーでは、イベント・ファイル用のファイル・フィルターを指定します。

表 71. 「パターンを使用したファイルの取得」の詳細

必須	はい
デフォルト	*.*
プロパティー・タイプ	String
使用法	ファイル・フィルターは正しく修飾された有効な正規表現で、英数字とワイルドカード文字「*」を使用できます。 *. 例えば `event*` と指定すると、`event` で始まる名前のファイルのみが処理されます。
グローバル化	はい
BIDI 対応	はい

接続が失敗した場合の再試行間隔 (RetryInterval)

このプロパティーでは、アダプターが Inbound 接続に関連したエラーを検出した場合に、アダプターが新規接続を確立しようとするまで待機する時間の長さを指定します。

表 72. 再試行間隔の詳細

必須	はい
デフォルト	`2000`
計測単位	ミリ秒
プロパティー・タイプ	Integer
使用法	正の値のみが有効です。このプロパティーでは、アダプターが Inbound 接続に関連したエラーを検出した場合に、アダプターが新規接続を確立しようとするまで待機する時間の長さを指定します。
グローバル化	はい
BIDI 対応	いいえ

ファイル内容を分割するための基準の指定

このプロパティーでは、イベント・ファイル内のビジネス・オブジェクトを分離する区切り文字またはイベント・ファイルの最大サイズのいずれかを指定します。

表 73. 「ファイル内容を分割するための基準の指定」の詳細

必須	いいえ
デフォルト	`0`
プロパティー・タイプ	String

表 73. 「ファイル内容を分割するための基準の指定」の詳細 (続き)

使用法	このプロパティーでは、イベント・ファイル内のビジネス・オブジェクトを分離する区切り文字またはイベント・ファイルの最大サイズのいずれかを指定します。このプロパティーの値は、SplittingFunctionClassName プロパティーに設定された値によって決まります。 • SplittingFunctionClassName プロパティーを com.ibm.j2ca.extension.utils.filesplit.SplitByDelimiter に設定した場合、SplitCriteria プロパティーにはイベント・ファイル内でビジネス・オブジェクトを分離する区切り文字が含まれている必要があります。 • SplittingFunctionClassName プロパティーを com.ibm.j2ca.utils.filesplit.SplitBySize に設定した場合、SplitCriteria プロパティーには最大ファイル・サイズをバイト単位で表す有効な数字が含まれている必要があります。イベント・ファイルのサイズがこの値よりも大きい場合、イベント・ファイルはこの値のチャンクに分割され、そのチャンクの数が送られます。イベント・ファイルのサイズがこの値よりも小さい場合、イベント・ファイル全体が送られます。 SplitCriteria プロパティーの値を 0 に設定すると、ファイル分割が無効になります。 **注:** Inbound パススルー・シナリオ中に、ファイルの分割がサイズに基づく場合で、FilePassByReference プロパティーが使用可能な場合、イベント・ファイルはチャンクに分割されません。
グローバル化	はい
BIDI 対応	はい

関数クラス名の分割

このプロパティーでは、イベント・ファイルの分割方法を決定します。

表 74. 「関数クラス名の分割」の詳細

必須	いいえ
使用可能な値	com.ibm.j2ca.extension.utils.filesplit.SplitByDelimiter – ファイルは、イベント・ファイル内のビジネス・オブジェクトを分離する区切り文字に基づいて分割されます。 com.ibm.j2ca.utils.filesplit.SplitBySize – ファイルは、イベント・ファイルのサイズに基づいて分割されます。
デフォルト	com.ibm.j2ca.utils.filesplit.SplitBySize
プロパティー・タイプ	String
使用法	区切り文字またはファイル・サイズは、SplitCriteria プロパティーで設定します。 **注:** EventContentType プロパティーが NULL の場合、SplittingFunctionClassName プロパティーは com.ibm.j2ca.utils.filesplit.SplitBySize に自動的に設定されます。
グローバル化	いいえ
BIDI 対応	いいえ

ポーリング時にエラーが検出された場合はアダプターを停止する (StopPollingOnError)

このプロパティーでは、ポーリング時にアダプターがエラーを検出した場合、アダプターがイベントのポーリングを停止するかどうかを指定します。

表 75. 「ポーリング時にエラーが検出された場合はアダプターを停止する」の詳細

必須	いいえ
使用可能な値	`True` `False`
デフォルト	`False`
プロパティー・タイプ	Boolean
使用法	このプロパティーを `True` に設定した場合、アダプターはエラーを検出するとポーリングを停止します。 このプロパティーを `False` に設定した場合、アダプターはポーリング時にエラーを検出すると例外をログに記録し、ポーリングを続行します。
グローバル化	いいえ
BIDI 対応	いいえ

アーカイブ用の成功ファイル拡張子

このプロパティーでは、処理に成功したビジネス・オブジェクトをアーカイブするのに使用するファイル拡張子を指定します。

表 76. 「アーカイブ用の成功ファイル拡張子」の詳細

必須	いいえ
デフォルト	`success`
プロパティー・タイプ	String
グローバル化	はい
BIDI 対応	はい

イベント DataSource に接続するのに使用されるユーザー名

このプロパティーでは、データ・ソースからの JDBC データベース接続を取得するためにイベント・パーシスタンス処理で使用されるユーザー名を指定します。

表 77. 「イベント DataSource に接続するのに使用されるユーザー名」の詳細

必須	いいえ
デフォルト	なし
プロパティー・タイプ	String
グローバル化	はい
BIDI 対応	はい

リソース・アダプター・プロパティー

リソース・アダプター・プロパティーでは、ビジネス・オブジェクトのネーム・スペースの指定など、アダプターの一般的な操作を制御します。リソース・アダプタ

ーのプロパティーは、アダプターの構成時に外部サービス・ウィザードを使用して設定します。アダプターをデプロイしたあとは、管理コンソールを使用してこれらのプロパティーを変更します。

以下に示すロギングおよびトレースのプロパティーは、バージョン 6.1.0 では必要なくなりました。ただし、旧バージョンとの互換性を維持するため、管理コンソールで表示することは可能です。

- LogFileMaxSize
- LogFileName
- LogNumberOfFiles
- TraceFileMaxSize
- TraceFileName
- TraceNumberOfFiles

以下の表に、リソース・アダプター・プロパティーとその目的のリストを示します。各プロパティーの完全な説明は、表に続くセクションで説明します。後続セクションのプロパティー詳細表の見方について詳しくは、133 ページの『プロパティーの詳細についてのガイド』を参照してください。

表 78. Adapter for Flat Files 用のリソース・アダプター・プロパティー

名前		
ウィザード内	**管理コンソール内**	**説明**
アダプター ID	AdapterID	CEI イベントおよび PMI イベントのアダプター・インスタンスをロギングおよびトレースを基準にして識別します。
(なし)	HA サポートの使用可能化	このプロパティーを変更しないでください。
(なし)	ログ・ファイル最大サイズ	旧バージョンとの互換性を維持するためにサポートされています。
(なし)	LogFilename	旧バージョンとの互換性を維持するためにサポートされています。
(なし)	ログ・ファイル数	旧バージョンとの互換性を維持するためにサポートされています。
(なし)	トレース・ファイル最大サイズ	旧バージョンとの互換性を維持するためにサポートされています。
(なし)	トレース・ファイル名	旧バージョンとの互換性を維持するためにサポートされています。
(なし)	トレース・ファイル数	旧バージョンとの互換性を維持するためにサポートされています。

ロギングおよびトレースで使用するアダプター ID (AdapterID)

このプロパティーは、アダプターの特定のデプロイメント (インスタンス) を識別する場合に使用します。

表 79. 「ロギングおよびトレースで使用するアダプター ID」の詳細

必須	はい
デフォルト	CWYFF_FlatFile

表 79. 「ロギングおよびトレースで使用するアダプター ID」の詳細 (続き)

プロパティー・タイプ	String
使用法	このプロパティーは、PMI イベントのアダプター・インスタンスを識別する場合に使用します。アダプターのインスタンスを複数デプロイする場合は、このプロパティーをアダプターのインスタンスごとに固有な値に設定します。 Inbound 処理の場合、このプロパティーはリソース・アダプター・プロパティーから取り出します。Outbound 処理の場合は、管理接続ファクトリー・プロパティーから取り出します。
グローバル化	はい
BIDI 対応	いいえ

高可用性サポートを使用可能にする (enableHASupport)

このプロパティーを変更しないでください。これは `true` に設定する必要があります。

グローバリゼーション

WebSphere Adapter for Flat Files は、複数の言語および国/地域別環境で使用することができる、国際化されたアプリケーションです。アダプターは、文字セット・サポートおよびホスト・サーバーのロケールに基づいて、メッセージ・テキストを適切な言語で送信します。アダプターは、統合コンポーネント間の双方向スクリプト・データの変換をサポートします。

グローバリゼーションおよび双方向データ変換

アダプターは、1 バイト文字セットとマルチバイト文字セットをサポートし、メッセージ・テキストを指定された言語で配信できるようにグローバル化されています。アダプターは双方向のスクリプト・データ変換も実行します。双方向変換とは、1 つのファイルに右から左 (ヘブライ語やアラビア語など) と左から右 (URL やファイル・パスなど) の両方の意味内容を含むデータを処理するタスクのことを指します。

グローバリゼーション

グローバル化されたソフトウェア・アプリケーションは、単一環境ではなく複数の言語環境や国/地域別環境で使用することを目的として設計され、開発されています。WebSphere Adapters、 WebSphere Integration Developer、WebSphere Process Server、および WebSphere Enterprise Service Bus は、Java で作成されています。Java 仮想マシン (JVM) 内の Java ランタイム環境は、Unicode 文字コード・セットでデータを表現します。Unicode には、知られているほとんどの文字コード・セット (1 バイトとマルチバイトの両方) の文字エンコードが含まれています。そのため、これらの統合システム・コンポーネント間でデータを転送するときに文字を変換する必要はありません。

エラー・メッセージや情報メッセージを適切な言語や個々の国や地域に合った形でログに記録するために、アダプターは、稼働しているシステムのロケールを使用します。

双方向スクリプト・データ変換

アラビア語やヘブライ語などの言語は右から左に書きますが、テキストには左から右に書かれる部分も埋め込まれるため、双方向スクリプトになります。ソフトウェア・アプリケーションで双方向スクリプト・データを扱う場合は、その表示と処理のためにさまざまな規格を使用します。双方向スクリプト・データ変換の適用対象は、ストリング・タイプのデータのみです。WebSphere Process Server および WebSphere Enterprise Service Bus では、Windows の標準形式が使用されますが、サーバーとデータを交換するアプリケーションまたはファイル・システムでは、異なる形式が使用される場合があります。 アダプターは、2 つのシステム間でやり取りされる双方向スクリプト・データの変換を行うことによって、トランザクションの両側でデータが正確に処理および表示されるようにします。スクリプト・データの変換は、スクリプト・データの形式を定義する 1 組のプロパティーと、変換の適用先となるコンテンツまたはメタデータを指定するプロパティーを使用して行われます。

双方向スクリプト・データ形式

WebSphere Process Server および WebSphere Enterprise Service Bus は、ILYNN (暗黙、左から右、オン、オフ、公称) の双方向形式を使用します。これは Windows の使用する形式です。エンタープライズ情報システムが別の形式を使用する場合、アダプターは、データを WebSphere Process Server または WebSphere Enterprise Service Bus に導入する前に形式を変換します。

双方向形式は 5 つの属性から構成されます。双方向プロパティーを設定する場合は、これらの各属性に値を割り当ててください。属性と設定を以下の表に示します。

表 80. 双方向データ形式の属性と値

文字の位置	目的	値	説明	デフォルト設定
1	スキーマの配列	I または V	暗黙 (論理的) または表示	I
2	方向	L R C D	左から右 右から左 コンテキスト上の左から右 コンテキスト上の右から左	L
3	対称スワッピング	Y または N	対称スワッピングのオン/オフ	Y
4	形状の指定	S N I M F B	形状が指定されているテキスト 形状が指定されていないテキスト 初期形状指定 中間形状指定 最終形状指定 分離形状指定	N
5	数字の形状指定	H C N	ヒンディ語 コンテキスト 公称	N

変換するデータを指定する双方向プロパティー

変換が行われるビジネス・データを指定するには、BiDiContextEIS プロパティーを設定します。 この操作を行うには、このプロパティーの 5 つの双方向形式属性 (169 ページの表 80 に記載) のそれぞれに対して値を指定します。 BiDiContextEIS プロパティーは、管理接続ファクトリーおよびアクティベーション・スペックに対して設定できます。

変換が行われるイベント・パーシスタンス・データを指定するには、BiDiFormatEP プロパティーを設定します。この操作を行うには、このプロパティーの 5 つの双方向形式属性 (表 1 に記載) のそれぞれに対して値を指定します。BiDiFormatEP プロパティーは、アクティベーション・スペックに対して設定できます。

変換の対象となるアプリケーション固有のデータを指定するには、ビジネス・オブジェクト内部の BiDiContextEIS プロパティーおよび BiDiMetadata プロパティーに注釈を付けます。この操作を行うには、WebSphere Integration Developer 内部のビジネス・オブジェクト・エディターを使用して、ビジネス・オブジェクトのアプリケーション固有のエレメントとしてプロパティーを追加します。

双方向データ変換で使用可能なプロパティー

双方向データ変換プロパティーは、アプリケーションまたはファイル・システムと統合ツールおよびランタイム環境の間で交換される双方向スクリプト・データの正しい形式を強制するものです。これらのプロパティーが設定されると、双方向スクリプト・データは、WebSphere Integration Developer および WebSphere Process Server または WebSphere Enterprise Service Bus で正しく処理および表示されます。

管理接続ファクトリー・プロパティー

以下の管理接続ファクトリー・プロパティーは双方向スクリプト・データ変換を制御します。

- FileSequenceLog
- OutputDirectory
- OutputFilename
- StagingDirectory

アクティベーション・スペック・プロパティー

以下のアクティベーション・スペック・プロパティーは双方向スクリプト・データ変換を制御します。

- ArchiveDirectory
- EventDirectory
- EventFileMask
- FailedArchiveExtension
- OriginalArchiveExtension
- SplitCriteria
- SuccessArchiveExtension

デプロイメント記述子構成プロパティー

以下のデプロイメント記述子構成プロパティーは双方向スクリプト・データ変換を制御します。

- EPDatabasePassword
- EPDatabaseSchemaName
- EPDatabaseUsername
- EPDataSourceJNDIName
- EPEventTableName

ビジネス・オブジェクト・ラッパー・プロパティー

以下のビジネス・オブジェクト・ラッパー・プロパティーは双方向スクリプト・データ変換を制御します。

- DirectoryPath
- FileName
- IncludeEndBODelimiter
- StagingDirectory
- ArchiveDirectoryForDeleteOnRetrieve
- ChunkFileName

アダプター・メッセージ

以下のロケーションで WebSphere Adapter for Flat Files により発行されたメッセージを表示します。

メッセージへのリンク: http://publib.boulder.ibm.com/infocenter/dmndhelp/v6r1mx/topic/com.ibm.wbit.610.help.messages.doc/messages.html

表示される Web ページには、メッセージ接頭語のリストが示されています。メッセージ接頭語をクリックすると、その接頭語を持つすべてのメッセージを表示できます。

- 接頭語 CWYFF を持つメッセージは、WebSphere Adapter for Flat Files により発行されます。
- 接頭語 CWYBS を持つメッセージは、すべてのアダプターが使用するアダプター・ファウンデーション・クラスにより発行されます。

関連情報

以下のインフォメーション・センター、IBM Redbooks、および Web ページには、WebSphere Adapter for Flat Files の関連情報が含まれています。

サンプルとチュートリアル

WebSphere Integration Developer のオンラインのサンプル/チュートリアル・ギャラリーには、WebSphere Adapters を使用する際に役立つサンプルとチュートリアルが含まれています。以下のようにしてオンラインのサンプル/チュートリアル・ギャラリーにアクセスできます。

- WebSphere Integration Developer を始動したときに開くウェルカム・ページから。WebSphere Adapter for Flat Files のサンプルとチュートリアルを表示するには、「**取得 (Retrieve)**」 をクリックします。次に、表示されたカテゴリーを参照して選択します。
- Web 上のロケーション http://publib.boulder.ibm.com/bpcsamp/index.html から。

情報リソース

- WebSphere Business Process Management 情報リソース Web ページ http://www14.software.ibm.com/webapp/wsbroker/redirect?version=pix&product=wps-dist&topic=bpmroadmaps には、WebSphere Adapters にはついて学習するのに役立つ項目、Redbooks、文書、および教育関連のオファリングへのリンクが含まれています。
- WebSphere Adapters ライブラリー・ページ http://www.ibm.com/software/integration/wbiadapters/library/infocenter/ には、すべてのバージョンの文書へのリンクが含まれています。

関連製品の情報

- WebSphere Business Process Management、バージョン 6.1.0、インフォメーション・センター http://publib.boulder.ibm.com/infocenter/dmndhelp/v6r1mx/index.jsp。このインフォメーション・センターには、WebSphere Process Server、 WebSphere Enterprise Service Bus、および WebSphere Integration Developer 情報が含まれます。
- WebSphere Adapters、バージョン 6.0.2、インフォメーション・センター http://publib.boulder.ibm.com/infocenter/dmndhelp/v6rxmx/topic/com.ibm.wsadapters602.doc/welcome_top_wsa602.html
- WebSphere Adapters、バージョン 6.0、インフォメーション・センター http://publib.boulder.ibm.com/infocenter/wbihelp/v6rxmx/topic/com.ibm.wsadapters.doc/welcome_wsa.html
- WebSphere Business Integration Adapters インフォメーション・センター http://publib.boulder.ibm.com/infocenter/wbihelp/v6rxmx/index.jsp?topic=/com.ibm.wbi_adapters.doc/welcome_adapters.htm

developerWorks® リソース

- WebSphere Adapter Toolkit
- WebSphere Business Integration ゾーン (WebSphere business integration zone)

サポートおよび支援

- WebSphere Adapters テクニカル・サポート http://www.ibm.com/software/integration/wbiadapters/support/
- WebSphere Adapters テクニカル・ノート http://www.ibm.com/support/search.wss?tc=SSMKUK&rs=695&rank=8

&dc=DB520+D800+D900+DA900+DA800+DB560&dtm 「**Product category**」 リストで、アダプターの名前を選択し、**検索ボタン**をクリックします。

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。

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。

〒106-8711
東京都港区六本木 3-2-12
日本アイ・ビー・エム株式会社
法務・知的財産
知的財産権ライセンス渉外

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。 IBM およびその直接または間接の子会社は、本書を特定物として現存するままの状態で提供し、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。

IBM Corporation
Department 2Z4A/SOM1
294 Route 100
Somers, NY 10589-0100
U.S.A.

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。

それぞれの複製物、サンプル・プログラムのいかなる部分、またはすべての派生した創作物にも、次のように、著作権表示を入れていただく必要があります。「(c) (お客様の会社名) (西暦年)」このコードの一部は、IBM Corp. のサンプル・プログラムから取られています。 (c) Copyright IBM Corp. _年を入れる_. All rights reserved.

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は、プログラムを使用してアプリケーション・ソフトウェアを作成する際に役立ちます。

一般使用プログラミング・インターフェースにより、お客様はこのプログラム・ツール・サービスを含むアプリケーション・ソフトウェアを書くことができます。

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。

警告:

診断、修正、調整情報は、変更される場合がありますので、プログラミング・インターフェースとしては使用しないでください。

商標

IBM、IBM LOGO、developerWorks、、Redbooks、Tivoli、ViaVoice および WebSphere は、International Business Machines Corporation の米国およびその他の国における登録商標です。

Java およびすべての Java 関連の商標およびロゴは Sun Microsystems, Inc.の米国およびその他の国における商標です。

Microsoft および Windows は、Microsoft Corporation の米国およびその他の国における登録商標です。

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。

他の会社名、製品名およびサービス名等はそれぞれ各社の商標です。

この製品には、Eclipse Project (http://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。

索引

日本語, 数字, 英字, 特殊文字の順に配列されています。なお, 濁音と半濁音は清音と同等に扱われています。

[ア行]

[カ行]

［サ行］

［タ行］

［ハ行］

［マ行］

［ヤ行］

［ラ行］

A

C

D

E

Printed in Japan

日本アイ・ビー・エム株式会社
〒106-8711　東京都港区六本木3-2-12