BEA WebLogic Integration™

WebLogic Integration Studio ユーザーズ ガイド

リリース 7.0
マニュアルの日付 : 2002 年 6 月

著作権

限定的権利条項

商標または登録商標

WebLogic Integration Studio ユーザーズ ガイド

パート番号	日付	ソフトウェアのバージョン
なし	2002 年 6 月	7.0

目次

このマニュアルの内容

1. WebLogic Integration Studio ーはじめに

2. Studio インタフェースの使用法

3. データの管理

4. ワークフロー リソースのコンフィグレーション

5. ワークフロー テンプレートの定義

6. アクションの定義

7. XML エンティティを操作

8. ワークフロー式の使用法

10. ワークフローのモニタリング

11. ワークフロー パッケージのインポートとエクスポート

索引

このマニュアルの内容

このマニュアルは BEA WebLogic Integration™ Studio で行うことができる BPM (Business Process Management) 機能のユーザーズ ガイドおよびリファレンスです。Studio にワークフローを定義する実践的で詳しい手順については、『*WebLogic Integration BPM ユーザーズ ガイド*』を参照してください。

このマニュアルの内容は以下のとおりです。

- 第 1 章「WebLogic Integration Studio ―はじめに」では、WebLogic Integration ワークフロー コンポーネントの概要、内部リソースと外部リソースの統合方法、Studio 設計ワーク モデルおよびタスク、そして Studio に付属の追加ツールを説明します。
- 第 2 章「Studio インタフェースの使用法」では Studio へのログインおよびその終了方法を説明します。また Studio ユーザ インタフェースについても説明します。
- 第 3 章「データの管理」では、オーガニゼーション、ユーザ、ロール、ビジネス カレンダー、タスク ルーティングの作成および保守方法を説明します。また、セキュリティの概念と、ユーザおよびロールのパーミッションの設定方法も説明します。
- 第 4 章「ワークフロー リソースのコンフィグレーション」では、外部とカスタムのコンポーネントおよびリソースを設定してワークフロー設計者が使用できるようにする方法を説明します。内容としては、カスタム プラグイン、EJB などの Java コンポーネント、XML ベースのイベントおよび WebLogic Integration XML リポジトリなどが含まれています。
- 第 5 章「ワークフロー テンプレートの定義」では、WebLogic Integration ワークフロー テンプレート、テンプレート定義、およびワークフロー コンポーネント（ノード、接続、変数、6 つの各ノード タイプのプロパティなど）の定義方法と保守方法を説明します。さらに、ワークフロー設計のガイドラインも提供します。
- 第 6 章「アクションの定義」では、タスク、ワークフロー、統合、およびその他多数のアクションを使用してワークフロー アクティビティを実行する方法を説明します。さらに、ワークフロー設計のガイドラインも提供します。

- 第 7 章「XML エンティティを操作」では、Studio 内蔵の XML エディタと、WebLogic Integration XML リポジトリなどのさまざまなソースに含まれる XML エンティティの取り出し、格納、構成および編集方法を説明します。
- 第 8 章「ワークフロー式の使用法」では、ワークフロー式の言語および Expression Builder および XPath Wizard を使用したワークフロー式の作成方法について説明します。また、この章ではワークフロー機能および式の構文の参照リストも提供します。
- 第 9 章「ワークフロー例外の処理」では、例外ハンドラの定義および呼び出し方法と、例外処理アクションの使用法を説明します。
- 第 10 章「ワークフローのモニタリング」では、ワークフロー モニタ機能についておよび状態を表示し、実行するワークフローおよびその変数をアップデートする方法について説明します。さらに、ワークロードのレポートや統計を生成することで、実行後データを収集する方法も説明します。
- 第 11 章「ワークフロー パッケージのインポートとエクスポート」では、ワークフロー パッケージ全体を Java アーカイブ ファイルとの間でインポート / エクスポートして異なるシステム間で共有する方法を説明します。

対象読者

このマニュアルは、WebLogic Integration Studio を使用してワークフローを設計、開発および管理する統合スペシャリスト、システム管理者、業務アナリストおよびアプリケーション開発者を対象としています。

このマニュアルでは、読者が J2EE™ (Java 2 Enterprise Edition: Java 2 エンタープライズ エディション) プラットフォーム、EJB™ (Enterprise JavaBeans: エンタープライズ JavaBeans)、BEA WebLogic Server™、XML (eXtensible Markup Language: 拡張マークアップ言語)、XPath 言語、JMS (Java Message Service: Java メッセージ サービス)、および Java プログラミング言語に関する多少の知識を有していることを前提としています。

関連情報

以下の BEA WebLogic Integration ドキュメントには、BPM に関連する情報が含まれています。

- 『*WebLogic Integration BPM ユーザーズ ガイド*』
- 『*BPM ワークフローの設計のベスト プラクティス*』
- 『*WebLogic Integration Worklist ユーザーズ ガイド*』
- 『*BPM クライアント アプリケーション プログラミング ガイド*』
- 『*WebLogic Integration BPM プラグイン プログラミング ガイド*』
- 『*BEA WebLogic Integration Javadoc*』
- 『*WebLogic Integration ソリューションの設計*』

Studio を使用してアプリケーション統合を行う方法については、『*Application Integration ユーザーズ ガイド*』を参照してください。

Studio を使用して B2B 統合を行う方法については、『*B2B Integration ワークフローの作成*』を参照してください。

Studio を使用してデータ統合を行う方法については、『*Data Integration プラグイン ユーザーズ ガイド*』を参照してください。

e-docs Web サイト

BEA 製品のドキュメントは、BEA Systems, Inc. の Web サイトで入手できます。BEA のホーム ページで [製品のドキュメント] をクリックするか、または「e-docs」という製品ドキュメント ページ（http://edocs.beasys.co.jp/e-docs/index.html）を直接表示してください。

このマニュアルの印刷方法

Web ブラウザの [ファイル | 印刷] オプションを使用することにより、Web ブラウザからこのマニュアルを一度に 1 ファイルずつ印刷できます。

このマニュアルの PDF 版は、Web サイトで入手できます。WebLogic IntegrationPDF を Adobe Acrobat Reader で開くと、マニュアルの全体（または一部分）を書籍の形式で印刷できます。PDF を表示するには、WebLogic Integration ドキュメントのホーム ページを開き、[PDF 版] ボタンをクリックして、印刷するマニュアルを選択します。

Adobe Acrobat Reader がない場合は、Adobe の Web サイト（http://www.adobe.co.jp/）で無料で入手できます。

サポート情報

BEA WebLogic Integration のドキュメントに関するユーザからのフィードバックは弊社にとって非常に重要です。質問や意見などがあれば、電子メールで **docsupport-jp@bea.com** までお送りください。寄せられた意見については、WebLogic Integration のドキュメントを作成および改訂する BEA の専門の担当者が直に目を通します。

電子メールのメッセージには、ご使用の WebLogic Integration のリリースをお書き添えください。

本バージョンの BEA WebLogic Integration についてご不明な点がある場合、または BEA WebLogic Integration のインストールおよび動作に問題がある場合は、BEA WebSupport (http://websupport.bea.com/custsupp) を通じて BEA カスタマ サポートまでお問い合わせください。カスタマ サポートへの連絡方法については、製品パッケージに同梱されているカスタマ サポート カードにも記載されています。

カスタマ サポートでは以下の情報をお尋ねしますので、お問い合わせの際はあらかじめご用意ください。

- 名前、電子メール アドレス、電話番号およびファックス番号
- 会社の名前と住所
- お使いの機種とコード番号
- 製品の名前とバージョン
- 問題の状況と表示されるエラー メッセージの内容

表記規則

このマニュアルでは、全体を通して以下の表記規則が使用されています。

表記法	適用
〔Ctrl〕 + 〔Tab〕	複数のキーを同時に押すことを示す。
斜体	強調または書籍のタイトルを示す。

等幅テキスト	コード サンプル、コマンドとそのオプション、データ構造体とそのメンバー、データ型、ディレクトリ、およびファイル名とその拡張子を示す。等幅テキストはキーボードから入力するテキストも示す。
	例
	`#include <iostream.h> void main ( ) the pointer psz`
	`chmod u+w *`
	`\tux\data\ap`
	`.doc`
	`tux.doc`
	`BITMAP`
	`float`

斜体の等幅テキスト	コード内の変数を示す。 *例* `String expr`

表記法	適用
すべて大文字のテキスト	デバイス名、環境変数、および論理演算子を示す。 *例* LPT1 SIGNON OR
{ }	構文の中で複数の選択肢を示す。実際には、この括弧は入力しない。
[ ]	構文の中で任意指定の項目を示す。実際には、この括弧は入力しない。 *例* `buildobjclient [-v] [-o name ] [-f file-list]...[-l file-list]...`
|	構文の中で相互に排他的な選択肢を区切る。実際には、この記号は入力しない。
...	コマンドラインで以下のいずれかを示す。 ■ 引数を複数回繰り返すことができる。 ■ 任意指定の引数が省略されている。 ■ パラメータや値などの情報を追加入力できる。 実際には、この省略記号は入力しない。 *例* `buildobjclient [-v] [-o name ] [-f file-list]...[-l file-list]...`
. . .	コード サンプルまたは構文で項目が省略されていることを示す。 実際には、この省略記号は入力しない。

1 WebLogic Integration Studio ー はじめに

WebLogic Integration は、ビジネス プロセスを自動化するために役立つワークフロー管理システムです。ビジネス プロセスは、所期の結果をもたらすための、相互に関連する一連のビジネス アクティビティです。ビジネス プロセスのグラフィック表現が *ワークフロー*です。ワークフローは、WebLogic Integration クライアント アプリケーションである WebLogic Integration Studio を使用して定義し、モニタすることができます。

以下の節で、Studio のワークフローにかかわる概念、モデリング コンストラクト、タスクおよびワーク モデルの概要について説明します。

- WebLogic Integration における Business Process Management
- Studio の概要
- ビジネス データのモデリング
- ビジネス プロセスのモデリング
- ユーザ、アプリケーション、およびデータの統合
- ワークフローの設計アプローチとタスク
- Studio ツール

WebLogic Integration における Business Process Management

e- ビジネスの世界では、ビジネスを迅速かつ効果的に進める必要があります。このようなレベルのパフォーマンスを達成するため、多くの企業では、Business Process Management システムによるビジネス プロセスの自動化が行われていま

す。これは、ソフトウェアを使用してビジネス プロセスを定義、管理、実行するシステムです。ビジネス アクティビティの実行順は、ワークフローと呼ばれる、プロセスのコンピュータ表現によって規定されています。

ワークフローによって、プロセス内のアクティビティのシーケンスが制御され、それらが必要とする適切なリソースが呼び出され、この仕組みを通じてビジネス プロセスが自動化されます。これらのリソースは、必要なアクティビティを遂行するソフトウェア コンポーネントの場合もあれば、ビジネス プロセスが生成するメッセージに応答する人員である場合もあります。

このようなレベルのワークフロー オートメーションを達成するため、ワークフロー管理システムでは、大きく分けて以下の 3 つの分野をサポートしています。

- ワークフローの定義と実行ーワークフローの定義の取り込み、実動環境におけるワークフロー プロセスの管理と実行、および実行すべきさまざまなアクティビティの順序付け。
- データ管理ーワークフローにかかわる人員の管理、必要に応じたタスクの再ルーティング、およびワークフロー アクティビティのスケジュールを規定するビジネス カレンダーの保守。
- ワークフロー モニターワークフロー プロセスの状態の追跡およびワークフローの実行時動的コンフィグレーション。

WebLogic Integration は以上 3 つの分野すべてのワークフロー管理をサポートします。WebLogic Integration は、実行時プロセス エンジンと WebLogic Integration Studio によるワークフローの定義とモニタを使用してワークフローの実行をサポートします。

ただし、自動化ビジネス プロセスの導入によって、すべての人員にコンピュータが取って代わるわけではありません。自動化プロセスにおいても、裁量に基づく決定を下す場合、あるいは例外や問題が発生した場合の処理など、依然として人員の関与が必要な場面があります。そこで、WebLogic Integration には、エンド ユーザに割り当てられた手動タスクを表示、実行するための Worklist アプリケーションも含まれています。Worklist アプリケーションの詳細については、『*WebLogic Integration Worklist ユーザーズ ガイド*』を参照してください。付属アプリケーションを使用しない場合は、プログラマが WebLogic Integration API を使用して独自にワークリスト アプリケーションを作成することもできます。詳細については、『*BPM クライアント アプリケーション プログラミング ガイド*』を参照してください。

注意： Worklist クライアント アプリケーションは、WebLogic Integration のこのリリースでは廃止になっています。置き換えられる機能の詳細については『*BEA WebLogic Integration リリース ノート*』を参照してください。

Studio の概要

WebLogic Integration Studio は、グラフィカル ユーザ インタフェースを備えたクライアント アプリケーションで、以下に示すとおり、ワークフロー設計、ワークフロー モニタおよびデータ管理の 3 つの分野の機能が備わっています。

- データ管理機能
  - システムのビジネス ユニットとユーザのモデリング
  - ユーザに対するパーミッション レベルのコンフィグレーション
  - 指定された期間の別のユーザへのタスクの再ルーティング
  - ワークフロー実行の制御に使用するビジネス カレンダーの作成
- ワークフロー設計機能
  - グラフィカル プロセス フロー ダイアグラムの作成
  - 実行時データの保存に使用する変数の定義
  - 実行時例外を処理するプロセスの定義
  - 外部コンポーネントに対するインタフェースの定義
- ワークフロー モニタ機能
  - 実行中のワークフローの状態の表示
  - 実行中のワークフローに関連するタスクの変更（タスクの再割り当て、作業の強制やり直しなど）
  - システムのワークロード状態のグラフィック表示
  - ボトルネックや効率の悪さを発見するためのワークフロー履歴データの表示
  - ワークフローの動的保守を行うためのユーザ ワークリストやロール ワークリストの表示

ビジネス データのモデリング

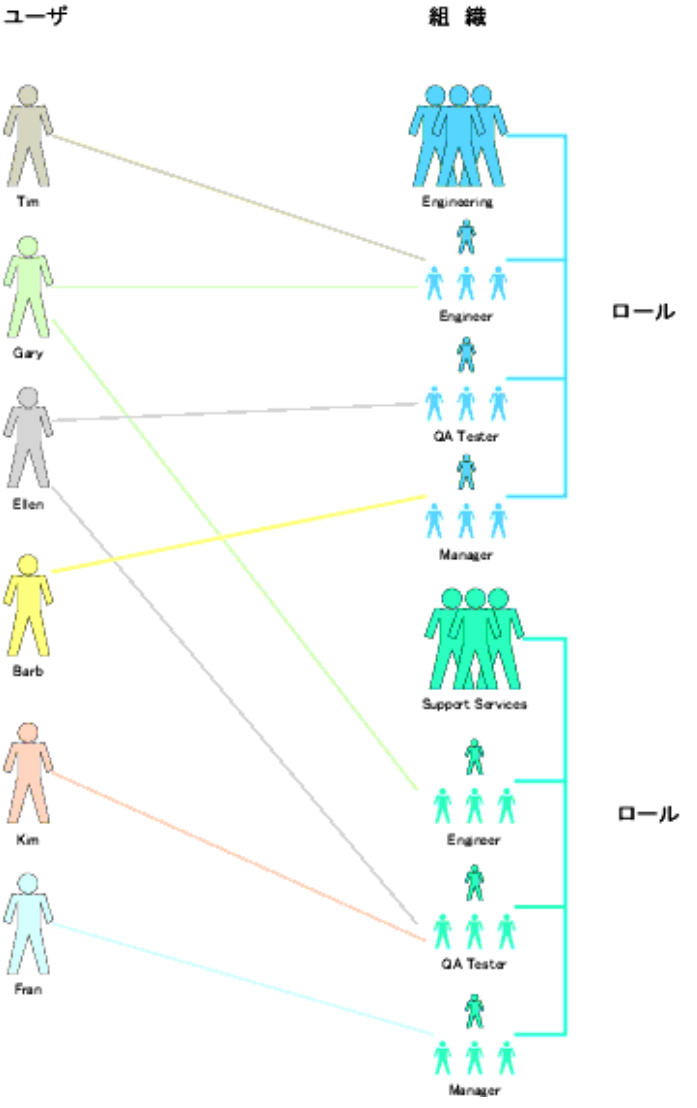
WebLogic Integration Studio では、オーガニゼーションのデータは以下の 3 つのカテゴリに分類されています。

- オーガニゼーション－1 つの企業全体、1 部門、1 地域または企業内の部署の所在地、その他その企業の特定の事業に関連するオーガニゼーション単位。
- ロール－特定のオーガニゼーションに所属する個人集団が担う、そのグループに共通の責任分担、能力、または権限レベル。
- ユーザ－ワークフローにより生成されたメッセージに応答するなどのワークフロー内の一定のタスクを実行するために必要なパーミッションを有する、一定のロールに割り当てられた個人。

ユーザをロールに割り当てることによって、ユーザのグループがオーガニゼーション内の汎用的なビジネス ロールに基づいて手動タスクを実行することが可能になります。ロールはオーガニゼーション内で一意に定義されます。これによりオーガニゼーションをオーガニゼーション単位に分け、異なるユーザのグループがアタッチされたロール名を再利用できます。

ロールはオーガニゼーション内では一意に定義されますが、次の図に示すように、ユーザは、1 つまたは複数のオーガニゼーションに所属でき、各オーガニゼーション内でも 1 つまたは複数のロールに所属できます。

図 1-1　オーガニゼーション、ユーザ、およびロールの相互関係



このサンプルでは、エンジニアリングとサポート サービスという 2 つのオーガニゼーションがあり、それぞれに、エンジニア（Engineer）、QA テスタ（QA Tester）、およびマネージャ（Manager）というロールがあります。ここに図示されている関係は、次の表にまとめられています。

表 1-1 オーガニゼーション、ロールおよびユーザ

オーガニゼーション	ロール	メンバー
エンジニアリング	エンジニア	Tim、Gary
	QA テスタ	Ellen
	マネージャ	Barbara
サポート サービス	エンジニア	Gary
	QA テスタ	Ellen、Kim
	マネージャ	Fran

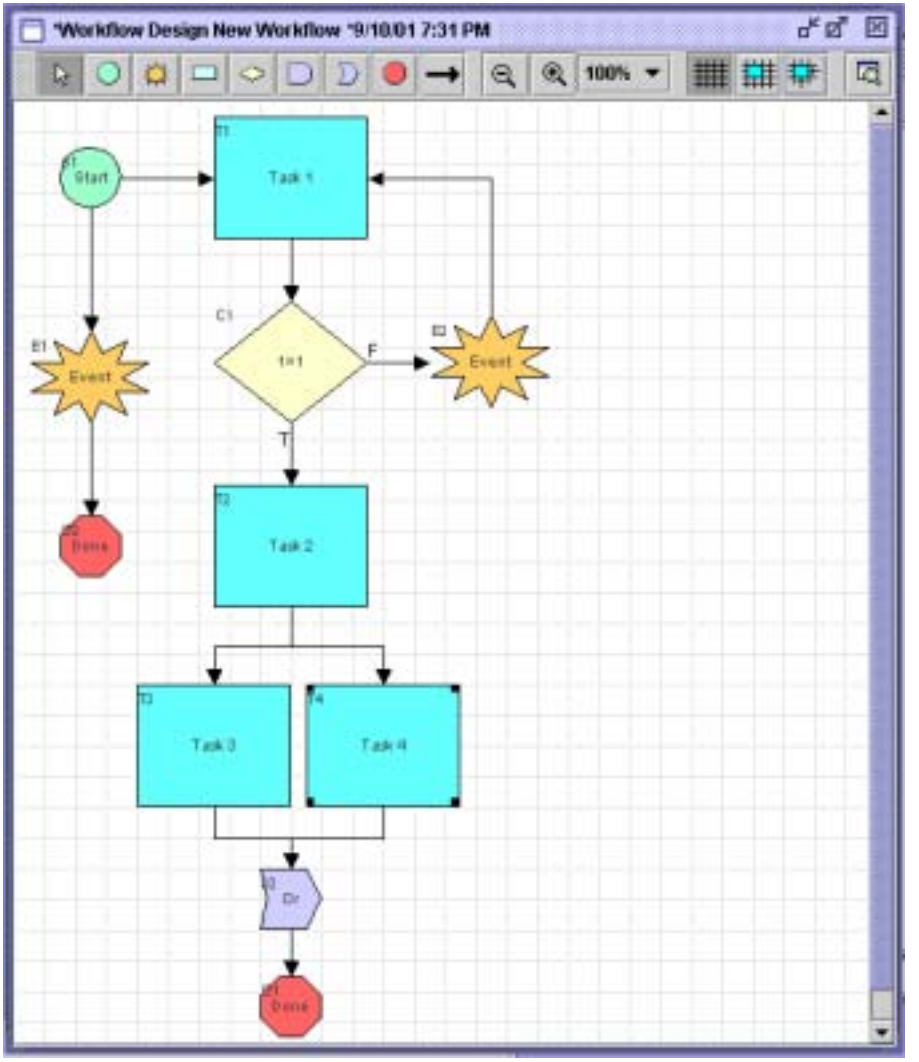
モデリングが可能なその他のビジネス データとしては、スケジュールやワークフロー ルーティングなどのビジネス ルールがあります。勤務時間およびスケジュールは、ビジネス *カレンダー*を使ってモデリングします。このビジネス カレンダーは、オーガニゼーション、ロールおよびユーザに関連付けることができます。また、*ルーティング*指定を作成して、アクティビティを一定期間、別のユーザまたはロールにリダイレクトすることもできます。

ビジネス プロセスのモデリング

Studio では、ビジネス プロセスは、モデリングおよびダイアグラム作成を経て、ワークフロー *テンプレート*としてデータベースに保存されます。これらのテンプレートは、複数のオーガニゼーションに関連付けることができ、異なるバージョンのワークフローを保存できるように、基本的には空のコンテナとなっています。テンプレートには、*テンプレート定義*が格納されていて、同じワークフローの異なるバージョンとして機能します。テンプレート定義は、各バージョンの*インスタンス化*（実行時環境への適用）が可能な期間を規定する有効日と終了日によって区別されます。

テンプレート定義では、フローの構成要素を表す図形の描画および結合によって、モデリングするビジネス プロセスを表現します。プログラム制御は、次の図に示すように、*ノード*と*コネクタ*を表すシェイプによって視覚的に表現されます。

図 1-2　Studio ワークフロー テンプレート定義 : 設計領域



さらに、テンプレート定義には、実行時に各種のアクティビティを実行する*アクション*と*例外ハンドラ*、および実行時データを収集、保存、配布するための*変数*も含まれています。テンプレート定義コンポーネントについては、以下の節でさらに詳しく説明します。

ノード

ノードは、ワークフローをグラフィックに表現するために使用する幾何学的図形です。Studio には、以下の 7 つのノードがあります。

- 開始ーワークフローの開始を表します。
- タスクーユーザが割り当てたタスクまたはある作業単位を構成するアクション（「アクション」を参照）のグループを表します。
- イベントー XML メッセージの受信によってトリガできる待機状態を表します。イベントがトリガされると、フローは先へ進みます。
- AND 結合ー複数のワークフロー パスの単一パスへの結合を表します。この結合では、結合ノードによってリンクされたすべてのワークフロー パスの処理が完了しない限り、フローが先へ進むことはありません。
- OR 結合ー複数のワークフロー パスの単一パスへの結合を表します。この結合では、結合ノードによってリンクされたワークフロー パスのうち 1 つのみ処理が完了するとフローは先へ進みます。
- 分岐ー有効または無効として評価されるべき条件（ノード内で指定される）を表します。ワークフローがたどる経路は、評価結果によって決まります。
- 完了ーワークフローの終了を表します。

アクション

ワークフローの動作はアクションによって定義されるため、アクションはある意味では、ワークフローの基本構成要素だと言えます。アクションには、タスクのユーザへの割り当てなどの単純なものもあれば、XML メッセージの送信や EJB (Enterprise JavaBean) メソッドの呼び出しなどの複雑なものもあります。アクションは、すべてのノード（結合ノードを除く）、例外ハンドラおよび他のアクションにも追加できます。

アクションには以下のように、いくつかのタイプがあります。

- [タスク] アクション―手動タスクのユーザまたはロールへの割り当て、およびタスク ノード実行の制御に使用します。
- [ワークフロー] アクション―現在のワークフローの停止やサブワークフローの開始など、ワークフロー全体を管理するために使用します。
- [統合] アクション―ワークフローと外部ソフトウェア コンポーネントや外部アプリケーションとの統合に使用します。この統合は、たとえば、実行可能プログラムの呼び出しまたは別のアプリケーションへの XML メッセージの送信によって実行できます。統合アクションによって実行される操作については、1-10 ページの「ユーザ、アプリケーション、およびデータの統合」でさらに詳しく説明します。
- [例外処理] アクション―例外ハンドラの呼び出しと終了に使用します。また、あるワークフローに対して例外ハンドラをアクティブにするためにも使用します。
- [その他] アクション―電子メールの送信、監査エントリの作成、またはワークフロー イベントの取り消しなどの追加のアクションの実行に使用します。
- [カスタム] アクション（plug-in）― WebLogic Integration プラグイン フレームワーク向けにプログラミングされるアクションで、Studio からアクセスできます。プラグイン フレームワークの詳細については、『*WebLogic Integration BPM プラグイン プログラミング ガイド*』を参照してください。

変数

変数には、実行時の値と、Java コンポーネントや着信 XML ドキュメントなど通常外部ソースから取得されるデータが格納されます。変数は、ワークフロー アクションによって定数値に設定することもできます。ワークフローで変数を使用する目的はいくつかあり、たとえば、分岐ノードでの条件の評価、テンプレート定義のラベル作成、ワークフローの実行時情報の格納などがそうです。

WebLogic Integration は、以下のタイプの変数をサポートしています。[ブール]、[日付]、[倍精度]、[エンティティ EJB]、[整数]、[Java オブジェクト]、[セッション EJB]、[文字列]、および [XML] の各型がある。日付型は、プラグイン フレームワーク用に開発されたプラグイン型を使用して拡張できます。

例外ハンドラ

例外ハンドラは、内部的または外部的に生成された例外条件の生成、トラップ、応答に使用します。例外は、ワークフローの設計時に見つけてトラップすべき異常条件である場合もあれば、トラップして適切な応答を行うべきサーバ例外である場合もあります。ワークフローには、指定すべき一連のアクションから成るカスタム定義の例外ハンドラを含めることができます。

ユーザ、アプリケーション、およびデータの統合

Studio は、WebLogic Integration の設計ツールとして、Business Process Management 以上の機能を備えていて、ビジネス サイクルにおける手動操作と自動操作を統合するために役立ちます。Studio を使用して、ファイアウォールの両側にあるクライアントと対話しながら、またデータの変換を行いながら、Web ベース アプリケーションとバックエンド アプリケーションを統合するアクションを実行できます。B2B Integration、Application Integration、Data Integration などの BPM 用プラグイン機能の詳細については、以下のドキュメントを参照してください。

- *Application Integration ユーザーズ ガイド*
- *B2B Integration ワークフローの作成*
- *Data Integration プラグイン ユーザーズ ガイド*

以下の節では、Studio の基本アクションによって可能な統合シナリオのサンプル、およびワークフローが XML メッセージング、ビジネス オペレーション、カスタム開発によるプラグインなどを介して外部コンポーネントとインタフェースを取る手段について説明します。

ユーザおよびクライアント アプリケーションを統合する

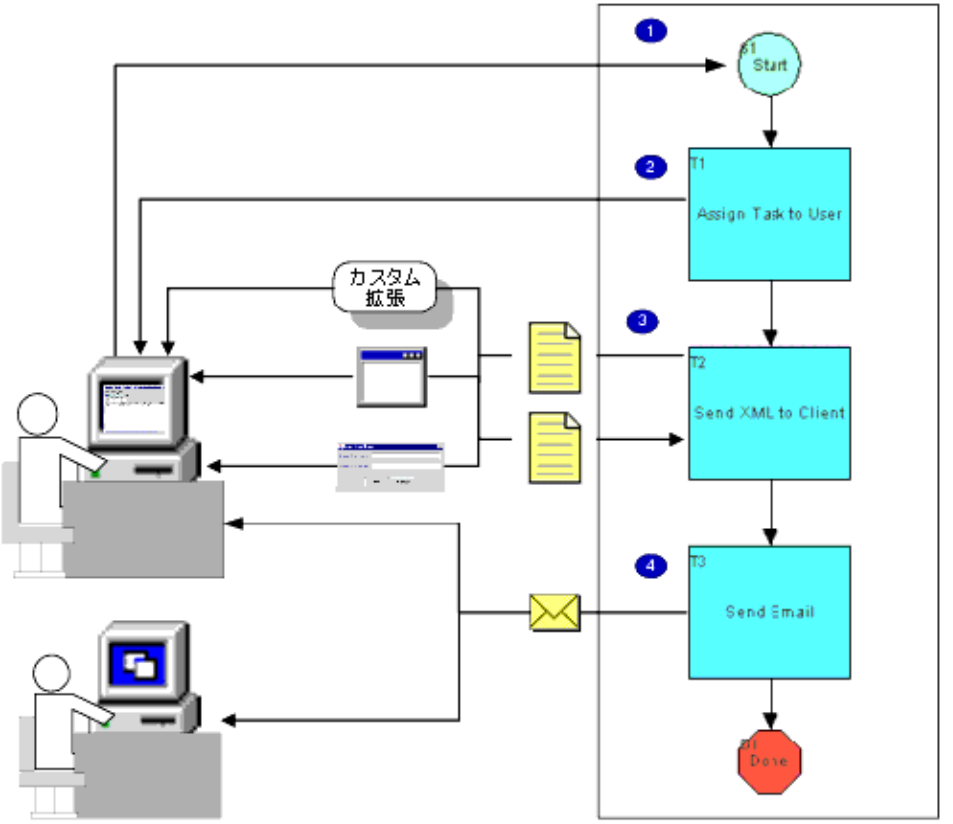
ワークフローは、Worklist アプリケーションまたはカスタムのクライアント アプリケーションを介して、以下の方法によってシステム ユーザと対話します。

- Worklist またはカスタム クライアントを通じて直接対話する方法
- Worklist またはカスタム クライアント アプリケーションとの内部 XML/JMS メッセージングによって、フォームの表示などの追加の操作を行う方法
- 電子メールによる方法

ワークフローでは、システムの外部のユーザとも電子メールで対話できます。

次の図は、ワークフロー、クライアント アプリケーションおよびユーザ間で実行されるさまざまな対話活動を示しています。図に続いて、それぞれのシナリオについて説明します。

図 1-3 ユーザおよびクライアント アプリケーションの統合



1. Worklist ユーザによってワークフローが開始されます。
2. タスクをユーザに割り当てると、Worklist ユーザにタスク実行の通知が送信されます。
3. クライアントに XML を送信すると、XML メッセージが Worklist アプリケーションに送信されて、メッセージ プロンプトまたはフォームの表示を指示したり、そのクライアント上の実行可能プログラムまたはカスタム コンポーネントを呼び出します。このメッセージに応答して、Worklist アプリケーションからワークフローに XML メッセージが返信されます。

4. 電子メール送信機能によって、WebLogic Integration システムの外部の Worklist ユーザやクライアントにプログラム エンジンから電子メールを送信することが可能になります。

外部コンポーネントおよびアプリケーションを統合する

ワークフローは、以下の種類の外部ソフトウェア コンポーネントと統合できます。

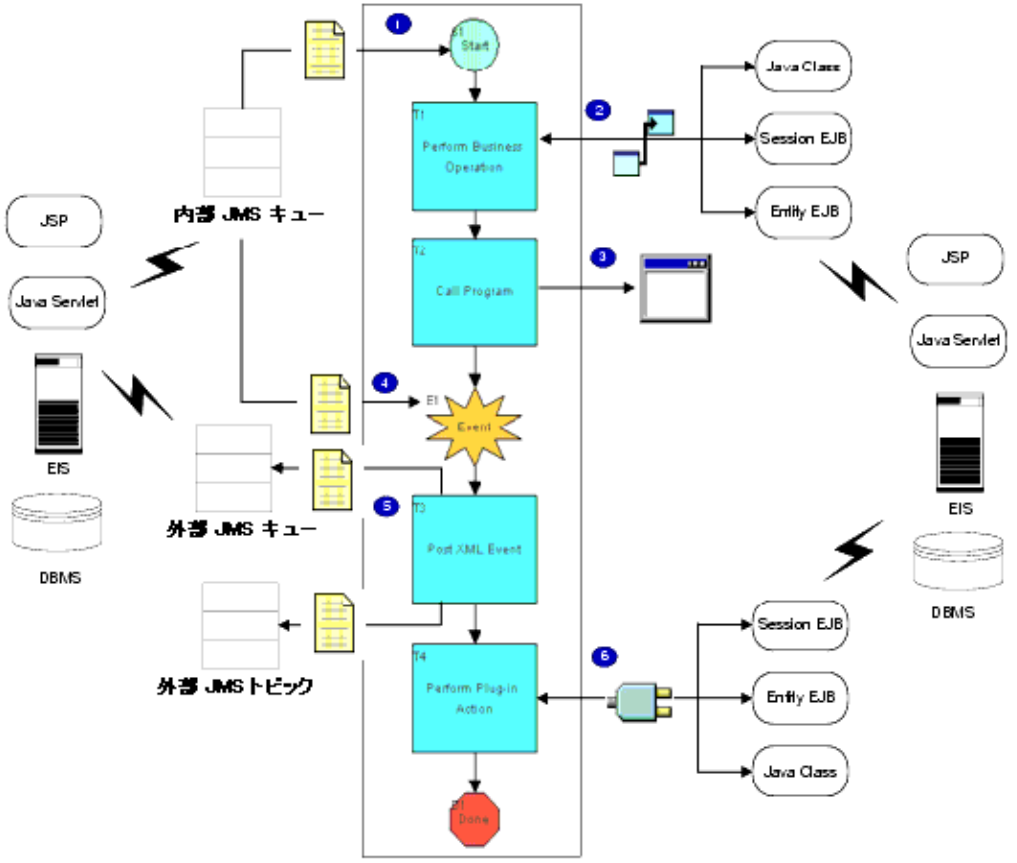
- JSP (Java Server Pages) やサーブレットなどの Web コンポーネントを含む外部システム
- 従来のシステム、パッケージ アプリケーションなどの企業情報システム
- データベース管理システム（DBMS）

これらの外部コンポーネントを統合しておくと、以下の方法によってワークフローとそれらとのデータ交換が可能になります。

- XML/JMS メッセージング
- EJB または Java のクラス メソッドによって表現するワークフローの *ビジネス オペレーション*
- プラグイン フレームワークを使用した、Java クラスおよび EJB で構成されるプラグイン コンポーネント

次の図は、ワークフローと外部コンポーネントとの対話を可能にするさまざまな手段を示しています。図に続いて、それぞれのシナリオについて説明します。

図 1-4 外部コンポーネントおよびアプリケーションの統合



1. ワークフローがトリガされ、送信元アプリケーションからの XML メッセージが内部 JMS キューで受信されることによってデータが取得されます。
2. ビジネス オペレーションを実行すると、以前からあった Java コンポーネントや、特に該当するワークフロー アプリケーション用に作成されたコンポーネントが呼び出されて、パラメータが直接ワークフローからコンポーネントに渡され、再びワークフローに戻されます。
3. プログラムを呼び出すと、サーバ上の実行可能プログラムが起動します。

4. ワークフローの進行中は、送信元アプリケーションからの XML メッセージが 内部 JMS キューで受信されることによって、イベントのトリガやデータの取得が行われます。
5. 外部 JMS トピックまたはキューに XML メッセージがポスティングされると、そのトピックにサブスクライブする、またはそのキューからメッセージを受信する外部アプリケーションに、通知やデータが送信されます。
6. プラグイン アクションが実行されると、他のアプリケーションやシステムを統合するために記述されたカスタムの Java コードが呼び出されます。

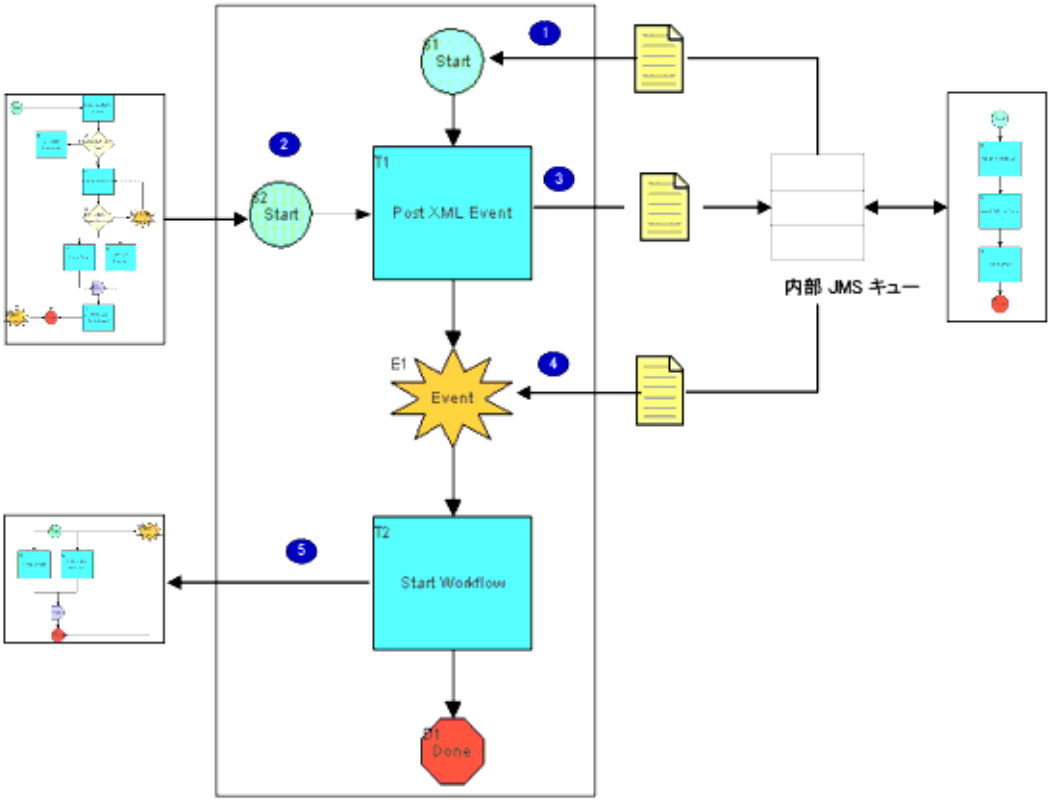
ワークフローを統合する

以下の方法で、ワークフロー間の交信を行うことができます。

- 相互に直接呼び出す方法
- XML/JMS メッセージングを介する方法

次の図は、ワークフロー間で行われるさまざまな対話を示しています。図に続いて、それぞれのシナリオについて説明します。

図 1-5 ワークフローの統合



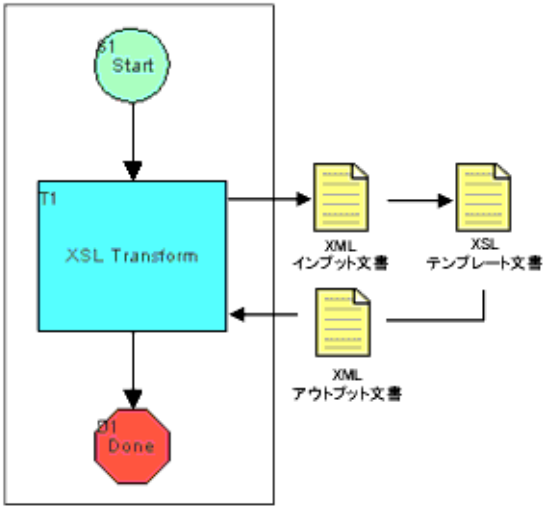
1. ワークフローがトリガされ、別の送信元ワークフローからの XML メッセージが内部 JMS キューで受信されることによってデータが取得されます。
2. ワークフローは、呼び出された開始ノードによって設定され、直接別のワークフローによって開始されます。これらのワークフローは直接パラメータをやりとりします。
3. 内部 JMS キューに XML メッセージがポスティングされると、別のワークフローに通知やデータが送信されて、その開始ノードまたはその中のイベントがトリガされます（1 の場合と同様）。

4. ワークフローの進行中は、別の、送信元ワークフローからの XML メッセージが 内部 JMS キューで受信されることによって、イベントのトリガやデータの取得が行われます（1 の場合と同様）。
5. ワークフローを開始すると、呼び出されたワークフローが開始され、直接そのワークフローとパラメータのやりとりが行われます（2 の場合と同様）。

データを統合する

XML メッセージの交換に加えて、ワークフローによって、XML ドキュメントをあるフォーマットから別のフォーマットに変換して、外部アプリケーションに渡せるようにすることができます。

図 1-6　データの統合

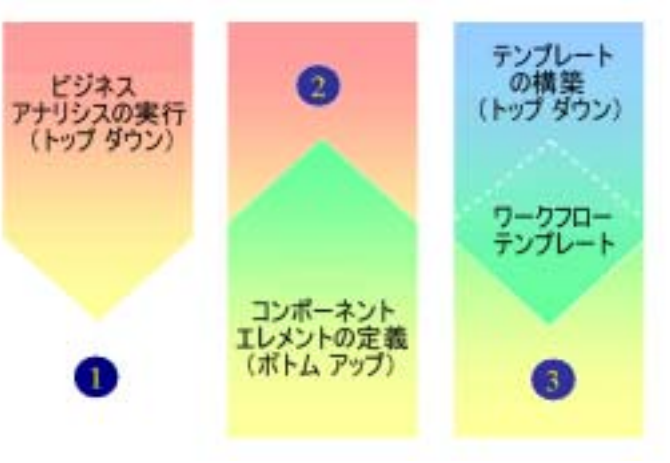


このシナリオでは、ワークフローで XSL (Extensible Stylesheet Language: 拡張スタイルシート言語) テンプレートを使用して、XML ドキュメントの他の構造への変換が行われます。

ワークフローの設計アプローチとタスク

システム設計理論では、理想的なシステム設計はトップダウンのアプローチであるとされています。現実には、システム設計にはトップダウン、ボトムアップの両方のアプローチが採用されています。次の図に示すように、WebLogic Integration ワークフローの設計にも、この 2 つのアプローチが使われています。

図 1-7 ワークフローの設計



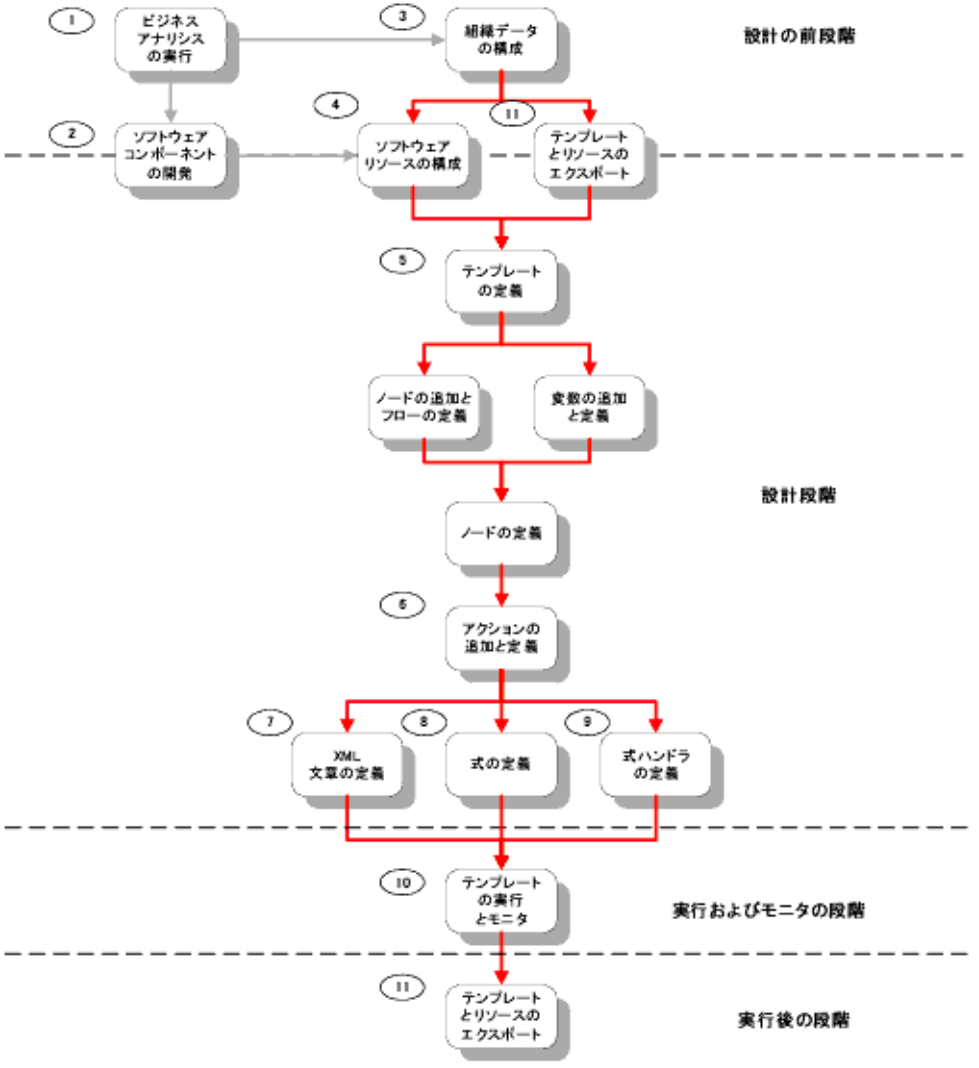
このドキュメントの組織原理として採用されているのがトップダウン アプローチです。ただし、トップダウン、ボトムアップの両方ともが有効であり必要でもあるため、実践的には、通常は両方が採用されます。Studio の特定の設計タスクおよび設計段階に対する両方のアプローチについては、以下の節で説明します。

トップダウン アプローチ

Studio のグラフィカル ユーザ インタフェースでは、トップダウン アプローチが採用されていて、ほとんどの外部コンポーネント要素はあらかじめ開発済みとなっています。このアプローチの場合、ワークフロー定義プロセスは、基本アクティビティの高レベルのグラフィック表現およびアプリケーションが実現するロジックのマッピングから始まって、より詳細な指定へと掘り下げていきます。

このワーク モデルは、次のフローチャートに示します。Studio での WebLogic Integration ワークフローのモデリング、設計、定義、テストの各段階で実行する主なタスクがまとめられています。このドキュメントで用いた構造のベースとなっているのがこのモデルです。手順 3 以降の各手順は、このドキュメントで取り上げるトピックに対応しています。この図については、この後、詳しく説明します。この設計プロセスには、タスクが繰り返されるという循環的な特徴がありますが、視覚的な単純化のため、フローチャートではこの性質を実際にループで表現していないので注意してください。

図 1-8 Studio の設計ワーク モデルトップダウン アプローチ



1. ビジネス アナリシスの段階では、アプリケーション要件およびデータ要件の特定、統合すべき既存コンポーネントおよびアプリケーションの確認、所属オーガニゼーションのビジネス ルールおよびビジネス構造を捕捉するための

データ モデルの定義を行います。これには、サードパーティの設計ツールでワークフローのモデリングに着手することもあります。この段階の詳細については、『*WebLogic Integration ソリューションの設計*』を参照してください。

2. リソース開発段階では、EJB やカスタム Java クラス、XML ドキュメントおよびワークフローから発信するメッセージおよび接続されたアプリケーションやコンポーネントから着信するメッセージで使用されるスタイル シートなどの Java コンポーネント、およびプラグイン コンポーネントの開発を行います。この段階の詳細については、『*WebLogic Integration ソリューションの設計*』を参照してください。
3. 設計の前段階では、Studio を使用してビジネス データのコンフィグレーションを行います。この段階のタスクについては、第 3 章「データの管理」で説明します。これらのタスクは、最初に 1 度だけ行い、後は、ビジネス ルールの変更や人事異動の際に必要に応じて行います。
4. この時点で、手順 2 で作成した任意の外部リソースに対して、ワークフローからグローバルにアクセスできるように、その設定に着手してください。この段階のタスクについては、第 4 章「ワークフロー リソースのコンフィグレーション」で説明します。これらの手順は、ワークフローの具体的要件の明確化に伴い、設計段階の前後で繰り返し行うことになります。
5. 実際の設計段階では、ワークフロー テンプレートおよびテンプレート定義の設定から着手します。次に、各ノードをワークフロー ダイアグラムに追加および接続し、変数およびノード プロパティを定義して、高レベルのプロセス フローを定義します。これらのタスクについては「ワークフロー テンプレートの定義」で説明しますが、手順 8 で説明するワークフロー式を何らかの形で使用する必要があります。
6. ノードの作成後、「アクションの定義」で説明するとおり、アクションの定義と追加に着手します。
7. アクションを定義する際、手順 2 および手順 3 で作成およびコンフィグレーションを行った XML ドキュメントのインポートが必要となる場合があります。また、ワークフロー アクション内で XML コンテンツを構成しなければならない場合もあります。これらのタスクについては、第 7 章「XML エンティティを操作」で説明します。
8. ワークフローの設計プロセス全体を通じて、ワークフロー式言語でフォーマットされたデータの入力が必要ですが、このフォーマットには、リテラル、定数、変数、実行時データを供給する埋め込み関数を格納できます。ワークフロー式のセマンティクスおよび構文については、第 8 章「ワークフロー式の使用法」で説明します。

9. テンプレート定義にカスタム例外ハンドラを追加するには、実行時例外の発生時に実行すべきアクションのサブフローおよび例外ハンドラを終了してメイン プログラム フローに戻る方法を定義します。これらのタスクについては、第 9 章「ワークフロー例外の処理」で説明します。例外ハンドラを定義した際、手順 6 で説明したように、ノードにアクションをもう 1 度追加して参照できるようにする必要があります。
10. ここで、ワークフローを実行してテストすることができます。Studio には、実行中のワークフロー インスタンスを表示して設計エラーを見つけるために役立ついくつかのモニタ機能が備わっています。モニタ機能については、第 10 章「ワークフローのモニタリング」で説明します。設計上の不具合と思われるものが見つかった場合は、アクションの再設定、式の再定義など、これまで説明してきた設計プロセスの手順をやり直す必要があります。
11. 設計実施の後段階では、つまり、ワークフローのテストが完了して、正常に実行している段階では、ワークフローや自分で作成、コンフィグレーションを行った任意のリソースを、Java アーカイブ パッケージにエクスポートできます。その後、エクスポートしたパッケージを再インポートして、プロセス サイクル全体を再開できます。インポートおよびエクスポートのタスクについては、第 11 章「ワークフロー パッケージのインポートとエクスポート」で説明します。

ボトムアップ アプローチ

Studio でのタスク設計に対するボトムアップ アプローチでは、テンプレート定義にコピーまたはインポートしてフローに組み込むことの可能な、変数、アクション、ノード、ノード グループで構成された再利用およびエクスポート可能な設計パターンのカタログを作成する必要があります。このアプローチでも、やはり、アプリケーションの要件を分析し、必要なリソースを整備するという事前タスクを完了しておく必要があります。ただし、このアプローチでは、高レベルのプログラム フローの概要を定義することから着手するのではなく、使用可能な Studio アクションをワークフローにおいて頻出する操作にマッピングし、それらのアクションの詳細を定義することから着手します。このアプローチによる推奨ワーク モデルは、以下の手順で構成されます。

1. アクションが参照する必要のあるすべてのグローバル エンティティを定義します。このエンティティには、ビジネス カレンダー（3-4 ページの「ビジネス カレンダーの管理」参照）およびビジネス オペレーションも含まれます（4-9 ページの「ビジネス オペレーションのコンフィグレーション」参照）。

2. ワークフロー テンプレートおよびテンプレート定義を作成またはインポートします。これらのタスクについては、5-3 ページの「テンプレートに関する作業」および 5-7 ページの「テンプレート定義に関する作業」で説明します。
3. ワークフローに頻出するアクティビティを実行する*ノード アーキタイプ*を追加、定義します。これらのタイプ、および各タイプを構成するアクションについては、6-1 ページの「アクションの概要」で説明します。
4. アドレス指定メッセージング、イベントによってトリガされる処理、ワークフロー間通信などの機能を利用するために使用する可能性のあるワークフロー関数およびその他の式コンポーネントを検索します。関数については、「ワークフロー式の使用法」の 8-5 ページの「関数の使用法」で説明します。
5. ビジネス オペレーション用のインスタンス変数、XML ドキュメントを格納する XML 変数など、呼び出されたワークフローに対する入力変数および出力変数など、アクションが頻繁に参照する必要のある変数を定義します。変数定義タスクについては、5-28 ページの「変数に関する作業」で説明します。
6. 式、ビジネス オペレーション用のパラメータ、XML イベントをポスティングするアクション用の JMS メッセージ オプション、XML ドキュメントを埋め込むアクション用の XML ドキュメント コンテンツなど、これらのノード タイプ に含まれるアクションに対するすべての低レベル詳細を定義します。ユーザ、ロール、またはオーガニゼーションを参照するアクションは、実際のビジネス要件を表現するエンティティを定義が済むまで、デフォルト アクションをプレースホルダとして使用することも可能です。
7. 開始、イベント、タスクの各ノードについて、具体的なノード プロパティを定義します。これらのタスクについては、5-34 ページの「ノードのプロパティの定義」で説明します。
8. 例外ハンドラとそのアクションを設定します。
9. 作成したコンポーネントを新しいワークフロー テンプレートにコピーします（5-26 ページの「ノードをコピーする」を参照）。あるいは、新しいテンプレートに再インポートできるように、テンプレート定義を Java アーカイブ ファイルにエクスポートします。
10. あらかじめ定義しておいたノードをフローに配置して、新しいテンプレート定義に関係する高レベルのオーガニゼーションのデータを定義します。

Studio ツール

Studio には、プロセス フロー設計に使用するグラフィック描画エンジンに加えて、以下のような、外部リソース管理およびワークフロー プロパティ定義に役立つ各種ツールも備わっています。

- XML ファインダ

  XML ファインダは、スキーマ、文書型定義、メッセージ フォーマットなど、データベース表やファイルに保存されている、タイプの異なる XML コンテンツにアクセスして整理するために役立つファイルおよびコンテンツの管理ツールです。XML ファインダについては、4-27 ページの「リポジトリにあるエンティティの管理」および 7-19 ページの「XML ファインダによる XML エンティティの取り出しとエクスポート」 で説明します。

- XML エディタ

  Studio アクションの多くは、XML ドキュメントをワークフローから構成、インポートおよびエクスポートするために役立ちます。これらのアクションには、XML ドキュメント テンプレートの新しい構成、既存ドキュメントの編集、および XML スキーマに照らしたコンテンツの検証に使用できる XML エディタが含まれています。XML 編集機能については、第 7 章「XML エンティティを操作」 で説明します。

- Expression Builder

  Studio のダイアログ ボックスの多くは、ワークフロー式の構文でデータを入力する必要があります。Expression Builder は、演算子、リテラル、ワークフロー関数、自分で作成した変数などのカタログから選択した要素を使用して、コンポーネントごとに式を作成するための編集ツールです。式が完成したときに、Expression Builder によって構文の確認が行われ、エラーがあれば通知されます。Expression Builder については、8-28 ページの「Expression Builder の使い方」 で説明します。

- XPath Wizard

  着信した XML ドキュメントからコンテンツを抽出するには、XPath 関数を使用して XML 要素や XML 属性に含まれているターゲット データを検索します。XPath Wizard は、サンプルの XML ドキュメントやスキーマ、DTD で生成した XML から選択したコンテンツから、XPath 言語の構文を習得することなく、XPath 式を自動的に生成させるポイント アンド クリック ツー

ルです。XPath Wizard については、8-31 ページの「XPath Wizard を使用する XPath 式の作成」で説明します。

- Import/Export ウィザード

  Import/Export ウィザードを使って、ワークフロー テンプレート、テンプレート定義、ビジネス オペレーション、イベント キー、プラグイン、XML エンティティなどのシステム内で定義済みの関連リソースのすべてをエクスポートできます。ワークフロー オブジェクトは Java アーカイブファイルにエクスポートされ、Studio が実行されている任意のシステムにインポートできます。Import/Export ウィザードについては、第 11 章「ワークフロー パッケージのインポートとエクスポート」で説明します。

2　Studio インタフェースの使用法

この章では、Studio の起動方法について説明し、Studio のグラフィカル ユーザ インタフェースの概要について説明します。

- Studio の起動とログオン
- Studio インタフェースの概要
- インタフェース ビューの使用法
- Studio の終了

Studio の起動とログオン

Studio は、以下のいずれかの方法で起動します。

- Windows システムで、[スタート | プログラム | BEA WebLogic Platform 7.0 | WebLogic Integration 7.0 | Studio] を選択します。
- UNIX システムの場合、*`WLI_HOME`*`/bin` ディレクトリに移動し、`studio` コマンドを実行します。

  たとえば、WebLogic Integration が `/home/bea/weblogic700/integration` ディレクトリにインストールされている場合、次を入力します。

  ```
  cd /home/bea/weblogic700/integration/bin
  . ./studio
  ```

[WebLogic Integration へのログオン] ダイアログ ボックスが表示されます。

図 2-1 [WebLogic Integration へのログオン] ダイアログ ボックス

WebLogic Integration にログオンする手順は、次のとおりです。

1. ユーザ名とパスワードを該当するフィールドに入力します。Studio のユーザ名およびパスワードの割り当てを受けていない場合は、デフォルトのユーザ名とパスワードを入力します。デフォルトのユーザ名とパスワードのリストについては、『*WebLogic Integration の起動、停止およびカスタマイズ*』の「はじめに」にある「WebLogic Integration ユーザおよびパスワード」を参照してください。

   注意 : ユーザ名とパスワードは大文字と小文字の区別が必要です。ユーザ名とパスワードは必ず小文字で入力してください。

2. [サーバ URL（プロトコル : ホスト : ポート）] フィールドで、WebLogic Integration サーバを実行しているシステムを以下のように指定します。

   `t3://host:port`

   - *host* は、WebLogic Integration サーバを実行しているシステムのコンピュータ名または IP アドレスです。サーバを Studio アプリケーションと同じコンピュータ上で実行している場合は、`localhost` を指定します。
   - *port* は、WebLogic Integration サーバのインストール時にリスン ポートに対して指定した番号です。デフォルトは 7001 です。

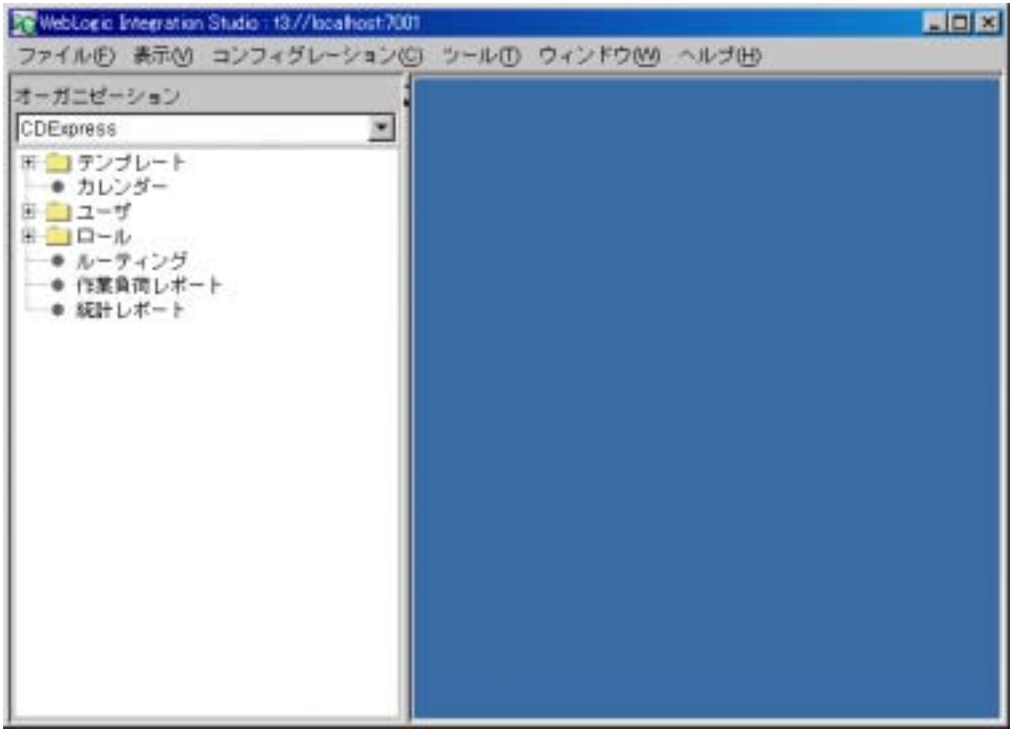
   クラスタ化サーバにログインするには、[サーバ URL（プロトコル : ホスト : ポート）] フィールドに次のように入力します。

t3://*host1*,*host2*,*host3*:*port*

この場合、*host1*、*host2*、および *host3* は WebLogic Integration クラスタ化サーバのコンピュータ名または IP アドレスです。

3. [OK] をクリックして Studio のメイン ウィンドウを表示します。

図 2-2 WebLogic Integration Studio メイン ウィンドウ



Studio インタフェースの概要

この節では、Studio ユーザ インタフェースの各部分およびそれぞれの機能について説明します。

メニュー オプション

以下の節では、Studio でのメニュー バーの使い方について説明します。

[ファイル] メニュー

[ファイル | *オプション*] の順に選択し、以下の表に示す機能を実行します。

メニュー オプション	機能
[ログオン]	WebLogic Integration サーバにログオンする。
[ログオフ]	WebLogic Integration サーバからログオフする。
[終了]	WebLogic Integration サーバからログオフして、WebLogic Integration Studio を終了する。

[表示] メニュー

[表示 | *オプション*] の順に選択し、以下の表に示す機能を実行します。

メニュー オプション	機能
[最新の情報に更新]	別のクライアント アプリケーションによってデータベースが変更された場合に、情報を更新する。
[フローチャートで色を使用]	ワークフロー図をカラーまたは白黒で表示する。
[選択内容とツリーを同期]	フォルダ ツリー階層内のワークフロー コンポーネントを、設計領域のワークフロー コンポーネントと同期させる。たとえば、ワークフロー図内の開始ノードをクリックすると、フォルダ ツリー階層構造内の対応する開始ノード フォルダが開く。

メニュー オプション	機能
[ルック & フィール]	Studio 表示のルック & フィールを変更する。オプションは次のとおり。 ■ Metal ■ CDE/Motif ■ Windows
[インタフェース ビュー . .]	ビジオネス オペーレーション、サブワークフロー、XML ドキュメントおよびプラグインを表す追加の視覚オブジェクトを表示する。このメニュー項目を設定すると [インタフェース ビュー 設定] ダイアログ ボックスが表示される。インタフェースの表示プリファレンスの設定方法については、2-13 ページの「インタフェース ビューの使用法」を参照。開始、イベント、または完了ノードに、プラグインによって定義されたカスタム プロパティが含まれる場合、ノードのアイコンの右上隅に小さなプラグイン アイコンが表示されます。

[コンフィグレーション] メニュー

[コンフィグレーション | *オプション*] の順に選択し、以下の表に示す機能を実行します。

メニュー オプション	機能
[オーガニゼーション]	ビジネス エンティティや地理的な位置など、企業の特定のビジネスを区別するオーガニゼーションを定義する。オーガニゼーションを定義する方法の詳細については、3-11 ページの「オーガニゼーションの保守」を参照。
[ビジネス オペレーション]	EJB インスタンスまたは Java Class インスタンスに関するメソッド呼び出しを表す、ビジネス オペレーションを定義する。ビジネス オペレーションを定義する方法の詳細については、4-9 ページの「ビジネス オペレーションのコンフィグレーション」を参照。

メニュー オプション	機能
[イベント]	イベント キーの式を定義する。イベント キー式を定義する方法の詳細については、4-21 ページの「イベント キーのコンフィグレーション」を参照。
[プラグイン]	使用可能なプラグインを、表示およびコンフィグレーションする。プラグインのコンフィグレーション方法の詳細については、4-3 ページの「プラグインのコンフィグレーション」を参照。
[パーミッション]	ユーザおよびロールのパーミッション レベルを定義する。パーミッション レベルを定義する方法の詳細については、3-29 ページの「ユーザに対してパーミッションを設定する」および 3-28 ページの「ロールに対してパーミッションを設定する」を参照。
[ロール マッピング]	現在定義されているロールを WebLogic Server グループに対してマップする。ロールのマッピングの詳細は、3-25 ページの「ロールに対するマッピングを変更する」を参照。

[ツール] メニュー

[ツール | *オプション*] の順に選択し、以下の表に示す機能を実行します。

メニュー オプション	機能
[パッケージをエクスポート . .]	ワークフロー オブジェクトを JAR ファイルとしてエクスポートする。パッケージをエクスポートする方法の詳細については、第 11 章「ワークフロー パッケージのインポートとエクスポート」を参照。
[パッケージをインポート . .]	ワークフロー オブジェクトを JAR ファイルからインポートする。パッケージをインポートする方法の詳細については、第 11 章「ワークフロー パッケージのインポートとエクスポート」を参照。

メニュー オプション	機能
[XML ファインダを表示..]	XML リポジトリの管理に使用する [XML ファインダ] ダイアログ ボックスを表示する。XML リポジトリを管理する方法の詳細については、4-27 ページの「リポジトリにあるエンティティの管理」を参照。

[ヘルプ] メニュー

[ヘルプ | *オプション*] の順に選択し、以下の表に示す機能を実行します。

メニュー オプション	機能
[ヘルプ トピック]	Studio のオンライン ヘルプにアクセスする。
[プラグイン ヘルプ]	ロードしたプラグインのオンライン ヘルプにアクセスする。
[WebLogic Integration Studio のバージョン情報]	WebLogic Integration Studio ソフトウェアのバージョン情報を表示する。

フォルダ ツリー表示

WebLogic Integration Studio インタフェースにはフォルダ ツリー表示が含まれていて、ワークフロー コンポーネントを標準ツリー構造で表示します。

図 2-3 WebLogic Integration フォルダ ツリー表示

異なるオーガニゼーションを選択するには、ウィンドウ最上部の [オーガニゼーション] ドロップダウン リストを使用します。

フォルダ ツリー表示に示される以下の各項目の詳細を表示するには、該当する項目をダブルクリックして展開します。

- テンプレート
- カレンダー
- ユーザ
- ロール
- ルーティング
- 作業負荷レポート

■ 統計レポート

たとえば、[テンプレート] フォルダをダブルクリックすると、ワークフロー テンプレートのリストが表示される。ワークフロー テンプレートをダブルクリックすると、すべてのワークフロー テンプレート定義のリストが表示される。特定のワークフロー テンプレート定義を展開すると、そのワークフロー テンプレート定義に対する タスク、分岐、イベント、結合、開始、完了、変数 などを格納しているフォルダが表示されます。

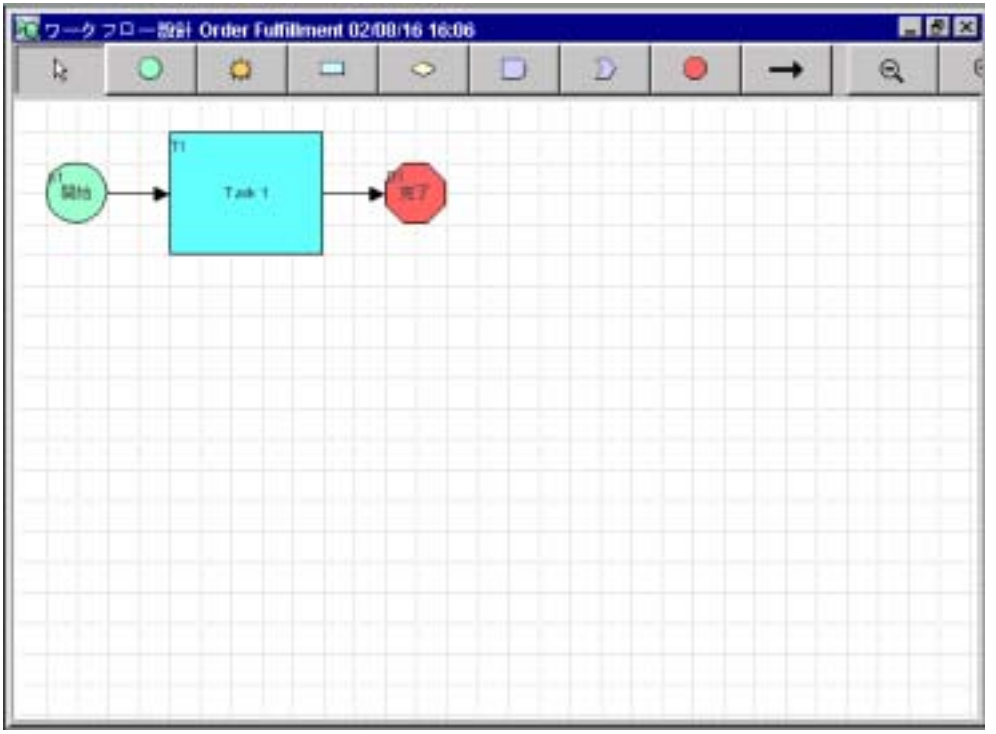
フォルダツリーにあるほとんどの項目は、右クリックすると強調表示され、その項目に関連するオプションを列挙するメニューが表示されます。たとえば、既存のワークフロー テンプレートを右クリックすると、次の図に示すオプションを列挙するメニューが表示されます。

図 2-4 ワークフロー テンプレートのメニュー オプション

ワークフローの設計領域とツールバー

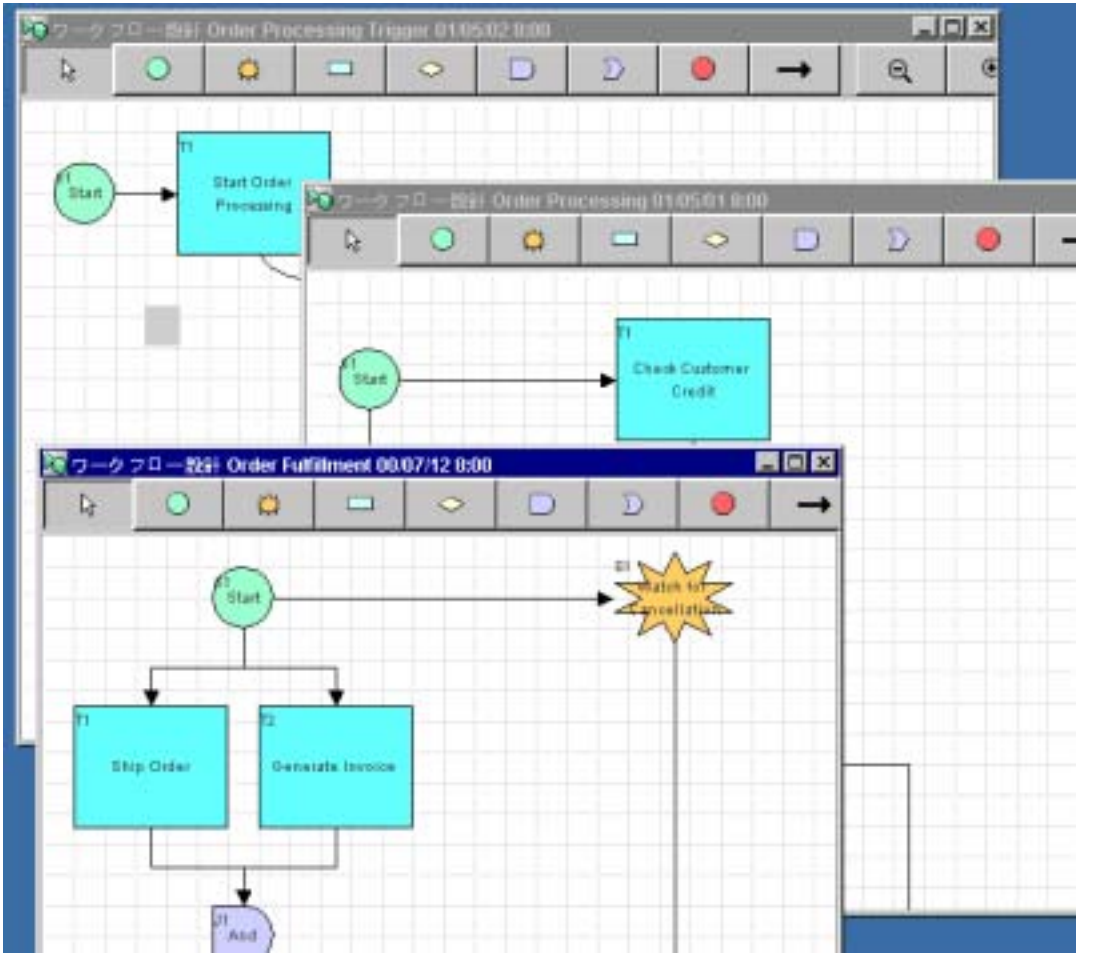
既存のワークフローテンプレート定義を作成またはオープンする（その手順については 5-7 ページの「テンプレート定義に関する作業」を参照）と、次の図のようなワークフロー設計領域でワークフロー設計を作成できます。ワークフロー ツールバーには、ワークフローの定義に使用する、ワークフローの各ノードとコネクタを表すシェイプがあります。

図 2-5 [ワークフロー設計] ウィンドウ



同時に複数のワークフロー設計を開くことができます。複数のワークフローを表示しておき、相互に切り替えることができます。

図 2-6　複数ワークフロー ダイアグラム



ツールバーの使い方

また、このツールバーには、次の切り替えコントロールもあります。

表 2-1 ツールバー ボタン

ツールバー ボタン	説明
	ダイアグラムのズーム アウト。複雑なダイアグラムに取り組んでいて、全体のフローを確認する必要がある場合に便利なボタン。
	ダイアグラムのズーム イン。
	ダイアグラムでグリッドを表示または非表示にする。
	ダイアグラム内の図形の整列。
	図形の自動整列をオンまたはオフに設定。
	インタフェース ビューのオン / オフ切り替え。インタフェース ビューの詳細については、2-13 ページの「インタフェース ビューの使用法」を参照。

デフォルトでは、ワークフロー ツールバーは、設計領域の最上部に現れます。[Studio] ウィンドウや設計領域のサイズによっては、ツールバー全体が表示されない場合があります。ツールバー全体を表示するには、ツールバーを Studio の他の設計領域またはデスクトップ上の他の場所に移動します。

ツールバーを移動するには、ツールバーの背景領域（[コネクタを描画] ボタンと [縮小] ボタンの間など）をクリックして、ツールバーを移動先にドラッグします。

ツールバーを、デスクトップ上の Studio の設計領域以外の場所に移動すると、ツールバーはデスクトップ上の別のウィンドウとなり、最小化、最大化、クローズの各操作が可能になります。

インタフェース ビューの使用法

テンプレート定義を作成して、ノードの定義を開始し、アクションの追加が完了した際には、一般的には、ワークフローがインタフェースを取る対象オブジェクトが、オブジェクトを参照するノード別に表示されていると便利です。インタフェース ビューを使って、ワークフロー ダイアグラム中のコンポーネント、およびそのワークフローがインタフェースを取る以下のオブジェクトを表すアイコンを表示できます。

- サブワークフロー―現在のワークフローが接続する他のワークフロー
- ビジネス オペレーション―関数を実行するためにワークフローが呼び出す EJB クラスや Java クラスなどのソフトウェア コンポーネント
- 着信および発信 XML ドキュメント―ワークフローが受信する、またはワークフローから他のワークフローやアプリケーションに送信される XML ドキュメント
- プラグイン―カスタムのワークフロー コンポーネント

インタフェース ビューは、デフォルトでは無効化されています。インタフェース ビューを使用するには、各テンプレート定義に対してインタフェース ビューを有効化し、そのテンプレート定義にアクセスする新しくなった各セッションも有効化する必要があります。

一方、表示されるオブジェクトを決定するインタフェース ビュー プリファレンスは、システム全体を対象として設定されます。すなわち、すべてのテンプレート定義に適用されます。設定は Studio におけるセッションを切り替えても保存されますが、テンプレート定義の一部としては保存されません。

表示する項目を指定する手順は、以下のとおりです。

1. [表示 | インタフェース ビュー] の順に選択して、[インタフェース ビュー設定] ダイアログ ボックスを表示します。

図 2-7 [インタフェース ビュー 設定] ダイアログ ボックス

2. 表示するオブジェクトのチェック ボックスにチェックし、表示しないオブジェクトのチェック ボックスのチェックを外します。

現在のテンプレートのノーマル ビューとインタフェース ビューを切り替えるには、ツールバーにある次のボタンをクリックします。 

受信 XML ドキュメント データを表示する

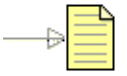
インタフェース ビューに表示される以下のアイコンは、ワークフロー図内の受信 XML ドキュメントを表します。

注意 : テンプレート ノード向きの矢印は、受信方向を示しています。

受信 XML ドキュメント アイコンは、受信 XML ドキュメントに応答する開始ノードとイベント ノードに対して表示されます。ワークフロー ダイアグラムの XML ドキュメント アイコンにマウス ポインタを保持すると、その XML ドキュメントのタイプ（受信または送信）、ルートおよびキーが示されているテキスト ボックスが表示されます。

送信 XML ドキュメント データを表示する

インタフェース ビューに表示される以下のアイコンは、ワークフロー図内の送信 XML ドキュメントを表します。



注意： テンプレート ノードを出発点とする向きになっている矢印は、送信方向を示しています。

オブジェクトに対して [XML をクライアントに送信] アクションまたは [XML イベントをポスト] アクションが定義されている場合に、送信 XML ドキュメントのアイコンが表示されます。XML ドキュメントに関する情報を取得する方法は次のとおりです。

- ワークフロー ダイアグラムの XML ドキュメント アイコンにマウス ポインタを保持すると、その XML ドキュメントのタイプ（受信または送信）、ルートおよびキーが示されているテキスト ボックスが表示されます。
- XML ドキュメント アイコンをダブルクリックまたは右クリックし、ポップアップ メニューから [プロパティ] を選択すると、定義されているアクション（[XML をクライアントに送信] または [XML イベントをポスト]）についてのダイアログボックス（読み取り専用）が表示されます。

[XML をクライアントに送信] アクションおよび [XML イベントをポスト] アクションの詳細は、6-62 ページの「クライアント アプリケーションに対し XML メッセージを送信する」および 6-88 ページの「JMS トピックまたはキューへの XML メッセージのポスト」を参照してください。

サブワークフロー データを表示する

インタフェース ビューでは、ワークフローダイアグラムにあるサブワークフローは次のアイコンで表されます。

サブワークフロー アイコンは、あるオブジェクトに対して ワークフローを開始アクションが定義されている場合に表示されます。サブワークフローに関する情報は、以下の手順で取得できます。

- サブワークフロー アイコンにマウス ポインタを置くと、呼び出されたサブワークフローの名前が示されているテキストボックスが表示されます。
- サブワークフロー アイコンをダブルクリックまたは右クリックし、ポップアップ メニューから [プロパティ] を選択すると、[ワークフローを開始] ダイアログボックス（読み取り専用）が表示されます。

[ワークフローを開始] アクションの詳細は、6-39 ページの「サブワークフローを呼び出す」 を参照してください。

ビジネス オペレーション データを表示する

インタフェース ビューに表示される以下のアイコンは、ワークフロー図内のビジネス オペレーションを表します。

オブジェクトに対して [ビジネス オペレーションを実行] アクションが定義されている場合に、ビジネス オペレーションのアイコンが表示されます。ビジネス オペレーションに関する情報を取得する方法は次のとおりです。

- ワークフロー ダイアグラムのビジネス オペレーション アイコンにマウス ポインタを保持すると、実行すべきビジネス オペレーションの名前が示されているテキスト ボックスが表示されます。
- ビジネス オペレーション アイコンをダブルクリックまたは右クリックし、ポップアップ メニューから [プロパティ] を選択すると、[ビジネス オペレーションを実行] ダイアログボックス（読み取り専用）が表示されます。

ビジネス オペレーションを実行アクションの詳細については、6-84 ページの「ビジネス オペレーションを呼び出す」 を参照してください。

プラグイン データを表示する

インタフェース ビューに表示される以下のアイコンまたはカスタム アイコンは、ワークフロー図内のプラグインを表します。

ノードにプラグイン アクションが含まれる場合、アクションに関する情報を取得する方法は次のとおりです。

- ワークフロー ダイアグラムのプラグイン アイコンにマウス ポインタを保持すると、そのプラグイン アクションの説明が示されているテキスト ボックスが表示されます。
- プラグイン アイコンをダブルクリックまたは右クリックし、ポップアップ メニューから [プロパティ] を選択すると、対応する [プラグイン] アクションのダイアログ ボックス（読み取り専用）が表示されます。

注意： 開始、イベント、完了の各ノードに、プラグインで定義された、カスタマイズされたプロパティが設定されている場合は、ノード アイコンの右上のコーナーに小さなプラグイン アイコンが表示されます。

Studio の終了

WebLogic Integration からログオフして Studio は表示したままにしておく手順は、以下のとおりです。

1. ワークフローに加えた変更を保存するには、ワークフロー テンプレート定義を右クリックするとメニューが表示され、そこから [保存] を選択します。
2. [ファイル | ログオフ] を選択します。

Studio を終了する手順は、以下のとおりです。

1. ワークフローに加えた変更を保存するには、ワークフロー テンプレート定義を右クリックするとメニューが表示され、そこから [保存] を選択します。

2. [ファイル | ログオフ] を選択します。削除の確認を求めるダイアログ ボックスが表示されます。[はい] をクリックすると終了します。

注意 : 変更内容を保存しないで Studio を終了すると、ダイアログ ボックスに保存を指示するメッセージが表示されます。

3 データの管理

この章では、Studio における以下のデータ管理にかかわるコンセプトおよびタスクについて説明します。

- データ コンフィグレーション タスクの概要
- セキュリティ レルム
- ビジネス カレンダーの管理
- オーガニゼーションの保守
- ユーザの保守
- ロールの保守
- ユーザおよびロールへのパーミッションの割り当て
- タスク ルーティングの管理

データ コンフィグレーション タスクの概要

Studio でのデータ コンフィグレーション タスクとしては、ビジネス カレンダーの定義、オーガニゼーションの作成、ユーザおよびロールの作成、セキュリティおよびパーミッションのコンフィグレーション、およびタスク ルーティングの定義があります。オーガニゼーション データのモデリングおよび最初に行うシステム コンフィグレーションの実行は、以下の手順で行うことをお勧めします。

1. オーガニゼーション、ユーザ、およびロールに関連付けることのできるカレンダーを作成します。手順の詳細は、3-6 ページの「カレンダーを作成する」を参照してください。別の方法として、あらかじめエクスポートしておいたカレンダーを既存のワークフロー パッケージからインポートします。その手順については、11-5 ページの「ワークフロー パッケージのインポート」を参照してください。

2. 1 つまたは複数のオーガニゼーションを追加します。手順の詳細は、3-12 ページの「オーガニゼーションを追加する」を参照してください。
3. システムにユーザを追加します。手順の詳細は、3-15 ページの「ユーザを作成する」を参照してください。
4. オーガニゼーションにユーザを追加します。手順の詳細は、3-17 ページの「オーガニゼーションにユーザを追加する」を参照してください。
5. （省略可能）WebLogic Server Administration Console を使用して、Studio で定義したロールに対応するグループを作成します。ロールの詳細については、3-21 ページの「ロールの保守」を参照してください。WebLogic Integration グループの作成に関する情報については、次の URL に掲載されている『WebLogic Security の管理』の「互換性セキュリティの使い方」にある「互換性レルムでのグループの定義」の節を参照してください。

   `http://edocs.beasys.co.jp/e-docs/wls/docs70/secmanage/security6.html`
6. オーガニゼーション内のロールまたは複数のロールを定義し、そのロールを WebLogic Server グループにマッピングして、メンバー ユーザをロールに関連付けます。手順の詳細は、3-21 ページの「ロールの作成」を参照してください。
7. 必要であれば、そのロールに対するパーミッション レベルを変更します。あるロールに対するパーミッション レベルを変更すると、そのグループのパーミッションも変更され、そのグループにマッピングされるすべてのロールに影響が及びます。その手順については、3-28 ページの「ロールに対してパーミッションを設定する」を参照してください。
8. ユーザに対する追加のパーミッション レベルを定義します。ユーザは各自が所属するロールのパーミッションを継承しますが、その他に、ユーザが所属するロールに対して定義されている以外のパーミッション レベルを追加できます。手順の詳細は、3-29 ページの「ユーザに対してパーミッションを設定する」を参照してください。

セキュリティ レルム

WebLogic Server は、セキュリティ レルムというサービスを通じて BPM アプリケーションのセキュリティを保証します。セキュリティ レルムはユーザとグループを論理グループにまとめたものです。ユーザは、プログラミングや営業などの一定のタスクを遂行する特定の個人です。グループは、同じタスクを遂行するユーザの集合です。この方式では、たとえば、グループ A はプログラマの集合を表し、グループ B は営業担当者の集合を表します。管理者はセキュリティ レルム内で、ワークフローやその他のリソースに対してユーザおよびグループに許可されるアクセスのレベルを指定できます。

注意： WebLogic Integration および BPM のセキュリティについての重要な基礎情報については、次を参照してください。

- 『*WebLogic Integration ソリューションのデプロイメント*』の「WebLogic Integration セキュリティの使い方」
- 『*WebLogic Integration の起動、停止およびカスタマイズ*』の「WebLogic Integration のカスタマイズ」にある「BPM セキュリティ モデルについて」
- 「*WebLogic Platform Security の紹介*」

リストの 3 番目に記載のマニュアルに、BPM アプリケーションと他の WebLogic Platform コンポーネントを併用する場合の重要な情報が提供されています。

WebLogic Integration では、ロールとユーザに関する情報が WebLogic Server セキュリティ レルム内に保持されます。WebLogic Integration のユーザおよびロールを定義する際、それらの WebLogic Server におけるユーザおよびグループに対する関係を指定する必要があります。このタスクは、Studio で定義したロールを WebLogic Server のセキュリティ レルムのグループにマッピングすることによって行います。

WebLogic Server のセキュリティ レルムは、管理が可能な場合と管理が不可能な場合があります。管理の可否は、WebLogic Server でのレルムの実装時に定義されます。管理可能レルムは、アプリケーションを通じてグループおよびユーザを更新できるレルムです。管理不可能レルムは、アプリケーションからはグループおよびユーザの一覧表示のみが可能なレルムです。

レルムが管理可能か管理不可能かは、WebLogic Integration によって自動的に検出されます。WebLogic Integration によって検出されるセキュリティ レルムのタイプによって、Studio で使用可能なデータ管理機能が決まります。

Studio の各ダイアログ ボックスでは、WebLogic Integration によって管理可能レルムが検出された場合に限り、以下のタスクを実行できます。

- ロールの WebLogic Server グループへのマッピング
- ユーザの追加または削除
- オーガニゼーションへのユーザの追加
- オーガニゼーションからのユーザの削除
- ロールへのユーザの追加
- ロールからのユーザの削除

WebLogic Integration によって管理不可能レルムが検出された場合は、Studio の各ダイアログ ボックスを使用して、以下のタスクを実行できます。

- ユーザとロールのリストの表示。Studio ダイアログ ボックスに備わっている、タスクの追加、削除、マッピングなどの機能は灰色表示となり、選択できなくなります。
- WebLogic Integration によって管理されるデータの更新。セキュリティ レルムの WebLogic Server によって管理されている、ビジネス カレンダーやオーガニゼーションなどのデータは更新できません。

管理不可能レルムにユーザを追加する場合は、WebLogic Server Administration Console で、そのレルムに対する適切なデータ管理機能を使用して追加する必要があります。Studio ダイアログ ボックスを使用して追加することはできません。

ビジネス カレンダーの管理

ビジネス カレンダー機能によって、ワークフローで表されるエンティティの稼働時間が定義されます。ビジネス カレンダーによって、「期日を本日より 3 稼働日後と設定する」など、時間が関係するビジネス計算が可能になります。ビジネ

ス カレンダーは、週末、祝日などの非稼働日や非稼働時間を除外して定義する必要があります。カレンダーを使用しないオーガニゼーションは、通常の 365 日の暦を使用します。

カレンダーは、以下のエンティティに関連付けることができます。その方法は（）内に示す節で説明します。

- オーガニゼーション（3-11 ページの「オーガニゼーションの保守」を参照）
- ユーザ（3-15 ページの「ユーザの保守」を参照）
- ロール（3-21 ページの「ロールの保守」を参照）

オーガニゼーション、ユーザ、ロール、アクションに対して、同じビジネス カレンダーを定義して割り当てることができます。同じオーガニゼーション内のユーザ、ロール、アクションに対して、別のビジネス カレンダーを割り当てることもできます。

カレンダーは、ワークフロー内の時限開始ノード（5-37 ページの「時限開始ノードを定義する」を参照）、時限イベント（6-34 ページの「時限シーケンスを埋め込む」を参照）、およびユーザが割り当てたタスクの期日（6-56 ページの「タスク期日を設定する」を参照）によっても使用されます。

注意： ビジネス カレンダーを管理するには、システムの構成パーミッションが必要です。詳細については、3-26 ページの「ユーザおよびロールへのパーミッションの割り当て」を参照してください。

カレンダーの割り当ては本質的に階層構造になっています。この階層構造では、時間に関係するアクションは、最も低いレベルに配置され、その上にロールおよびユーザが配置されます。最も高いレベルにはオーガニゼーションが配置されます。時間に関係するアクションにカレンダーが割り当てられていない場合、デフォルトでは、そのアクションに割り当てられたユーザまたはロールのカレンダーが割り当てられます。ユーザまたはロールにカレンダーが割り当てられていない場合は、デフォルトでは、ビジネス カレンダー（オーガニゼーション レベル）が割り当てられます。言い換えれば、カレンダーの割り当ては最も詳細なコンポーネント レベルで行われます。

ビジネス カレンダーはルール ベースになっています。カレンダー機能は、各ルールの定義に従って実行されます。

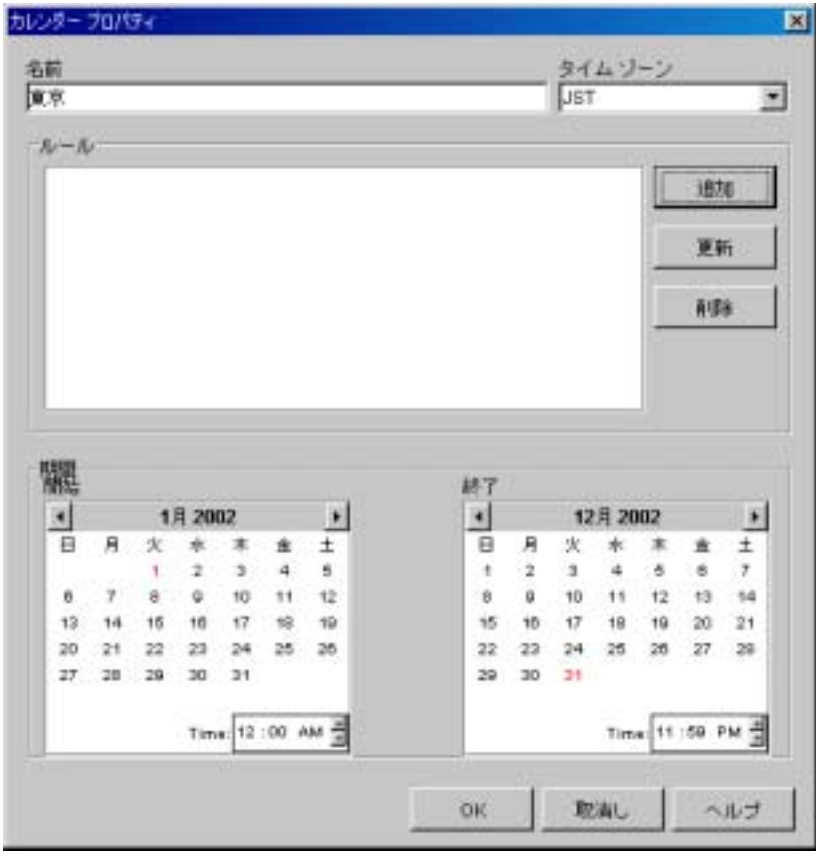
カレンダーを作成する

作成済みのカレンダーは、システム内のすべてのオーガニゼーション、ユーザ、ロールにとってグローバルに利用可能になります。

新しいカレンダーを作成する手順は、以下のとおりです。

1. 任意のアクティブなオーガニゼーションについて、フォルダ ツリーにある [カレンダー] を右クリックし、[カレンダーを作成] を選択して [カレンダー プロパティ] ダイアログ ボックスを表示します。

図 3-1 [カレンダー プロパティ] ダイアログ ボックス



2. [名前] フィールドに、類推できるカレンダー名を入力します。
3. [タイム ゾーン] ドロップ ダウン リストから、カレンダーのタイム ゾーンを選択します。
4. [期間] ボックスで、[開始] および [終了] のボックスから日付を選択して、カレンダーの対象期間を指定します（デフォルトは現在の年度の 1 月から 12 月まで）。
5. カレンダーのルールを追加する場合は、[追加] をクリックし、[ルール] ダイアログ ボックスを表示させます。

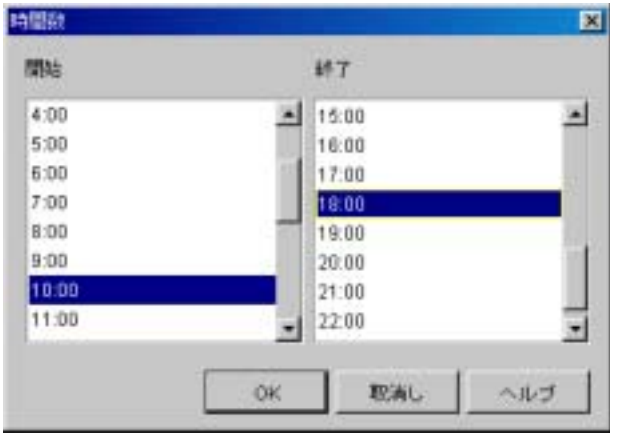
図 3-2 [ルール] ダイアログ ボックス

6. [除外] または [包含] を選択します。これらのボタンを使用して、カレンダー ルールを定義する方法（除外または包含）を決定します。ルール定義のプロセスでは、両方の方法を使用しないで、どちらか一方だけを使用することをお勧めします。
7. 次のボタンのいずれかをクリックしてルールを定義します。
   - 曜日－除外または包含する 1 つまたは複数の日付を選択するための [曜日] ダイアログ ボックスを表示します。〔Ctrl〕 を押さえながらクリックすると複数の曜日を選択できます。[OK] をクリックします。

図 3-3 [曜日] ダイアログ ボックス

- 時間数－除外または包含する時間帯を選択するための [時間数] ダイアログ ボックスを表示します。[開始] リストと [終了] リストからそれぞれ時間を選択して、[OK] をクリックします。

図 3-4 [時間数] ダイアログ ボックス



- 日付－特定の日付を選択するための [日付] ダイアログ ボックスを表示します。除外または包含する月および日を選択して、[OK] をクリックします。

図 3-5 [曜日] ダイアログ ボックス

- 月－除外または包含するその年度内の特定の月を選択するための [月] ダイアログ ボックスを表示します。〔Ctrl〕を押さえながらクリックすると、[月] ダイアログ ボックスから複数の月を選択できます。

図 3-6 [月] ダイアログ ボックス

- 期間－除外または包含する一連の日付を選択するための [期間] ダイアログ ボックスを表示します。[開始] ボックスと [終了] ボックスからそれぞれ月と日を選択して、[OK] をクリックします。

図 3-7 [期間] ダイアログ ボックス

8. [ルール] ダイアログ ボックスで [OK] をクリックしてルールに追加します。追加した新しいルールが、[カレンダー ルール] ダイアログ ボックスの [ルール] 領域に表示されます。
9. 手順 7 と 8 を繰り返して、カレンダーへのルールの追加を続けます。
10. [OK] をクリックしてカレンダーを保存します。

カレンダーを更新する

既存カレンダーを更新する手順は、以下のとおりです。

1. 任意のアクティブなオーガニゼーションについて、フォルダ ツリーにある [カレンダー] フォルダを展開し、更新するカレンダーを右クリックして [プロパティ] を選択します。
2. [カレンダー プロパティ] ダイアログ ボックスで、そのカレンダーの時間帯と対象期間を必要に応じて変更します。

3. カレンダーのルールを変更するには、[ルール] リストから適切なルールを選択し、[削除] ボタンまたは [更新] ボタンをクリックして既存のルールを削除または更新します。新しいルールを追加する場合は、[追加] をクリックして、3-6 ページの「カレンダーを作成する」の手順に従ってルールを定義します。
4. [OK] をクリックしてカレンダーに加えた変更を保存します。

カレンダーの削除

注意 : カレンダーを削除する場合、他のワークフロー オブジェクトによるそのカレンダーへの参照について警告は行われません。ユーザ、ロール、およびオーガニゼーションがカレンダーに割り当てられていること、同様にカレンダーを参照できる以下のワークフロー コンポーネントの割り当ても完了していることを確認してください。

時間指定付き開始ノード（5-37 ページの「時限開始ノードを定義する」を参照）

時間指定付きイベント（6-34 ページの「時限シーケンスを埋め込む」を参照）

タスクの期日（6-56 ページの「タスク期日を設定する」を参照）

既存カレンダーを削除する手順は、以下のとおりです。

1. 削除するカレンダーを右クリックして、ポップアップ メニューから [削除] を選択します。
2. 「カレンダーを削除」という警告メッセージが表示されたら [はい] をクリックします。削除を取り消すには [いいえ] をクリックします。

オーガニゼーションの保守

オーガニゼーションは、オーガニゼーション機能を使用して定義され、さまざまな業務エンティティ、地域、その他その企業の特定の事業に関連するオーガニゼーションを表すことができます。

あるオーガニゼーション内の各種ユニットを異なるオーガニゼーションとしてモデリングすると、同一ロール名を再利用して、別のグループにマッピングできます。たとえば、Supervisor というロールを複数個作成できます。実際は、このロールには、オーガニゼーションごとに異なるメンバーが所属します（3-21 ページの「ロールの保守」を参照）。オーガニゼーションは、WebLogic Integration に固有のもので、WebLogic Server のグループには対応しないため注意してください。

ユーザも、1 つまたは複数のオーガニゼーションに割り当てます。ユーザは、所属するオーガニゼーション内のワークフローのみを実行できます。

注意： オーガニゼーションを追加、更新、または削除するには、管理ユーザパーミッションが必要です。パーミッション レベルの詳細については、3-26 ページの「ユーザおよびロールへのパーミッションの割り当て」を参照してください。

フォルダ ツリーの上にある [オーガニゼーション] ドロップダウン リストには、現在アクティブなオーガニゼーションが表示されます。このリストにあるオーガニゼーションが選択されると、フォルダ ツリーにそのオーガニゼーションに対して定義されたロール、ユーザ、およびワークフローが表示されます。

オーガニゼーションを追加する

Add Organization 機能によって、オーガニゼーションを WebLogic Integration データベースに追加できます。

オーガニゼーションを追加する手順は、以下のとおりです。

1. [コンフィグレーション | オーガニゼーション] を選択して、[オーガニゼーションを定義] ダイアログ ボックスを表示します。

図 3-8 [オーガニゼーションを定義] ダイアログ ボックス

2. [オーガニゼーションを定義] ダイアログ ボックスで、[追加] をクリックして [オーガニゼーションのプロパティ] ダイアログ ボックスを表示します。

図 3-9 [オーガニゼーションのプロパティ] ダイアログ ボックス

3. [オーガニゼーション ID] フィールドにオーガニゼーションの名前として有意な名前を入力します。
4. カレンダーの作成完了後、[カレンダー ID] ドロップダウン リストから、オーガニゼーションに割り当てるカレンダーを選択します。この機能の詳細は、3-4 ページの「ビジネス カレンダーの管理」 を参照してください。
5. [OK] をクリックしてオーガニゼーションを作成します。

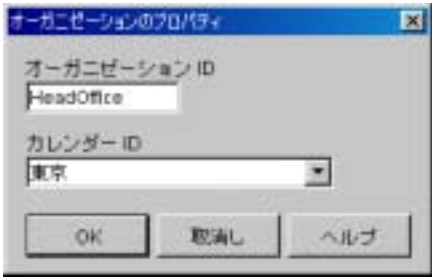
オーガニゼーションを更新する

Update Organization 機能によって、既存オーガニゼーションのビジネス カレンダーを更新できます。

オーガニゼーションを更新する手順は、以下のとおりです。

1. [コンフィグレーション | オーガニゼーション] を選択して、[オーガニゼーションを定義] ダイアログ ボックスを表示します。
2. [オーガニゼーションを定義] ダイアログ ボックスで、更新するオーガニゼーションを強調表示します。
3. [更新] をクリックします。[オーガニゼーションのプロパティ] ダイアログ ボックスが表示されます。

図 3-10 [オーガニゼーションのプロパティ] ダイアログ ボックス



4. [オーガニゼーション ID] フィールドまたは [カレンダー ID] フィールドに必要に応じて変更を加え、[OK] をクリックします。

オーガニゼーションを削除する

オーガニゼーションを削除機能によって、WebLogic Integration データベースからオーガニゼーションを削除できます。ただし、削除できるのは、ワークフローが定義されていないオーガニゼーションに限られます。オーガニゼーションにワークフローが定義されている場合は、まずワークフローを削除する必要があります。その手順については、5-18 ページの「テンプレート定義を削除する」 を参照してください。

オーガニゼーションを削除する手順は、以下のとおりです。

1. [コンフィグレーション | オーガニゼーション] を選択して、[オーガニゼーションを定義] ダイアログ ボックスを表示します。
2. [オーガニゼーションを定義] ダイアログ ボックスで、削除するオーガニゼーションを強調表示します。
3. 「オーガニゼーションを削除」という警告メッセージが表示されたら、削除する場合は [はい] を、削除しない場合は [いいえ] をクリックします。

ユーザの保守

ユーザは、一定のタスクを遂行するパーミッションを与えられている個人です。ユーザ機能を使用して、WebLogic Server のセキュリティ レルムでユーザを作成、更新、削除します。オーガニゼーションへのユーザの追加やオーガニゼーションからのユーザの削除も可能です。

[ユーザ] フォルダは、Studio のフォルダ ツリーにあります。このフォルダを展開すると、現在のオーガニゼーションに対して既に定義されているユーザのリストが表示されます。

注意 : ユーザを追加、更新、または削除するには、管理ユーザ パーミッションが必要です。パーミッション レベルの詳細については、3-26 ページの「ユーザおよびロールへのパーミッションの割り当て」を参照してください。

ユーザを作成する

ユーザを作成機能によって、現在の WebLogic Server のセキュリティ レルムにユーザを追加します。この機能は、セキュリティ レルムが管理可能レルムである場合に限り使用可能です。

セキュリティ レルムおよび WebLogic Integration データベースでユーザを作成する手順は、以下のとおりです。

1. 任意のアクティブなオーガニゼーションについて [ユーザ] を右クリックし、[ユーザを作成] を選択して [ユーザを作成] ダイアログ ボックスを表示します。

図 3-11 [ユーザを作成] ダイアログ ボックス

2. 以下の各フィールドに値を入力して、[OK] をクリックします。
   - [ユーザ ID] ―ユーザを一意に定義します。ユーザはこの ID を入力して Studio および Worklist クライアント アプリケーションにログインします。
   - [パスワード] と [パスワードを再入力] ―ユーザのパスワードを入力します。
   - [電子メール アドレス] ―ユーザの電子メール アドレスを入力します。
   - [デフォルトのオーガニゼーション] ―ユーザのデフォルトのオーガニゼーションを選択します。このオーガニゼーションは、ユーザがクライアント アプリケーションにログオンするときにデフォルトで表示される。ユーザの作成がいったん完了すれば、そのユーザを他のオーガニゼーションに追加できる。
   - [カレンダー] ―（省略可能）そのユーザのビジネス カレンダーを選択します。

   注意： ユーザ ID とパスワードには、JDK がサポートする文字セット（各国文字を含む）中の英数字を指定することができます。

オーガニゼーションにユーザを追加する

ユーザを追加機能を使用して、WebLogic Server のセキュリティ レルムで既に定義されているユーザを追加します。この機能は、WebLogic Integration Studio が管理可能なレルム内で動作している場合にのみ利用できます。

オーガニゼーションに追加されたユーザは、実行時にそのオーガニゼーション内でワークフローを実行できます。ただし、設計時にワークフロー テンプレートへのユーザのアクセスが制限されないため注意が必要です。テンプレートを開くため必要なパーミッションが与えられている限り、そのユーザは、複数のオーガニゼーションに関連付けられたテンプレートにアクセスできます。

現在のオーガニゼーションにユーザを追加する手順は、以下のとおりです。

1. フォルダ ツリーの上の [オーガニゼーション] フィールドから、ユーザを追加するオーガニゼーションを選択します。
2. フォルダ ツリーで [ユーザ] フォルダを右クリックし、[ユーザを追加] を選択して [ユーザを追加] ダイアログ ボックスを表示します。

図 3-12 [ユーザを追加] ダイアログ ボックス

3. 追加するユーザの左のチェック ボックスをオンにして、[OK] をクリックします。
4. フォルダ ツリーにユーザが追加されます。

ユーザを更新する

ユーザの ID、電子メール アドレス、デフォルトのオーガニゼーションおよびビジネス カレンダーを変更して、ユーザを更新できます。

注意： 作成済みのユーザのパスワードを変更するには、WebLogic Server Administration Console を使用する必要があります。詳細については、『*WebLogic Integration の起動、停止およびカスタマイズ*』の「WebLogic Integration のカスタマイズ」にある「パスワードの更新」の節を参照してください。

ユーザを更新する手順は、以下のとおりです。

1. フォルダ ツリーの上の [オーガニゼーション] フィールドから、ユーザを定義するオーガニゼーションを選択します。
2. フォルダ ツリーで、[ユーザ] フォルダを展開してユーザ名を右クリックし、ポップアップ メニューから [プロパティ] を選択して [ユーザのプロパティ] ダイアログ ボックスを表示します。

図 3-13 [ユーザのプロパティ] ダイアログ ボックス

3. 以下のフィールドを必要に応じて編集します。
   - [ユーザ ID]
   - [電子メール アドレス]

- [デフォルトのオーガニゼーション]
- [カレンダー] ―そのユーザに割り当てるビジネス カレンダーを選択します。この機能の詳細は、3-4 ページの「ビジネス カレンダーの管理」を参照。

4. [OK] をクリックします。または、この操作を取り消す場合は、[取消し] をクリックします。

オーガニゼーションからユーザを削除する

WebLogic Server のセキュリティ レルムでユーザを消去機能を使用して、既に定義されているユーザを削除します。この操作によって、セキュリティ レルムからユーザが削除されるわけではありません。

現在のオーガニゼーションからユーザを削除する手順は、以下のとおりです。

1. フォルダ ツリーの上の [オーガニゼーション] フィールドからユーザを削除するオーガニゼーションを選択します。
2. フォルダ ツリーで [ユーザ] フォルダを展開し、ユーザ名を右クリックして、ポップアップ メニューから [削除] を選択します。
3. 「ユーザを消去」という警告メッセージが表示されたら [はい] をクリックします。削除を取り消すには [いいえ] をクリックします。

ユーザを削除する

ユーザを削除すると、WebLogic Server セキュリティ レルムおよび WebLogic Integration データベースからユーザが削除されます。

注意 : ユーザを削除する場合、そのユーザを参照する可能性のある他のワークフロー コンポーネントについては警告は行われません。以下のアクションは必ず更新しておいてください。

タスク ルーティング指定（3-30 ページの「タスク ルーティングの管理」を参照）

[ユーザにタスクを割り当て] アクション（6-49 ページの「タスクをユーザに割り当てる」を参照）

[ルーティング テーブルを使用してタスクを割り当て] アクション（6-53 ページの「ルーティング テーブルによりタスクを割り当てる」を参照）

[電子メール メッセージを送信] アクション（6-78 ページの「電子メール メッセージを送信」を参照）

ユーザを削除する手順は、以下のとおりです。

1. 任意のアクティブなオーガニゼーションについて [ユーザ] フォルダを右クリックし、[ユーザを削除] を選択して [ユーザを削除] ダイアログ ボックスを表示します。このダイアログボックスにはシステム内で定義されているすべてのユーザが表示されます。

図 3-14 [ユーザを削除] ダイアログ ボックス

2. 削除するユーザの左のチェック ボックスをオンにして [OK] をクリックします。
3. 削除の確認を求めるメッセージが表示されたら [はい] をクリックします。

ロールの保守

ロールは、あるグループに属する個人が共通して担う一定の範囲の責任分担、能力、または権限レベルです。ロールがメンバーとして属せるのは 1 つのオーガニゼーションに限られますが、同じ名前を複数のオーガニゼーションで使用することは可能です。たとえば、Org1 と Org2 という 2 つのオーガニゼーションで *Supervisor* という名前のロールを定義できます。これらは、ロール名は同じですが、異なるロールです。名前は同じでも、Org1 の *Supervisor* と Org2 の *Supervisor* は同じものではありません。

ロールは、WebLogic Server 内のグループにマップされます。先に説明したロールとオーガニゼーションを例にとると、Org1 の *Supervisor* を *SupervisorOrg1* という名前のグループに、Org2 の *Supervisor* を *SupervisorOrg2* という名前のグループにマップできます。

ロールは各オーガニゼーション内で定義および表示されます。特定のオーガニゼーションに所属するロールを表示するには、WebLogic Integration のメイン ウィンドウの [オーガニゼーション] ドロップダウン リストからオーガニゼーションを選択し、[ロール] フォルダを展開します。ロールのプロパティを表示するにはそのロールをダブルクリックします。

注意： ロールを追加、更新、または削除するには 管理ユーザ パーミッションが必要です。パーミッション レベルの詳細については、3-26 ページの「ユーザおよびロールへのパーミッションの割り当て」を参照してください。

ロールの作成

ロールを作成機能によって、現在の WebLogic Server のセキュリティ レルムにおける新しいロールの作成、およびそのロールの WebLogic Server のグループへのマッピングが可能になります。この機能は WebLogic Integration Studio が管理可能レルム内で実行されている場合にのみアクセスできます。

ロールを作成したら、WebLogic Server 上のグループにマッピングする必要があります。そのグループは、ロールの作成時に自動的に作成されるように設定できます。または、あらかじめ WebLogic Server Administration Console で作成しておくことも可能です。WebLogic Integration グループの作成に関する情報について

は、次の URL に掲載されている『*WebLogic Security の管理*』の「互換性セキュリティの使い方」にある「互換性レルムでのグループの定義」の節を参照してください。

```
http://edocs.beasys.co.jp/e-docs/wls/docs70/secmanage/security6.html
```

新しいロールを追加する手順は、以下のとおりです。

1. フォルダ ツリーの上の [オーガニゼーション] フィールドから、ロールを作成するオーガニゼーションを選択します。
2. フォルダ ツリーで [ロール] フォルダを右クリックし、メニューから [ロールを作成] を選択して [ロールを作成] ダイアログ ボックスを表示します。

図 3-15 [ロールを作成] ダイアログ ボックス

3. [ID] フィールドにロールの名前として有意な名前を入力します。

4. （省略可能）[カレンダー] ドロップダウン リストを使用して、ロールにカレンダーを割り当てます。この機能の詳細は、3-4 ページの「ビジネス カレンダーの管理」を参照。
5. 以下のいずれか 1 つを実行します。
   - [WLS グループ] ドロップダウン リストを使用して、WebLogic Server で定義されている既存のグループにそのロールを割り当てます。
   - WebLogic Server で [ロールと同じ名前のグループにマップ] チェック ボックスをオンにして、そのロールと同じ名前の新しいグループを作成します。
6. ダイアログ ボックスの [メンバー] セクションで、このロールのメンバーにするユーザ（複数可）の左のボックスをチェックします。
7. [OK] をクリックして作成した新しいロールを保存します。操作を取り消すには、[取消し] をクリックします。

ロールの更新

ロールは、カレンダーとメンバーの変更によって更新できます。

既存のロールを更新する手順は、以下のとおりです。

1. フォルダ ツリーの上の [オーガニゼーション] フィールドから、ロールを定義するオーガニゼーションを選択します。
2. フォルダ ツリーで [ロール] フォルダを展開してロール名を右クリックし、ポップアップ メニューから [プロパティ] を選択して、[ロール プロパティ] ダイアログ ボックスを表示します。

図 3-16 [ロール プロパティ] ダイアログ ボックス

3. 必要に応じてカレンダーおよびロール メンバーに変更を加えます。
4. [OK] をクリックして更新した内容を保存します。操作を取り消すには、[取消し] をクリックします。

ロールを削除する

ロールを削除すると、WebLogic Server セキュリティ レルムおよび WebLogic Integration データベースからロールが削除されます。

注意 : ロールを削除する場合、そのロールを参照する可能性のある他のワークフロー アクションについては警告は行われません。以下のアクションはロールを参照する可能性があるため、実行時におけるサーバ例外の発生を防ぐため必ず更新してください。

- タスク ルーティング指定（3-30 ページの「タスク ルーティングの管理」を参照）

[ユーザにタスクを割り当て] アクション（6-49 ページの「タスクをユーザに割り当てる」を参照）

[ロールにタスクを割り当て] アクション（6-51 ページの「ロールに対するタスクを割り当てる」を参照）

[ルーティング テーブルを使用してタスクを割り当て] アクション（6-53 ページの「ルーティング テーブルによりタスクを割り当てる」を参照）

[電子メール メッセージを送信] アクション（6-78 ページの「電子メール メッセージを送信」を参照）

ロールを削除する手順は、以下のとおりです。

1. フォルダ ツリーの上の [オーガニゼーション] フィールドから、ロールを定義するオーガニゼーションを選択します。
2. フォルダ ツリーで [ロール] フォルダを展開してロール名を右クリックして、ポップアップ メニューから [削除] を選択します。
3. 「カレンダーを削除」という警告メッセージが表示されたら [はい] をクリックします。削除を取り消すには [いいえ] をクリックします。

ロールに対するマッピングを変更する

ロールに割り当てられたタスクを再ルーティングするには、ロールのマッピングを変更する必要があります。ロールに割り当てられたタスクを再ルーティングするには、対象ロールがすでにマッピングされた WebLogic Server グループを選択します。

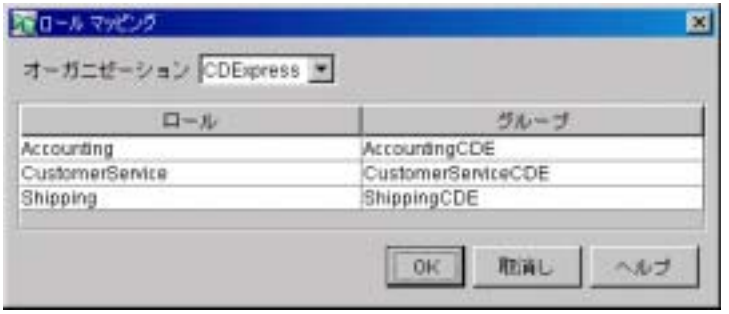
WebLogic Server グループの作成に関する情報の詳細については、次の URL に掲載されている『WebLogic Server 管理者ガイド』の「WebLogic Security の管理」にある「グループの定義」の節を参照してください。

```
http://edocs.beasys.co.jp/e-docs/wls/docs70/secmanage/security7.h
tmll
```

ロールに対するマッピングを変更する手順は、以下のとおりです。

1. Studio のメイン ウィンドウから [コンフィグレーション | ロール マッピング] を選択します。[ロール マッピング] ダイアログ ボックスが表示されます。

図 3-17 [ロール マッピング] ダイアログ ボックス



2. [オーガニゼーション] ドロップダウン リストから、WebLogic Server グループに割り当てるロールを含むオーガニゼーションを選択します。現在のマッピングが表形式で表示されます。
3. マッピングを変更するロールまたはグループを選択します。変更するグループ名の右に下向きの矢印が表示されます。
4. [グループ] フィールドでドロップダウンの矢印をクリックし、ドロップダウン リストで定義済みの WebLogic Server グループからロールのマッピング先となる新しい WebLogic Server グループを選択します。
5. [OK] をクリックして手順を完了します。[取消し] をクリックすると操作はキャンセルされます。

ユーザおよびロールへのパーミッションの割り当て

パーミッション レベルを指定すると、Studio 機能を保護し、アクセスを制御することができます。ロールとユーザは、対応するパーミッション レベルを設定されている場合にのみ、以下の表に示すタスクを実行できます。

表 3-1 パーミッション レベル

パーミッション レベル	ユーザまたはロールに許可される作業
システムの構成	■ ビジネス カレンダーの追加、更新、削除。 ■ イベント キー インタフェースの追加、更新、削除。
コンポーネントの構成	■ ビジネス オペレーションの定義、更新、削除。 ■ プラグイン のロードおよびコンフィグレーション。
管理ユーザ	■ ユーザ、ロール、およびオーガニゼーションの追加、更新、削除。 ■ ロールのグループへのマップ。 ■ ユーザおよびロールに対するパーミッション レベルの指定。 ■ タスクのルーティング指定の処理。
インスタンスのモニタ	■ ワークフロー変数、タスクのプロパティの更新を含むワークフローインスタンスのモニタ。 ■ 作業負荷レポートと統計レポートの処理。
テンプレートの作成	■ テンプレートの作成とテンプレート定義。 ■ テンプレート定義のオープン。 ■ テンプレートおよびテンプレート定義プロパティの設定。 ■ テンプレートおよびテンプレート定義変数の設定。
テンプレートの削除	テンプレートとテンプレート定義の削除。
テンプレートの実行	■ 特定のオーガニゼーション内での、Worklist クライアントまたはカスタム クライアントからのワークフローの開始。 ■ ワークフロー インスタンス変数およびプロパティの設定。
インスタンスのモニタ	■ インスタンス タスクの割り当ておよび割り当て解除。 ■ インスタンス プロパティの取得および設定。

ユーザは所属ロールのパーミッションを継承するため、ロールのパーミッションを定義し、ユーザを定義することをお勧めします。その手順については以下の節で説明します。

注意： ユーザおよびロールにパーミッションを割り当てるには、管理ユーザパーミッションが必要です。パーミッション レベルの詳細については、3-26 ページの「ユーザおよびロールへのパーミッションの割り当て」を参照してください。

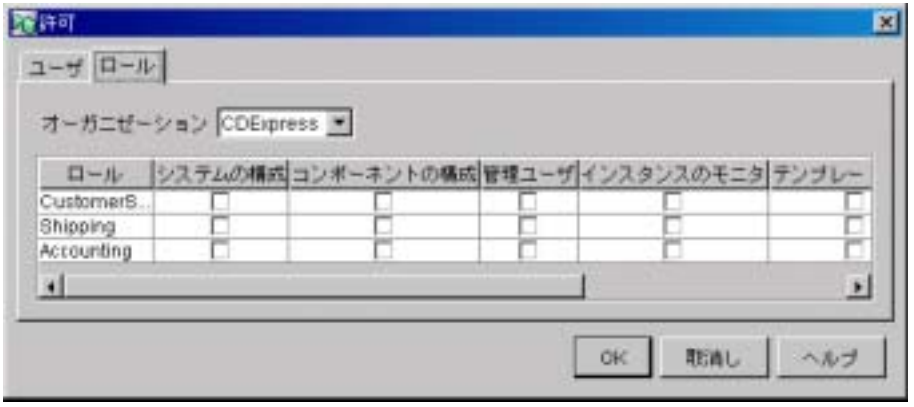
ロールに対してパーミッションを設定する

ロールは、マッピング先の WebLogic Server グループに対するパーミッション レベルを継承します。ロールに対して設定されたパーミッション レベルは、追加および削除できます。ロールに対して加えた変更はすべて、そのロールのマッピング先のグループにも反映されます。

ロールに対してパーミッション レベルを設定する手順は、以下のとおりです。

1. Studio のメイン ウィンドウから、[コンフィグレーション | パーミッション] の順に選択します。[パーミッション] ダイアログ ボックスが表示されます。
2. [ロール] タブを選択します。

図 3-18 [パーミッション] ダイアログ ボックス : [ロール] タブ



3. [オーガニゼーション] ドロップダウン リストから、ロール パーミッション を設定するオーガニゼーションを選択します。[パーミッション] ダイアログ ボックスに、オーガニゼーション内のすべてのロールと現在割り当て済みの すべてのパーミッションが表示されます。
4. 必要に応じてチェック ボックスにチェックする、またはチェックを外しま す。
5. [OK] をクリックして変更を適用します。[取消し] をクリックすると操作は キャンセルされます。

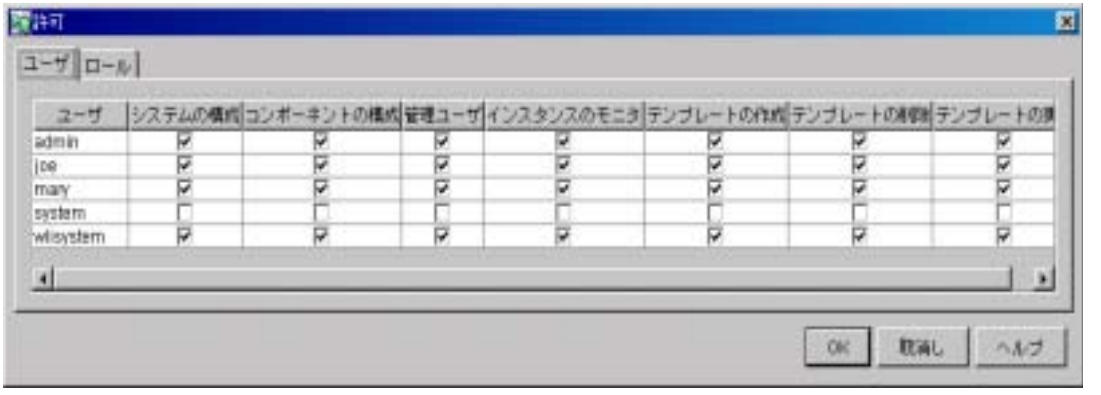
ユーザに対してパーミッションを設定する

ユーザは、所属するロールに対するパーミッションのレベルを継承します。ユーザに、その所属ロールに対して定義されているパーミッションのレベル以外のレベルを追加することは可能ですが、ユーザが所属ロールから継承したパーミッションを削除できません。ユーザに追加するパーミッションはユーザに固有ですので、そのユーザが所属するロールには反映されません。

ユーザに対してパーミッション レベルを設定する手順は、以下のとおりです。

1. Studio のメイン ウィンドウから、[コンフィグレーション | パーミッション] の順に選択します。[パーミッション] ダイアログ ボックスが表示されます。
2. [ユーザ] タブを選択します（選択されていない場合）。

図 3-19 [パーミッション] ダイアログ ボックス : [ユーザ] タブ



[パーミッション] ダイアログ ボックスに、現在定義済みのすべてのユーザと現在割り当て済みのすべてのパーミッションが表示されます。パーミッションがチェックされているがグレー表示されている場合は、ユーザが所属するロールから継承しているパーミッションですので、そのパーミッションを削除することはできません。

3. 必要に応じてチェック ボックスにチェックする、またはチェックを外します。
4. [OK] をクリックして変更を適用します。[取消し] をクリックすると操作はキャンセルされます。

タスク ルーティングの管理

タスク ルーティングは、オーガニゼーションごとに定義して、現在割り当てられているタスクを指定された期間、一時的に別のユーザまたはロールに再ルーティングできます。タスク ルーティング機能は、指定するユーザに割り当てられたすべてのタスクの再ルーティングに使用されます。タスクは、ユーザ、ロール、ロール内のユーザからルーティングできます。これらの区別、および割り当てられたタスクに関する詳細については、6-48 ページの「手動タスクの設定」を参照してください。

注意： ルーティング機能では、ユーザに割り当てられたタスクのみ再ルーティングされ、ロールに割り当てられたタスクは再ルーティングされません。再ルーティングされたタスクは、別のユーザ、ロールのユーザ、またはロールに送ることができます。ロールに割り当てられたタスクの再ルーティングについては 3-25 ページの「ロールに対するマッピングを変更する」を参照してください。

また、個々のタスクを特定の条件に基づいて再ルーティングできます。これは、タスク ノード内で指定されるアクションを使って実行されます。詳細については、6-53 ページの「ルーティング テーブルによりタスクを割り当てる」を参照してください。

注意： タスク ルーティングを管理するには、管理ユーザ パーミッションが必要です。パーミッション レベルの詳細については、3-26 ページの「ユーザおよびロールへのパーミッションの割り当て」を参照してください。

タスク ルーティング指定を表示する

オーガニゼーションに対するタスク ルーティング指定を表示する手順は、以下のとおりです。

1. フォルダ ツリーの上の [オーガニゼーション] フィールドから、ルーティングを表示するオーガニゼーションを選択します。
2. フォルダ ツリーで [ルーティング] フォルダを右クリックして、ポップアップ メニューから [開く] を選択して [ルーティング] ウィンドウを表示します。

図 3-20 [ルーティング] ダイアログ ボックス

[ルーティング] ウィンドウには、ルーティング指定ごとに次の情報が表示されます。

[ユーザ]	すべてのタスクのルーティング元のユーザに与えられたユーザ ID
[ルーティング先]	全タスクのルーティング先のユーザ、ロール、またはルーティング先ロール内のユーザの ID

[開始]	タスク ルーティングを開始すべき日付と時間
[終了]	タスク ルーティングを終了すべき日付と時間

ルーティング指定を追加する

タスク ルーティング指定を追加する手順は、以下のとおりです。

1. フォルダ ツリーの上の [オーガニゼーション] フィールドから、ルーティング指定を削除するオーガニゼーションを選択します。
2. フォルダ ツリーで [ルーティング] フォルダを右クリックし、ポップアップ メニューから [開く] を選択して [ルーティング] ウィンドウを表示します。
3. [ルーティング] ダイアログ ボックスで、[追加] をクリックして [タスクを再ルーティング] ダイアログ ボックスを表示します。

図 3-21 [タスクを再ルーティング] ダイアログ ボックス

4. [ルーティング元] ドロップダウン リストから、全タスクのルーティング元のユーザを選択します。
5. [ルーティング先] ドロップダウン リストから、全タスクのルーティング先のユーザ、ロール、またはロール内のユーザを選択して、対応するラジオ ボタンを選択します。

   注意 : ロール内のユーザ（[ロール内のユーザ] ラジオ ボタン）を選択すると、そのロール内の各ユーザに割り当てられているタスクの数が点検され、その数が最も少ないユーザが選択されて、再ルーティングされるすべてのタスクがこのユーザに割り当てられる、という方法でワークロード バランシングが行われます。

6. 発効日を指定するには [開始] ペインで年度と月を選択し、月表示から日付をクリックします。

7. 同様に、タスク再ルーティングの有効期限終了日を [終了] ペインで指定します。
8. [OK] をクリックして再ルーティング指定を保存します。操作を取り消すには、[取消し] をクリックします。

タスク ルーティング指定を更新する

タスク 再ルーティング指定を更新する手順は、以下のとおりです。

1. フォルダ ツリーの上の [オーガニゼーション] フィールドから、ルーティング指定を削除するオーガニゼーションを選択します。
2. フォルダ ツリーで [ルーティング] フォルダを右クリックし、ポップアップ メニューから [開く] を選択して [ルーティング] ウィンドウを表示します。
3. タスク ルーティングのリストが表示されるので、そのリストから更新するルーティングを選択します。
4. [更新] をクリックして、[タスクを再ルーティング] ダイアログ ボックスを表示します。
5. 必要に応じて、[ルーティング先]、[開始]、および [終了] の値を変更します。
6. [OK] をクリックして、更新した内容を保存します。操作を取り消すには、[取消し] をクリックします。

タスクのルーティング指定の削除

タスク 再ルーティング指定を削除する手順は、以下のとおりです。

1. フォルダ ツリーの上の [オーガニゼーション] フィールドから、ルーティング指定を削除するオーガニゼーションを選択します。
2. フォルダ ツリーで [ルーティング] フォルダを右クリックし、ポップアップ メニューから [開く] を選択して [ルーティング] ウィンドウを表示します。
3. タスク ルーティングのリストが表示されるので、そのリストから更新するルーティングを選択します。

4. 「再ルーティングを削除」という警告メッセージが表示されたら [はい] をクリックします。削除を取り消すには [いいえ] をクリックします。

再ルーティング タスク リストをリフレッシュする

[ルーティング] ダイアログ ボックスの [最新の情報に更新] をクリックして、再ルーティング タスク リストをリフレッシュし、[ルーティング] ダイアログ ボックスを最初に呼び出してから現在までに加えた変更を表示します。

4 ワークフロー リソースのコンフィグレーション

この章では、システムおよびアプリケーション コンポーネントのコンフィグレーション方法について説明します。

- リソース コンフィグレーション タスクの概要
- プラグインのコンフィグレーション
- ビジネス オペレーションのコンフィグレーション
- イベント キーのコンフィグレーション
- リポジトリにあるエンティティの管理

リソース コンフィグレーション タスクの概要

この節で説明するすべてのタスクは、ワークフロー設計の前段階またはその設計中に実行できますが、ここで触れるリソースは、ワークフロー テンプレートにアクセスすることなくコンフィグレーションを行えます。場合によってはワークフロー設計作業は、これらのリソースが既に設定されていることを前提としています。たとえば、ビジネス オペレーションなどは、ワークフローによって呼び出される前に定義しておく必要があります。定義済みのリソースは、システム内のすべてのワークフロー、ユーザ、およびオーガニゼーションにとってグローバルに利用可能になります。これらのタスクの実行については、特に決まった順序はありません。

注意： また、イベント キーのコンフィグレーションもタスクとして予定されている場合は、ワークフロー式言語と Studio Expression Builder および XPath Wizard の各ツールについての学習も必要です。ワークフロー式に関する詳細については、第 8 章「ワークフロー式の使用法」を参照してください。

- プラグインのロードとコンフィグレーション Studio を通じてアクセスできる機能を追加するクライアント コンポーネントまたはサーバ コンポーネントをカスタムで開発した場合は、それらのコンポーネントのコンフィグレーションを先に完了してください。プラグインの開発方法については、『*WebLogic Integration BPM プラグイン プログラミング ガイド*』を参照してください。プラグインのロードとコンフィグレーションについては、4-3 ページの「プラグインのコンフィグレーション」を参照してください。
- ビジネス オペレーションの定義。カスタム クラスや EJB (Enterprise JavaBeans) などの Java コンポーネントを開発した場合は、それらに対するインタフェースを*ビジネス オペレーション*の形で設定する必要があります。ワークフローによって呼び出されたこのインタフェースを通じて、Java コンポーネントのメソッドに付与された機能が呼び出されます。ビジネス オペレーションが定義済みでなければ、ワークフローによる呼び出しは不可能です。手順の詳細は、4-9 ページの「ビジネス オペレーションのコンフィグレーション」を参照してください。別の方法としては、あらかじめエクスポートしておいたビジネス オペレーションを既存のワークフロー パッケージからインポートします。その手順については、11-5 ページの「ワークフロー パッケージのインポート」を参照してください。
- イベント キーのコンフィグレーション。ワークフローまたはそこに含まれるノードが、JMS (Java Message Service) キューへの着信 XML メッセージなどの*イベント*によってトリガされる仕組みとなっている場合は、着信ドキュメントから適切なデータを取り出してイベントをトリガするためのイベント キーを設定できます。また、イベント キーのコンフィグレーションは、ワークフローの設計中に行うこともできるため、それらを事前に設定しておくこともできます。その手順については、4-21 ページの「イベント キーのコンフィグレーション」を参照してください。別の方法としては、あらかじめエクスポートしておいたイベント キーを既存のワークフロー パッケージからインポートします。その手順については、11-5 ページの「ワークフロー パッケージのインポート」を参照してください。
- リポジトリの設定。XML データ 変換操作、XML メッセージの内部キュー、または外部トピックや外部キューへの送信が行われる場合は、XML ドキュ

メント、スタイル シート、その他のエンティティをあらかじめリポジトリにインポートしておくと便利です。そうすることによって、データベースの一元管理が可能になり、接続されている任意の Studio クライアントから自在にアクセスできるようになります。詳細については、4-27 ページの「リポジトリにあるエンティティの管理」を参照してください。別の方法としては、あらかじめエクスポートしておいたリポジトリ エンティティを既存のワークフロー パッケージからインポートします。その手順については、11-5 ページの「ワークフロー パッケージのインポート」を参照してください。

プラグインのコンフィグレーション

プラグインは、EJB として実装されている Java クラスの集合で、一部のワークステーション コンポーネントに備わる機能を拡張するものです。プラグインは、実際の環境に合わせて既存の WebLogic Integration 機能をカスタマイズする手段となり、また、環境固有の機能を追加する手段ともなります。

プラグインは以下のワークフロー コンポーネントの機能を拡張します。

- ワークフロー テンプレート
- ワークフロー テンプレート定義
- 開始ノード
- イベント ノード
- 完了ノード
- 変数
- アクション
- 関数（式の構成部分）

プラグインが以上のワークフロー コンポーネントのいずれかに対して開発される場合は、Studio の対応するダイアログ ボックスも、そのプラグイン機能にアクセスできるように変更されます。たとえば、Studio では開始ノードの [プロパティ] ダイアログ ボックスに、ワークフローの開始をトリガするためのデフォルトの方法がいくつか用意されています。時限方式、手動操作、呼び出し、そしてイベントによるトリガです。デフォルトの方法を拡張するため、開発者は電子メール メッセージの受信など、現実の環境でワークフローを開始するために最

適なカスタムのワークフロー トリガ イベントを指定するプラグインを作成できます。このプラグインによる方法は、[開始のプロパティ] ダイアログ ボックスにオプションとして表示されます。

開発されたプラグインが WevLogic Server にデプロイされると、WebLogic Integration での使用が可能になります。WebLogic Integration の起動時に、WebLogic Server 上で利用可能なプラグインの有無を調べるためサーバのチェックが行われます。

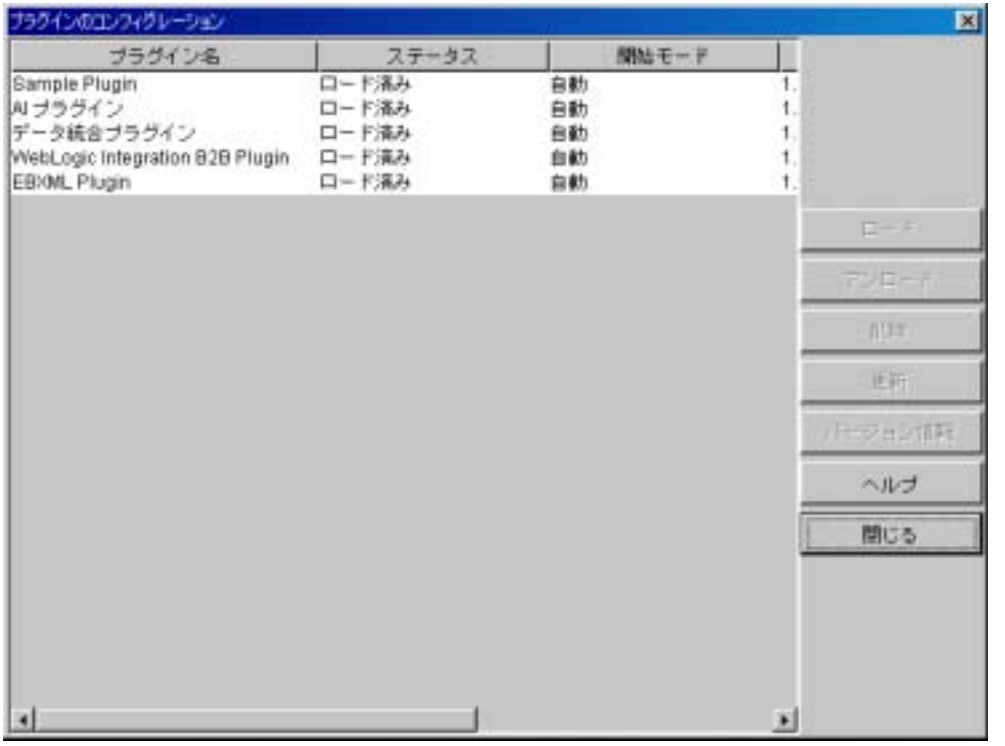
利用可能なプラグインは、Studio を使用してロードしアクティブにしない限り使用できません。また、使用する前にプラグインに対して一定のコンフィグレーション設定を指定しなければならない場合もあります。

注意 : プラグインのロードまたはコンフィグレーションを行うには、コンポーネントの構成パーミッションが必要です。パーミッション レベルの詳細については、3-26 ページの 「ユーザおよびロールへのパーミッションの割り当て」 を参照してください。

プラグインを表示する

プラグインを表示するには、[コンフィグレーション | プラグイン] を選択して [プラグインのコンフィグレーション] ダイアログ ボックスを表示します。

図 4-1 [プラグインのコンフィグレーション] ダイアログ ボックス



各プラグインについて表示される情報については次の表で説明します。

プラグイン名	プラグイン ソフトウェアで指定されているプラグインの名前。
ステータス	ロード済み－プラグイン ロード済み。 初期化済み－プラグインは利用可能であるがロード未完。 欠如－プラグインは登録済みのコンフィグレーションを備えているが、デプロイされておらず利用不可能。 エラー－プラグインの呼び出しのときに例外が送出された状態、または新しいバージョンのプラグイン フレームワークが必要な状態。

開始モード	自動－サービス開始時に毎回プラグインをロード。 手動－サーバ起動時に毎回手動でプラグインをロード。 無効化－プラグインのロード不可能。
バージョン	プラグインのソフトウェア バージョン番号。
ベンダ	プラグイン供給元の会社名。

また、リストからプラグインを選択して [バージョン情報] をクリックしても、そのプラグインの情報を表示できます。

プラグインをロードする

プラグインの開始モード（Start mode）が手動（Manual）であれば、WebLogic Integration サーバ セッションの起動時に毎回手動でロードできます。プラグインの開始モードが無効（Disabled）であれば、まず起動モードを手動または自動に変更してロードします。

初期化されたプラグインをロードする手順は、次のとおりです。

1. [プラグインのコンフィグレーション] ダイアログ ボックスで必要なプラグインを選択します。
2. [ロード] をクリックします。リスト内のプラグインのステータスが [ロード済み] に変わります。

使用不可能にしたプラグインをロードする手順は、以下のとおりです。

1. [プラグインのコンフィグレーション] ダイアログ ボックスで必要なプラグインを選択し、[更新] をクリックして [コンフィグレーション] ダイアログ ボックスを表示します。
2. [コンフィグレーション] ダイアログ ボックスで [開始モード] を [自動] または [手動] にして、[OK] をクリックします。
3. [プラグインのコンフィグレーション] ダイアログ ボックスでそのプラグインを選択して [ロード] をクリックします。

リスト内のプラグインのステータスが [ロード済み] に変わります。

注意 : プラグインの開始モードに加えた変更は、WebLogic Integration サーバを再起動するまで有効になりません。

プラグイン コンフィグレーションを更新する

プラグインのコンフィグレーションを行う手順は、以下のとおりです。

1. [プラグインのコンフィグレーション] ダイアログ ボックスでコンフィグレーションを行うプラグインを選択します。
2. [更新] をクリックします。コンフィグレーションを行うダイアログ ボックスが表示されます。
3. （省略可能）以下のうちいずれかのボタンをクリックする方法でプラグインの開始モードを選択します。
   - [自動] ― このオプションは、サーバの起動時に毎回プラグインをロードする場合に選択します。
   - [手動] ― このオプションは、必要に応じて毎回プラグインをロードする場合に選択します。
   - [無効化] ― このオプションは、プラグイン機能を無効にする場合に選択します。

   注意 : 設定した開始モードは、WebLogic Integration サーバが再起動されるまで有効になりません。
4. プラグインのコンフィグレーション設定を目的に合わせて指定します。

   プラグインのコンフィグレーション設定を定義する方法の詳細については、そのプラグインのオンライン ヘルプを参照してください。プラグインのヘルプにアクセスするには〔F1〕を押します（コンテキスト センシティブなプラグイン ヘルプの場合）。または、Studio のメイン メニューで [ヘルプ | プラグイン ヘルプ] を選択し、適切なプラグイン ヘルプをメニューから選択します。
5. [OK] をクリックしてコンフィグレーションを完了します。または、この操作を取り消す場合は [取消し] をクリックします。

プラグイン コンフィグレーションを削除する

プラグインのコンフィグレーションは、不要になった時点で削除できます。コンフィグレーションを削除してもプラグイン自体は削除されません。登録されているコンフィグレーションのみが削除されます。

注意 : プラグインのステータスが [欠如] でない場合は、プラグインのコンフィグレーションは削除できません。

コンフィグレーションを削除する手順は、次のとおりです。

1. [プラグインのコンフィグレーション] ダイアログ ボックスで、コンフィグレーションを削除するプラグインを選択します。
2. [削除] をクリックします。選択したプラグインに登録されていたコンフィグレーションは削除されます。しかし、プラグインは引き続き [プラグインのコンフィグレーション] ダイアログ ボックスに表示されます。

   次の表では、プラグインのコンフィグレーションを削除してから、WebLogic Integration サーバを再起動した際に発生するアクションについて説明します。

WebLogic Integration サーバが再起動された際のプラグインの状態	結果
デプロイされていない	プラグイン マネージャによってプラグインが検索されないので、プラグインは [プラグインのコンフィグレーション] ダイアログ ボックスに表示されない。
デプロイされている	プラグインは、プラグインで定義されたデフォルトのコンフィグレーション値で自動的にロードされる。

ビジネス オペレーションのコンフィグレーション

ワークフローで、Java クラスや Enterprise JavaBeans (EJB) などのビジネス ロジックを実行するソフトウェア コンポーネントを開くには、ビジネス オペレーションを定義します。ビジネス オペレーションは、パラメータとして渡された変数も含め、EJB または Java クラスのメソッド呼び出しと、その結果としてワークフローに返された値を表しています。ビジネス オペレーション機能を使用して、既存アプリケーションまたは特にそのワークフロー用にビルドされたアプリケーションを呼び出すための、カスタマイズされた関数を作成できます。ビジネス オペレーション機能を使用して、WebLogic Server に登録されている EJB および Java クラスのすべてを、そのメソッドおよびパラメータと共に表示できます。

注意： Java クラスと EJB を、Studio で表示できるようにデプロイする方法については、『*WebLogic Integration の起動、停止およびカスタマイズ*』の「WebLogic Integration のカスタマイズ」にある「ビジネス オペレーションに対する EJB と Java クラスのデプロイ」を参照してください。

ビジネス オペレーションをいったん定義すれば、システム内のすべてのワークフローでグローバルに利用できるようになります。ビジネス オペレーションを、Java アーカイブ パッケージ ファイルの一部としてエクスポートおよびインポートすることもできます。その際、それらを参照するワークフローの有無は問題にはなりません（詳細については、第 11 章「ワークフロー パッケージのインポートとエクスポート」を参照）。

個々のワークフロー内では、ビジネス オペレーションを実行アクションを使用してビジネス オペレーションを呼び出します。また、必要に応じてメソッド呼び出しの結果をワークフロー変数に割り当てます。詳細については、6-84 ページの「ビジネス オペレーションを呼び出す」を参照してください。

ワークフローによって EJB のメソッドまたは Java クラスの非静的メソッドを呼び出すには、まず、ワークフローによってそのコンストラクタを呼び出して、サーバ上に EJB または Java クラスのインスタンスを作成する必要があります。したがって、`create()` EJB のメソッドおよび Java クラスのコンストラクタ メソッドに対してビジネス オペレーションを作成し、また、インスタンスへの参照を格納する変数を必ず作成してください。各 Java コンポーネント タイプに対

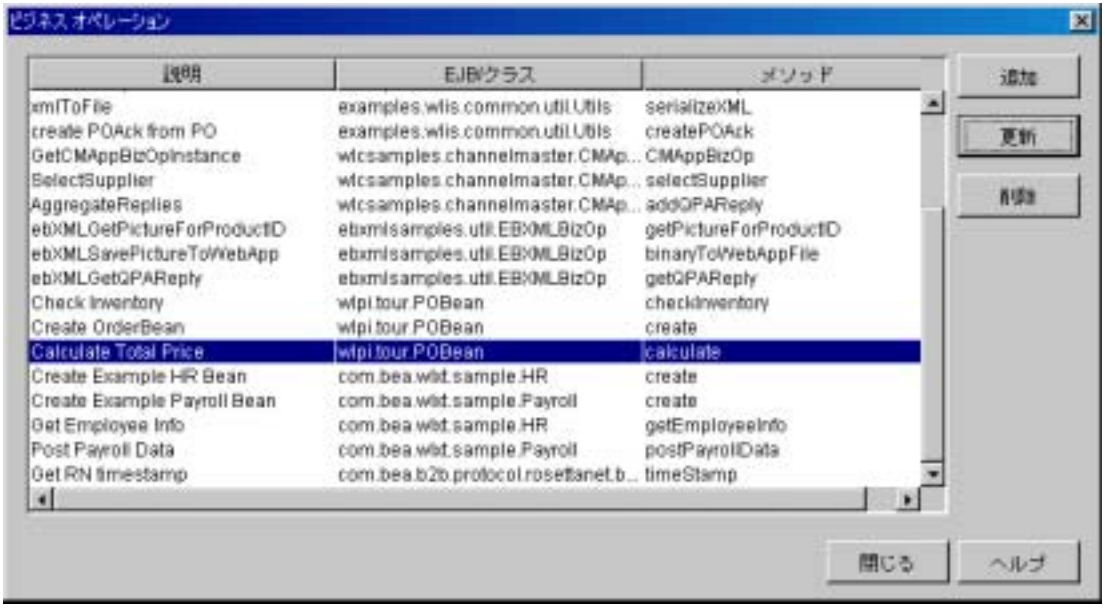
するビジネス オペレーションの定義と追加の方法など、詳細については、以下の節で説明します。また、6-84 ページの「ビジネス オペレーションを呼び出す」では、必要なメソッドを呼び出す手順および変数を割り当てる手順について説明します。また、『*WebLogic Integration BPM ユーザーズ ガイド*』の「ビジネス オペレーションの作成と実行 : Check Inventory タスクの定義」には、ビジネス オペレーションを定義するコード例が記載されています。.

注意 : ビジネス オペレーションを追加、定義、または削除するには、コンポーネントの構成パーミッションが必要です。パーミッション レベルの詳細については、3-26 ページの「ユーザおよびロールへのパーミッションの割り当て」を参照してください。

ビジネス オペレーションを表示する

ビジネス オペレーションを表示するには、[コンフィグレーション | ビジネス オペレーション] を選択して [ビジネス オペレーション] ダイアログ ボックスを表示します。

図 4-2 [ビジネス オペレーション] ダイアログ ボックス



各ビジネス オペレーションに関して表示される情報については、次の表で説明します。

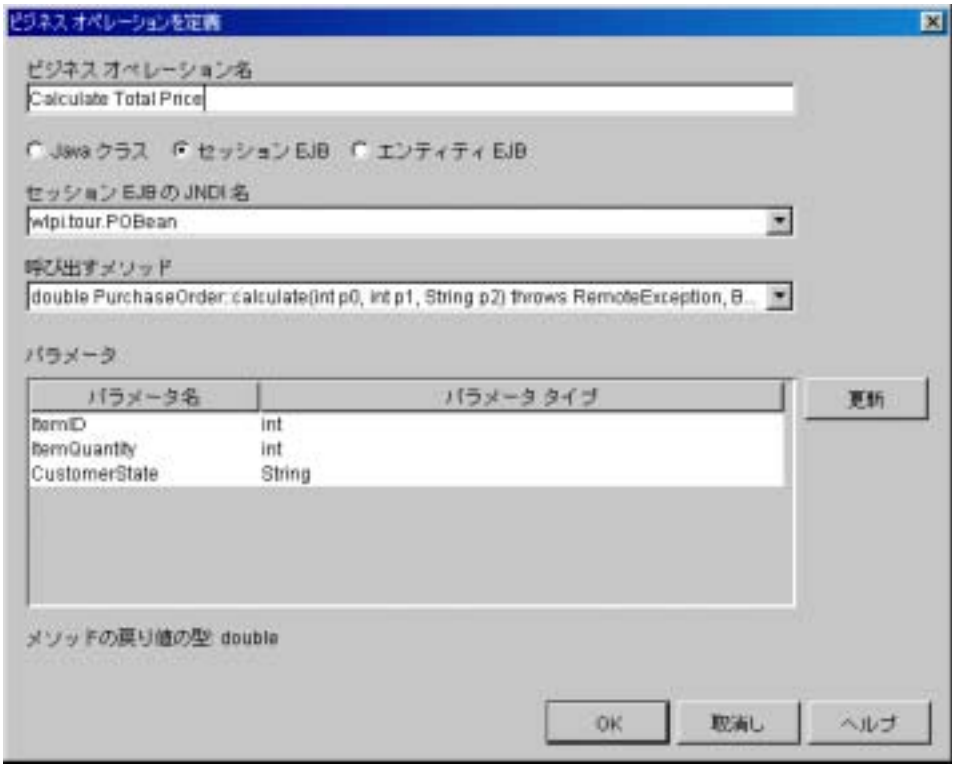
説明	ビジネス オペレーションに対して定義する名前。
EJB/ クラス	呼び出す EJB または Java クラス。
メソッド	呼び出す EJB または Java クラスのメソッド。

ビジネス オペレーションの追加

ビジネス オペレーションを作成、定義する手順は、以下のとおりです。

1. [ビジネス オペレーション] ダイアログ ボックスから、[追加] をクリックして [ビジネス オペレーションを定義] ダイアログ ボックスを表示します。

図 4-3 [ビジネス オペレーションを定義] ダイアログ ボックス



2. [名前] フィールドにビジネス オペレーションの名前として有意な名前を入力します。たとえば、`Check Inventory` など、在庫にある利用可能な品目の数を返すビジネス オペレーションが呼び出される場合があります。実行時に Java クラスまたは EJB のインスタンスを作成するメソッドには、`Create Order Processing EJB Instance` などのように、メソッドの目的や作成するクラスまたは Bean の名前を示すビジネス オペレーション名を付ける。

   注意 : また、インスタンスを参照するために、対応する変数も作成する必要があります。

3. そのビジネス オペレーションが呼び出すソフトウェアコンポーネントを指定します。オプションは次のとおり。

   - Java クラス―ビジネス オペレーションは、WebLogic Server 上の Java クラスのメソッドを呼び出します。Java クラスはシリアライズ可能または

シリアライズ可能ではなくてもかまいません。シリアライズ可能ではない Java クラス参照は、トランザクションの間のみ存続します。ワークフロー インスタンスが静止状態に到達する度に再び作成する必要があります。4-13 ページの「Java クラスを呼び出すビジネス オペレーションを追加する」で説明する手順に従ってください。

- セッション EJB ービジネス オペレーションは、WebLogic Server 上のセッション EJB のメソッドを呼び出します。ステートフル セッション EJB 参照はトランザクションの間のみ存続します。ワークフロー インスタンスが静止状態に到達する度に再び作成する必要があります。ステートレス EJB 参照は、ワークフロー インスタンスの期間存続します。4-16 ページの「セッション EJB を呼び出すビジネス オペレーションを追加する」で説明する手順に従ってください。
- エンティティ EJB ービジネス オペレーションは、WebLogic Server 上のエンティティ EJB のメソッドを呼び出します。エンティティ EJB は永続的なデータで、少なくともワークフロー インスタンスの有効期間中は存続するが、多くはワークフロー インスタンスの有効期間を過ぎても存続する。4-18 ページの「エンティティ EJB を呼び出すビジネス オペレーションを追加する」で説明する手順に従ってください。

4. [ビジネス オペレーションを定義] ダイアログ ボックスで、[OK] をクリックしてビジネス オペレーションを保存します。このビジネス オペレーションは [ビジネス オペレーション] ダイアログ ボックスの有効なビジネス オペレーションのリストに追加されます。

Java クラスを呼び出すビジネス オペレーションを追加する

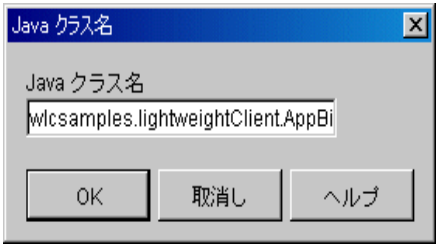
Java クラスの非静的メソッドを呼び出すビジネス オペレーションを作成する場合は、そのクラスのコンストラクタ メソッドを呼び出すビジネス オペレーションも作成する必要があります（詳細については、6-85 ページの「EJB または Java クラス インスタンスを作成するためのビジネス オペレーションを呼び出す」を参照）。このビジネス オペレーションは、そのクラスの非静的メソッドを呼び出す前にワークフローから呼び出す必要があるため、必ずその機能を識別できる有意な名前を付けてください。Java クラス インスタンスが実行時に作成される際に、そのインスタンスへの参照を格納する Java Object 型の変数も作成する必要があります。変数については、5-28 ページの「変数に関する作業」を参照してください。

最後に述べておきますが、ビジネス オペレーションに名前を付ける際、ビジネス オペレーションが静的メソッド、非静的メソッドのいずれを呼び出すのかを明示する名前にします。それにより、ワークフロー デザイナは、最初にコンストラクタ メソッドを呼び出す必要があるのかどうか判断します。

Java クラスを呼び出すビジネス オペレーションを定義する手順は、以下のとおりです。

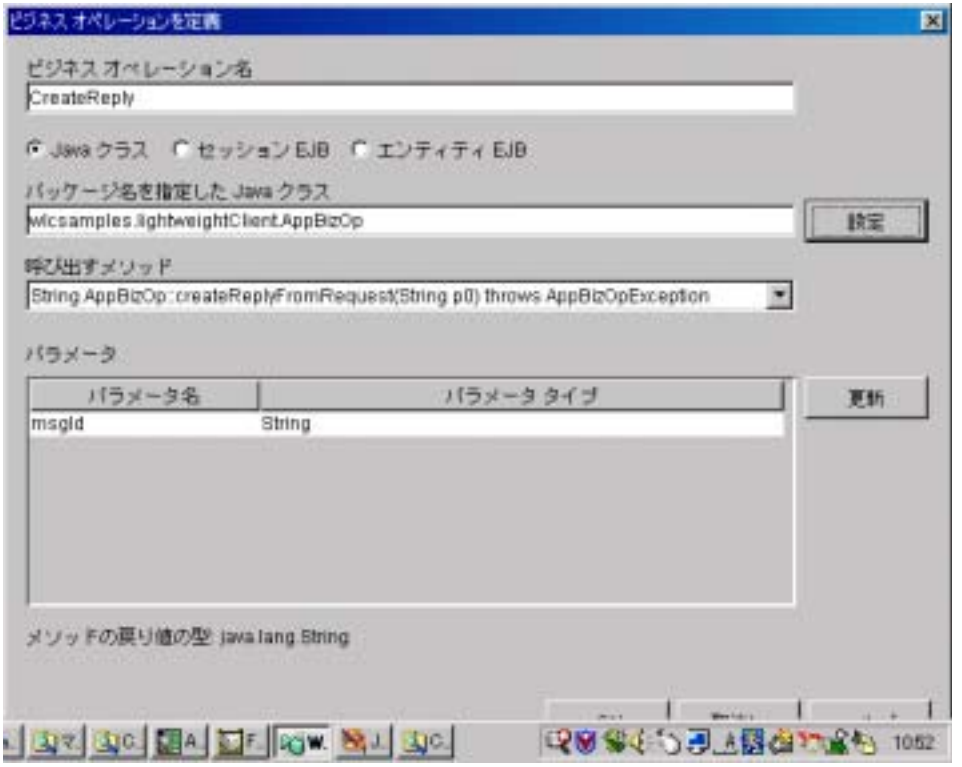
1. [Java クラス] ラジオ ボタンを選択します。
2. [設定] をクリックして [Java クラス名] ダイアログ ボックスを表示します。

図 4-4 [Java クラス名] ダイアログ ボックス



3. Java クラスの完全修飾名（たとえば、`java.lang.string`）を入力し、[OK] をクリックする。この Java クラス名は [ビジネス オペレーションを定義] ダイアログ ボックスの [パッケージ名を指定した Java クラス] フィールドに設定されます。

図 4-5 [ビジネス オペレーションを定義] ダイアログ ボックス : [Java クラス] オプション



4. [呼び出すメソッド] ドロップダウン リストで、ワークフローでビジネス オペレーションが呼び出された際に呼び出すメソッドを選択します。このリストには、次の 3 種類の Java クラス メソッドが表示される。

   - コンストラクタ型ーこのメソッド タイプはリストの冒頭に記載されます。コンストラクタ メソッドは [Java クラス名] フィールドに入力した型のオブジェクトを返します。
   - メソッド型ーこのメソッド タイプはリストの 2 番目に記載されます。Java オブジェクト上で開始され、任意のタイプを返すことができる。
   - Static 型ーこのメソッド タイプは `Static` という語を伴い、リストの最後に記載します。静的メソッドに対してはオブジェクトを作成する必要はなく、任意の型を返すことができます。

5. (省略可能) メソッドのパラメータに有意な名前を付けるには、Parameters リストでそのパラメータを強調表示し、[更新] をクリックして [パラメータ] ダイアログ ボックスを表示し、そこで名前を変更します。

図 4-6 [パラメータ] ダイアログ ボックス

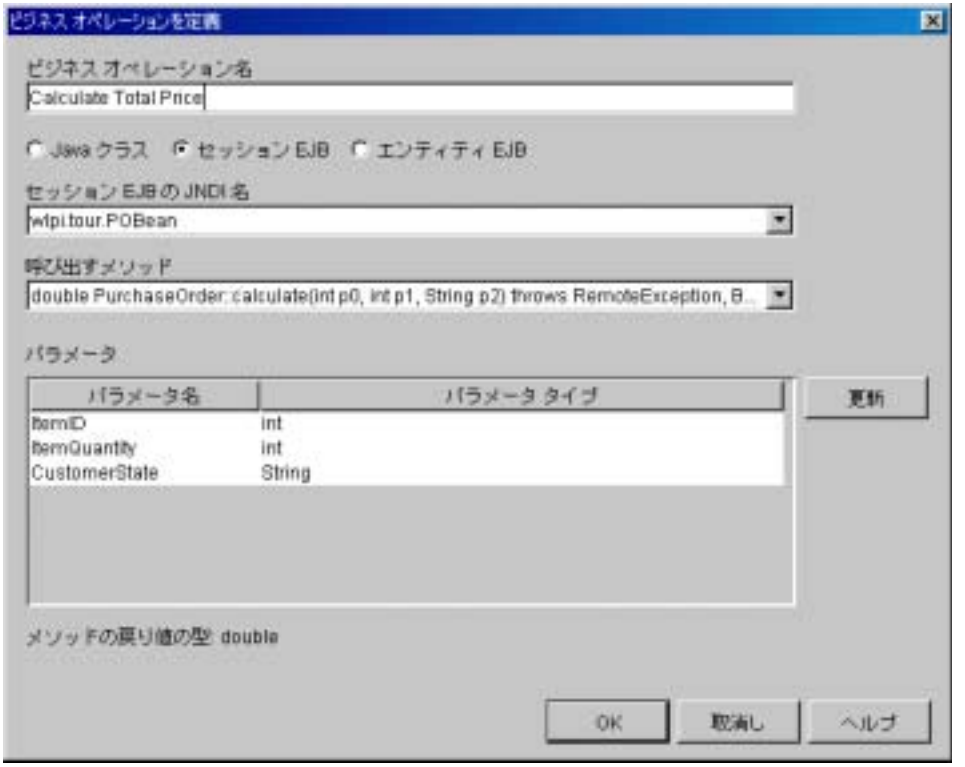
6. [名前] フィールドにパラメータの説明となる名前を入力して [OK] をクリックします。
7. Parameters リストにあるすべてのパラメータについて、手順 5 および 6 を繰り返します。

セッション EJB を呼び出すビジネス オペレーションを追加する

ワークフローに対してビジネス ロジックを提供するメソッドを呼び出すビジネス オペレーションの作成に加えて、セッション EJB (メソッドはビジネス オペレーションで呼び出す) の `create()` メソッドを呼び出すビジネス オペレーションも作成する必要があります (詳細については、6-85 ページの「EJB または Java クラス インスタンスを作成するためのビジネス オペレーションを呼び出す」を参照)。このビジネス オペレーションは、他のメソッドを呼び出す各トランザクションのワークフローから呼び出す必要があるため、必ずその機能を識別できる有意な名前を付けてください。EJB インスタンスが実行時に作成される際に、そのインスタンスへの参照を格納するセッション EJB 型の変数も作成する必要があります。変数については、5-28 ページの「変数に関する作業」を参照してください。

WebLogic Server にデプロイされるすべてのセッション EJB は、[ビジネス オペレーションを定義] ダイアログ ボックスにあるリストに、それぞれの JNDI (Java Naming and Directory Interface) 名に基づいた名前が表示されます。

図 4-7 [ビジネス オペレーションを定義] ダイアログ ボックス : [セッション EJB] オプション



セッション EJB を呼び出すビジネス オペレーションを定義する手順は、以下のとおりです。

1. [ビジネス オペレーションを定義] ダイアログ ボックスで [セッション EJB] ラジオ ボタンを選択して、セッション EJB に関係するフィールドを表示します。
2. [セッション EJB の JNDI 名] ドロップダウン リストから、セッション EJB の JNDI 名を選択します。
3. [呼び出すメソッド] ドロップダウン リストで、ワークフローでビジネス オペレーションが呼び出された際に呼び出すメソッドを選択します。

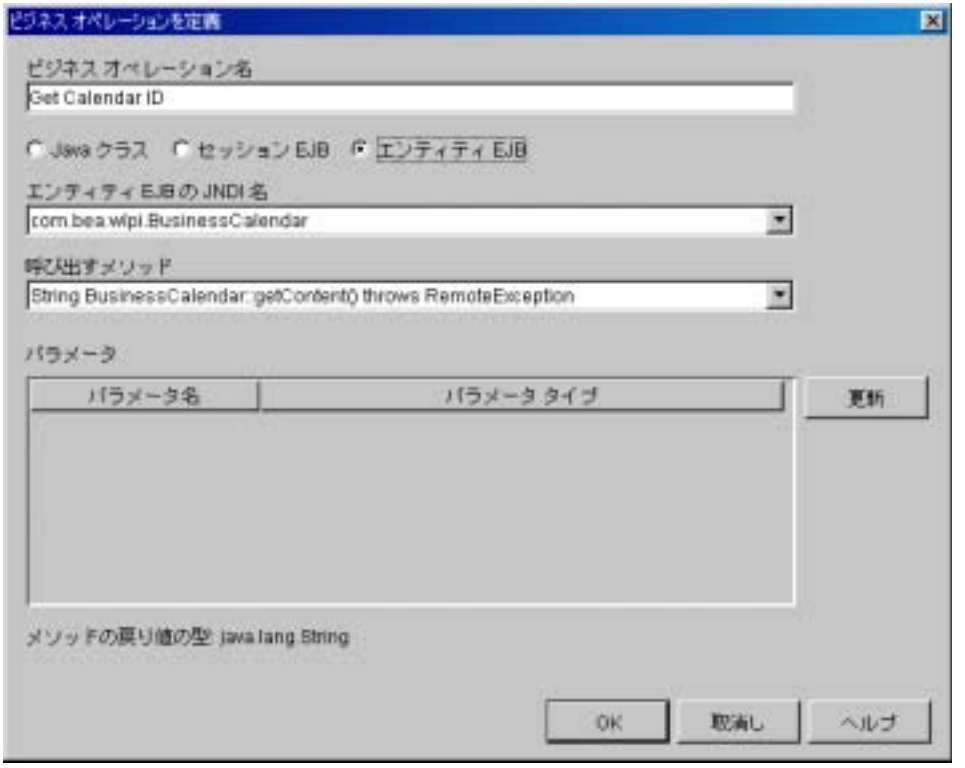
4. （省略可能）メソッドのパラメータに有意な名前を付けるには、Parameters リストでそのパラメータを強調表示し、[更新] をクリックして [パラメータ] ダイアログ ボックスを表示し、そこで名前を変更します。
5. [名前] フィールドにパラメータの説明となる名前を入力して [OK] をクリックします。
6. Parameters リストにあるすべてのパラメータについて、手順 4 および 5 を繰り返します。

エンティティ EJB を呼び出すビジネス オペレーションを追加する

ワークフローに対してビジネス ロジックを提供するメソッドを呼び出すビジネス オペレーションの作成に加えて、エンティティ EJB (メソッドはビジネス オペレーションで呼び出す) の `create()` メソッドを呼び出すビジネス オペレーションも作成する必要があります（詳細については、6-85 ページの「EJB または Java クラス インスタンスを作成するためのビジネス オペレーションを呼び出す」を参照）。このビジネス オペレーションは、その EJB の他のメソッドを呼び出す前にワークフローから呼び出す必要があるため、必ずその機能を識別できる有意な名前を付けてください。EJB インスタンスが実行時に作成される際に、そのインスタンスへの参照を格納するエンティティ EJB 型の変数も作成する必要があります。変数については、5-28 ページの「変数に関する作業」を参照してください。

WebLogic Server にデプロイされるすべてのエンティティ EJB は、[ビジネス オペレーションを定義] ダイアログ ボックスにあるリストに、それぞれの JNDI 名に基づいた名前が表示されます。

図 4-8 [ビジネス オペレーションを定義] ダイアログ ボックス : [エンティティ EJB] オプション



エンティティ EJB を呼び出すビジネス オペレーションを定義する手順は、以下のとおりです。

1. [ビジネス オペレーションを定義] ダイアログ ボックスで [エンティティ EJB] ラジオ ボタンを選択して、エンティティ EJB に関係するフィールドを表示します。
2. [エンティティ EJB の JNDI 名] ドロップダウン リストからエンティティ EJB の JNDI 名を選択します。
3. [呼び出すメソッド] ドロップダウン リストで、ワークフローでビジネス オペレーションが呼び出された際に呼び出すメソッドを選択します。

4. （省略可能）メソッドのパラメータに有意な名前を付けるには、Parameters リストでそのパラメータを強調表示し、[更新] をクリックして [パラメータ] ダイアログ ボックスを表示し、そこで名前を変更します。
5. [名前] フィールドにパラメータの説明となる名前を入力して [OK] をクリックします。
6. Parameters リストにあるすべてのパラメータについて、手順 4 および 5 を繰り返します。

ビジネス オペレーションの更新

ビジネス オペレーションを更新する場合は、必ずワークフローからビジネス オペレーションを参照するビジネス オペレーションを実行アクションをすべて更新してください。このアクションの詳細については、6-84 ページの「ビジネス オペレーションを呼び出す」 を参照してください。

ビジネス オペレーションを更新する手順は、以下のとおりです。

1. [ビジネス オペレーション] ダイアログ ボックスから更新するビジネス オペレーションを選択して、[更新] をクリックします。[ビジネス オペレーションを定義] ダイアログ ボックスが表示されます。
2. 必要な変更を加え、完了したときに [OK] をクリックします。

ビジネス オペレーションの削除

注意 : ビジネス オペレーションを削除する前に、そのビジネス オペレーションがビジネス オペレーションを実行アクションを使用したワークフローによって参照されていないことを確認してください。参照しているとワークフローをアクティブにできません。削除を実行する際、参照がある場合も警告は表示されないため、削除操作に対応したビジネス オペレーションを実行アクションの更新を確実に実行してください（6-84 ページの「ビジネス オペレーションを呼び出す」 を参照）。

ビジネス オペレーションを削除する手順は、以下のとおりです。

1. [ビジネス オペレーション] ダイアログ ボックスから削除するビジネス オペレーションを選択して、[削除] をクリックします。
2. 削除の警告メッセージが表示されたら、削除する場合は [OK] を、取り消す場合は [取消し] をクリックします。

イベント キーのコンフィグレーション

ワークフローの開始やワークフロー内のノードのトリガは、イベントによって行うことができます。イベントは別のワークフローや別のアプリケーションなどの外部ソースからの非同期の通知です。開始ノードはイベント トリガ型として定義され、イベント ノードは常に外部イベントのみによってトリガできる非同期ノードです。

イベント通知は、一般的には JMS (Java Message Service) メッセージに含まれ、JMS キューで受信される XML ドキュメントの形をとります。ただし、プラグインで定義することもでき、その場合イベント通知は XML ドキュメントではなく、カスタムのトリガとなります（詳細については、『*WebLogic Integration BPM プラグイン プログラミング ガイド*』を参照）。

XML イベントの型では、実際のトリガは XML メッセージのプロローグで定義された文書型（DOCTYPE）の宣言であるか、または XML メッセージのルート要素であるかのいずれかです。開始またはイベント ノードのプロパティ ダイアログ ボックスで、イベントのトリガまたはワークフローの開始に使用する DOCTYPE またはルート要素を指定します。イベントは、ノードのプロパティ ダイアログ ボックスで指定された DOCTYPE、またはルート要素が着信 XML メッセージのそれと一致しない限りトリガされません。

DOCTYPE またはルート要素の使用に加え、イベント キーを使用してイベントをさらに修飾できます。イベント キーを使用して、開始ノードやイベント ノードを開始する着信 XML メッセージ、JMS ヘッダ、プロパティ フィールドなどの内容を指定できます。すなわち、当該ノードをトリガする特定の DOCTYPE またはルート要素を含むすべての着信 XML ドキュメントを受け入れるのではなく、ドキュメントやヘッダに含まれる一定の値を持つメッセージのみが実行中のワークフロー内のノードをトリガできるように、そのような値に基づいて着信 XML メッセージのインスタンスをフィルタ処理できます。

イベント キーは以下の 2 つの部分で構成されます。

- *キー値の式*

  キー値の式は開始ノードまたはイベント ノードの [プロパティ] ダイアログ ボックスで指定します。キー値の式は実行時に評価されるワークフロー式で、そのノードのトリガが可能となるように着信メッセージに含まれている正確なデータを指定します。開始ノードでは、この式は一般的には、着信 XML ドキュメントに含まれる特定の再帰データを参照する定数を含んでいます。それが JMS ヘッダの値である場合もあります。イベント ノードの場合、キー式には通常、実行時に一意の値を受け取る変数または関数が含まれています。キー値の式のサンプルは 5-40 ページの「イベント キーを理解する」にあります。また、キー値の式を定義する方法は、5-48 ページの「イベント トリガ型開始のプロパティを定義する」および 5-51 ページの「イベント プロパティを定義する」で説明します。

- *イベント キー式*

  これは、実行時に着信メッセージのヘッダまたは本文からキー値を返し、それを、開始ノードまたはイベント ノードの対応するキー値の式が必要とするデータ型にコンバートする式です。イベント キー式は、[コンフィグレーション] メニューからアクセスするイベント キー式のダイアログ ボックスで指定します。この式には一般的には、XML ドキュメントを解析するための XPath 言語の式、または JMS メッセージのヘッダから値を抽出するための EventAttribute() 関数の式が含まれています。

イベント キーのコンフィグレーションで、開始ノードまたはイベント ノードのプロパティ ダイアログボックスで指定した DOCTYPE またはルート要素に対応する *イベント記述子*を指定します。次にイベント キー式を指定します。これは、開始ノードまたはイベント ノードのプロパティ ダイアログ ボックスで定義されたキー値の式に対応する式で、プロセス エンジンによって実行時にこの 2 つの値を比較して一致するかどうかの判断が可能になります。イベント記述子と DOCTYPE またはルート要素の関係、およびイベント キー式とキー値の式の関係は、5-40 ページの「イベント キーを理解する」で、コード例を使用してさらに詳しく説明します。

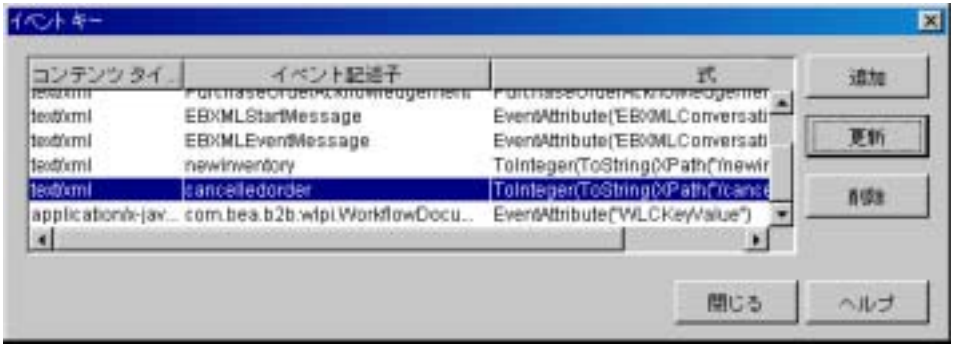
ただし、イベント キーのコンフィグレーションは、ワークフローとは独立して行うことができるため、その手順は以下で説明します。コンフィグレーションが完了したイベント キーは、すべてのオーガニゼーションのワークフローで使用可能となります。着信メッセージの内容がわかっている場合は、開始ノードかまたはイベント ノードで対応するキー値の式を設定するワークフロー デザイナが利用できるように、あらかじめイベント キー式のコンフィグレーションを済ましておくことが可能です。イベント キーを、Java アーカイブ パッケージ ファイ

ルとの間でエクスポートおよびインポートすることもできます。その際、それらを参照するワークフローの有無は問題にはなりません（詳細は、第 11 章「ワークフロー パッケージのインポートとエクスポート」を参照）。

イベント キーのコンフィグレーションを表示する

イベント キーのコンフィグレーションを表示するには [コンフィグレーション | イベント] を選択して、[イベント キー] ダイアログ ボックスを表示します。

図 4-9 [イベント キー] ダイアログ ボックス



各イベント キーに対して表示される情報について、次の表で説明します。

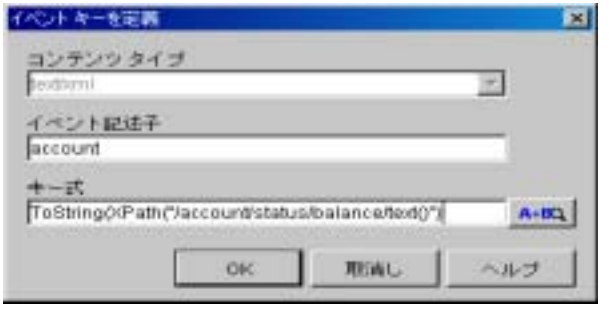
コンテンツ タイプ	デフォルトで text/xml に設定済みで編集できない。ただし、ロードされるプラグインについてプラグイン イベント タイプが使用可能な場合のみ編集できる。
イベント記述子	XML/JMS イベントでは、これは受信した XML ドキュメントの DOCTYPE またはルート要素であり、JMS メッセージ内に含まれる。
式	実行時に受信メッセージのヘッダまたは本体からキー値を生成し、それを対応する開始ノードまたはイベント ノード内のキー値の式で必要なデータ型に変換する式。

イベント キーのコンフィグレーションを追加する

イベント キーのコンフィグレーションを追加する手順は、以下のとおりです。

1. [イベント キー] ダイアログ ボックスから [追加] をクリックして [イベント キーを定義] ダイアログ ボックスを表示します。

図 4-10 [イベント キーを定義] ダイアログ ボックス



2. [コンテンツ タイプ] フィールドで、XML/JMS メッセージと対応する text/xml を選択します。
3. [イベント記述子] フィールドに、受信 XML ドキュメントの DOCTYPE またはルート要素を入力します。
4. [式] フィールドに、以下のうち 1 つを入力します。
   - JMS ヘッダまたはプロパティ フィールドから値を抽出するには、`EventAttribute()` 関数を使用して、かっこの中のフィールド名で式を作成します。
   - XML 本文から値を抽出するには、`XPath()` 関数（8-8 ページの「XPath()」を参照）を含む式、または XML 要素が文字列として返されるためのドット（）表記（8-11 ページの「XML 要素のドット表記」を参照）を使用します。また [式] ボタン  を使用して XPath Wizard を呼び出し、このウィザードを使ってサンプルの着信ドキュメントから XPath の式を自動的に生成できます。詳細については、8-31 ページの「XPath Wizard を使用する XPath 式の作成」を参照してください。

イベント キー式の構文についての詳細は、8-6 ページの「実行時のイベント データを抽出する」を参照してください。また、対応する開始ノードまたはイベント ノードで定義されているキー値の式によって返される型と一致する

データ型を返すには、XPath() 関数または EventAttribute() 関数が型キャスト関数にラップされている必要があるため注意してください。型キャスト関数については、8-16 ページの「データ型を変換する」を参照してください。

ここに入力したイベント キーの式が、開始ノードまたはイベント ノードのキー値の式によって指定された値に一致する値を返す必要があることにも注意してください。開始ノードまたはイベント ノードのキー値の式を定義する方法の詳細については、5-39 ページの「イベントおよびイベントトリガ型開始のプロパティを定義する」を参照してください。

5. [OK] をクリックします。イベント キーは、WebLogic Integration データベースのイベント キー表に保存され、[イベント キー] ダイアログボックスに表示されます。
6. [閉じる] ボタンをクリックします。

イベント キーのコンフィグレーションを更新する

イベント キーのコンフィグレーションを更新する場合、式のみの更新が可能で、イベント記述子は更新できません。

イベント キーを更新する手順は、以下のとおりです。

1. [イベント キー] ダイアログ ボックスから更新するイベント キーを選択し、[更新] をクリックして [イベント キーを定義] ダイアログ ボックスを表示します。
2. 必要に応じて式を編集します。
3. 完了したときに [OK] をクリックします。

イベント キーのコンフィグレーションを削除する

イベント キーを削除する場合は、キー値がワークフローの開始ノードまたはイベント ノードのキー値の式によって参照されないように注意してください。参照される場合は、これらのイベントはトリガされません。

イベント キーを削除する手順は、以下のとおりです。

1. [イベント キー] ダイアログ ボックスから削除するイベント キーを選択し、[削除] をクリックします。
2. 警告メッセージが表示されたら、削除する場合は [OK] を、取り消す場合は [取消し] をクリックします。

リポジトリにあるエンティティの管理

WebLogic Integration のリポジトリには、XML ドキュメント、DTD (Document Type Definition: 文書型定義) ファイル、XSL ドキュメントなどの XML エンティティを保存するために使用されるデータベース表が格納されています。Studio を使用して、リポジトリの表示、整理、リポジトリへのファイルの格納を行い、また、既存 XML エンティティをシステム内のすべてのワークフローによってグローバルに利用、再利用できるように設定できます。作成したワークフロー内で、以前格納した XML ドキュメントを参照する場合、システムにログオンしたあらゆるクライアントでこれらのドキュメントを使用できるように、リポジトリを設定できます。

たとえば、XML をクライアントに送信アクションを使用して Worklist ユーザ（6-66 ページの「Worklist アプリケーションに対する XML メッセージを送信する」を参照）と対話する場合、各ユーザが DTD ファイルに一元管理下にある場所で容易にアクセスできるように、DTD ファイルをリポジトリに格納できます。また、XSL 変換アクションを使用して XML ドキュメントを実行時に変換する場合は（6-103 ページの「XML ドキュメントの変換」を参照）、XML スタイルシート の変換ドキュメントをリポジトリに格納して、そのアクションを定義する際に容易にアクセスできるようにすると便利です。

この節では、リポジトリの初期設定について説明します。ただし、リポジトリおよびこの節で説明するすべての機能は、第 7 章「XML エンティティを操作」で述べたとおり、ワークフロー ダイアログボックスからもアクセスできます。また、リポジトリに格納されているエンティティを、ディスクのファイルをエクスポート（このオプションについては、4-38 ページの「エンティティをファイルシステムにエクスポートする」で説明）および Java アーカイブ パッケージ ファイルをエクスポートして、別のシステムに再インポートすることもできます（インポートおよびエクスポートの詳細については、第 11 章「ワークフロー パッケージのインポートとエクスポート」を参照）。

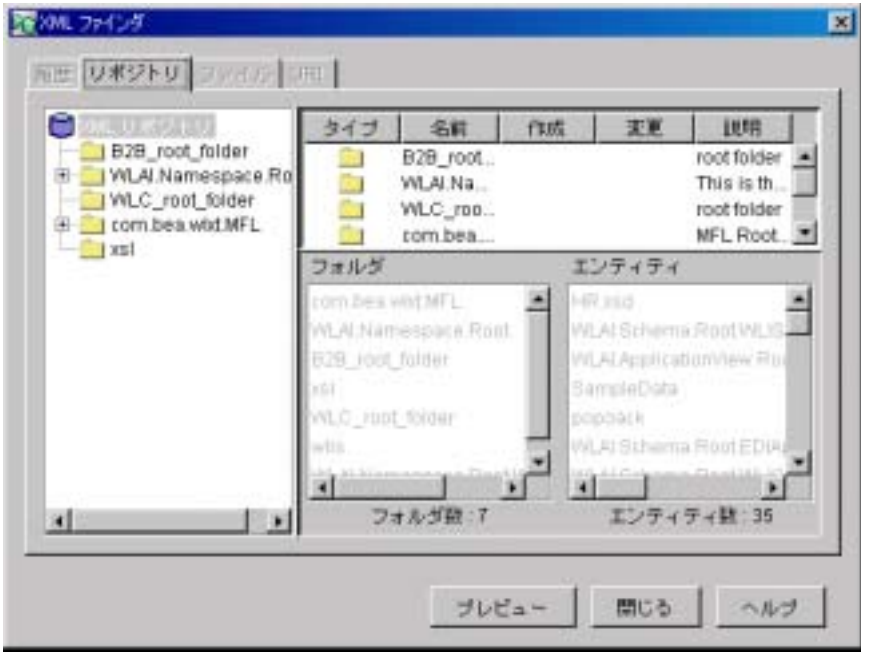
リポジトリにある XML エンティティを表示する

リポジトリにある XML エンティティを表示する手順は、以下のとおりです。

1. [ツール | XML ファインダを表示] を選択します。[XML ファインダ] ダイアログ ボックスが表示されます。[リポジトリ] タブが選択されています。

2. 左ペインで XML リポジトリを選択します。リポジトリにある XML エンティティが表示されます。

図 4-11 リポジトリ内の XML エンティティ



[リポジトリ] ウィンドウに表示されている情報については次の表で説明します。

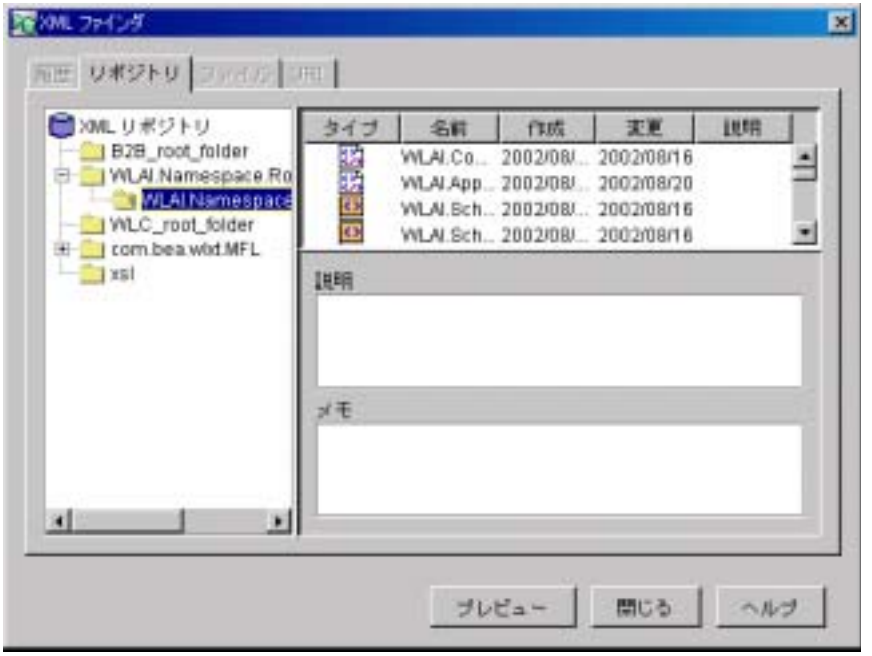
タイプ	フォルダ、ドキュメント、その他の XML エンティティなどのエンティティの型（XML エンティティの型については 4-32 ページの「XML エンティティを操作する」を参照）。
名前	フォルダまたはエンティティの名前。
作成	リポジトリ内のエンティティの最初の作成日。
変更	エンティティの最終変更日。
説明	エンティティ作成時に入力された、フォルダまたはエンティティの説明。

フォルダ	リポジトリにあるすべてのフォルダを網羅するリスト。
エンティティ	リポジトリにあるすべてのエンティティを網羅するリスト。

フォルダの内容を表示する手順は、以下のとおりです。

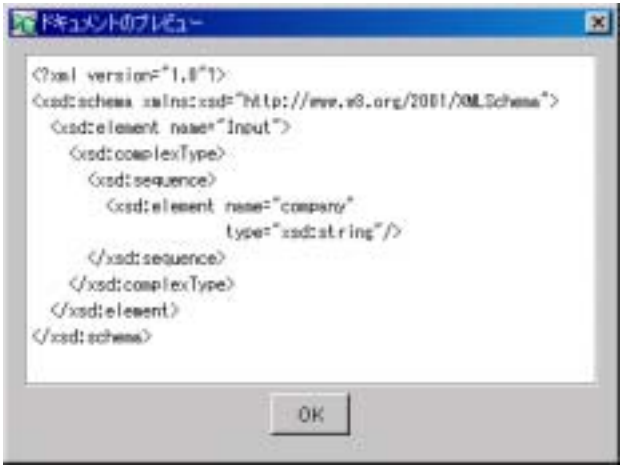
1. 左側のペインですべてのフォルダを展開し、内容を表示するフォルダを選択します。フォルダに格納されているエンティティのリストが、そのフォルダ作成時に入力された説明やメモと共に右のペインに表示されます。

図 4-12 選択したフォルダ内の XML エンティティ



2. エンティティ作成時に入力された説明やメモを表示するには、リストからエンティティを選択します。
3. （省略可能）エンティティを選択した状態で [プレビュー] をクリックし、[ドキュメントのプレビュー] ウィンドウを表示してドキュメントの内容を表示します。

図 4-13 [ドキュメントのプレビュー] ウィンドウ



4. [OK] をクリックして、[ドキュメントのプレビュー] ダイアログ ボックスを閉じます。

エンティティの詳細については、4-32 ページの「XML エンティティを操作する」 を参照してください。

左側のパネルには、フォルダやサブフォルダが階層状に配列されたリポジトリのツリー ビューが表示されます。右最上のパネルには、選択したフォルダの内容が表示されます。[説明] フィールドには、選択したフォルダの説明が表示されます。[メモ] フィールドには、選択したフォルダに関する注意が表示されます。

フォルダを操作する

1 つのフォルダに対して、追加、更新、削除など、いくつかのアクションを実行できます。フォルダに対してこれらのアクションを実行する方法について以下で説明します。

フォルダを追加する

フォルダを追加する手順は、以下のとおりです。

1. [リポジトリ] ウィンドウの左のペインで [XML リポジトリ] アイコンまたは任意のサブフォルダを右クリックし、ポップアップ メニューから [フォルダを追加] を選択して、[フォルダを追加] ダイアログ ボックスを表示します。

図 4-14 [フォルダを追加] ダイアログ ボックス

2. [名前] フィールドにフォルダ名を入力します。
3. （省略可能）[説明] および [メモ] フィールドに、フォルダの説明と注意をそれぞれ入力します。
4. [OK] をクリックします。[XML ファインダ] ダイアログ ボックスに新しいフォルダが表示されます。

フォルダ情報を更新する

フォルダを更新する手順は、以下のとおりです。

1. [リポジトリ] ウィンドウの左のペインで更新するフォルダを右クリックし、ポップアップ メニューから [フォルダ情報を更新] を選択して、[フォルダ情報を更新] ダイアログ ボックスを表示します。

図 4-15 [フォルダ情報を更新] ダイアログ ボックス

2. 必要に応じて、[名前]、[説明]、および [メモ] フィールドの内容を変更します。
3. [OK] をクリックします。

フォルダを削除する

フォルダはサブフォルダが存在する間は削除できません。

フォルダを削除する手順は、以下のとおりです。

1. [リポジトリ] ウィンドウの左のペインで削除するフォルダを右クリックし、ポップアップ メニューから [フォルダを削除] を選択します。
2. メッセージが表示されたら削除の操作を確認します。

XML エンティティを操作する

リポジトリにはさまざまなタイプの XML エンティティが保存されており、各エンティティは次の表に示すとおり、記号で表されています。

記号	XML エンティティの型
	DTD (Documnet Type Definition: 文書型定義) ファイル
	MFL (Message Format Language: メッセージ フォーマット言語) ファイル
	スキーマ（XSD）ファイル
	テキスト ファイル
	XML ドキュメント
	XSL (Extensible Stylesheet Language: 拡張スタイルシート言語) テンプレート ドキュメント

リポジトリに格納されているフォルダ内の XML エンティティに対しては、エンティティの追加、更新、移動、削除など、いくつかのアクションを実行できます。

リポジトリに XML エンティティをインポートする

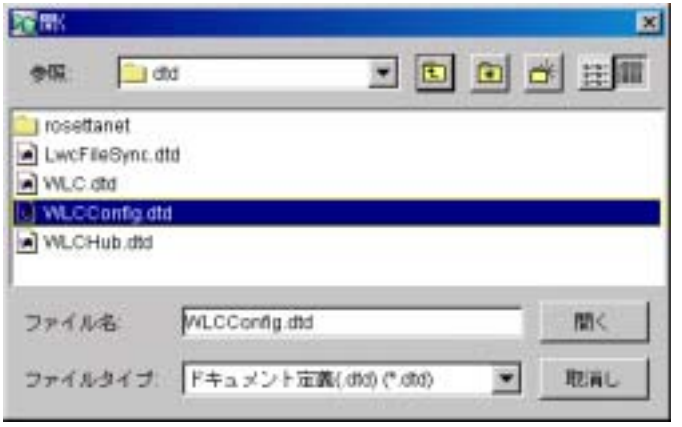
XML エンティティを追加する手順は、以下のとおりです。

1. [リポジトリ] ウィンドウの左のペインでフォルダを展開して、エンティティを追加するフォルダを右クリックし、ポップアップ メニューから [エンティティを追加] を選択して [エンティティを追加] ダイアログ ボックスを表示します。

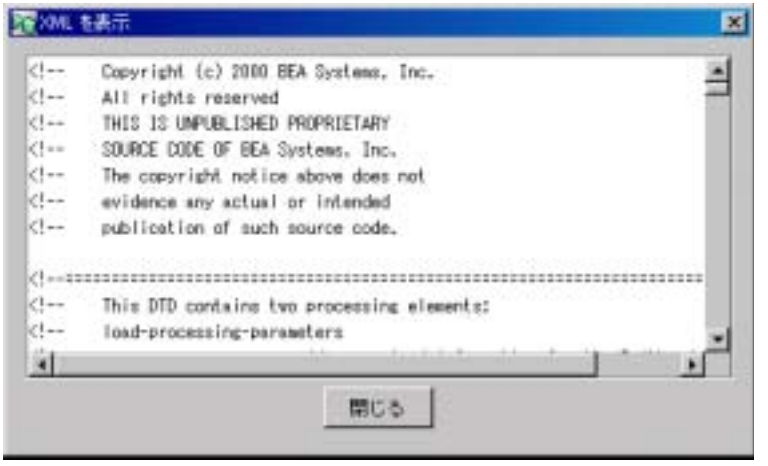
図 4-16 [エンティティを追加] ダイアログ ボックス

2. [名前] フィールドに、追加するエンティティに固有な名前を入力します。
3. [タイプ] ドロップダウン リストから追加するエンティティの型を選択します。
4. （省略可能）[説明] フィールドおよび [メモ] フィールドに、そのエンティティに関する説明とメモをそれぞれ入力します。
5. [コンテンツ URL] フィールドに追加するエンティティの URL を入力します。または、[参照] を使用して、ローカル ネットワーク ドライブまたはマッピングされたネットワーク ドライブ上のエンティティを見つけます。[開く] ダイアログ ボックスが表示されます。

図 4-17 [開く] ダイアログ ボックス



6. [参照] ドロップダウン リストから、インポートする内容を含むファイルが格納されたフォルダを選択します。
7. [ファイル名] フィールドにファイル名と拡張子を入力、あるいはファイルを選択し [開く] をクリックします。
8. [エンティティを追加] ダイアログ ボックスにそのファイルの URL が返されます。
9. （省略可能）[表示] をクリックして、追加するエンティティのコンテンツを表示します。



10. [取消し] をクリックするとウィンドウが閉じ、コンテンツは [エンティティを追加] ダイアログ ボックスに戻されます。

11. [OK] をクリックします。選択したフォルダの [XML ファインダ] ダイアログ ボックスに新しいエンティティが表示されます。

エンティティを更新する

エンティティを更新機能を使用して、定義済みのエンティティのコンテンツを変更することができます。ただし、エンティティの型の変更はできません。エンティティの型を変更するには、コンテンツ型を持つ新しいエンティティを作成する必要があります。詳細については、4-33 ページの「リポジトリに XML エンティティをインポートする」を参照してください。

エンティティを更新する手順は、以下のとおりです。

1. [リポジトリ] ウィンドウの左側のペインでフォルダを展開して、更新するエンティティが格納されているフォルダを選択します。
2. ウィンドウの右側のペインで更新するエンティティを右クリックし、ポップアップ メニューから [エンティティ定義を更新] を選択して、[エンティティ定義を更新] ダイアログ ボックスを表示します。

図 4-18 [エンティティ定義を更新] ダイアログ ボックス

3. 必要に応じて、[名前]、[説明]、および [メモ] フィールドの内容を変更します。
4. [コンテンツ URL] フィールドに、追加する新しいコンテンツのソースの URL を入力します。または、[参照] を使用して、ローカル ネットワーク ドライブまたはマッピングされたネットワーク ドライブ上のエンティティを探します。
5. (省略可能) [表示] をクリックして、そのエンティティの新しいコンテンツを表示します。
6. [OK] をクリックして [エンティティ定義を更新] ダイアログ ボックスを閉じます。これで選択したエンティティは更新されます。

エンティティを移動する

エンティティは、ソース フォルダから切り取って移動先フォルダに貼り付ける方法で、フォルダからフォルダへと移動できます。

エンティティを移動する手順は、以下のとおりです。

1. [リポジトリ] ウィンドウの左側のペインでフォルダを展開して、切り取るエンティティが格納されているフォルダを選択します。
2. ウィンドウの右のペインで切り取るエンティティを右クリックし、ポップアップ メニューから [切り取り] を選択します。
3. [リポジトリ] ウィンドウの左のペインでフォルダを展開し、そのエンティティを貼り付けるフォルダを右クリックし、ポップアップ メニューから [貼り付け] を選択します。これでそのエンティティは移動先フォルダに貼り付けられました。

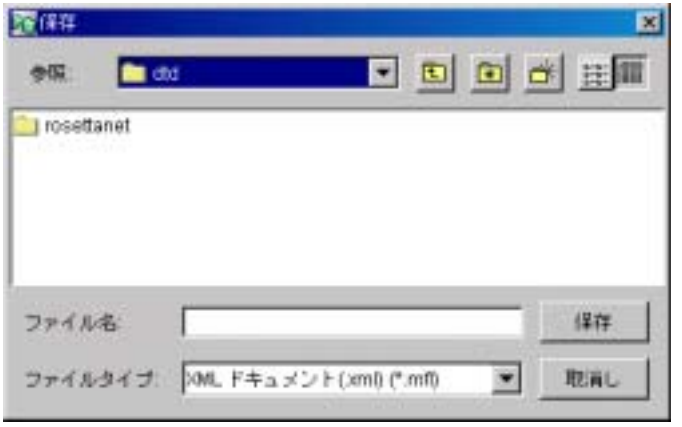
エンティティをファイル システムにエクスポートする

リポジトリのデータベース表から取得した XML エンティティは、ローカルファイル システム上のローカル ファイル システムまたはローカル マシンにマッピングされたネットワーク ドライブに保存できます。

エンティティのファイルをエクスポートする手順は、以下のとおりです。

1. [リポジトリ] ウィンドウの左側のペインでフォルダを展開して、エクスポートするエンティティが格納されているフォルダを選択します。
2. ウィンドウの右のペインでエクスポートするエンティティを右クリックし、ポップアップ メニューから [エンティティをエクスポート] を選択して、[保存] ダイアログ ボックスを表示します。

図 4-19 [保存] ダイアログ ボックス



3. [参照] ドロップダウン リストから、そのエンティティのエクスポート先のドライブとフォルダを選択します。
4. [ファイル名] フィールドでエンティティのエクスポート先ファイルの名前を指定するか、既存のファイルを選択します。名前を指定しない場合は、そのエンティティにはデフォルトの名前が割り当てられます。既存ファイルを選択する場合は、上書きを求めるメッセージが表示されます。
5. [保存] をクリックします。ファイルは、ファイルのタイプに合った拡張が付けられてディスクに保存されます。

エンティティを削除する

ワークフロー内の XSL 変換アクションによって参照されるエンティティを削除する場合は、実行時に WebLogic Integration サーバ例外が発生しないように必ずこのアクションを更新してください（このアクションの詳細については、6-103 ページの「XML ドキュメントの変換」を参照）。

エンティティを削除する手順は、以下のとおりです。

1. [リポジトリ] ウィンドウの左側のペインでフォルダを展開して、削除するエンティティが格納されているフォルダを選択します。
2. ウィンドウの右のペインで削除するエンティティを右クリックし、ポップアップ メニューから [エンティティを削除] を選択します。
3. メッセージが表示されたら削除の操作を確認します。

5　ワークフロー テンプレートの定義

この章では、ワークフロー テンプレート、テンプレートの定義、ワークフロー変数およびノードの定義と保守に関するコンセプトやタスクについて説明します。構成は以下のとおりです。

- テンプレート定義タスクの概要
- テンプレートに関する作業
- テンプレート定義に関する作業
- ノードに関する作業
- 変数に関する作業
- ノードのプロパティの定義
- 例外ハンドラに関する作業

テンプレート定義タスクの概要

詳細なワークフロー テンプレートの定義は、テンプレートの作成、テンプレート定義の作成、フローの設計、変数の定義、ノードのプロパティの指定および例外ハンドラの定義（省略可能）で構成されます。ワークフローの定義は、各レベルでリビジョンおよび改良を要する反復的なプロセスですが、新しいテンプレートを定義する場合、これらのタスクの実行は以下の手順をお勧めします。

注意： また、ワークフロー テンプレートの定義を開始する前に、ワークフロー式の言語や Studio の Expression Builder および XPath Wizard ツールについての学習も必要です。テンプレート定義のラベル作成、分岐ノードの条件の定義、イベントの定義など、この節で説明するタスクの多くではダイアログボックス フィールドに式を入力する必要があります。ワークフロー式に関する詳細については、第 8 章「ワークフロー式の使用法」を参照してください。

1. ワークフロー テンプレートの作成手順の詳細は、5-4 ページの「ワークフロー テンプレートを作成する」を参照してください。あるいは、エクスポート済みのワークフロー パッケージからテンプレートをインポートします。手順の詳細は、11-5 ページの「ワークフロー パッケージのインポート」を参照してください。
2. テンプレート内で、テンプレート定義を作成します。手順の詳細は、5-4 ページの「ワークフロー テンプレートを作成する」を参照してください。あるいは、エクスポート済みのワークフロー パッケージまたは XML ファイルからテンプレート定義をインポートします。その手順については、11-5 ページの「ワークフロー パッケージのインポート」および 11-12 ページの「XML からワークフロー テンプレート定義をインポートする」を参照してください。
3. 設計領域にシェイプとコネクタを追加して高レベルのワークフローを作成します。手順の詳細は、5-19 ページの「ノードに関する作業」を参照してください。
4. 機能を認識しやすくするため、タスク、イベントおよび開始ノード型の名前を変更します。手順の詳細は、5-23 ページの「ノードの名前を変更する」を参照してください。
5. 変数の作成を開始します。手順の詳細は、5-31 ページの「変数を作成する」を参照してください。
6. 条件を定義して分岐ノードの名前を変更します。分岐のプロパティの定義の手順については、5-54 ページの「分岐のプロパティを定義する」を参照してください。
7. トリガ型とプロパティ、変数の初期設定を定義し、必要に応じてイベント トリガ開始の場合のイベント キーおよび時限開始の場合のカレンダーを設定（省略可能）して、開始ノードのプロパティを指定します。イベント キーの詳細については、4-21 ページの「イベント キーのコンフィグレーション」を参照してください。カレンダーの詳細については、3-4 ページの「ビジネス カレンダーの管理」を参照してください。

   あるいは、エクスポート済みのイベント キーおよびカレンダーを既存のワークフロー パッケージからインポートします。詳細については、11-5 ページの「ワークフロー パッケージのインポート」を参照してください。開始のプロパティ定義の手順については、5-48 ページの「イベント トリガ型開始のプロパティを定義する」を参照してください。

8. イベント ノードのプロパティを指定し、イベントのイベント キーのコンフィグレーション（省略可能）を行います。詳細については、4-21 ページの「イベント キーのコンフィグレーション」を参照してください。

   あるいは、エクスポート済みのイベント キーを既存のワークフロー パッケージからインポートします。詳細については、11-5 ページの「ワークフロー パッケージのインポート」を参照してください。イベント ノード プロパティ定義の手順については、5-51 ページの「イベント プロパティを定義する」を参照してください。

9. 手動割り当てタスクに対するタスク ノード プロパティを指定します。タスク プロパティの説明および手順については、5-55 ページの「タスクのプロパティを定義する」を参照してください。

10. （省略可能）タスク ノードおよび他のノードにアクションを追加します。アクションの追加と定義の手順については、第 6 章「アクションの定義」を参照してください。

11. （省略可能）テンプレート定義の例外ハンドラを定義し、それらのハンドラを呼び出すアクションを追加します。例外ハンドラについては、第 9 章「ワークフロー例外の処理」で説明します。

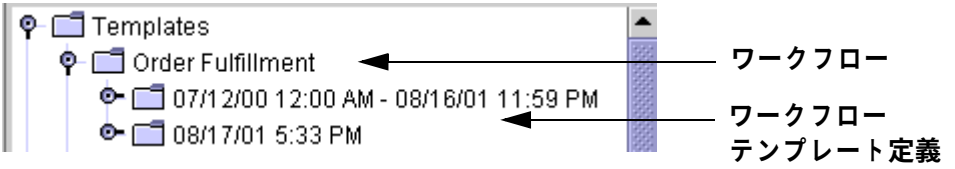
12. テンプレート定義を保存します。手順の詳細は、5-12 ページの「テンプレート定義を保存し終了する」を参照してください。

13. ワークフローを実行する準備ができたら、5-13 ページの「テンプレート定義を更新、ラベリングおよびアクティブ化する」で説明するようにテンプレート定義をアクティブにし、それを保存します。

テンプレートに関する作業

ワークフロー テンプレートとは、つまり WebLogic Integration ワークフロー テンプレート定義用のフォルダまたはコンテナです。各ワークフロー テンプレートには、1 つまたは複数のワークフロー テンプレート定義を保持できます。ワークフロー テンプレート定義は、5-7 ページの「テンプレート定義に関する作業」で説明するように、フォルダ ツリーで Effective (有効) および Expiry (終了) 日時によって識別されます。

図 5-1 ワークフロー テンプレートとワークフロー テンプレート定義



テンプレートは、オーガニゼーションに対して 1 対複数の関係を持ちます。つまり、テンプレートはそのシステムにおいて固有のものとなりますが、複数のオーガニゼーションに対して定義できます。テンプレートは、定義されている各オーガニゼーションのフォルダ ツリーに、そのすべてのテンプレート定義と共に表示されます。1 つのオーガニゼーションのフォルダ ツリーでテンプレート定義に加えられた変更（削除も含む）は、そのテンプレートが関連付けられている他のすべてのオーガニゼーションのフォルダ ツリーに自動的に表示されます。

Import/Export 機能を用いてテンプレートおよびテンプレート定義をインポートする場合、テンプレートは上書きされますが、テンプレート定義は上書きされません。既存のものと同じ日付のテンプレート定義をインポートする場合、既存のテンプレート定義は上書きされず、別の定義が作成されます（テンプレートのインポートの詳細については、11-5 ページの「ワークフロー パッケージのインポート」 を参照）。

ワークフロー テンプレートを作成する

注意： テンプレートを作成するためにはテンプレートの作成パーミッションが必要です。パーミッション レベルの詳細については、3-26 ページの「ユーザおよびロールへのパーミッションの割り当て」 を参照してください。

ワークフロー テンプレートを作成する手順は、以下のとおりです。

1. オーガニゼーションがアクティブな状態で、Studio のメイン ウィンドウの [テンプレート] フォルダを右クリックします。
2. ポップアップ メニューから [テンプレートを作成] を選択します。[テンプレートのプロパティ] ダイアログ ボックスが表示されます。

図 5-2 [テンプレートのプロパティ] ダイアログ ボックス

3. [名前] フィールドで、ワークフロー テンプレートに対する有意な固有名を入力します。このワークフロー テンプレート内で定義されたすべてのワークフロー テンプレート定義は、実行時にワークフロー インスタンスになると識別にこの名前を用います。
4. ダイアログ ボックスの [オーガニゼーション] セクションで、このワークフロー テンプレートを割り当てたい単一あるいは複数のオーガニゼーションを選択します。ワークフロー テンプレートをすべてのオーガニゼーションに対して有効にするには [すべてのオーガニゼーション] を、オーガニゼーションをクリアしてリセットするには [オーガニゼーションをクリア] をクリックします。

   注意 : 1 つのテンプレートを複数のオーガニゼーションに関連付ける場合、そのテンプレートに加えたすべての変更は、他のオーガニゼーションがそのテンプレートを表示すると自動的に反映されます。
5. [OK] をクリックします。フォルダ ツリーの [テンプレート] フォルダの下に新しいワークフロー テンプレートが表示されます。

テンプレート プロパティを更新する

[テンプレートのプロパティ] ダイアログ ボックスで、作成後またはインポート後にテンプレートを追加されたオーガニゼーションに割り当てることができます（手順については、11-5 ページの「ワークフロー パッケージのインポート」 を参照）。

テンプレート プロパティを更新する手順は、以下のとおりです。

1. フォルダ ツリーの上の [オーガニゼーション] フィールドで、更新するテンプレートが定義されているオーガニゼーションを選択します。
2. フォルダ ツリーで [テンプレート] フォルダを展開し、テンプレートを右クリックして、ポップアップ メニューから [プロパティ] を選択し、[テンプレートのプロパティ] ダイアログ ボックスを開きます。
3. このテンプレートが割り当てられているオーガニゼーションに必要な変更を加えます。
4. [OK] をクリックして変更を保存し、ダイアログ ボックスを閉じます。

テンプレートを削除する

ワークフロー テンプレートを削除すると以下のことが起こります。

- テンプレートは割り当てられているすべてのオーガニゼーションから削除されます。
- テンプレートに含まれるすべてのテンプレート定義は削除されます。
- テンプレートの定義のすべてのインスタンス（現在のステータスにかかわらずすべてのタスクを含む）は削除されます。
- テンプレートの定義に関連するすべての履歴が削除され、統計あるいは作業ロード レポートでの使用が不可能となります。

注意 : テンプレートを削除するためにはテンプレートの削除パーミッションが必要です。パーミッション レベルの詳細については、3-26 ページの「ユーザおよびロールへのパーミッションの割り当て」 を参照してください。

ワークフロー テンプレートを削除する手順は、以下のとおりです。

1. テンプレートがオープン テンプレート定義を含む場合、描画ウィンドウの右上隅にある [X] をクリックするか、フォルダ ツリーのテンプレート定義を右クリックしてポップアップ メニューから [閉じる] を選択してこれらを閉じます。
2. フォルダ ツリーのワークフロー テンプレート名を右クリックします。
3. ポップアップ メニューから [削除] を選択します。
4. 「テンプレートの削除」警告メッセージが表示された場合は [はい] をクリックして、ワークフロー テンプレートとそのすべてのワークフロー テンプレート定義を削除するか、[いいえ] をクリックして削除をキャンセルします。

   テンプレートのインスタンスが存在する場合、インスタンスも削除するよう要求されます。インスタンスを削除してよい場合は [はい] をクリックし、削除をキャンセルする場合は [いいえ] をクリックします。

テンプレート定義に関する作業

フォルダ ツリーのワークフロー テンプレート定義名は、その有効日時および終了日時で構成されます。有効日時および終了日時は、ワークフロー テンプレート定義が*インスタンス*あるいは実行に有効となる時間の範囲を表します。同じ有効日時および終了日時を持つ複数のテンプレート定義が存在する場合もあります。

インスタンスに有効なテンプレート定義を作成するためには、まずそれを*アクティブにする*必要があります。設計時に、テンプレート定義を、いくつでもアクティブにできますが、唯一、同じ*有効*（開始）日時を持つテンプレート定義はアクティブにできないという制限があります。

有効および終了日時の機能の利点は、1 年を通じて異なるテンプレート定義を手動で非アクティブあるいはアクティブにすることなく、ある期間に対して正確に有効なワークフロー テンプレート定義を作成できるという点です。たとえば、特定のワークフロー テンプレート定義を 1 月から 3 月まで実行し、異なるタスクや変数を持つ次のワークフロー テンプレート定義を 4 月から 6 月まで実行しなければならず、さらに次の定義を 7 月から 9 月まで実行しなければならない周期的ビジネスの場合などです。1 つのテンプレート定義を手動で非アクティブに

して、別のテンプレート定義を四半期ごとにアクティブにする代わりに、異なる有効日時および終了日時を指定し、それらすべてをアクティブとしてマークすると、サーバがワークフローのインスタンス化の際に正しいバージョンを自動的に選択します。

一方、複数のアクティブなテンプレート定義を持つことは可能ですが、実際に実行時にインスタンス化されるのは 1 つのテンプレート定義のみです。つまり、有効および終了日時は、*ローリング*方式で設計するよう注意し、日付のオーバーラップを回避する必要があります。日付がオーバーラップするアクティブなテンプレート定義がある場合（たとえば 1 月から 3 月までの定義と 2 月から 4 月までの定義が混在する場合）、プロセス エンジンはデータベースから検索した最初の定義を取り上げます。これは通常、設計時に先に作成された定義です。

注意： 異なるビジネス条件に従って異なるフローを指定する場合、同じテンプレート定義で複数の開始ノードを使用します。詳細については、5-34 ページの「開始のプロパティを定義する」を参照してください。

ワークフロー テンプレート定義を作成する

テンプレート定義を作成する場合、その有効および終了の日付を指定します。

監査エントリを作成アクションを用いて、クライアント アクセスおよび実行時間など、実行時のワークフロー情報のロギングを有効にし、ワークフロー全体を通じて各ポイントにおけるカスタム エントリを可能にする監査を有効にすることもできます（詳細については、6-45 ページの「監査エントリを作成する」を参照）。監査情報は、XML メッセージとしてデフォルトの JMS 監査トピック com.bea.wlpi.AuditTopic にポスティングされ、サーバ上のアクティブな WebLogic Integration ドメインの `logs` ディレクトリにあるテキスト ファイル `myserver.log` に書き込まれます。

注意： テンプレート定義を作成するためには、テンプレートを作成パーミッションが必要です。パーミッション レベルの詳細については、3-26 ページの「ユーザおよびロールへのパーミッションの割り当て」を参照してください。

ワークフロー テンプレート定義を作成する手順は、以下のとおりです。

1. フォルダ ツリーの上の [オーガニゼーション] フィールドで、定義の追加を行うテンプレートが定義されているオーガニゼーションを選択します。

2. フォルダ ツリーで [テンプレート] フォルダを展開し、テンプレートを右クリックし、ポップアップ メニューから [テンプレート定義を作成] を選択して [テンプレート定義] ダイアログ ボックスを表示します。ダイアログ ボックスにはテンプレート定義が属するテンプレートの名前が表示されます。

図 5-3 [テンプレート定義] ダイアログ ボックス

3. [一般] タブで以下を指定します。

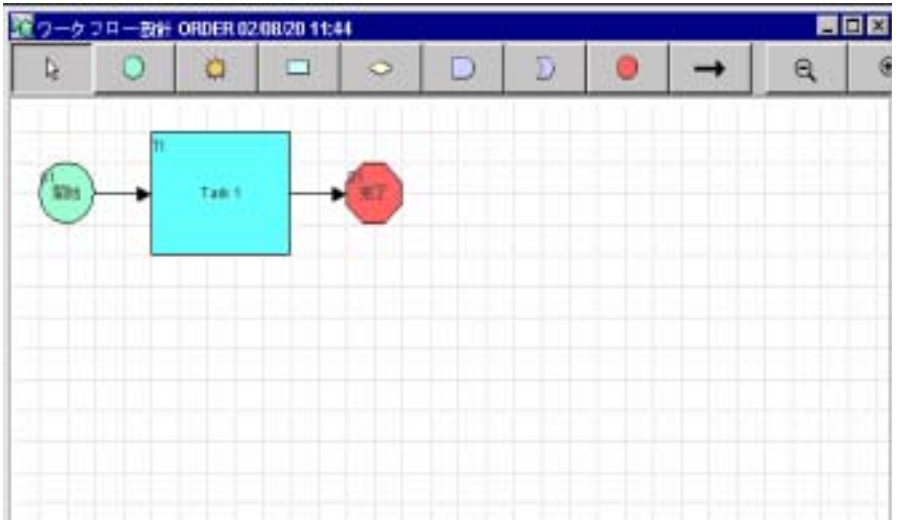
   - 開始ー このワークフロー テンプレート定義が有効になる正確な日付を選択します。時間はシステムにより自動的に供給され、テンプレート定義が最初に作成された時間に対応します。
   - 終了ー（省略可能）このチェック ボックスを選択して、ワークフロー テンプレート定義の終了日を設定します。時間はシステムにより自動的に供給され、必ず終了日の午後 11 時 59 分となります。

   注意 : [終了] オプションが選択されていない場合、ワークフロー テンプレート定義は常に有効となります。

4. (省略可能) [監査を有効化] を選択します。これにより、実行時のワークフロー情報をロギングできます。
5. (省略可能) [メモ] フィールドにテンプレート定義を説明する全般的なコメントを入力します。
6. [OK] をクリックしてワークフロー設計領域を表示します。

ワークフロー設計領域ではデフォルトのワークフロー テンプレート定義がツールバーと共に表示されます。このツールバーにはワークフロー テンプレートの定義に用いられる描画シェイプが含まれます。デフォルトのワークフロー テンプレート定義には、開始、タスク、完了の 3 つのシェイプが含まれます。

図 5-4 ワークフロー設計領域



5-19 ページの「ノードに関する作業」で説明するようにノードやコネクタを追加し、5-28 ページの「変数に関する作業」で説明する変数を追加してテンプレート定義を行うことができます。

既存のテンプレート定義を開く

テンプレート定義の読み取り専用プロパティは、フォルダ ツリーでいずれかのフォルダを右クリックしてポップアップ メニューから [プロパティ] を選択して見ることができますが、テンプレート定義は開かれていない限り修正を加えることはできません。テンプレート定義は開くとロックされるので、他のユーザは読み取り専用モードでしか開くことができません。

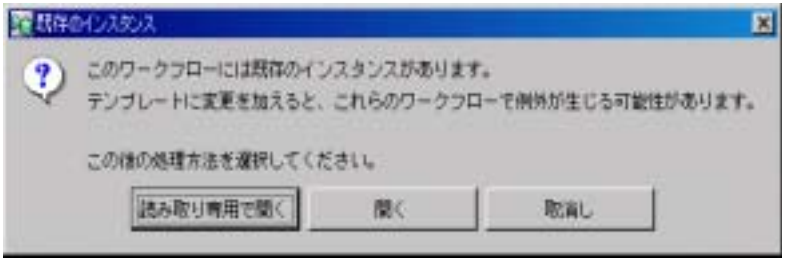
注意 : テンプレート定義を開くためにはテンプレートを作成パーミッションが必要です。パーミッション レベルの詳細については、3-26 ページの「ユーザおよびロールへのパーミッションの割り当て」を参照してください。

既存のワークフロー テンプレート定義を開く手順は、以下のとおりです。

1. フォルダ ツリーの上の [オーガニゼーション] フィールドで、開きたいテンプレートが定義されているオーガニゼーションを選択します。
2. フォルダ ツリーで [テンプレート] フォルダを展開して、目的のテンプレート定義を含むテンプレート フォルダを展開し、さらにテンプレート定義を右クリックしてポップアップ メニューから [開く] を選択します。

   選択したワークフロー テンプレート定義のインスタンスが既に存在する場合、つまり選択したワークフロー テンプレート定義が開始済みでテンプレート定義のインスタンスがサーバに存在する場合は、次の図に示すように警告メッセージが表示されます。

図 5-5 [既存のインスタンス] ダイアログ ボックス



3. 以下のオプションのうち 1 つを選択する。
   - 読み取り専用で開く ― 選択したワークフロー テンプレート定義を閲覧目的のみで開きます。ワークフローへの変更にかかわるすべてのインタ

フェース オプションはアクセス不可能となっており、ワークフローに修正を加えることはできません。

- 開く－選択したワークフロー テンプレート定義を閲覧と更新の目的で開きます。
- 取消し－このダイアログ ボックスを閉じ、ワークフロー テンプレート定義を開きません。

注意： 実行中のインスタンスを持つテンプレート定義の修正は行わないでください。このような定義を修正すると、これらのインスタンスに予測不可能な例外が発生する場合があります。そのようなテンプレート定義に変更を加える場合は以下を行ってください。

- プロダクション環境で、まず実行中のすべてのインスタンスを終了させ、別の有効および終了日を持つ新しいテンプレート定義を作成し、アクティブにします。既存のワークフロー情報を新しいテンプレート定義にコピーする場合は、5-15 ページの「ワークフロー テンプレート定義をコピーする」の手順に従ってください。
- 開発環境で、変更を加える前にテンプレート定義のすべてのインスタンスを削除します。その手順については、10-11 ページの「ワークフロー インスタンスを削除する」を参照してください。

テンプレート定義を保存し終了する

Studio のフォルダ ツリーでは、保存が必要な各ワークフロー テンプレート定義の左側にアスタリスク（*）が表示されます。

テンプレート定義を保存するには以下のいずれかを行います。

- フォルダ ツリーでテンプレート定義を右クリックし、ポップアップ メニューから [保存] を選択します。
- ワークフロー設計エリアの任意の場所で右クリックし、ポップアップ メニューから [保存] を選択します。

テンプレート定義を閉じると、ロックが解除され別のユーザによる使用が可能になります。

ワークフロー テンプレート定義を終了するには以下のいずれかを行います。

- フォルダ ツリーでテンプレート定義を右クリックし、ポップアップ メニューから [閉じる] を選択します。
- テンプレート定義のワークフロー設計ウィンドウの右上隅の [X] をクリックします。

テンプレート定義を更新、ラベリングおよびアクティブ化する

テンプレート定義を作成し、開いたら（5-8 ページの「ワークフロー テンプレート定義を作成する」で説明）、作成時に指定したプロパティを更新し、ラベルを追加してアクティブ化できます。

ここで作成するワークフロー ラベルは、実行時にタスク リストのワークフロー ラベル カラムに表示され Worklist ユーザに示されます。また、実行時に Studio の [ワークフロー インスタンス] ダイアログ ボックスの [ワークフロー ラベル] フィールドにも表示されます（詳細については、10-5 ページの「ワークフロー インスタンスの状態を表示する」を参照）。これは、ワークフローのインスタンスを識別するために用いられ、日時、請求書番号、顧客名あるいは他と区別するための関連情報が含まれます。ラベルはワークフロー式言語で公式化され、定数、変数、演算子およびその他の式コンポーネントを含むことが可能です。ワークフロー式機能および構文に関する詳細については、第 8 章「ワークフロー式の使用法」を参照してください。

新しいテンプレート定義を作成するとデフォルトで非アクティブな状態になります。現在開発中のテンプレート定義は、不用意に開かれることを避けるため通常は非アクティブとなっています。ただし、ワークフローがインスタンス化される前、あるいは実行環境に置かれる前に、いずれか 1 つのテンプレート定義をアクティブにする必要があります。

ワークフロー テンプレート定義のプロパティを更新する手順は、次のとおりです。

1. ワークフロー テンプレート定義が開いた状態で設計ウィンドウの任意の場所を右クリックするか、フォルダ ツリーのテンプレート定義を右クリックして、ポップアップ メニューから [プロパティ] を選択し、[テンプレート定義] ダイアログ ボックスを開きます。

図 5-6 テンプレート定義のプロパティの更新

2. [テンプレート定義] ダイアログ ボックスの [一般] タブの [ワークフロー ラベル] フィールドに、実行時にラベルを生成するために評価される式を入力します。ワークフロー式の構築に関する詳細については、第 8 章「ワークフロー式の使用法」を参照してください。
3. テンプレート定義をアクティブにするには [アクティブ] チェック ボックスにチェックします。
4. （省略可能）[開始] および [終了] を更新し、監査を有効または無効にします。
5. （省略可能）[テンプレート定義] ダイアログ ボックスの [例外ハンドラ] タブで、例外ハンドラを追加、更新または削除します。例外のハンドリングに関する詳細については、、第 8 章「ワークフロー式の使用法」を参照してください。
6. [OK] をクリックして変更を保存し、ダイアログ ボックスを閉じます。

ワークフロー テンプレート定義をコピーする

Studio では、同一テンプレート内の既存のワークフロー テンプレート定義をコピーできます。コピーを行うと、割り当てられたすべてのワークフロー テンプレート定義プロパティもコピーされます。これは、同様のプロパティを持つ複数のワークフロー テンプレート定義を作成する必要がある場合に時間を節約できます。新しいワークフロー テンプレート定義のプロパティは必要に応じて修正できます。

注意： コピーされた（新しい）ワークフロー テンプレート定義はアクティブにマークされていません。これをインスタンス化のために有効にしたい場合は、まずアクティブにする必要があります。詳細については、5-13 ページの「テンプレート定義を更新、ラベリングおよびアクティブ化する」を参照してください。

既存のワークフロー テンプレート定義をコピーする手順は、以下のとおりです。

1. コピーするワークフロー テンプレート定義を右クリックし、ポップアップ メニューから [コピー] を選択します。選択したワークフロー テンプレート定義のコピーは、フォルダ ツリーの最後のワークフロー テンプレート定義として直ちに貼り付けられます。このコピーはオリジナルのワークフロー テンプレート定義と同じ有効および終了日を持ち、設計領域に展開されます。
2. テンプレート定義の名前を変更し、プロパティを変更する場合は、5-13 ページの「テンプレート定義を更新、ラベリングおよびアクティブ化する」の手順に従います。

テンプレート定義を印刷する

Studio から、ワークフロー テンプレート定義ダイアグラムを印刷できます。印刷機構を呼び出す方法は 2 通りあります。

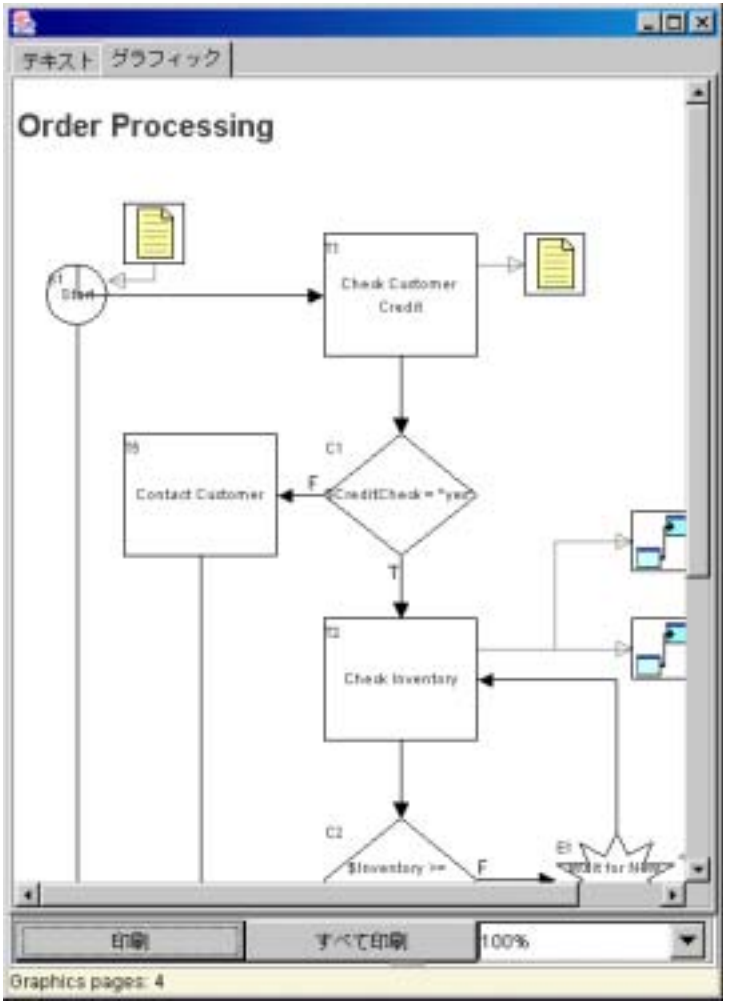
- Studio のフォルダ ツリーでワークフロー テンプレート定義を右クリックし、表示されるポップアップ メニューから [印刷] を選択します。
- 描画領域のワークフロー テンプレート定義ダイアグラムのどこかを右クリックし、ポップアップ メニューから [印刷] を選択します。

図 5-7 ワークフロー テキストの印刷

[テキスト] タブには、タスク、イベント、分岐および完了ノード内のアクション（アクション内のサブアクションを含む）の詳細に加え、アクションおよびノード メモなど、各ワークフロー ノードに関連した情報が含まれます。

[グラフィック] タブにはワークフロー テンプレート定義のダイアグラムが含まれます。

図 5-8 ワークフロー グラフィックの印刷



[印刷] をクリックして選択したタブの情報を印刷するか、[すべて印刷] をクリックして [テキスト] タブに含まれる情報と [グラフィック] タブに含まれるダイアグラムの両方を印刷します。

[印刷] ダイアログ ボックスで、ワークフローを印刷するための適切な設定を行います。

テンプレート定義を削除する

ワークフロー テンプレート定義を削除すると以下のことが起こります。

- テンプレート定義は、それが含まれるテンプレートを割り当てたすべてのオーガニゼーションから削除されます。
- そのワークフロー テンプレート定義のすべてのインスタンス（現在のステータスにかかわらずすべてのタスクを含む）が削除されます。
- そのワークフロー テンプレート定義に関連するすべての履歴が削除され、統計レポートでの使用が不可能となります。
- 削除されたワークフロー テンプレート定義を具体的に参照する作業ロード レポートは、代わりに残ったすべてのワークフロー テンプレート定義を参照するよう変更されます。

注意 : テンプレート定義を削除するためには、テンプレートの削除パーミッションが必要です。パーミッション レベルの詳細については、3-26 ページの「ユーザおよびロールへのパーミッションの割り当て」を参照してください。

ワークフロー テンプレート定義を削除する手順は、以下のとおりです。

1. テンプレート定義が開かれている場合、描画ウィンドウの右上隅にある [X] をクリックするか、フォルダ ツリーのテンプレート定義を右クリックしてポップアップ メニューから [閉じる] を選択してこれらを閉じます。
2. フォルダ ツリーのワークフロー テンプレート定義を右クリックします。
3. ポップアップ メニューから [削除] を選択します。
4. 「ワークフローを削除」警告メッセージが表示されたら、[はい] をクリックしてワークフロー テンプレート定義を削除するか、[いいえ] をクリックして削除を取り消します。

   テンプレートの定義にインスタンスがあれば、そのインスタンスも削除するように要求されます。インスタンスを削除してよい場合は [はい] をクリックし、削除をキャンセルする場合は [いいえ] をクリックします。

ノードに関する作業

最初にワークフロー テンプレート定義を作成した後、デフォルトでは、設計領域に開始、タスクおよび完了の 3 つのシェイプ（ノード）が存在します。開始ノードはデフォルトでは手動による開始に設定されており、タスク ノードはワークフローを開始する Worklist ユーザにタスクを割り当てます。これらは、実行可能なワークフローを作成するために必要な最小限のプロパティです。（開始ノード プロパティの編集の詳細については 5-48 ページの「イベント トリガ型開始のプロパティを定義する」を、タスク ノード プロパティの編集の詳細については、5-55 ページの「タスクのプロパティを定義する」を参照）。

次の表に、ワークフロー シェイプ、そのノード名および目的を示します。

表 5-1 ワークフローシェイプおよびコネクタ

記号	ノード タイプ	目的
	開始	ワークフローの開始を示し、異なる方法（手動、特定の時間、イベント、あるいは別のワークフローにより）でトリガすることが可能。開始ノードに関する詳細については、5-34 ページの「開始のプロパティを定義する」を参照。
	イベント	外部アプリケーションや別のワークフロー、あるいはプラグイン定義イベントからの内部 JMS キューで受け取った XML メッセージによってトリガされるイベントを表す。イベント ノードに関する詳細については、5-51 ページの「イベント プロパティを定義する」を参照。
	タスク	さまざまなアクションを定義するノードを表す。ユーザ割り当てタスクも定義する。タスク ノードに関する詳細については、5-55 ページの「タスクのプロパティを定義する」を参照。
	分岐	True または False を評価するワークフローにおける条件を表す。True か False かにより、異なるワークフロー パスに分岐する。詳細については、5-54 ページの「分岐のプロパティを定義する」を参照。

表 5-1 ワークフローシェイプおよびコネクタ

記号	ノード タイプ	目的
	And 結合	2 つの別々のパスを AND ゲートで結合する。どちらのパスもフローを処理する前に実行を終了している必要がある。また、ワークフローに追加した後で AND を OR 結合に変更することも可能。詳細については、5-60 ページの「結合のプロパティを定義する」を参照。
	Or 結合	2 つの別々のパスを OR ゲートで結合する。どちらかのパスがフローを処理する前に実行を終了している必要がある。コントロールが単一のパスから結合に続いて起こるノードに渡されると、付随するすべての未実行タスクは実行されない。また、ワークフローに追加した後で OR を AND 結合に変更することも可能。詳細については、5-60 ページの「結合のプロパティを定義する」を参照。
	完了	ワークフローの終了を示す。詳細については、5-62 ページの「完了のプロパティを定義する」を参照。
	コネクタ	ワークフロー ノードの接続に用いられる。矢印は、フロー内で次に実行されるノードを示す。

ノードを追加、配置および接続する

設計領域のシェイプを操作するために、以下を行うことができます。

- ツールバー内のシェイプをクリックして、カーソルを設計領域に置き、再度クリックしてシェイプを設計領域にドロップして設計領域にシェイプを配置できます。設計領域にシェイプを配置すると、フォルダ ツリー内にノードも作成されます。
- マウスでシェイプをクリック アンド ドラッグし、設計領域に移動できます。
- ツールバーの [コネクタを描画] ボタンをクリックし、ソース ノードをクリックしてターゲット ノードにドラッグし、マウス ボタンを離してシェイプを接続できます。分岐シェイプからコネクタを作成すると、[コネクタを作成] ダイアログ ボックスが表示され、コネクタが [True] か [False] かを指定するよう要求されます。

注意： ツールバーに関する詳細については、2-12 ページの「ツールバーの使い方」を参照してください。

設計領域にノードを追加すると、ノードは直ちにフォルダ ツリーにも表示されます。ノードにはデフォルト名が付けられます。この名前は設計領域に配置した順番を示す番号を含み、タスク ノードの場合は T1、T2、T3 のようになります。ノードの名前を変更し、プロパティを編集する場合は、5-23 ページの「ノード プロパティに関する作業を行う」を参照してください。

ノードまたはコネクタを削除する

ノードを削除する手順は、以下のとおりです。

1. 以下のいずれか 1 つを実行します。
   - 設計領域で削除するノードを右クリックし、ポップアップ メニューから [削除] を選択します。
   - フォルダ ツリーで削除するノードを含むフォルダを展開し、ノードを右クリックして、ポップアップ メニューから [削除] を選択します。
2. 警告メッセージが表示されたら、削除する場合は [OK] を、この操作を取り消す場合は [取消し] をクリックします。

コネクタを削除する手順は、以下のとおりです。

1. コネクタを右クリックし、ポップアップ メニューから [削除] を選択します。
2. メッセージが表示されたら削除を確認します。そのノードとの間のすべてのコネクタが削除されます。

ワークフロー設計のガイドラインとヒント

シェイプを追加、接続、配置する際、以下の設計ガイドラインを覚えておくと便利です。

- ワークフローには少なくとも 1 つの開始ノードを含む必要があり、また同一のテンプレート定義に複数の開始ノードを含んで異なるパスを開始することもできます。

- ワークフローに完了ノードが含まれない場合、ワークフローは終了しません。
- フロー内の異なるパスを終了させるために複数の完了ノードの追加ができますが、実行時に最初の完了ノードに達すると、他のパスが実行を終了しているかどうかにかかわらずワークフローは終了します。
- アクションによって実行される各主要アクティビティに対し 1 つのタスクノードを作成し、ロジックをできる限り図式的に見やすく保つと効果的です。アクションとタスクの関係に関する詳細については、第 6 章「アクションの定義」を参照してください。
- フローを複数のパスに分割する場合は、単一ノードを複数ノードに接続します。
- 複数パスを単一フローに戻す場合、複数ノードを単一の結合ノードに接続します。AND 結合を用いて、パスを結合する前にすべてのパスを実行するようにします。OR 結合を用いて、パスを結合する前にいずれか 1 つのパスを実行するようにします。
- BPM は、並列する複数パスは処理しません。ノードは、作成された順序で処理されます。
- ワークフローにループを作成する場合、以下のガイドラインに従います。
  - ループを作成する場合、ノードをフローの前のノードに逆方向に接続し、分岐ノードをループに含めてカウンタの現在の値を調べます。カウンタまたは分岐を実装しない限り無限のループになります。
  - 同一ノードにループを作成する場合 [コネクタを描画] ボタンを使うことはできません。ノードの [プロパティ] ダイアログ ボックスの [次] タブを用いて、サクセサ ノードを現在のノードとして指定します。詳細については、5-24 ページの「サクセサ ノードを指定または更新する」を参照してください。この場合のカウンタの値を評価する場合、ノードに埋め込まれた条件を評価アクションを用いる必要があります（このアクションの詳細については、6-44 ページの「条件付きシーケンスを埋め込む」を参照）。
- 非常に複雑なワークフローの場合、フローをお互いに呼び出すことのできるいくつかのワークフローに分割することも可能です。サブフローに関する詳細については、6-39 ページの「サブワークフローを呼び出す」を参照してください。XML メッセージを用いたワークフロー間通信に関する詳細については、5-39 ページの「イベントおよびイベントトリガ型開始のプロパティを

定義する」および 6-88 ページの「JMS トピックまたはキューへの XML メッセージのポスト」を参照してください。

設計ガイドラインの追加情報については、『*BPM ワークフローの設計ベスト プラクティス ガイド*』を参照してください。

ノード プロパティに関する作業を行う

ワークフロー設計領域にノード型を配置し、そのプロパティを指定することができます。たとえば、最初のタスクとしてノードの名前を変更し、ワークフローにおけるノードの機能を認識しやすくできます。以下の節では、すべてのタイプのノードに共通するプロパティについての説明、および設定手順について説明します。ノードの各タイプ固有のプロパティについては各ノード タイプについて説明する節を参照してください。

ノードのプロパティにアクセスするには以下のいずれかを行います。

- 設計領域でノードをダブルクリックします。
- 設計領域でノードを右クリックし、ポップアップ メニューから [プロパティ] を選択します。
- フォルダ ツリーでノード タイプのフォルダを展開し、目的のノードを右クリックして、ポップアップ メニューから [プロパティ] を選択します。

ノードの名前を変更する

結合および完了を除くすべてのノード タイプに有意な名前を付けることができます。また、分岐に対しては識別のための条件式を入力する必要があります。詳細については、5-54 ページの「分岐のプロパティを定義する」を参照してください。

ノードの名前を変更する手順は、以下のとおりです。

1. イベント、開始またはタスク ノードに対する [プロパティ] ダイアログ ボックスを開きます。
2. [名前] フィールドに `Check Inventory` など、実行するアクションを表す有意な名前を入力します。
3. [OK] をクリックして変更を保存します。

サクセサ ノードを指定または更新する

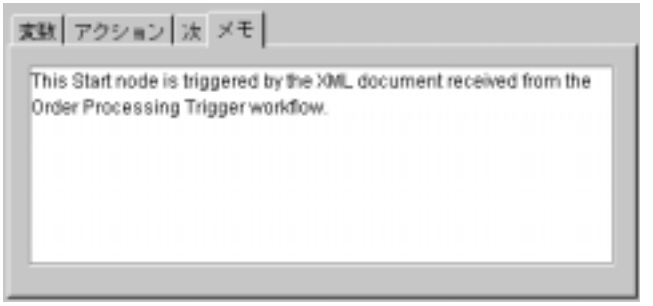
すべてのノードの [プロパティ] ダイアログ ボックスには、現在のワークフローの全ノードを一覧表示する [次] タブがあります。ワークフローにおける次のノードを、ノード名の横にあるチェックによって指定する。このタブを使って、現在のノードに後続ノード（複数可）を指定する、または設計領域で定義された後続ノードを変更できる。タブ内のチェック ボックスにチェックするかチェックを外すと、設計領域内に表示されたノード間をつなぐ線が自動的に引き直される。

図 5-9 ノードの [プロパティ] ダイアログ ボックスの [次] タブ

ノードへの注意を追加する

すべてのノードの [プロパティ] ダイアログ ボックスには、ノードやノードに含まれるアクションに関するコメントを入力できる [メモ] テキスト ボックスがあります。これは、同じワークフローにアクセスする他のユーザが、そのワークフローのロジックや設計を理解する必要のある場合に役立ちます。

図 5-10　ノードの [プロパティ] ダイアログ ボックスの [メモ] タブ



さらに、分岐ノードおよびタスク ノードには、アクションに対して定義されたメモを見ることのできる [アクション メモ] タブがあります（6-22 ページの「アクションへコメントを追加する」を参照）。[タスクのプロパティ] または [分岐のプロパティ] ダイアログ ボックス左のペインでアクションを選択すると、そのアクション定義に対するメモが表示されます。

ワークフロー アクションを追加、更新、並べ替えおよび削除する

結合（AND および OR）を除くすべてのノードでは、そのプロパティ ダイアログ ボックスでアクションの追加、更新、並べ替えそして削除が可能です。

図 5-11　ノードの [プロパティ] ダイアログ ボックスの [アクション] タブ

アクションは、ノードがアクティブになった場合に実行する操作を定義するものです。ノードのプロパティ ダイアログ ボックスの [アクション] タブで指定したアクションは、ワークフローが次のノード（およびそれに含まれるアクション）に進む前に実行されます。

タスク ノードではアクションの追加が必須ですが、その他のノードではサクセサ タスク ノードの追加によって同一ロジックが実装される場合が多いため、アクションの追加は任意であり、推奨されません。アクションの追加、更新、削除、並べ替えそして定義についての詳細については、6-18 ページの「アクションの操作」で説明します。

ノードをコピーする

ワークフロー テンプレート定義内のノードを表すシェイプをコピーして、現在のワークフロー テンプレート定義、または別の開いているワークフロー テンプレート定義に貼り付けることができます。ノード内で定義されたアクションおよびプロパティもコピーされるため、コピー機能を用いて若干の修正のみで再利用が可能な設計パターンを作成できます。

注意 : テンプレート定義間でノードをコピーする場合には、そのノードが参照する変数とそのアクションがコピー先のテンプレート定義内で作成されていること、その他の参照オブジェクトであるロール、ユーザ、ビジネス カレンダーなどがそのテンプレートに関連付けられたオーガニゼーションについて定義されていることを確認する必要があります。

ノード内に定義されているすべてのプロパティとアクションもコピーされます。

テンプレート定義間でノードとそのプロパティをコピーする手順は、以下のとおりです。

1. 以下のいずれか 1 つを実行します。
   - 設計領域でコピーするノードを右クリックし、表示されるポップアップ メニューから [コピー] を選択します。
   - フォルダ ツリーで、コピーするノードを含むフォルダを展開します。コピーするノードを右クリックしてポップアップ メニューの [コピー] を選択します。
2. 以下のいずれか 1 つを実行します。

- カーソルをコピー先テンプレート定義の設計領域に置き、右クリックします。表示されたポップアップメニューから [貼り付け] を選択します。
- フォルダ ツリーで、該当するノード タイプのフォルダを右クリックします。表示されるポップアップ メニューから [貼り付け] を選択します。

コピーしたノードが設計領域とフォルダ ツリーに表示されます。このアクションの [プロパティ] ダイアログ ボックスが表示されます。すべての設定が元のアクションからコピーされています。

3. 必要に応じて、ノードのプロパティを変更します。

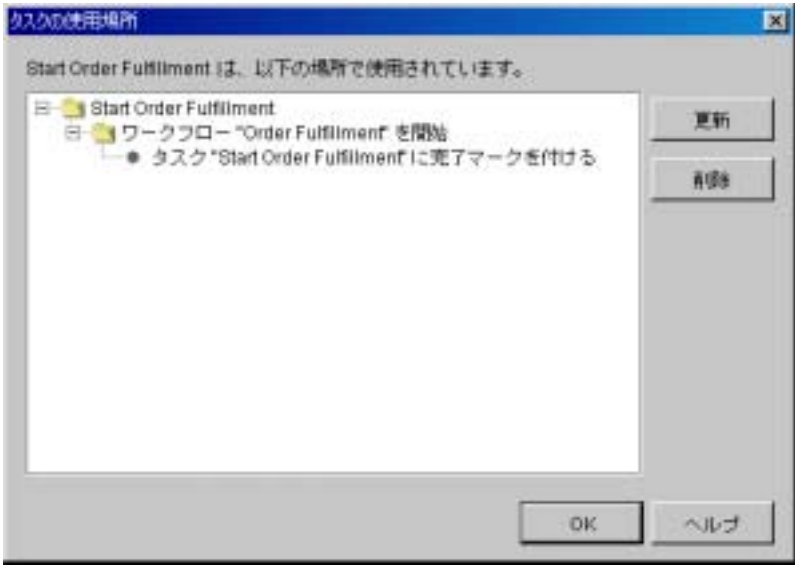
タスクおよびイベント用途を表示する

タスク アクションやワークフロー イベントを取消しアクションなどによりイベント またはタスク ノードが参照される別の場所を閲覧できます。詳細については、6-2 ページの「アクション カテゴリ」を参照してください。

タスク またはイベント ノードの使用場所を閲覧する手順は、以下のとおりです。

1. 設計領域またはフォルダ ツリーで目的のイベントまたはタスク ノードを右クリックし、ポップアップ メニューから [使用場所] を選択して [タスクの使用場所] または [イベントの使用場所] ダイアログ ボックスを開きます。

図 5-12 [タスクの使用場所] ダイアログ ボックス



2. 選択したノードを参照する項目が表示されるまでフォルダを展開します。
3. （省略可能）以下に示すダイアログ ボックスのボタンを使用して操作を行います。
   - [更新] ―ノードを参照する選択オブジェクトに対するダイアログ ボックスが開きます。
   - [削除] ―選択したオブジェクトを削除します。
4. [OK] をクリックして、[タスクの使用場所] または [イベントの使用場所] ダイアログ ボックスを閉じます。

変数に関する作業

各ワークフロー テンプレート定義には、関連した一連の変数を割り当てることができます。変数は、ビジネス オペレーションから返された値、XML ドキュメントから抽出された値、あるいはワークフロー アクションにより明示的に設定

された値を持つことができます。また、変数は、分岐ノードの条件の評価、Worklist クライアント アプリケーションからのレスポンス結果の保存など、ワークフローにより他の目的に用いられることもあります。

すべてのワークフロー テンプレート定義に変数が必要なわけではありません。しかし、変数が必要な処理を含むワークフロー テンプレート定義については、ノード プロパティやアクションなどの他のワークフロー コンポーネントの定義を開始する前に変数を定義したり、または設計プロセスで変数を追加したりできます。

ワークフロー変数の適用範囲はワークフロー全体です。つまり、単一の変数はワークフロー テンプレート定義インスタンス内のすべてのオブジェクトによって共有されます。よって、変数はフォルダ ツリーのワークフロー テンプレート定義レベルで定義され、ワークフロー変数と呼ばれます。

変数は、次のいずれかのタイプとなります。

表 5-2 ワークフロー変数のタイプおよび初期値

変数タイプ	内容	初期値
Boolean	ブール値 True または False	false
日付	Java date オブジェクト	現在の日付
Double	倍精度浮動小数点の数	0.0
エンティティ EJB	ビジネス オペレーションを実行アクションによって呼び出されるエンティティ EJB への参照（6-84 ページの「ビジネス オペレーションを呼び出す」を参照）	null
Integer	長整数	0
Java オブジェクト	ビジネス オペレーションを実行アクションによって呼び出される Java クラスへの参照（6-84 ページの「ビジネス オペレーションを呼び出す」を参照）。	null
セッション EJB	ビジネスオペレーションを実行アクションによって呼び出されるセッション EJB への参照（6-84 ページの「ビジネス オペレーションを呼び出す」を参照）。	null
文字列	文字列	空の文字列

表 5-2 ワークフロー変数のタイプおよび初期値

変数タイプ	内容	初期値
XML	XML ドキュメント（詳細については、6-22 ページの「変数値の設定」を参照）。	null

注意： 変数タイプに対して plug-in が定義されている場合、追加の変数タイプが選択可となる場合があります。

注意： 上記の表に示した初期値は、サーバのスタートアップ スクリプトの設定によって決められたものです。この設定を修正して、すべてのデータ型の初期値を NULL に変更できます。詳細については、『*WebLogic Integration の起動、停止およびカスタマイズ*』の「WebLogic Integration のカスタマイズ」にある「Null 変数をサポートする BPM のコンフィグレーション」を参照してください。

さらに、ワークフローが別のワークフローによって呼び出される場合（詳細については、5-34 ページの「開始のプロパティを定義する」を参照）、変数が*入力*または*出力*どちらのパラメータとして機能するかを指定する必要があります。入力パラメータは、変数がその値を呼び出し側、つまり親ワークフローから受け取ることを示します。出力パラメータは、呼び出し元のワークフローに返された値を変数が持つことを示します。

さらに、入力パラメータに対しては、このパラメータが*必須*であるかどうかも指定することもできます。入力パラメータが必須の場合、呼び出し側（親）ワークフローから変数の値が受け取られるまでワークフローは開始されません。値が受け取られない限りワークフローは開始されず、例外が発生します。

呼び出されたワークフローへの値の供給に関する詳細については、6-39 ページの「サブワークフローを呼び出す」を参照してください。

変数の初期値を設定する場合は、ノード内にあるワークフロー変数を設定アクションを用いる（詳細については、6-22 ページの「変数値の設定」を参照）か、開始またはイベント ノードの [変数] タブ、例外ハンドラ、あるいは XML をクライアントに送信アクションを用います（詳細については、5-46 ページの「イベント データからの変数を初期化する」を参照）。

変数がワークフロー式で用いられる場合、変数名の前にドル記号（$）またはコロン（:）、あるいは他の文字が入ります。変数表記の詳細については、8-3 ページの「変数の使い方」を参照してください。

変数を作成する

変数を作成する手順は、以下のとおりです。

1. フォルダ ツリーで適切なワークフロー テンプレート定義の [変数] を右クリックし、[変数を作成] を選択して [変数プロパティ] ダイアログ ボックスを開きます。

図 5-13 [変数プロパティ] ダイアログ ボックス

2. [名前] フィールドで `OrderID` などの有意な変数名を入力します。

   注意 : 変数名にはスペースは入れません。

3. [タイプ] ドロップダウン リストから、表 5-2 に示す変数タイプを選択します。

4. （省略可能） ワークフローが呼び出されたワークフローの場合、Parameter とそれが入力パラメータか出力パラメータか指定することもできます。入力パラメータには、パラメータが必須かどうかを指定します。

   パラメータは、値を渡すために呼び出し側ワークフローによって使用され、呼び出し側サブワークフローから値を受け取ります。入力パラメータは、サ

ブワークフローに渡される値を含み、出力パラメータはサブワークフローからの戻り値を含みます。

注意： ワークフローをプログラムによりインスタンス化する場合は、入力変数としてのみ指定された変数を設定できます。ワークフローをプログラムによってインスタンス化する方法については、『*BPM クライアント アプリケーション プログラミング ガイド*』の「手動によるワークフローの開始」を参照してください。

1 度ワークフローをインスタンス化すると、『*BPM クライアント アプリケーション プログラミング ガイド*』の「実行時変数のモニタリング」で説明されているように入力および出力変数を含むあらゆる変数の値を設定できます。

別のワークフローの開始の詳細は、6-39 ページの「サブワークフローを呼び出す」を参照してください。

5. （省略可能）変数に関するコメントを [メモ] テキスト ボックスに入力します。
6. [OK] をクリックして変数の定義を保存します。新しい変数がフォルダ ツリーの [変数] フォルダの下に表示されます。

変数を更新する

既存の変数を更新する手順は、以下のとおりです。

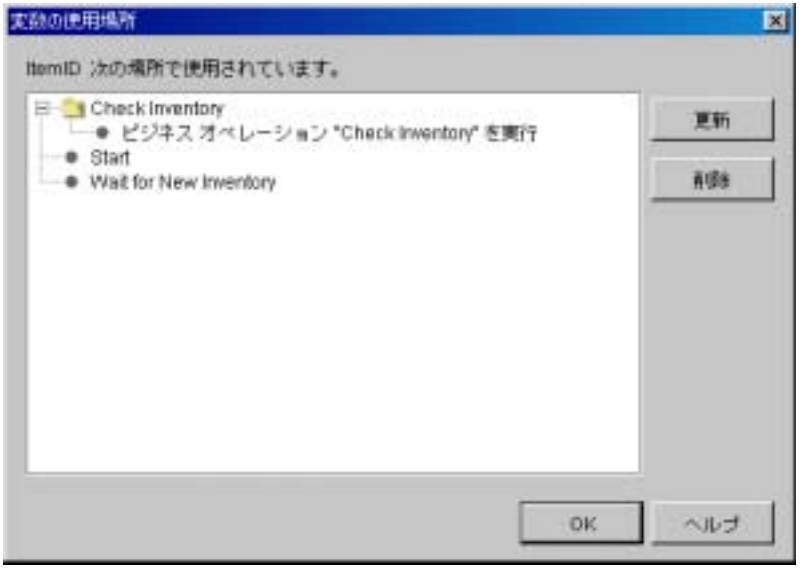
1. フォルダ ツリーで既存の変数を右クリックし、ポップアップ メニューから [プロパティ] を選択します。[変数プロパティ] ダイアログ ボックスが表示されます。
2. 必要に応じて変数を変更し、[OK] をクリックします。

変数の用途を表示する

変数がワークフローのどこで用いられるかを閲覧する手順は、以下のとおりです。

1. フォルダ ツリーで変数を右クリックし、ポップアップ メニューから [使用場所] を選択して [変数の使用場所] ダイアログ ボックスを表示します。このダイアログ ボックスには選択した変数が使用されている場所や値が割り当てられている場所が一覧表示されます。

図 5-14 [変数の使用場所] ダイアログ ボックス



2. 選択した変数を参照する項目が表示されるまでフォルダを展開します。
3. （省略可能）以下に示すダイアログ ボックスのボタンを使用して操作を行います。
   - [更新] －変数を参照する選択オブジェクトに対するダイアログ ボックスが開きます。
   - [削除] －選択したオブジェクトを削除します。
4. [OK] をクリックして、[変数の使用場所] ダイアログ ボックスを閉じます。

変数を削除する

どのワークフロー ノード、アクション、または式からも参照されていない変数のみ削除できます。参照変数が使用されている場所を閲覧する場合は、5-32 ページの「変数の用途を表示する」の手順に従います。

変数を削除する手順は、次のとおりです。

1. フォルダ ツリーで [変数] フォルダを展開し、削除する変数を右クリックして、ポップアップ メニューから [削除] を選択します。
2. 警告メッセージが表示されたら、[OK] をクリックして削除を確認するか、[取消し] をクリックして削除を取り消します。

ノードのプロパティの定義

この節では、特定の各ノード タイプによって、ノードのプロパティを定義する方法について説明します。

- 開始のプロパティを定義する
- イベントおよびイベント トリガ型開始のプロパティを定義する
- 分岐のプロパティを定義する
- タスクのプロパティを定義する
- 結合のプロパティを定義する
- 完了のプロパティを定義する

開始のプロパティを定義する

各ワークフローには、ワークフローの始まりを示す開始シェイプが 1 つ以上設定されています。開始後、最初のノードは、ワークフローで最初にアクティブになるノードで、タスク、分岐またはイベント ノードのどちらかです。

注意： 開始ノードが指定されていない場合、ワークフローではノードのアクティブ化は行われません。

開始ノードには 4 つのタイプのトリガがあります。

- 手動開始－ワークフローは Worklist あるいはクライアント アプリケーションからのエンド ユーザによって手動で開始されます。このタイプの開始には、エンド ユーザがワークフローを開始する時期を決定するための特定のビジネス条件を知っていることが必須となります。すべての要件を取り込むトリガを作成できない場合に、手動開始を作成する場合もあります。
- 呼び出し－ワークフローはワークフローを開始アクションを用いて別のワークフローから呼び出されることで開始されます。詳細については、6-39 ページの「サブワークフローを呼び出す」を参照してください。
- イベント － ワークフローは JMS キューによる XML ドキュメントの受け取りなどの外部イベントや、プラグイン定義イベントによるトリガによって開始されます。詳細については、5-48 ページの「イベント トリガ型開始のプロパティを定義する」を参照してください。
- 時限－ワークフローは時間によりスケジュールされたジョブとして、定義した日時に実行されます。詳細については、5-37 ページの「時限開始ノードを定義する」を参照してください。

注意： テンプレート定義の作成時、デフォルトの開始ノードは手動開始に設定されています。

以下のような多種の目的で複数の開始ノードを指定することもできます。

- 複数の個別作業パスが同時に開始されるよう指定する場合。これは、すべての開始ノードを手動開始に、同じ開始時間に、あるいは同じイベント トリガに設定することで実施できます。
- 複数の個別作業パスが別々に開始されるよう指定する場合。これは、各ノードで異なるトリガ タイプを指定することで実施できます。
- 同一の作業パスを開始させるために異なる条件、つまり異なるトリガ イベントを指定する場合。

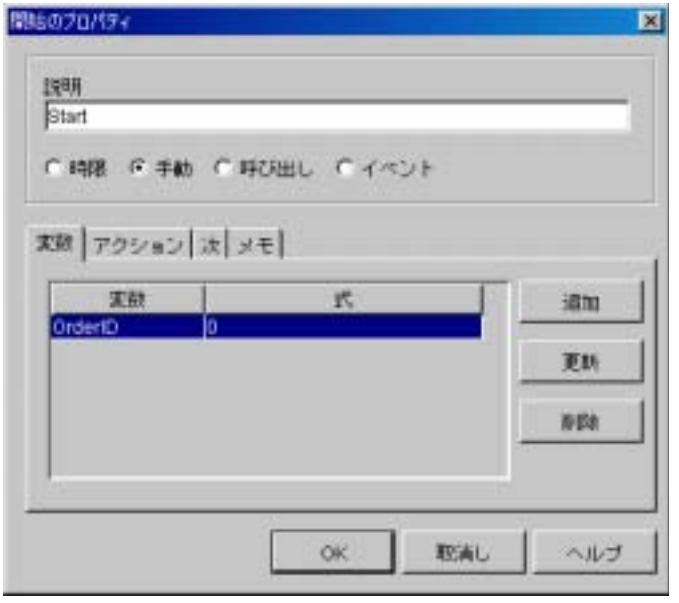
注意： シーケンシャルなローリング日時を採用するワークフローを指定して、1 つのフローが終了になると別のフローが開始されるようにする場合は、適切な有効日と終了日を設定して個別のテンプレート定義を行います。詳細については、5-7 ページの「テンプレート定義に関する作業」を参照してください。

最後に開始ノードを用いて、そのワークフローに対して作成された変数を初期化できます。たとえば、開始時の値として 1 を設定したいワークフローにカウンタを設定する場合があります。この値を開始ノードがアクティブになる際にカウンタ変数として割り当てることができます。詳細については、5-46 ページの「イベント データからの変数を初期化する」を参照してください。

開始ノードを定義する手順は、以下のとおりです。

1. 開始ノードをダブルクリックするか、フォルダ ツリーで開始ノードを右クリックして [プロパティ] を選択して、[開始のプロパティ] ダイアログ ボックスを表示します。

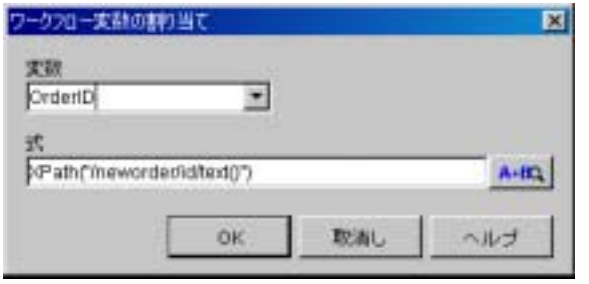
図 5-15 [開始のプロパティ] ダイアログ ボックス



2. (省略可能) [説明] フィールドで開始ノードの名前を固有で識別しやすい名前に変更します。
3. ワークフローのトリガ メソッドを選択します。時限を選択した場合、5-37 ページの「時限開始ノードを定義する」で説明する手順に従って追加のオプションを指定します。イベントを選択した場合、5-48 ページの「イベント トリガ型開始のプロパティを定義する」で説明する手順に従います。

4. （省略可能）ワークフロー開始時に変数を初期化するために [変数] タブを選択し、[追加] をクリックして [ワークフロー変数の割り当て] ダイアログ ボックスを表示します。

図 5-16 [ワークフロー変数の割り当て] ダイアログ ボックス



5. [変数] ドロップダウン リストから初期化する変数を選択します。
6. [式] フィールドに式を入力します。入力した式は実行時に評価され、結果として変数の値が得られます。定数の指定に使用する構文については、8-2 ページの「リテラルの使い方」を参照してください。
7. （省略可能）開始ノードの起動時に実行されるアクションを追加します。アクションに関する詳細については、第 6 章「アクションの定義」を参照してください。
8. [OK] をクリックして変更を保存します。

時限開始ノードを定義する

開始日式を指定することで、ワークフローを正確な日時に開始させることができます。ワークフローを開始するには、ワークフローが開始されるオーガニゼーションも指定する必要があります。

たとえば 2 日ごとなどのように、ワークフローが再開されるインターバルも指定できます。この場合、ワークフローのインスタンスはテンプレートの終了日になるまで 2 日 ごとに開始されます。テンプレートの終了日になると、ワークフローはそれ以上開始されません。

図 5-17 [開始のプロパティ] ダイアログ ボックス : [時限] オプション

時限開始ノードを定義する手順は、以下のとおりです。

1. [開始のプロパティ] ダイアログ ボックスで [時限] オプションを選択します。

2. [開始日の式] フィールドに絶対値または相対値として、ワークフローの開始日時を指定する式を入力します。式は必ず Date オブジェクトを返さなければならないため、以下のように日付関数を使用する必要があります。
   - 絶対日時を指定するには `StringToDate()` を使用します。詳細については、8-18 ページの「StringToDate()」を参照してください。
   - 関数 `DateAdd()` を用いて定数ベースの日時に相対する値を指定します。詳細については、8-20 ページの「DateAdd()」を参照してください。
3. （省略可能）[再スケジューリング] フィールドに値を入力し、ドロップダウン リストから時間の単位を選択することで、ワークフローが再開されるまでのインターバルを指定します。

   サーバが指定した開始時刻に実行されていない場合、指定した時刻に開始されるワークフローを回復するか（つまりサーバが再開始されるまで遅らせる）またはスキップするかを指定するには [回復可能] チェックボックスを設定します。
4. （省略可能）開始日を評価するために用いるビジネス カレンダーを選択します。
5. [開始オーガニゼーション] フィールドで、以下のいずれかを行ってワークフローを開始するオーガニゼーションを選択します。
   - ドロップダウン リストからオーガニゼーションを選択します。
   - [ワークフロー式を使用] チェックボックスをオンにし、[開始オーガニゼーション] フィールドで引用符で囲まれた文字列を入力するか、実行時に評価されオーガニゼーションの名前となる式を入力してオーガニゼーションを指定します。
6. [OK] をクリックして開始ノードを保存します。

イベントおよびイベントトリガ型開始のプロパティを定義する

ワークフローの開始や、ワークフロー内のノードのトリガはイベントによって行うことができます。イベントは別のワークフローや別のアプリケーションなどの外部ソースからの非同期の通知です。開始ノードはイベントトリガ型として定義され、イベント ノードは外部イベントによってのみトリガできます。

イベント通知は、一般的には JMS (Java Message Service) メッセージに含まれ、JMS キューで受信される XML ドキュメントの形をとります。ただし、プラグインで定義することもでき、その場合イベント通知は XML ドキュメントではなく、カスタムのトリガとなります（詳細については、『*WebLogic Integration BPM プラグイン プログラミング ガイド*』を参照）。

ワークフローによるメッセージの消費先である WebLogic Integration のデフォルトの内部 JMS キューの JNDI 名は、`com.bea.wlpiEventQueue` です。ただし、別のメッセージ キューの設定も可能です。詳細については、『*WebLogic Integration の起動、停止およびカスタマイズ*』の「WebLogic Integration のカスタマイズ」にある「カスタム Java Message Service キューのコンフィグレーション」を参照してください。

XML イベントの型では、実際のトリガは XML メッセージのプロローグで定義された文書型（DOCTYPE）の宣言であるか、または XML メッセージのルート要素であるかのいずれかです。開始またはイベント ノードのプロパティ ダイアログ ボックスで、イベントのトリガまたはワークフローの開始に使用する DOCTYPE またはルート要素を指定します。イベントは、ノードのプロパティ ダイアログ ボックスで指定された DOCTYPE、またはルート要素が着信 XML メッセージのそれと一致しない限りトリガされません。

DOCTYPE またはルート要素の使用に加え、イベント *キー*やイベント *条件*で、イベントをさらに修飾できます。そのようなサブワークフローについて以下で説明します。

イベント キーを理解する

イベント キーを使用すれば、開始ノードまたはイベント ノードをトリガすることになる受信 XML メッセージの内容、JMS ヘッダ、またはプロパティの値を指定できます。つまり、特定の DOCTYPE またはルート要素を含むすべての受信 XML ドキュメントにノードをトリガできるのではなく、XML 本体または JMS ヘッダ フィールドに含まれる特定の値に従って受信 XML メッセージのインスタンスをフィルタリングすることで、特定の値を含む XML メッセージ（複数）だけが、実行中のワークフローに含まれるノードをトリガできます。

イベント キーは以下の 2 つの部分で構成されます。

- *キー値の式*

  キー値の式は開始ノードまたはイベント ノードの [プロパティ] ダイアログ ボックスで指定します。キー値の式は実行時に評価されるワークフロー式

で、そのノードのトリガが可能となるように着信メッセージに含まれている正確なデータを指定します。開始ノードでは、式には一般的に着信 XML ドキュメント あるいは JMS ヘッダに含まれる特定の再帰データを参照する定数が含まれます。イベント ノードの場合、キー式には通常、実行時に一意の値を受け取る変数または関数が含まれています。イベント キーを用いる場合、各イベント ノードは固有のキーを指定します。

キー値の式のサンプルは以下の節で示しており、キー値の式の定義手順は、5-48 ページの「イベントトリガ型開始のプロパティを定義する」および 5-51 ページの「イベント プロパティを定義する」で説明します。

- *イベント キー式*

  これは、実行時に着信メッセージのヘッダまたは本文からキー値を返し、それを、開始ノードまたはイベント ノードの対応するキー値の式が必要とするデータ型にコンバートする式です。イベント キー式は、[コンフィグレーション] メニューからアクセスするイベント キー式のダイアログ ボックスで指定します。この式には一般的には、XML ドキュメントを解析するための XPath 言語の式、または JMS メッセージのヘッダから値を抽出するための EventAttribute() 関数の式が含まれています。イベント キー式のコンフィグレーションが行われると、すべてのオーガニゼーションのすべてのワークフローで使用できます。イベント キー式のサンプルは、以下の節を参照してください。また、イベント キー式の設定手順については、4-21 ページの「イベント キーのコンフィグレーション」を参照してください。

注意 : ワークフロー式言語の詳細については、第 8 章「ワークフロー式の使用法」で、XPath および `EventAttribute()` 関数の詳細については、8-6 ページの「実行時のイベント データを抽出する」で説明します。

XML コンテンツのイベント キーとして使用する

簡単なサンプルとして、受取勘定アプリケーションから定期的に顧客アカウント情報などを報告する XML メッセージを着信すると想定します。これらのメッセージには他のデータと並んで次の要素を含みます。

コード リスト 5-1 着信 XML ドキュメントのサンプル

```
<account>
        .
        .
```

```
        .
        <number>847365</number>
        <customer>John Doe</customer>
        <balance>
                <status>past due</status>
                <date_due>7-11-2001</date_due>
                <amount_due>5670.85</amount_due>
        </balance>
        <credit_limit>7500.00</credit_limit>
</account>
```

残高状態が（「OK」、あるいは「ペンディング」に対し）「期日到来済み（past due）」である着信ドキュメントのみがワークフローをトリガするよう、期限経過勘定を処理するワークフローがあるとします。最初に、着信ドキュメントのルート要素は、ドキュメントがこのワークフローのトリガとしてみなされるよう、必ず <account> となるよう指定します。次に past due の値に対応するイベント キーを作成します。実行時にイベント プロセッサは、着信 XML ドキュメントの残高状態要素から返された値を、開始ノードで指定した値と比較します。一致した場合、ワークフローがトリガされます。

通常開始ノードは、ワークフローの複数のインスタンスが着信 XML ドキュメントの複数のインスタンスによって開始できるよう、*定数*をイベント キーとして使用します。一方、ワークフロー内のイベント ノードは、通常、現在のワークフローの別の場所（たとえば、開始ノードの変数初期化時）で取り込まれた特定のデータを含む XML ドキュメントの特定のインスタンスによってのみトリガされる必要があります（5-46 ページの「イベント データからの変数を初期化する」を参照）。この値は設計時には決定できないため、ワークフロー*変数*または実行時に目的の値を返す関数として表す必要があります。たとえば、イベント インスタンスが正しいアカウント番号（この場合 847365）を含む XML インスタンスによってのみトリガされるように、コード リスト 5-1 のドキュメントを用いてイベント キーがアカウント番号の値を指定することもできます。実行時に、イベント プロセッサは、着信 XML ドキュメントのアカウント番号から返された値を、イベント ノードで指定した式から返された値と比較します。一致した場合イベントがトリガされます。

ここで、キー値およびイベント キー式の構築方法について、このサンプルの状況内で見てみましょう。開始ノードの場合、キー値の式は次のように（ワークフロー式構文で要求されるように引用符で囲まれた）定数で構成されます。

```
"past due"
```

イベント キー コンフィグレーションで設定した XML ドキュメントからこの値を返すよう要求されたイベント キー式は次のようになります。

```
ToString(XPath("/account/balance/status/text()"))
```

イベント ノードの場合、キー値の式はワークフロー インスタンスを実行することで早期に設定されたはずの値を持つ（ワークフロー式構文においてドル記号で示される）ワークフローで作成した変数で構成されます。

```
$AccountNumber
```

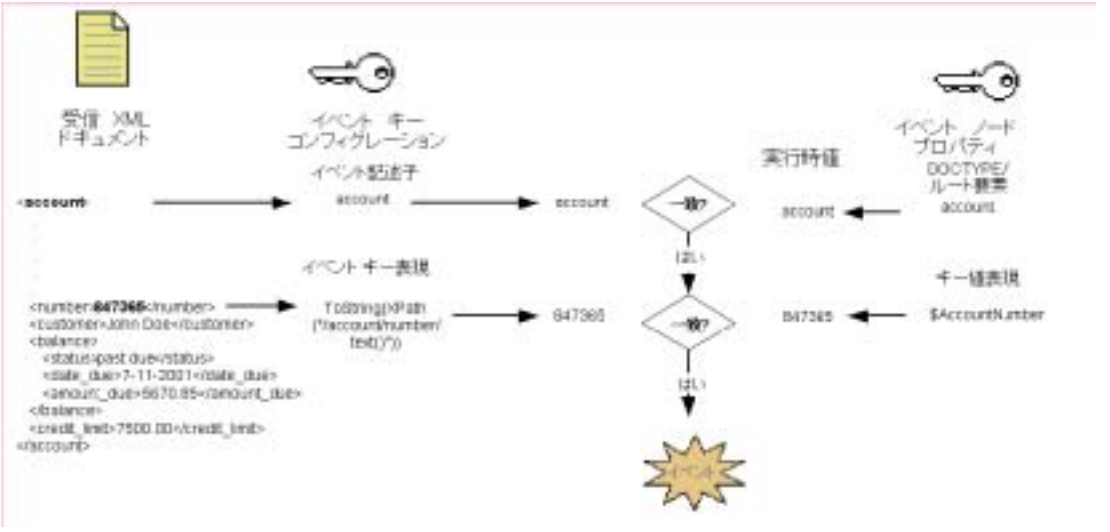
アカウント番号はワークフロー変数の中で文字列として格納され、XML ドキュメントからこの変数を返すためにイベント キー コンフィグレーションで必要になる式は次のとおりです。

```
ToString(XPath("/account/number/text()"))
```

注意： イベント キー式は、開始あるいはイベント ノードでキー値の式によって用いられたものと同じデータ型として評価される必要があります。XPath 式は*ノード リスト*のタイプを返すため、通常、正しいデータ型を返すには、ワークフロー式の言語で記述される型キャストを実行する関数を使用する必要があります。これらの関数の詳細については、8-16 ページの「データ型を変換する」を参照してください。返されたデータ型が文字列の場合、XML ドット表記を用いることもできます。詳細については、8-11 ページの「XML 要素のドット表記」を参照してください。

次の図に XML ドキュメントのイベント キーのメカニズムをまとめています。

図 5-18 イベント キー メカニズム



イベント キーとして JMS ヘッダまたはプロパティ データを使用する

`EventAttribute()` 関数を用いて、JMS ヘッダまたはプロパティから特定の値を検索することもできます。このメカニズムは XML ドキュメントに対しての場合とまったく同様に機能しますが、XML ドキュメントを解析してターゲット値を探すのではなく、値は JMS プロパティから抽出されます。

たとえば、送信元のアプリケーションがプロパティ フィールドを用いてメッセージの送信元の国を示し（これを `Country` と呼ぶ）、送信元の国によって開始される複数の異なるワークフローがあると想定します。この場合、イベント キー式は次のようになります。

```
ToString(EventAttribute(“Country”))
```

注意： イベント キー コンフィグレーション式は、開始あるいはイベント ノードでキー値の式によって用いられたものと同じデータ型として評価される必要があります。`EventAttribute()` 式は*オブジェクト*型を返すため、正しいデータ型を返すよう、ワークフロー式言語で与えられる型キャスト関数を用いる必要があります。これらの関数の詳細については、8-16 ページの「データ型を変換する」を参照してください。

カナダに関連する情報のみを処理するワークフローの場合、開始ノードのキー値式は次のようになります。

```
“Canada”
```

このメッセージには、Province と呼ばれるプロパティも含まれているとします。開始ノード内でこの情報を抽出し、これを州名を含むよう変数に保存します。ワークフローがインスタンス化されると、その特定の州向けに送られたメッセージのみがワークフローの別のイベント インスタンス（たとえば、売上税の値を算出するビジネス オペレーションの呼び出しなど）のトリガに用いられるようにします。この場合、州のイベント キーを作成します。イベント キー式は次のようになります。

```
ToString(EventAttribute(“Province”))
```

キー値の式は変数名で構成されます。

```
$ProvinceName
```

イベント条件を理解する

イベントまたは開始ノードのトリガをさらに修飾するために評価する必要のある条件を指定できます。これによりイベント プロセッサがイベント キーの一致を識別しても、条件が満たされるまではイベントがトリガされません。

注意： イベント キーなしでイベント条件を用いることはできますが、これは推奨されません。イベント キーのメカニズムでは、メモリに目的の値を格納し、着信 XML ドキュメントにおける DOM 解析が軽減されるため、単純なイベント条件に比べ、はるかに優れたパフォーマンスを提供します。イベント条件は、イベントをトリガする XML メッセージ インスタンスをさらに制限する場合に、イベント キーと共に*追加フィルタ*としてのみ使用します。

コード リスト 5-1 で示した XML ドキュメントのサンプルを用いて、ワークフロー内に期日到来済みで特定の額（たとえばアカウントのクレジット限度の 75%）以上の残高があるアカウントのクレジット凍結を発行するイベントがあると想定します。ワークフローでは、既に <amount_due> および <credit_limit> 要素から値を抽出し、2 つの変数 Amount_Due および Credit_Limit に格納しています。残高がクレジット限度の 75% を超えた場合にのみイベントをトリガする（および、クレジットの凍結を実行する）ための条件として次の式を用いることができます。

```
$Amount_Due > .75 * $Credit_Limit
```

コード リスト 5-1 に示すドキュメント インスタンス サンプルの場合、条件は true として評価され、イベントがトリガされます。

また、イベント条件で `XPath()` および `EventAttribute()` 関数を用いて、XML または JMS ヘッダまたはプロパティ データからコンテンツを直接抽出して、それを定数、変数あるいは他の関数を含む他のデータと比較することもできます。前出の国の例では、送信元の国が指定した国である場合にのみ、イベントが実行されました。国情報が <country> などの XML 要素に埋め込まれていることを想定すると、条件は次のようになります。

```
ToString(XPath("/root_element/child_element/country/text()")) =
"Canada"
```

国の値が `Country` と呼ばれる JMS プロパティに埋め込まれている場合は、条件は次のようになります。

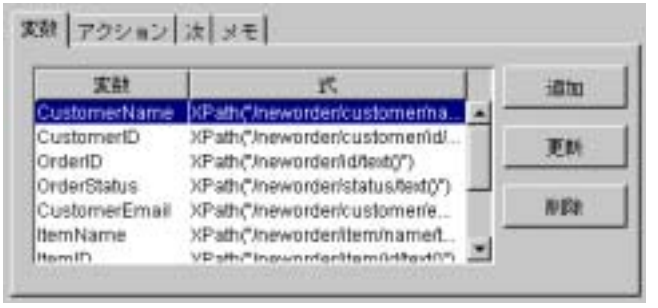
```
ToString(EventAttribute("Country")) = "Canada"
```

注意： `XPath()` または `EventAttribute()` などの関数で条件を用いる場合は、両方程式にある式は同じデータ型として評価されなければならず、同じデータ型でない場合、条件が処理できないという点に注意してください。Studio の型キャスト関数の詳細については、8-16 ページの「データ型を変換する」を参照してください。また、文字列値に XML ドット表記を用いることもできます。詳細については、8-11 ページの「XML 要素のドット表記」を参照してください。

イベント データからの変数を初期化する

開始 およびイベント ノードのプロパティ ダイアログ ボックスには、ワークフロー開始時またはイベントのトリガ時に値を設定する変数の追加、更新あるいは削除に用いることのできる [変数] タブがあります（変数の詳細については、5-28 ページの「変数に関する作業」を参照）。

図 5-19 開始およびイベント ノードの [プロパティ] ダイアログ ボックスの [変数] タブ



[追加] をクリックすると [ワークフロー変数の割り当て] ダイアログ ボックスが表示されます。ここでは、ワークフローに対し既に定義された変数に値を割り当てることができます。定義済みの変数は [変数] ドロップダウン リストに表示され、ここから初期化する変数を選択できます。

図 5-20 [ワークフロー変数の割り当て] ダイアログ ボックス

注意 : [アクション] タブのワークフロー変数を設定アクションを用いても同じことができます。実際、開始あるいはイベントを除くすべてのノードに対して、あるいは XML ドキュメントを XML 変数に割り当てる場合は、*必ず*このアクションを用いる必要があります（詳細については、6-22 ページの「変数値の設定」を参照）。ただし、ワークフローの実行時、開始またはイベント ノードでは [変数] タブに指定されている変数はアクションが実行される *前* に [アクション] タブに加えられた順番で初期化されます。

この機能は、開始ノードで用いてワークフローの開始時に変数を定数値に初期化することもできますが、複数の変数値を設定するために用いられる着信イベントデータの取り込みに用いることが最も便利と言えるでしょう。たとえば、ノードが JMS メッセージの着信 XML ドキュメントによってトリガされる場合、複数の式を用いて、変数をドキュメントに含まれる値あるいは JMS ヘッダまたはプロパティ フィールドで指定された値に初期化できます。

また、XML ドキュメントや JMS プロパティから現在のワークフローに渡されるインスタンス ID やテンプレート名など、他のワークフローのトラッキング属性を用いることもできます。たとえば、後に対話を開始したワークフローへのメッセージ応答で使用したい場合、これらの値を抽出して、それを変数に格納する必要があります。また、このメカニズムによって JMS トピックまたはキューを通じて外部アプリケーションに配信されるメッセージの単一の JMS プロパティ ヘッダで送ることのできる複数のワークフロー属性を集めることもできます（アドレス メッセージングおよび JMS プロパティとしてのワークフロー属性の挿入に関する詳細については、6-88 ページの「JMS トピックまたはキューへの XML メッセージのポスト」を参照）。

変数の値を更新する場合は、リスト内の変数名を強調表示させ、[更新] をクリックして [ワークフロー変数の割り当て] ダイアログ ボックスを開きます。

変数割り当てを削除する場合は、リストから変数名を選択し [削除] をクリックします。

イベントトリガ型開始のプロパティを定義する

イベントトリガ型での開始を定義する場合、ワークフローが開始されるオーガニゼーションを指定する必要があります。設計時にオーガニゼーションを指定するか、実行時に、たとえば着信イベント メッセージで指定されたデータを抽出するなどしてオーガニゼーションを決定する式を用いることが可能です。

イベントトリガ型開始ノードを定義する手順は、以下のとおりです。

1. [開始のプロパティ] ダイアログ ボックスで [イベント] オプションを選択します。

図 5-21 [開始のプロパティ] ダイアログ ボックス : [イベント] オプション

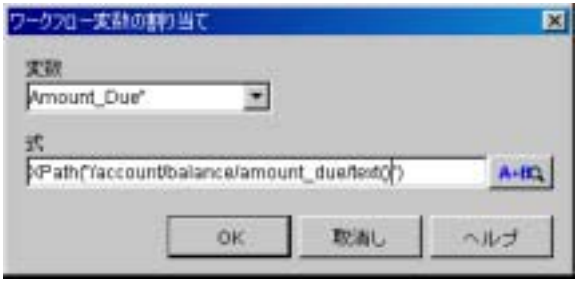
2. [ドキュメント タイプ / ルート要素] フィールドで、ワークフローの開始をトリガする XML メッセージの DOCTYPE またはルート要素を入力します。
3. (省略可能) [キー値の式] フィールドに、実行時にイベントをトリガする正確な XML コンテンツ、JMS ヘッダ、あるいはプロパティ フィールド値の評価となるワークフロー式を入力して XML メッセージのキー値を定義します。

式は通常、定数リテラルで構成されます。キー値式に関する詳細については、5-40 ページの「イベント キーを理解する」を参照してください。式の作成方法の詳細については、第 8 章「ワークフロー式の使用法」を参照してください。

注意 : また、プロセス エンジンがこのフィールドに指定したキー値と比較できるよう、着信 XML メッセージのキー値を配置するイベント キー コンフィグレーションを定義する必要があります。詳細については、4-21 ページの「イベント キーのコンフィグレーション」を参照してください。

4. (省略可能) [条件] フィールドで、ワークフローの開始前に評価する必要がある条件を定義します。イベント条件の詳細については、5-45 ページの「イベント条件を理解する」を参照してください。
5. [開始オーガニゼーション] フィールドで、以下のいずれかを行ってワークフローを開始するオーガニゼーションを選択します。
   - ドロップダウン リストからオーガニゼーションを選択します。
   - [ワークフロー式を使用] チェックボックスをオンにし、[開始オーガニゼーション] フィールドで引用符で囲まれた文字列を入力するか、実行時に評価されオーガニゼーションの名前となる式を入力してオーガニゼーションを指定します。これはたとえば、`XPath()` あるいは `EventAttribute()` 関数で着信 XML メッセージからオーガニゼーション情報を抽出する式です。
6. [変数] タブを選択して、着信イベント データなどからの変数を初期化し、[追加] をクリックして [ワークフロー変数の割り当て] ダイアログ ボックスを表示します。

図 5-22 [ワークフロー変数の割り当て] ダイアログ ボックス



7. [変数] ドロップダウン リストで、受信するデータを格納する変数を選択します。
8. 以下のいずれかを実行して、実行時に評価して変数の値を生成する式を [式] フィールドに入力します。
   - 定数の指定に使用する構文については、8-2 ページの「リテラルの使い方」を参照してください。
   - 着信 JMS ヘッダ データを取り込む場合、`EventAttribute()` 関数を使用します（詳細は、8-7 ページの「EventAttribute()」を参照）。
   - 着信 XML コンテンツを取り込む場合、`XPath()` 関数を用いる（詳細については、8-8 ページの「XPath()」を参照）か、文字列として返される XML 要素のドット表記を用います（詳細については、8-11 ページの「XML 要素のドット表記」を参照）。また [式] ボタン  を使用して XPath Wizard を呼び出し、このウィザードを使ってサンプルの着信ドキュメントから XPath の式を自動的に生成できます。詳細については、8-31 ページの「XPath Wizard を使用する XPath 式の作成」を参照してください。
9. [OK] をクリックします。変数の初期化が [開始のプロパティ] ダイアログ ボックスの [変数] タブのリストに表示されます。
10. 初期化するすべての変数に対し、手順 6 から 9 を繰り返します。
11. [OK] をクリックして変更を保存します。

イベント プロパティを定義する

イベント ノードを定義する手順は、以下のとおりです。

1. イベント ノードをダブルクリックするか、フォルダ ツリーでイベント ノードを右クリックして [プロパティ] を選択して、[イベントのプロパティ] ダイアログ ボックスを開きます。

図 5-23 [イベントのプロパティ] ダイアログ ボックス

2. [説明] フィールドでデフォルト名を修正して、Wait for New Inventory など、イベントに識別しやすい固有の名前を付けます。
3. [ドキュメント タイプ / ルート要素] フィールドで、イベントをトリガする XML メッセージの DOCTYPE またはルート要素を入力します。
4. (省略可能) [キー値の式] フィールドに、実行時にイベントをトリガする正確な XML コンテンツ、JMS ヘッダ、あるいはプロパティ フィールド値の評価となるワークフロー式を入力して XML メッセージのキー値を定義します。式は通常、変数またはワークフロー関数で構成されます。キー値の式に関する詳細については、5-40 ページの「イベント キーを理解する」を参照してください。式の作成方法の詳細については、第 8 章「ワークフロー式の使用法」を参照してください。

注意： また、プロセス エンジンがこのフィールドに指定したキー値と比較できるよう、着信 XML メッセージのキー値を配置する式を定義する必要があります。詳細については、4-21 ページの「イベント キーのコンフィグレーション」を参照してください。

5. (省略可能) [条件] フィールドで、イベントのトリガ前に評価する必要のある条件を定義します。イベント条件の詳細については、5-45 ページの「イベント条件を理解する」を参照してください。
6. [変数] タブを選択して、着信イベント データなどからの変数を初期化し、[追加] をクリックして [ワークフロー変数の割り当て] ダイアログ ボックスを表示します。

図 5-24 [ワークフロー変数の割り当て] ダイアログ ボックス

7. [変数] ドロップダウン リストで、受信するデータを格納する変数を選択します。
8. 以下のいずれかを実行して、実行時に評価して変数の値を生成する式を [式] フィールドに入力します。
   - 定数の指定に使用する構文については、8-2 ページの「リテラルの使い方」を参照してください。
   - 着信 JMS ヘッダ データを取り込む場合、`EventAttribute()` 関数を使用します（詳細は、8-7 ページの「EventAttribute()」を参照）。
   - 着信 XML コンテンツを取り込む場合、`XPath()` 関数を用いる（詳細については、8-8 ページの「XPath()」を参照）か、文字列として返される XML 要素のドット表記を用います（詳細については、8-11 ページの「XML 要素のドット表記」を参照）。また [式] ボタン  を使用して XPath Wizard を呼び出し、このウィザードを使ってサンプルの着信ドキュメントから XPath の式を自動的に生成できます。詳細については、

8-31 ページの「XPath Wizard を使用する XPath 式の作成」を参照してください。

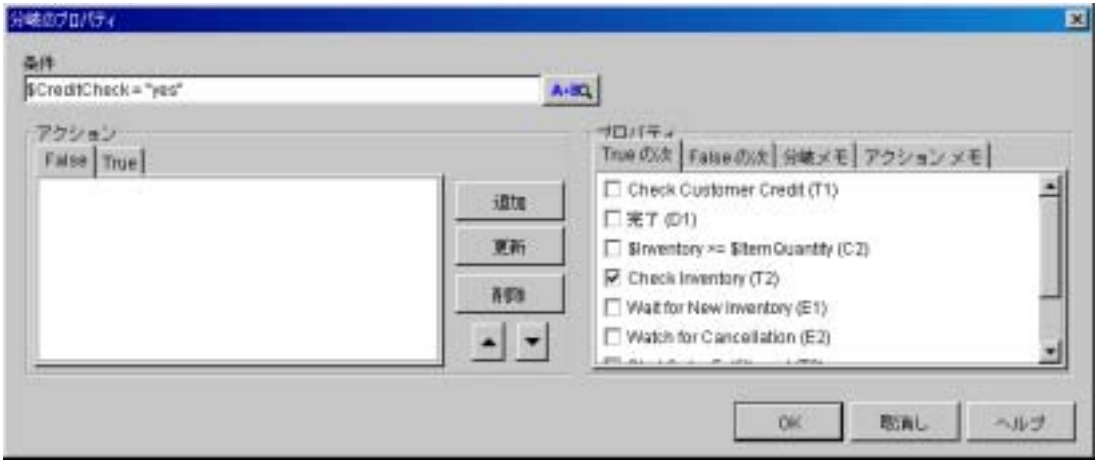
9. [OK] をクリックします。変数の初期化が [イベントのプロパティ] ダイアログ ボックスの [変数] タブのリストに表示されます。
10. 初期化するすべての変数に対し、手順 6 から 9 を繰り返します。
11. 〔省略可能〕 イベントのトリガ時に実行されるアクションを追加します。アクションに関する詳細については、第 6 章「アクションの定義」を参照してください。
12. [OK] をクリックして変更を保存します。

分岐のプロパティを定義する

ワークフローでは分岐をいくつでも使用できます。各分岐ノードには条件が指定されています。条件は、分岐ノードに移行するときに評価されます。結果は True または False となり、その結果に基づいて異なるパスに渡されたコントロールのサブシーケンス フローを持ちます。

また、条件の評価の結果が True または False のどちらの場合でも実行するアクションを指定することもできます。[True] タブおよび [False] タブに定義したアクションは、true または false の分岐先に指定されたノードの前に実行されます。また、フォルダ ツリーにおいて、これらのアクションは [分岐] フォルダ内の [True] および [False] フォルダに表示されます。

図 5-25 [分岐のプロパティ] ダイアログ ボックス



分岐ノードを定義する手順は、以下のとおりです。

1. 分岐ノードをダブルクリックするか、フォルダ ツリーで分岐ノードを右クリックして [プロパティ] を選択して、[分岐のプロパティ] ダイアログ ボックスを開きます。
2. [条件] フィールドで実行時に評価される条件式を指定します。条件には、定数、変数、関数を含めることができます。式の作成方法の詳細については、第 8 章「ワークフロー式の使用法」 を参照してください。
3. （省略可能）[False] や [True] タブにアクションを追加して、実行時の条件の結果（true または false）に基づいて実行されるアクションを指定します。これらのアクションは、後続ノードに指定されたアクションよりも前に実行される。

   アクションに関する詳細については、第 6 章「アクションの定義」 を参照してください。
4. [OK] をクリックして変更を保存します。

タスクのプロパティを定義する

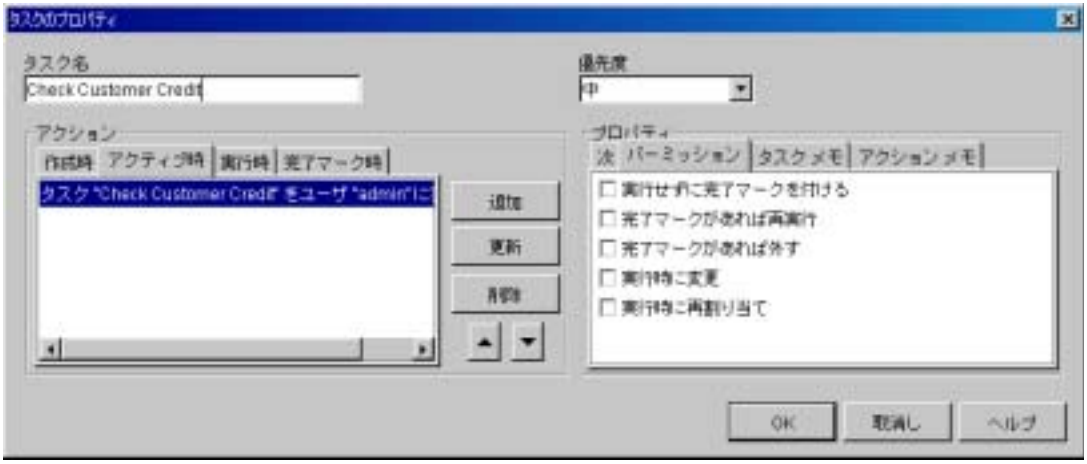
タスク ノードは、以下のように定義されます。

- ソフトウェア コンポーネントによって手動または自動で実行されるアクションを含む作業ユニット。
- 手動で実行するために Worklist または カスタム クライアント アプリケーション ユーザに割り当てられたタスク。

タスク ノードはワークフローの基本構成要素であり、フロー内の単一の操作（複数のワークフロー アクションによって実装される必要はある）を表します。

ワークフロー トランザクション モデルでは、ワークフロー ノードは、『*BPM クライアント アプリケーション プログラミング ガイド*』の「BPM トランザクション モデルの理解」で説明するように、トランザクションがどのように処理されるかを決定するアクティブ化および実行の状態になります。タスク以外のすべてのノードでは、これらの状態は Studio で明示的に表されているものではありません。ただし、タスク ノードでは 4 つのタスク状況のうちのいずれかに従ってアクションを指定する必要があり、フローのコントロールを効果的に操作するには、これらの意味を理解することが重要となります。

図 5-26 [タスクのプロパティ] ダイアログ ボックス



タスクの状態を理解する

各タスク ノードは、作成時、アクティブ時、実行時 および Marked Done の 4 つの異なる状態を通じて進行します。各ステータスに対してアクションのリストを指定できる。タスクのステータスが変わると、それに応じて指定したアクションが実行される。次の表では、各タスク状態とそのタスクの到達時に実行されるアクションを示します。

表 5-3 タスク状態

タスク状態	アクションの設定および実行される状況	一般的に指定されるアクション
作成時	新しいワークフロー インスタンスが作成されたとき。 タスクは、ワークフローでタスク ノードに到達するまで作成時状態が維持される。	開始ノードで変数を初期化できるため、この状態に対してはアクションを指定する必要はほとんどない。
アクティブ時	前のノードが処理を終了したとき。 タスクは、以下によって実行されるまでアクティブ時状態が維持される。 ■ Worklist またはカスタム クライアント ユーザ ■ タスクを実行アクション ■ プログラム的な API 呼び出し	■ ユーザ割り当てタスク アクションに先立つ、あるいはそれを含まないアクション。 ■ ユーザ割り当てタスク アクション。
実行時	■ ユーザが Worklist または カスタム クライアント アプリケーションを用いて手動でタスクを実行したとき ■ タスクを実行アクションが指定されたとき ■ プログラム的な WebLogic Integration API の使用 タスクが完了するまでワークフローは進行せず、タスクのステータスは [実行時] のままである。	ユーザ割り当てタスク アクションに続くアクション。

表 5-3 タスク状態

タスク状態	アクションの設定および実行される状況	一般的に指定されるアクション
完了マーク時	■ ユーザが Worklist または Studio アプリケーションで手動でタスクに完了マークを付けたとき ■ タスクに完了マークを付けるアクションが指定されたとき ■ プログラム的な WebLogic Integration API の使用	以下のノードでアクションを指定できるため、この状態に対してはアクションを指定する必要はほとんどない。

他のノードと異なり、すべてのタスク ノードにノードの完了を知らせるために完了マークを付ける必要があります。マークされない限りワークフローは次に進みません。手動割り当てタスクの場合、実行時にタスクに完了マークを付けることを可能にするパーミッション（以下を参照）を設定できます。ただし、多くの場合、設計時にタスクに完了マークを付けるアクションの追加によって明示的にタスクに完了マークを付けます。

タスクに完了マークを付けるアクションを置くタブは、タスクが 表 5-3 に示されるどの手段によって実行されるかによって異なります。タスクに完了マークを付けるアクションを実行する場合、タスクに完了マークを付けるアクションを [実行時] タブに指定し、実行しない場合は [アクティブ時] タブに指定します。詳細については、6-11 ページの「タスクに完了マークを付ける」を参照してください。

[タスクのプロパティ] ダイアログ ボックスで定義されたアクションは、その配置に従ってフォルダ ツリーの [作成時]、[アクティブ時]、[実行時] あるいは [完了マーク時] フォルダの下に表示されます。

タスク パーミッションについて

パーミッションをタスクに割り当て、Worklist (またはカスタム クライアント) ユーザ、あるいは Studio でインスタンスをモニタしている管理者による、実行時のタスクに対して行われる操作の型をコントロールできます。Studio での実行時におけるタスク操作実行に関する詳細については、10-14 ページの「タスクのパーミッションと優先度の変更」および 10-16 ページの「タスクのステータスと

割り当ての変更」を参照してください。Worklist での実行時におけるタスク操作実行に関する詳細については、『*WebLogic Integration Worklist ユーザーズ ガイド*』を参照してください。

タスク ノードを作成する場合、デフォルトでは初めから割り当てられているパーミッションはありませんが、次の表に示す有効なタスク パーミッションを有効化できます。

表 5-4 タスク パーミッション

パーミッション	説明
実行せずに完了マークを付ける	Worklist ユーザまたは Studio 管理者に、実行されていないタスクに完了マークを付けることを許可し、タスクの完了日付を手動で設定できる。
完了マークがあれば再実行	Worklist ユーザは、完了済みのタスクを再実行できる。
完了マークがあれば外す	Worklist ユーザまたは Studio 管理者に、完了マークが付いているタスクの状態を未完了に戻すことを許可する（ユーザが完了済みのタスクに完了済みではないというマークを付けると、タスクのステータスは Active に変更されるが、実行済みのアクションによる効果が取り消されることはない）。
実行時に変更	Worklist ユーザまたは Studio 管理者に、タスク実行前にタスクに対するパーミッションを変更することを許可する。
実行時に再割り当て	Worklist ユーザに、タスクを受けるか再割り当てすることを許可する。あるいは、Studio 管理者に、タスク実行前にタスクを別のユーザまたはロールに再り割当てすることを許可する。

タスクの優先順位について

手動割り当てタスクの場合、優先順位レベルを割り当てることができます。優先度は実行時におけるノードまたはノードの実行方法には影響を与えない。単にタスクを実行したりソートしたりできる Worklist のユーザに応じて表示される。

優先順位オプションは、低、中、および高です。デフォルト値は中です。

タスク ノードを定義する

タスク ノードを定義する手順は、以下のとおりです。

1. タスク ノードをダブルクリックするか、フォルダ ツリーで [タスク] を右クリックして [プロパティ] を選択して、[タスクのプロパティ] ダイアログ ボックスを表示します。
2. [タスク名] フィールドでデフォルト名を修正して、`Confirm Order` など識別しやすい固有の名前を付けます。この値は、主に Worklist アプリケーションで用いられますが、さまざまな Studio のモニタリング レポートから見ることができます。
3. 適切なタブで各タスク状態に対して実行されるアクションを追加します。これらの状態に関する詳細については、表 5-3 を参照してください。アクションの指定の詳細は、第 6 章「アクションの定義」を参照。
4. （省略可能）手動割り当てタスクに対し、表 5-4 で説明するように、[パーミッション] タブでチェック ボックスにチェックしてパーミッションを割り当てます。
5. （省略可能）手動割り当てタスクに対し、[低]、[中] あるいは [高] を選択して優先順位を割り当てます。
6. 非手動割り当てタスク、あるいは実行せずに完了マークを付けるパーミッションが無効な手動割り当てタスクに対しては、タスクに完了マークを付けます。詳細については、6-11 ページの「タスクに完了マークを付ける」を参照してください。
7. [OK] をクリックして変更を保存します。

結合のプロパティを定義する

結合ノードを使って、タスク、イベント、および分岐ノードの複数の経路を 1 つの経路に結合します。AND 結合の場合、ワークフローはすべてのパスの実行の終了を待ってから次のノードに進みます。OR 結合を使用すると、先行する 1 つの経路の実行が完了した後、その他の先行ノードは実行されずにワークフローは次のノードに進みます。

結合ノードを作成した後で、AND から OR、または OR から AND に変更できます。設計領域内の結合ノードと対応するシェイプは、自動的に更新されます。

図 5-27 [結合のプロパティ] ダイアログ ボックス

結合ノードを定義する手順は、以下のとおりです。

1. 結合ノードをダブルクリックするか、フォルダ ツリーで 結合ノードを右クリックして [プロパティ] を選択して、[結合のプロパティ] ダイアログ ボックスを開きます。
2. 以下のオプションのうち 1 つを選択します。
   - And ― ノードを AND 結合に変更します。
   - Or ― ノードを OR 結合に変更します。
3. [OK] をクリックして変更を保存します。設計領域内のシェイプが更新されます。

完了のプロパティを定義する

完了ノードはワークフロー内のプロセス終了ポイントを示します。ワークフロー内の 1 つの完了ノードに到達すると、他の完了ノードに到達したかどうかとは無関係に、ワークフロー内で実行中のインスタンスは完了とマークされます。

ワークフローでは完了ノードをいくつでも指定できます。ワークフローがさまざまなポイントから終了できる場合は、必要に応じてどこにでも個別の完了ノードを配置することが可能になります。

完了ノードにアクションを追加することはできますが、推奨はされません。

完了ノードを定義する手順は、以下のとおりです。

1. 完了ノードをダブルクリックするか、フォルダ ツリーで完了ノードを右クリックして [プロパティ] を選択して、[完了のプロパティ] ダイアログ ボックスを開きます。

図 5-28 [完了のプロパティ] ダイアログ ボックス

2. （省略可能）完了ノードの到達時に実行されるアクションを追加します。
3. [OK] をクリックして変更を保存します。

例外ハンドラに関する作業

例外ハンドラはサーバの例外の発生によってトリガされ、例外ハンドラの呼び出しアクションによって呼び出されて、メイン ワークフロー内のアクションのサブフローのように機能します。例外ハンドラの定義と呼び出しに関する詳細については、第 9 章「ワークフロー例外の処理」を参照してください。

6　アクションの定義

この章では、WebLogic Integration アクションを使用して各種ワークフロー アクティビティを遂行する方法について説明します。

- アクションの概要
- アクション定義タスクの概要
- アクションの操作
- 変数値の設定
- プログラム フローの制御
- 時限オペレーションの使用法
- サブワークフローの使用法
- 実行時状態のモニタリング
- 手動タスクの設定
- 電子メール メッセージを送信
- コンポーネントの呼び出し
- JMS トピックまたはキューへの XML メッセージのポスト
- XML ドキュメントの変換
- 例外処理

アクションの概要

ビジネス プロセスの論理的順序と制御を指定するにはノードとコネクタが使用されますが、ワークフロー定義の実際の作業を実行するのはアクションです。アクションは、何らかのアクティビティを実行できる最も基本的なワークフローの単位です。このアクティビティには、変数の初期化のような単純なものから、ク

ライアント システム上でカスタム アプリケーションを呼び出すような複雑なものまであります。アクションは、すべてのタイプのノード（結合ノードを除く）、例外ハンドラ、および他のアクションに追加できます。

以下の節では、ワークフローでアクティビティを実行するためにアクションを使用する場合の重要な基礎情報を示します。構成は以下のとおりです。

- アクション カテゴリ
- アクション タイプと配置を理解する
- タスク ノードにアクションを置く

アクション カテゴリ

Studio では、アクションはいくつかのカテゴリに分類されます。ただし、このガイドでは、実行できる主要アクティビティにアクションを分類します。次の表に、カテゴリ、カテゴリに含まれるアクション、実行されるアクティビティおよびこのガイドで詳細に説明されている節をリストします。

カテゴリ	アクション	使用目的
タスク	■ [タスクに完了マークを付ける] ■ [タスクの完了マークを外す] ■ [タスクを実行]	メイン プログラム フローの制御。 詳細は、6-25 ページの 「プログラム フローの制御」 を参照。
	■ [ユーザにタスクを割り当て] ■ [ロールにタスクを割り当て] ■ [ルーティング テーブルを使用してタスクを割り当て] ■ [タスクの割り当てを解除] ■ [タスク コメントを設定] ■ [タスク優先度を設定]	手動タスクの割り当てとそのプロパティの設定。 詳細については、6-48 ページの 「手動タスクの設定」 を参照。
	■ [タスク期日を設定]	手動タスクおよび非手動タスクの期日の設定、あるいは時間遅延の導入。 詳細については、6-48 ページの 「手動タスクの設定」 を参照。

カテゴリ	アクション	使用目的
ワークフロー	■ [ワークフローに完了マークを付ける] ■ [ワークフローを中断]	メイン プログラム フローの制御。 詳細は、6-25 ページの「プログラム フローの制御」を参照。
	■ [ワークフローを開始]	サブワークフローの呼び出し。 詳細については、6-38 ページの「サブワークフローの使用法」を参照。
	■ [ワークフロー変数を設定]	変数の初期化。 詳細については、6-22 ページの「変数値の設定」を参照。
	■ [ワークフロー コメントを設定]	実行時状態のモニタ。 詳細については、6-45 ページの「実行時状態のモニタリング」を参照。
統合	■ [ビジネス オペレーションを実行] ■ [プログラムの呼び出し]	ワークフローを、EJB や Java クラスのような外部ソフトウェア コンポーネント、あるいは実行可能プログラムと統合。 詳細については、6-82 ページの「コンポーネントの呼び出し」を参照。
	■ [XML をクライアントに送信]	Worklist 上またはカスタム クライアント システム上で別のオペレーションを実行。 詳細については、6-48 ページの「手動タスクの設定」を参照。
	■ [XML イベントをポスト]	ワークフロー、コンポーネント、アプリケーションの間で XML メッセージを交換。 詳細については、6-88 ページの「JMS トピックまたはキューへの XML メッセージのポスト」を参照。
	■ [XSL 変換]	スタイル シートに従って、XML ドキュメントをあるフォーマットから別のフォーマットに変換。 詳細については、6-103 ページの「XML ドキュメントの変換」を参照。

カテゴリ	アクション	使用目的
例外処理	■ [ワークフロー例外ハンドラを設定] ■ [例外ハンドラを編集]	ワークフローのために使用する例外ハンドラの管理。 詳細については、第 9 章「ワークフロー例外の処理」を参照。
	■ [例外ハンドラの呼び出し]	例外を処理するサブワークフローの呼び出し。 詳細については、第 9 章「ワークフロー例外の処理」を参照。
その他	■ [処理なし] ■ [ワークフロー イベントを取消し]	メイン プログラム フローの制御。 詳細については、6-25 ページの「プログラム フローの制御」を参照。
	■ [条件を評価]	条件付きサブワークフローの埋め込み。 詳細については、6-38 ページの「サブワークフローの使用法」を参照。
	■ [時限イベント]	時限サブワークフローの埋め込みと時間遅延の導入。 詳細については、6-33 ページの「時限オペレーションの使用法」を参照。
	■ [電子メール メッセージを送信]	電子メールをユーザ、ロール、または外部クライアントに送信。 詳細については、6-78 ページの「電子メール メッセージを送信」を参照。
	■ [監査エントリを作成]	実行時状態のモニタとワークフロー設計のデバッグ。 詳細については、6-45 ページの「実行時状態のモニタリング」を参照。

アクション タイプと配置を理解する

タスク、ワークフロー、統合、例外処理、その他の Studio により表わされるアクション カテゴリだけでなく、アクションとその動作の間に存在する以下のような区別も知っておく必要があります。

■ 端末アクションと非端末アクション（後者は他のアクションを含めることができる）

- アクションまたはサブアクションの同期的実行と非同期的実行（後者は並行処理が可能）

これらは、ワークフロー オペレーションが実行される順序に影響することがあるため、その区別は重要です。それぞれの区別の詳細について、以下の節で説明します。

端末アクションと非端末アクション

大部分のアクションは端末アクションです。つまり、アクションの中に他のアクションを含めることはできません。一方、一部のアクションは、その内部にサブアクションを定義できます。サブアクションは、親アクションの性質に応じて特定の条件に従って実行されます。以下のアクションは、その内部で定義されるサブアクションを持つことができるため非端末アクションです。

- [条件を評価] ー定義された条件の結果が True であるか False であるかに応じて、サブアクションを実行できます。
- [時限イベント] ーサブアクションは日付と時刻に従って実行され、スケジュールに従って再実行できます。
- [XML をクライアントに送信] ーワークフローがクライアント アプリケーションからの応答を受け取った際に、サブアクションを実行できます。
- [ワークフローを開始] ー呼び出されたワークフローの完了時に、サブアクションを実行できます。
- [タスク期日を設定] ータスクの期日が過ぎた後に、サブアクションを実行できます。

このような非端末アクションの場合、サブアクションは [プロパティ] ダイアログ ボックスにのみ表示され、フォルダ ツリーには表示されません。

非端末アクションに関する特記事項として、大部分の端末アクションは常に*同期*方式で実行される、つまりワークフローは端末アクションの実行が終了するまで待機しますが、非端末アクションは同期方式または*非同期*方式のどちらでもオペレーションを実行できます。非端末アクションの非同期的実行とは、ワークフローがアクションまたはそのサブアクションのオペレーションの完了を待たずに先に進み、並行して処理を続けることです。以下で、この区別について詳しく説明します。

アクションの同期的実行と非同期的実行

すべてのアクションは同期的です。ワークフローは現在のアクションにより実行されているオペレーションが完了するまで次のアクションに進みません。ただし、以下の例外があります。

- [プログラムの呼び出し] ―このアクションは、ワークフローと常に並行実行される実行可能プログラムを呼び出します。
- [XML イベントをポスト] ―このアクションは「発信後削除（fire and forget）」 モデルで XML メッセージを送信するため、メッセージの受信者によりトリガされるすべてのワークフローまたはアプリケーションは並行実行されます（このアクションの非同期的機能を強制する方法についての詳細については、6-88 ページの「JMS トピックまたはキューへの XML メッセージのポスト」 を参照）。

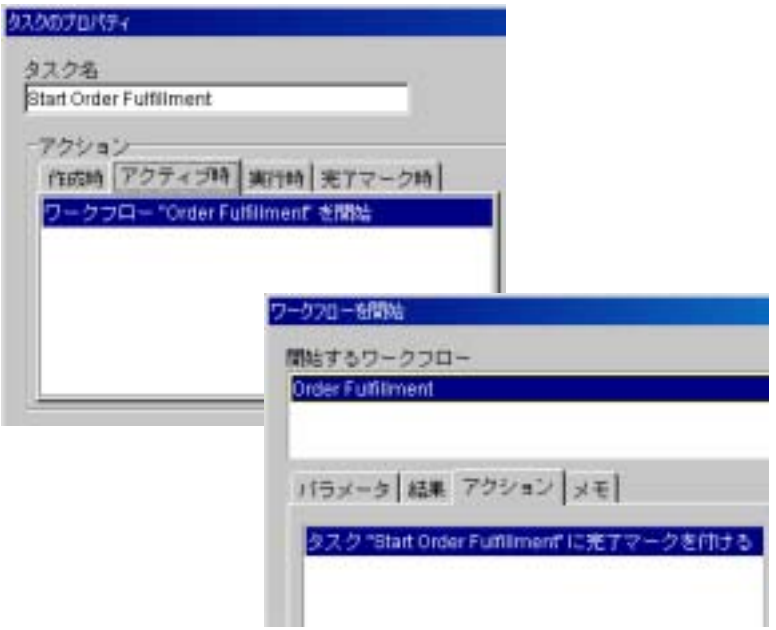
ただし、以下のアクションと、それに含まれるすべてのサブアクションは、アクションがどのように配置されているかに応じて同期方式 *または* 非同期方式のいずれかで実行できます。

- [時限イベント] ―サブアクションを非同期的または同期的に実行できます。
- [XML をクライアントに送信] ― XML メッセージ伝送、応答の受領および任意のコールバックの各サブアクションを同期的または非同期的に実行できます。
- [ワークフローを開始] ―サブワークフローおよび任意のサブアクションを非同期的または同期的に実行できます。
- [タスク期日を設定] ―任意の期日超過サブアクションを非同期的または同期的に実行できます。

上記の非端末アクションとそのサブアクションは以下の条件で非同期的に実行されます。

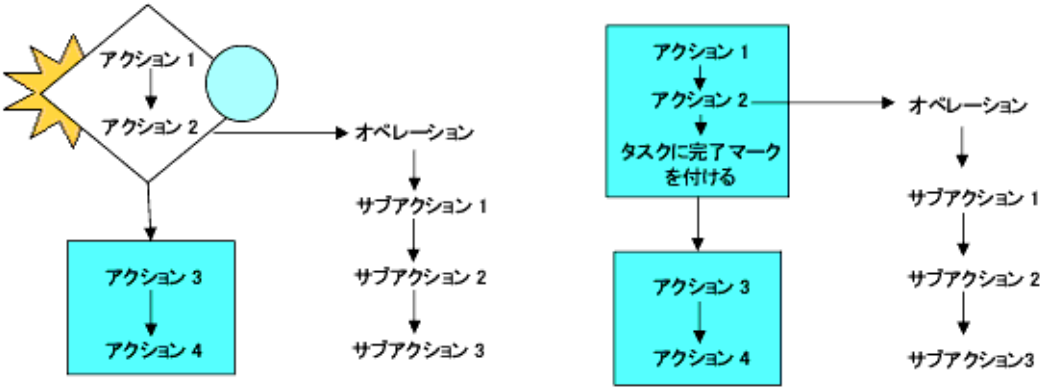
- アクションが開始、イベント、または分岐のいずれかのノードに置かれている場合。
- 次の図に示すように、アクションがタスク ノードに置かれており、タスクに完了マークを付けるアクションが [タスクのプロパティ] に置かれている場合。

図 6-1 [タスクのプロパティ] から完了マークを付けられたタスクを持つタスク ノードの非端末アクション



上記の場合、アクションのオペレーションおよび指定されたサブアクションは、ワークフロー内の後続のノードで並行実行されます。これを次の図に示します。

図 6-2 ワークフロー アクションの非同期的実行

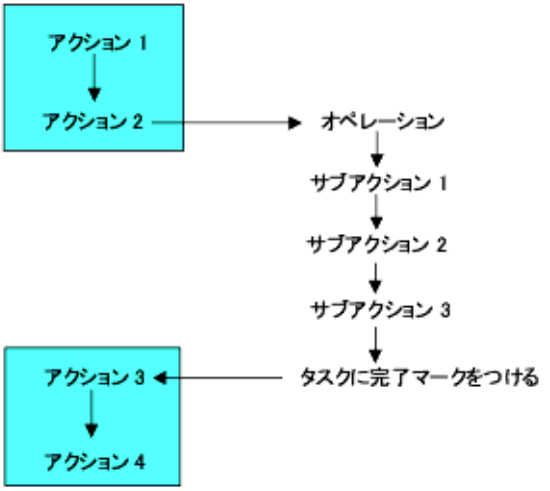


一方、非端末アクションとそのサブアクションは、アクションのプロパティでそれらのアクションの置かれているタスク ノードに完了マークが付けられている場合、同期的に実行されます。これを次の図に示します。

図 6-3 [アクションのプロパティ] で完了マークが付いたタスクを持つ非端末アクション

つまり、ワークフローは次の図に示すように、すべてのオペレーションとサブアクションが完了するまで待ってから次のノードに進みます。

図 6-4 ワークフロー アクションの同期的実行



アクション設計問題に関する詳細については以下のトピックを参照してください。

- 6-39 ページの「サブワークフローを呼び出す」
- 6-34 ページの「時限シーケンスを埋め込む」
- 6-56 ページの「タスク期日を設定する」
- 6-62 ページの「クライアント アプリケーションに対し XML メッセージを送信する」

タスクに完了マークを付ける方法の詳細については以下の節を参照してください。

タスク ノードにアクションを置く

タスク ノードを使用する場合、特定のアクティビティを実行するために、アクションをどのように置くことがよいか知っておく必要があります。そのためには、まず、[タスクのプロパティ] ダイアログ ボックスの適切なタブ（[アクティブ時] または [実行時]）にアクションを置き、次に適切なポイントでタス

クに完了マークを付けます。[タスクのプロパティ] ダイアログ ボックスの各タブの詳細については、5-57 ページの 「タスクの状態を理解する」 を参照してください。

[アクティブ時] タブと [実行時] タブを使用する

タスクは以下の方法でのみ実行されます。

- ユーザにタスクを割り当てアクションを使用して、Worklist またはタスクを手動で実行するカスタム クライアント ユーザにタスクを割り当てます。
- タスクを実行アクションを使用します。
- プログラム的に API コールを通じて実行します。

したがって、一般的にはアクション配置を考える場合に以下のガイドラインに従います。

- ユーザが手動でタスクを実行する前に実行される必要のあるアクション（あるいは上のリスト以外の手段により実行されるアクション）は、[アクティブ時] タブに置きます。
- ユーザにタスクを割り当てアクションは、常に [アクティブ時] タブのリストの最後に置きます。[アクティブ時] タブでこのアクション以降にリストされたアクションは実行されません。
- [実行時] タブでは、ユーザにタスクを割り当てアクションが最初に来るように配置します。

[タスクのプロパティ] ダイアログ ボックスの各タブを使用して、このガイドで説明する主なアクティビティを実行する場合のガイドラインをまとめた詳しい表は、6-13 ページの「タスク ノードでのアクション配置に関するガイドライン」にあります。

タスクに完了マークを付ける

すべてのタスクには完了マークを付ける必要があります。これは、タスクに対して実行せずに完了マークを付けるパーミッションが割り当てられている場合は Worklist またはカスタム クライアント ユーザにより手動で（詳細については 5-58 ページの「タスク パーミッションについて」 を参照）、あるいはタスクに完了マークを付けるアクションを使用することにより設計時に明示的に行います。

タスクに完了マークが付けられていない場合、ワークフローはタスク ノードから先に進みません。逆に、[タスクのプロパティ] ダイアログ ボックスでタスクに完了マークを付けるアクション以降にリストされたアクションは、[タスクのプロパティ] ダイアログ ボックスの [完了マーク時] タブに置かれている場合を除き実行されません。

注意 : 同様に、非端末アクションの場合、[アクションのプロパティ] ダイアログ ボックスの [タスクに完了マークを付ける] アクション以降に置かれたアクション（6-7 ページの「アクションの同期的実行と非同期的実行」に説明）は実行されません。

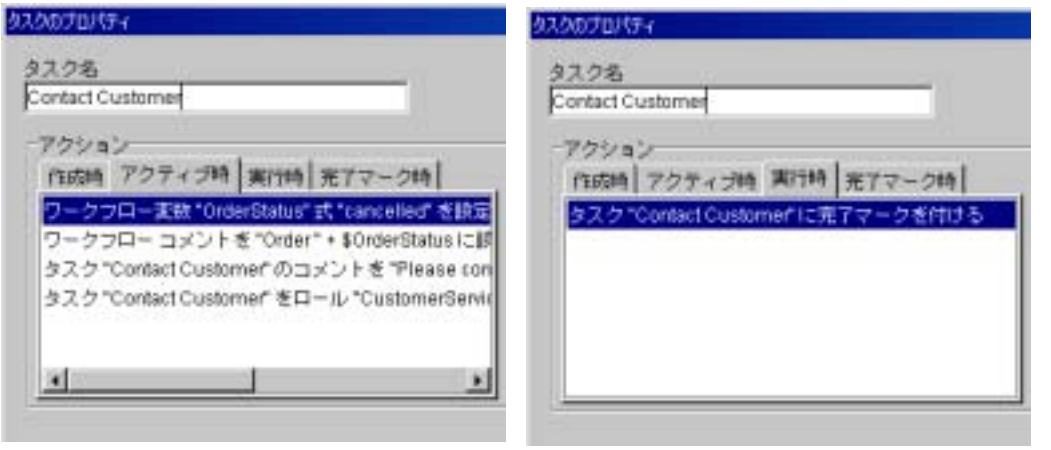
タスクに完了マークを付けるために使用できる一般的ガイドラインは以下のとおりです。

- タスクにそれを実行できるアクションが含まれていない場合、次の図に示すように、[アクティブ時] タブの最後のアクションとしてタスクに完了マークを付けるアクションを置きます。

図 6-5 実行されないタスクに完了マークを付ける

- ユーザにタスクを割り当てやタスクを実行など、タスクにそれを実行できるアクションを含まれている場合、[実行時] タブの最後のアクションとしてタスクに完了マークを付けるアクションを置きます。

図 6-6　実行されるタスクに完了マークを付ける



- 非端末アクションとそのサブアクションを非同期的に実行するには、[タスクのプロパティ] ダイアログ ボックスで、[アクティブ時] タブまたは [実行時] タブのうち適切な方の最後のアクションとしてタスクに完了マークを付けるアクションを置きます（図 6-1 を参照）。
- 非端末アクションとそのサブアクションを同期的に実行するには、アクションのプロパティ ダイアログ ボックスの最後のアクションとしてタスクに完了マークを付けるアクションを置きます（図 6-3 を参照）。

[タスクのプロパティ] ダイアログ ボックスの各タブを使用して、このガイドで説明する主なアクティビティを実行する場合のガイドラインをまとめた詳しい表は以下の節にあります。

タスク ノードでのアクション配置に関するガイドライン

通常はタスク ノードにのみアクションを置き、他のタイプのノードでアクションを使用することは避けてください。また、少数のタスク ノードに長いアクション リストを置くよりも、短いアクション リストを多数のタスク ノードに置くようにしてください。浅いまたはフラットな設計アプローチを使用すると、グラフィック表現の中で、ワークフローのロジックやトランザクション単位を一目で把握できます。

ただし、場合によっては、同じノードに複数のアクションを指定しなければならないこともあります。たとえば、ユーザに割り当てられるタスクにさまざまなプロパティを指定する必要がある場合などです（詳細については、6-48 ページの「手動タスクの設定」を参照）。同様に、タスク以外のノードにアクションを追加する方が効率的な場合や、新たなタスク ノードを作成してグラフィカル フローを複雑にするよりも、1 つのタスク ノードにアクション リストを置く方が効率的な場合もあります。たとえば、変数値の設定、タスクまたはワークフローのコメントの設定、監査エントリの作成のような単純なアクティビティの場合が該当します。

次の表に、主な一般的アクティビティを実行する再利用可能なノード パターンを設計する場合の参考となるガイドラインを示します。

表 6-1 アクション配置のガイドライン

アクティビティ	タスク ノードに置くアクション	アクションを置くタブ	タスクに完了マークを付けるタブ	詳細の参照先
Java コンポーネントの呼び出し	[ビジネス オペレーションを実行]	[アクティブ時]	[アクティブ時]	6-84 ページの「ビジネス オペレーションを呼び出す」
実行可能プログラムの呼び出し	[プログラムの呼び出し]	[アクティブ時]	[アクティブ時]	6-82 ページの「サーバ上の実行可能なプログラムを呼び出す」
XML ドキュメントの変換	[XSL 変換]	[アクティブ時]	[アクティブ時]	6-103 ページの「XML ドキュメントの変換」
電子メール送信	[電子メール メッセージを送信]	[アクティブ時]	[アクティブ時]	6-78 ページの「電子メール メッセージを送信」
JMS トピックまたはキューへの XML メッセージのポスト	[XML イベントをポスト]	[アクティブ時]	[アクティブ時]	6-88 ページの「JMS トピックまたはキューへの XML メッセージのポスト」

表 6-1 アクション配置のガイドライン

アクティビティ	タスク ノードに置くアクション	アクションを置くタブ	タスクに完了マークを付けるタブ	詳細の参照先
プラグイン アクション の使用	[カスタム アクション]	[アクティブ時]	[アクティブ時]	
手動タスクの割り当て	(省略可能) [タスク コメントを設定] (省略可能) [タスク期日を設定] [Assign Task to User/Role/Using Routing Table]	[アクティブ時]	[実行時]	6-48 ページの「手動タスクの設定」
	(省略可能) [XML をクライアントに送信]	[実行時]	[XML をクライアントに送信] ダイアログ ボックスの [コールバック アクション] タブ	6-62 ページの「クライアント アプリケーションに対し XML メッセージを送信する」
サブワークフローの同期的呼び出し	[ワークフローを開始]	[アクティブ時]	[ワークフローを開始] ダイアログ ボックスの [アクション] タブ	6-39 ページの「サブワークフローを呼び出す」

表 6-1 アクション配置のガイドライン

アクティビティ	タスク ノードに置くアクション	アクションを置くタブ	タスクに完了マークを付けるタブ	詳細の参照先
時間遅延の導入	[タスク期日を設定]	[アクティブ時]	[タスク期日を設定] ダイアログ ボックスの [期日に実行するアクション] タブ	6-56 ページの「タスク期日を設定する」
	[時限イベント]	[アクティブ時]	[時限イベント] ダイアログ ボックスの [トリガ時のアクション] タブ	6-34 ページの「時限シーケンスを埋め込む」

アクション定義タスクの概要

ワークフローを作成し、必要なリソースをコンフィグレーションした後、ノードなどのオブジェクトにアクションを追加できます。アクションの追加と定義は反復的プロセスです。新しいノードを追加する、新しい変数を作成する、データとリソースを再コンフィグレーションする、XML ドキュメントを定義するなどのタスクが必要です。アクションの定義には次のタスクが関係します。各タスクはどの順序で実行してもかまいません。

注意： 大部分のアクションでは、ダイアログ ボックス フィールドに式の入力が必要になるため、アクションの定義を開始する前に、ワークフロー式言語および Studio のツールである Expression Builer と XPath Wizard の学習も必要です。ワークフロー式に関する詳細については、第 8 章「ワークフロー式の使用法」を参照してください。

- ビジネス オペレーションを実行アクションを定義するには、4-9 ページの「ビジネス オペレーションのコンフィグレーション」に説明されているように、ビジネス オペレーションを定義します。
- （省略可能）ユーザにタスクを割り当て、ルーティング テーブルを使用してタスクを割り当て、または電子メール メッセージを送信の各アクションのためのユーザを指定する場合は、3-15 ページの「ユーザの保守」に説明されているように、ユーザを定義します。
- （省略可能）ロールにタスクを割り当て、ルーティング テーブルを使用してタスクを割り当て、または電子メール メッセージを送信の各アクションのためのロールを指定する場合は、3-21 ページの「ロールの保守」に説明されているように、ロールを定義します。
- （省略可能）時限イベント アクションまたはタスク期日を設定アクションの場合、3-4 ページの「ビジネス カレンダーの管理」に説明されているように、ビジネス カレンダーを定義します。あるいは、既存のワークフロー パッケージからエクスポート済みのカレンダーをインポートすることもできます。その順序については、11-5 ページの「ワークフロー パッケージのインポート」を参照してください。
- テンプレート定義にノードを追加します。手順の詳細は、5-19 ページの「ノードに関する作業」を参照してください。
- アクションにより参照される変数を作成します。手順の詳細は、5-28 ページの「変数に関する作業」を参照してください。
- タスク ノードにアクションを追加します。必要に応じて、開始、イベント、分岐ノードにも追加します。アクションを追加するための手順については、6-18 ページの「アクションを追加する」を参照してください。
- アクションを非端末アクションに追加します。
- タスクに完了マークを付けるアクションをタスク ノードまたは非端末アクションに追加します。手順の詳細は、6-26 ページの「タスクに完了マークを付ける」を参照してください。

- 例外ハンドラを追加します。例外ハンドラを定義するための手順については、9-3 ページの「例外ハンドラの定義」を参照してください。
- アクションを例外ハンドラに追加します。例外ハンドラを定義するための手順については、9-3 ページの「例外ハンドラの定義」を参照してください。
- 独立した各アクティビティがノードにより表わされるように、新しいノードをワークフローに追加し、設計を再定義します。ノードを追加するための手順については、5-20 ページの「ノードを追加、配置および接続する」を参照してください。
- ワークフローに XML ドキュメントを埋め込むアクション（XML イベントをポスト、XML をクライアントに送信、ワークフロー変数を設定など）が必要とする XML ドキュメントを作成またはインポートします。[アクションのプロパティ] ダイアログ ボックスを使用して XML ドキュメントを操作するための手順については、第 7 章「XML エンティティを操作」を参照してください。

アクションの操作

アクションの追加および更新は、[タスク]、[分岐]、[イベント]、[完了]、[開始のプロパティ] の各ダイアログ ボックス、ならびにカスタム例外ハンドラ、および 6-6 ページの「端末アクションと非端末アクション」にリストされた [アクションのプロパティ] ダイアログ ボックスで行います。各ダイアログ ボックスには、アクションの配置と保守のために使用する [アクション] タブがあります。この節では、すべてのアクションに共通な機能について説明します。

アクションを追加する

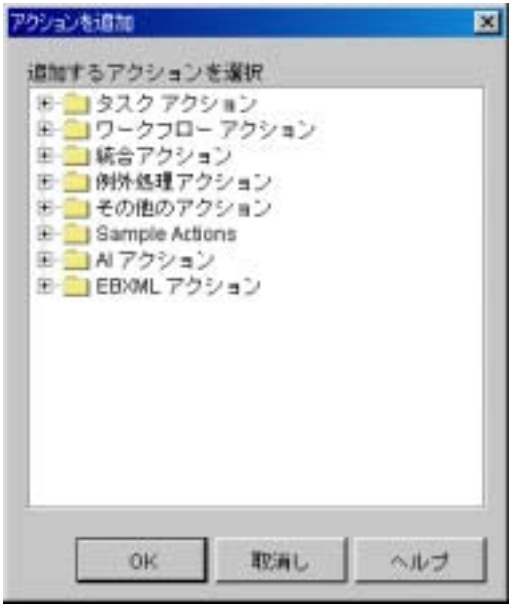
アクションを追加する手順は、以下のとおりです。

1. 以下のいずれか 1 つを実行します。
   - 設計領域で、アクションを追加するノードをダブルクリックし、表示された [プロパティ] ダイアログ ボックスで該当する [アクション] タブを選択して [追加] をクリックします。

- フォルダ ツリーで、アクションを追加するオブジェクトを表すフォルダを展開し、[アクション] フォルダを右クリックして、表示されたダイアログ ボックスで [アクションを作成] を選択します。

[アクションを追加] ダイアログ ボックスが表示されます。

図 6-7 [アクションを追加] ダイアログ ボックス



注意 : アクションに対してプラグインが定義されている場合、このダイアログ ボックスに新しいアクションが入ります。

2. アクション タイプのフォルダをダブルクリックして展開し、アクションを選択し、[OK] をクリックします。選択したアクションに固有な [プロパティ] ダイアログ ボックスが表示されます。

3. ダイアログ ボックスのフィールドに入力し、[OK] をクリックします。各アクション タイプに関する詳細な手順は、このガイドの残りの部分で説明します。

   アクションは、選択したノードの [プロパティ] ダイアログ ボックスに追加され次第、フォルダ ツリー内でそのアクションを含むノードまたは例外ハンドラを表すフォルダの下に表示されます。

アクションを更新する

アクションを更新する手順は、以下のとおりです。

1. 以下のいずれか 1 つを実行します。
   - 更新するアクションの入った [プロパティ] ダイアログ ボックスで、目的のアクションを選択し、[更新] をクリックします。
   - フォルダ ツリーで、更新するアクションの入ったフォルダを展開し、アクションを右クリックし、ポップアップ メニューから [プロパティ] を選択します。

   選択したアクションの [プロパティ] ダイアログ ボックスが表示されます。
2. アクション定義の変更が完了した後 [OK] をクリックします。

アクションを削除する

アクションを削除する手順は、以下のとおりです。

1. 以下のいずれか 1 つを実行します。
   - 削除するアクションの入った [プロパティ] ダイアログ ボックスで、目的のアクションを選択し、[削除] をクリックします。
   - フォルダ ツリーで削除するアクションの入ったフォルダを展開し、アクションを右クリックし、ポップアップ メニューから [削除] を選択します。
2. 警告メッセージが表示されたら、削除する場合は [OK] を、この操作を取り消す場合は [取消し] をクリックします。

アクションをコピーする

ワークフロー ノードの内部または相互間で、あるいはワークフロー テンプレート定義（およびテンプレート）の内部または相互間でアクションをコピーすることにより、再利用可能な設計パターンを作成できます。アクション内で定義されたプロパティもコピーされるので、アクションをわずかに変更するだけで済み、時間を節約できます。

注意： テンプレート定義間でアクションをコピーする場合は、そのアクションが参照する変数がコピー先のテンプレート定義内にすでに作成されていること、およびその他の参照オブジェクトであるロール、ユーザ、ビジネス カレンダーなどが、そのテンプレートと関連付けられているオーガニゼーションについて定義されていることを必ず確認してください。

ノード内またはノード間でアクションとそのプロパティをコピーする手順は、次のとおりです。

1. 以下のいずれか 1 つを実行します。
   - 設計領域でコピーするアクションの入っているノードをダブルクリックします。[プロパティ] ダイアログ ボックスで、適切な [アクション] タブを選択し、アクションを右クリックし、ポップアップ メニューから [コピー] を選択します。
   - フォルダ ツリーで、コピーするアクションを含むフォルダを展開し、コピーするアクションを右クリックしてポップアップ メニューの [コピー] を選択します。
2. 別のノードまたはテンプレート定義に貼り付ける場合は、次の操作のうち 1 つを実行します。
   - [OK] をクリックしてノードの [プロパティ] ダイアログ ボックスを閉じ、ターゲット テンプレート定義の設計領域で、定義済みアクションをペーストするノードのシェイプをダブルクリックします。[プロパティ] ダイアログ ボックスで適切な [アクション] タブを選択し、右クリックし、ポップアップ メニューから [貼り付け] を選択します。
   - フォルダ ツリーで、アクションをペーストするフォルダを展開し、[アクション] フォルダを右クリックし、ポップアップ メニューから [貼り付け] を選択します。

   このアクションの [プロパティ] ダイアログ ボックスが表示されます。すべての設定が元のアクションからコピーされています。
3. 必要に応じて、アクションの [プロパティ] ダイアログ ボックスの設定を変更します。
4. [OK] をクリックすると変更内容が保存され、コピー先のオブジェクトにアクションが追加されます。

アクションの順序を変更する

アクションは実行される順序で表示されます。アクションの順序を変更するには、[プロパティ] ダイアログ ボックスのリストからアクションを選択し、上矢印または下矢印を押してリスト内での位置を移動してください。

アクションへコメントを追加する

すべてのアクションの [プロパティ] ダイアログ ボックスには、アクションに関するコメントを入力できる [メモ] テキスト ボックスがあります。これは、同じワークフローにアクセスする他のユーザが、そのワークフローのロジックや設計を理解する必要のある場合に役立ちます。

図 6-8 [メモ] テキスト ボックス

変数値の設定

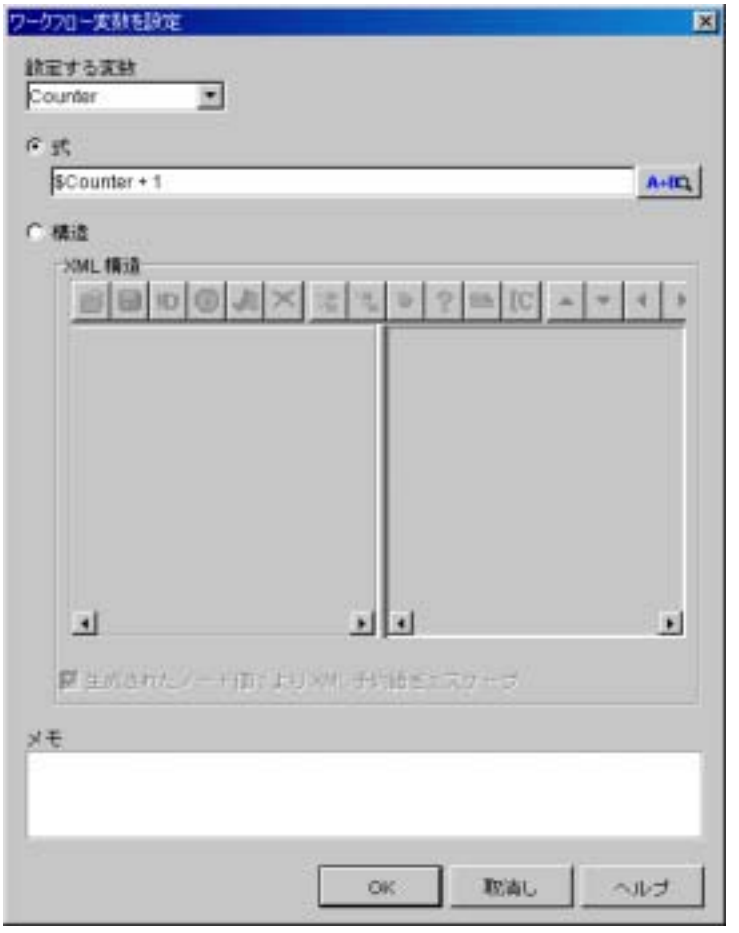
どのタイプのノードについても、ワークフローのどこかのポイントで既存のワークフロー変数に値を割り当てるには、ワークフロー変数を設定アクションを使用します。たとえば、ループを設計するには、このアクションを使用してカウンタをインクリメントできます。変数の定義方法の詳細は、5-28 ページの「変数に関する作業」を参照。

変数に対して割り当てる値は、式または定数にできます。式は、アクションの実行時に評価され、結果は指定された変数に割り当てられます。

このアクションを使用して、既存の XML ドキュメントの作成やインポートおよび XML または文字列型の変数への定数の格納を行うこともできます。この後、その変数を再利用して XML イベント アクションのポストなどのために、ワークフローのさまざまな場所でこの XML の内容を参照できます（6-88 ページの「JMS トピックまたはキューへの XML メッセージのポスト」参照）。

注意： 非 XML 変数の場合、開始ノード、イベント ノード、[XML をクライアントに送信]、[例外ハンドラのプロパティ] の各ダイアログ ボックスの [変数] タブを使用しても変数値を設定できます。

図 6-9 [ワークフロー変数を設定] ダイアログ ボックス



変数に値を割り当てる手順は、以下のとおりです。

1. [アクションを追加] ダイアログ ボックスの [ワークフロー アクション] フォルダから [ワークフロー変数を設定] を選択し、[OK] をクリックすると、[ワークフロー変数を設定] ダイアログ ボックスが表示されます。
2. [設定する変数] ドロップダウン リストから値を割り当てる既存の変数を選択するか、変数を入力します（詳細は、5-28 ページの「変数に関する作業」を参照）。
3. 以下のいずれかのオプションを選択します。

- 式－値を生成するために実行時に評価する式を入力します。ワークフロー式コンポーネントと構文の詳細については、第 8 章「ワークフロー式の使用法」を参照してください。定数を入力するには、8-2 ページの「リテラルの使い方」で説明されている構文を使用します。
- 構造－ XML ドキュメントを指定することにより、XML または文字列型の変数の値をセットします。新たに自由形式ドキュメントを作成するには [子を追加] ボタンをクリックし、ノードの追加を開始します。既存の XML ドキュメントを指定するには [インポート] ボタンをクリックし、ドキュメントをロードし、必要に応じて編集します。新たなタイプ指定 XML ドキュメントを作成するには [コンテンツ タイプを設定] ボタンをクリックし、スキーマ ドキュメントをロードします。上記のオプションの手順の詳細については、7-2 ページの「XML ドキュメントの作成と編集」を参照してください。

ユーザ定義の XML ドキュメントは、ワークフロー テンプレート定義内に保存される。

注意 : 文字列型の変数に格納された XML ドキュメントは、まず文字列に変換されます。

4. [OK] をクリックすると [ワークフロー変数を設定] アクションが追加されます。

プログラム フローの制御

通常は、プログラム フローを制御するにはノードとコネクタを使用してください。ただし、場合によってはノード内から代替実行パスを指定しなければならないこともあります。たとえば、ワークフローの特定のノードまでに限ってトリガを許可し、それ以降は禁止するようなイベントを定義することがあります。あるいは、条件に応じて別のノード内から自動的にタスクを実行しなければならないこともあります。最も重要なことは、タスクに完了マークを付けるアクションの位置により、サブアクションあるいはサブワークフロー全体を同期的に実行するか、非同期的に実行するかを制御できることです。

したがって、以下のアクションをプログラム制御のために使用できます。

- [タスクに完了マークを付ける] －ユーザが実行する手動タスクだけでなく、完全に自動化されたアクションの入ったタスク ノードに完了マークを付けることも、フローが正しく進行するには必要です。ワークフローが意図したと

おりに実行されるにはこのアクションが必要です。6-26 ページの「タスクに完了マークを付ける」で説明します。

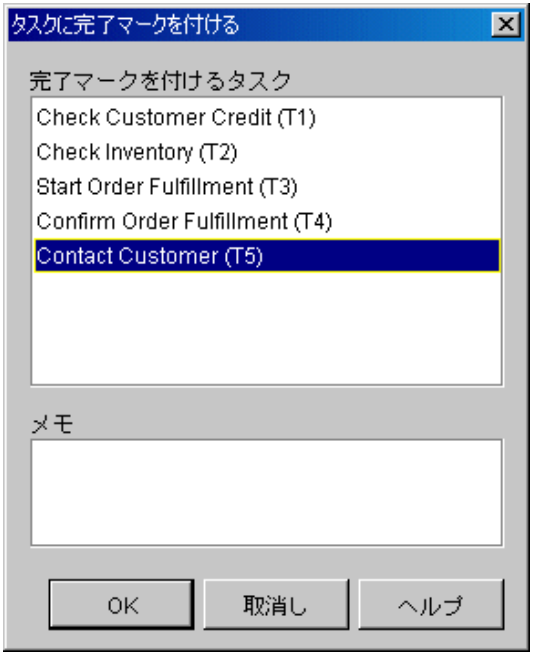
- [タスクの完了マークを外す] ―既に完了マークが付けられたタスクの完了マークを外します。6-28 ページの「タスクから完了マークを外す」で説明します。
- [ワークフロー イベントを取消し] ―指定したイベントをキャンセルし、ワークフローのこのポイント以降でこのイベントがトリガされないようにします。6-29 ページの「ワークフロー イベントをキャンセルする」で説明します。
- [ワークフローに完了マークを付ける] ―このアクションは完了ノードに進むことと同じです。6-30 ページの「ワークフローに完了マークを付ける」で説明します。
- [ワークフローを中断] ―実行中のインスタンスを終了し、データベースからそれを削除します。6-31 ページの「ワークフローを中断する」で説明します。
- [タスクを実行] ―タスク ノードを自動的に実行し、ノードの [実行時] タブにリストされたアクションを実行します。6-31 ページの「タスクを自動実行する」で説明します。
- [処理なし] ―このアクションは、実際にはワークフローに対しての影響はなく、ワークフロー設計者のためのプレースホルダとして機能します。6-33 ページの「プレースホルダ アクションを追加する」で説明します。
- [時限イベント] ―ワークフローで時間遅延を作成するために使用できます。6-34 ページの「時限シーケンスを埋め込む」で説明します。
- [タスク期日を設定] ―ワークフローで時間遅延を作成するために使用できます。6-56 ページの「タスク期日を設定する」で説明します。

タスクに完了マークを付ける

ユーザに割り当てられたタスクの定義で、ユーザが実行時にタスクに手動で完了マークを付けることを許可されている場合を除き（5-55 ページの「タスクのプロパティを定義する」を参照）、すべてのタスク ノードには、その [タスクのプロパティ] ダイアログ ボックス、または非端末アクションの [アクション] タブ

のいずれかでタスクに完了マークを付けるアクションを指定しておく必要があります。このアクションの使用法の詳細については、6-5 ページの「アクション タイプと配置を理解する」を参照してください。

図 6-10 [タスクに完了マークを付ける] ダイアログ ボックス



タスクに完了マークを付ける手順は、次のとおりです。

1. [アクションを追加] ダイアログ ボックスの [タスク アクション] フォルダで [タスクに完了マークを付ける] を選択し、[OK] をクリックして [タスクに完了マークを付ける] ダイアログ ボックスを表示します。
2. [完了マークを付けるタスク] フィールドで、アクション実行時に完了マークを付けるタスクを選択します。一度に選択できるタスクは、1 つのみ。
3. [OK] をクリックすると [タスクに完了マークを付ける] アクションが追加されます。

タスクから完了マークを外す

タスクの完了マークを外すアクションは、タスクが完了していないものとしてマークします。このアクションを割り当てられたタスクは、Worklist でアクティブになり再度実行できます。ループや並行ワークフローなどで、タスクを再度実行できるようにするには、このアクションを使用してください。

注意： 完了していないものとしてマークされたタスクは、再度トリガされて起動状態にはなりません。タスクを起動状態にトリガするには、ワークフローでタスクを再度開始する必要があります。タスク状態の詳細については、5-57 ページの「タスクの状態を理解する」を参照してください。

図 6-11 [タスクの完了マークを外す] ダイアログ ボックス

タスクから完了マークを外す手順は、以下のとおりです。

1. [アクションを追加] ダイアログ ボックスの [タスク アクション] フォルダから [タスクの完了マークを外す] を選択し、[OK] をクリックすると、[タスクの完了マークを外す] ダイアログ ボックスが表示されます。

2. [完了マークを外すタスク] フィールドで、アクション実行時に完了マークを付けるタスクを選択します。一度に選択できるタスクは、1 つのみ。
3. [OK] をクリックすると [タスクの完了マークを外す] アクションが追加されます。

ワークフロー イベントをキャンセルする

ワークフロー内で定義された 1 つまたは複数のイベント ノードをキャンセルする場合は、ワークフロー イベントを取消しアクションを使用してください。このアクションは、ワークフローの特定のポイント以降イベントがトリガされないようにするため、イベント ノードの終了メカニズムとして使用できます。たとえば、注文処理ワークフローで、顧客からのキャンセル通知を待つイベントがあり、その結果に応じて何らかのアクションを実行するとします。出荷後に注文のキャンセルを発行できないようにするには、ワークフローで出荷タスクが実行されるポイントでワークフロー イベントを取消しアクションを定義し、注文キャンセル イベントのトリガを禁止できます。

図 6-12 [ワークフロー イベントを取消し] ダイアログ ボックス

イベントをキャンセルする手順は、以下のとおりです。

1. [アクションを追加] ダイアログ ボックスの [その他のアクション] フォルダで、[ワークフロー イベントを取消し] を選択し [OK] をクリックして、[ワークフロー イベントを取消し] ダイアログ ボックスを表示します。
2. [取り消すイベント] フィールドで、キャンセルするワークフローを強調表示します（複数のイベントを強調表示するには〔Ctrl〕を押したままで選択）。
3. [OK] をクリックすると [ワークフロー イベントを取消し] アクションが追加されます。

ワークフローに完了マークを付ける

ワークフローに完了マークを付けるアクションは、現在のワークフローに完了マークを付け、ワークフローが完了ノードに到達したのと同じ状態にします。これは分岐ノードにおいて、完了ノードより前のすべてのアクションをスキップし、そのポイントでワークフローを終了したい場合などに使用できます。

図 6-13 [ワークフローに完了マークを付ける]

ワークフローに完了マークを付ける手順は、次のとおりです。

1. [アクションを追加] ダイアログ ボックスの [ワークフロー アクション] フォルダから [ワークフローに完了マークを付ける] を選択し、[OK] をクリックすると、[ワークフローに完了マークを付ける] ダイアログ ボックスが表示されます。
2. [OK] をクリックすると [ワークフローに完了マークを付ける] アクションが追加されます。

ワークフローを中断する

ワークフローを中断アクションは、現在処理中のワークフローを恒久的に停止します。このアクションは、インスタンスを終了しなければならない例外的な状況で使用できます。

注意 : 中断されたワークフロー インスタンスは、データベースから削除され、モニタできなくなります。ワークフローを終了させてもインスタンスのレコードは保持したい場合は、完了ノードまたは [ワークフローに完了マークを付ける] アクションを使用してください。

図 6-14 [ワークフローを中断] ダイアログ ボックス

ワークフローを中断する手順は、以下のとおりです。

1. [アクションを追加] ダイアログ ボックスの [ワークフロー アクション] フォルダから [ワークフローを中断] を選択し、[OK] をクリックすると、[ワークフローを中断] ダイアログ ボックスが表示されます。
2. [OK] をクリックすると [ワークフローを中断] アクションが追加されます。

タスクを自動実行する

タスクを実行アクションを使用して、ユーザではなくワークフローにワークフロー内のタスクを明示的に実行させることができます。このアクションによって、タスクのステータスは実行済みになり、[タスクのプロパティ] ダイアログ ボックスの [実行時] タブに一覧表示されているすべてのアクションが実行されます。タスク状態の詳細については、5-57 ページの 「タスクの状態を理解する」を参照してください。

このアクションを使用して、たとえば、例外ハンドラ内から例外ハンドラのアクションの完了後にタスク ノードのアクションを再実行させることができます。あるいは、グラフィックに表現するのが難しいループを作成するためにも使用できます。

図 6-15 [タスクを実行] ダイアログ ボックス

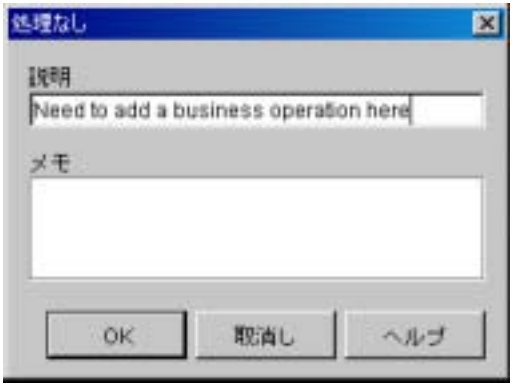
タスクを自動的に実行する手順は、以下のとおりです。

1. [アクションを追加] ダイアログ ボックスの [タスク アクション] フォルダから [タスクを実行] を選択し、[OK] をクリックすると、[タスクを実行] ダイアログ ボックスが表示されます。
2. [実行するタスク] フィールドで実行するタスクを選択します。一度に選択できるタスクは、1 つのみ。
3. [OK] をクリックすると [タスクを実行] アクションが追加されます。

プレースホルダ アクションを追加する

処理なしアクションはワークフローには影響しません。後でアナリストがアクションの追加を忘れないようにするためのプレースホルダとしての役割だけです。

図 6-16 [処理なし] ダイアログ ボックス



プレースホルダを追加する手順は、以下のとおりです。

1. [アクションを追加] ダイアログ ボックスの [その他のアクション] フォルダから [処理なし] を選択し、[OK] をクリックすると、[処理なし] ダイアログ ボックスが表示されます。
2. [説明] フィールドに処理なしアクションの内容を表わす名前を入力します。後でアクションを追加することを思い出させるような名前がよいでしょう。
3. [OK] をクリックすると [処理なし] アクションが追加されます。

時限オペレーションの使用法

以下のアクションにより時限オペレーションを設定できます。

- [時限イベント] ―実行する一連のサブアクションを指定します。任意で、正確なタイム スケジュールに従って再実行させることもできます。また、ワー

クフローで時間遅延を作成するためにも使用できます。6-34 ページの「時限シーケンスを埋め込む」で説明します。

- [タスク期日を設定] ―タスクを実行する期日を指定できます。必要であれば期日後に実行する一連のサブアクションも指定できます。また、ワークフローで時間遅延を作成するためにも使用できます。6-56 ページの「タスク期日を設定する」で説明します。

時限シーケンスを埋め込む

時限イベント アクションを使用して、正確な日時にトリガされる時限アクションのシーケンスを作成できます。必要に応じて指定されたスケジュールに従って再実行することもできます。

実行スケジュールについて

時限イベントを再スケジューリングする場合、2 つのオプションに従って実行停止メソッドを指定する必要があります。

- ワークフローが完了した場合。この場合、時限イベントはワークフローが完了するまで再実行されます。
- タスクが完了した場合。この場合、時限イベントは、そのイベントの入っているタスク ノードに完了マークが付くまで再実行されます。このオプションは、以下の 2 つの条件を満たす場合にのみ使用してください。
  - 時限イベント アクションがタスク ノードに指定されています。
  - そのタスク ノードがタスク ノード自体の外部、たとえば分岐ノードや他のアクションから完了マークを付けられます。タスク ノードをタスク ノードの内部から、あるいは [時限イベント] ダイアログ ボックスの [サブ アクション] タブ内から完了マークを付けることはできません。この方法でタスクに完了マークを付けることは、実際上、時限イベントのすべての反復実行を終了することになるからです。

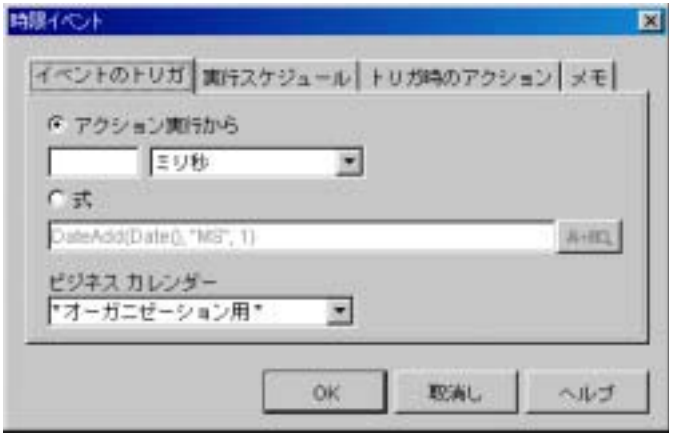
非同期的および同期的にトリガされたアクションを実行する

時限シーケンスに指定されたアクションは、同期モードまたは非同期モードのいずれでも実行できます。非同期モードではサブアクションは並行実行され、その間にワークフローは次のノードに進みます。この方法でアクションを設定するには、タスク ノードで時限イベントを使用し、タスク ノード自体で、または他のタイプのノードで、アクションの入っているタスクに完了マークを付けます。

同期モードでは、このアクションを使用してワークフローで時間遅延の作成ができます。ワークフローは、トリガ日時に達するまで待ってからサブアクションを実行し、その後、先に進みます。この方法でアクションを設定するには、タスク ノードでアクションを使用し、[時限イベント] ダイアログ ボックスの [トリガ時のアクション] タブに [タスクに完了マークを付ける] アクションを追加します。

どの場合でも、時限イベントを再スケジューリングするには、ワークフローが完了した場合に限り実行を停止してください。

図 6-17 [時限イベント] : [イベントのトリガ] タブ



時限イベントを定義する

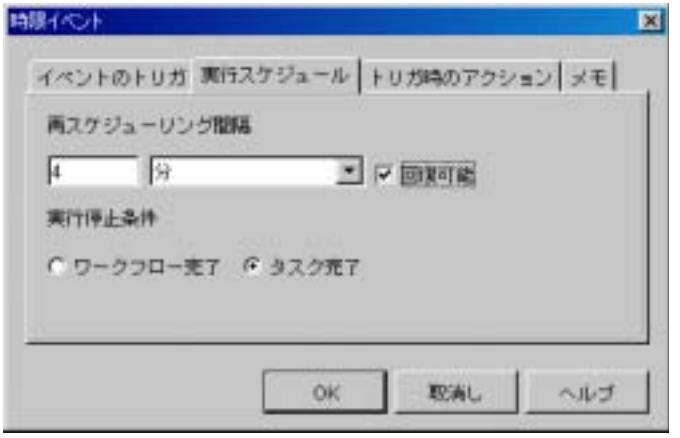
時限イベントを定義する手順は、以下のとおりです。

1. [アクションを追加] ダイアログ ボックスの [その他のアクション] フォルダで、[時限イベント] を選択し [OK] をクリックして、[時限イベント] ダイアログ ボックスを表示します。
2. [イベントのトリガ] タブで以下のいずれかのオプションを選択することにより、サブアクションを実行する時刻を指定します。
   - アクション実行から－時限イベントが実行された後、つまり、ノードのアクション リストで時限イベント アクションに到達した場合に、一定間隔でトリガされるアクションを開始するには、このオプションを選択します。右側のドロップダウン リストで時間単位を選択し、左側のフィールドに数値を入力する。たとえば、[分] を選択して 4 を入力すると、ワークフロー内で [時限イベント] アクションに達してから 4 分後にサブアクションが実行される。
   - 式－絶対日時または相対日時を指定するにはこのオプションを選択します。日時はフィールドに式を入力することにより定義します。
3. [式] を選択した場合、以下のように Java Date オブジェクトを戻す日付関数をフィールドに入力します。
   - 絶対日時を指定するには `StringToDate()` を使用します。詳細については、8-18 ページの「StringToDate()」 を参照してください。
   - 定数に対する相対的な値または変数ベースの日時を指定するには `DateAdd()` 関数を使用します。詳細については、8-20 ページの「DateAdd()」 を参照してください。
4. （省略可能）DateAdd() 関数のパラメータとしてビジネス カレンダーを指定しない場合、以下のようにビジネス カレンダーを指定する方法もあります。
   - オーガニゼーション用－ワークフロー テンプレート定義インスタンスが実行されるオーガニゼーションに割り当てられたビジネス カレンダーを使用します。
   - 割り当て対象用－時限イベントがユーザに対して手動タスクを割り当てるタスク ノードで定義されている場合に限り、トリガされるイベントの入っているタスクを割り当てられたユーザまたはロールに属するカレンダーを使用します。

5. （省略可能）トリガされたアクションを反復するため、[実行スケジュール] タブの [再スケジューリング間隔] フィールドに、トリガされたアクションの再実行インターバルを表わす値を入力し、ドロップダウン リストから時間の単位を選択します。たとえば、「4 分」を選択すると、指定された実行が停止するまで、最初のトリガ後、4 分ごとにサブアクションが実行されます。

   サーバが指定した開始時刻に実行されていない場合、指定した時刻に開始されるイベントを回復する（つまりサーバが再開始されるまで遅らせる）またはスキップするかを指定するには [回復可能] チェックボックスを設定します。

図 6-18 [時限イベント] : [実行スケジュール] タブ



6. 実行スケジュールを指定した場合、[実行停止条件] オプションを選択し、トリガを停止する時期を指定します。
   - ワークフロー完了－ワークフロー全体の完了時にトリガを停止する場合に選択します。
   - タスク完了－このアクションが指定されているタスクに完了マークが付けられたら、トリガを停止する場合に選択します。このオプションは、[時限イベント] アクションをタスク ノード内で指定していて、そのタスク ノードに対してノードの外部およびアクションの外部から完了マークが付けられている場合にのみ選択すること。詳細については、6-34 ページの「実行スケジュールについて」を参照してください。

7. [トリガ時のアクション] タブで [追加] をクリックすると、[アクションを追加] ダイアログ ボックスが表示され、イベントがトリガされた際に実行するサブアクションを選択および定義します。サブアクションが同期的に実行されるようにする場合、またはこのアクションを使用してワークフローで時間遅延を作成する場合は、このアクション リストの最後にタスクに完了マークを付けるアクションを追加します。[タスクに完了マークを付ける] ダイアログ ボックスで、時限イベント アクションの入ったタスク ノードを選択します。
8. [OK] をクリックすると [時限イベント] アクションが追加されます。

サブワークフローの使用法

一部のアクションではサブワークフローを指定できます。これは設計領域に表示されません。そのようなサブワークフローについて以下で説明します。

注意 : イベント トリガされたワークフローを開始する XML メッセージを内部 JMS キューに対してポストすることにより、サブワークフローを呼び出すこともできます。詳細については、6-88 ページの「JMS トピックまたはキューへの XML メッセージのポスト」を参照してください。

- ワークフローを開始ー Called Start で定義されているまったく別のワークフローを呼び出します。6-39 ページの「サブワークフローを呼び出す」で説明します。
- 条件を評価ー条件の真偽に従って実行される代替サブアクションを指定することにより、単一ノード内に決定分岐を埋め込みます。6-44 ページの「条件付きシーケンスを埋め込む」で説明します。
- 時限イベントー実行する一連のサブアクションを指定します。任意で、正確なタイム スケジュールに従って再実行させることもできます。6-34 ページの「時限シーケンスを埋め込む」で説明します。
- 例外ハンドラの呼び出しー例外が発生したかどうかにかかわらず、ワークフロー内の特定のポイントで例外ハンドラ内に定義された一連のサブアクションを呼び出します。このアクションの詳細については、第 9 章「ワークフロー例外の処理」を参照してください。

サブワークフローを呼び出す

大きく複雑なワークフローを複数の部分に分割するために、プロセスを複数のワークフローで構成したい場合や、特定の条件に従ってのみ呼び出されるプロセスのためにサブワークフローを使用する場合があります。

ワークフローを開始アクションを使用することにより、現在のワークフローから別のワークフローを開始できます。開始されるワークフローは*サブワークフロー*と呼ばれます。*呼び出される*ワークフローまたは*子*ワークフローと呼ばれることもあります。サブワークフローを開始する側のワークフローは、*呼び出し側のワークフロー*または*親ワークフロー*と呼びます。

ワークフローを呼び出すには、Called Start ノードを含み、アクティブとしてマークされ、現在有効な開始日および終了日を指定しているテンプレート定義が、ワークフロー テンプレートに少なくとも 1 つ入っている必要があります。そのワークフロー テンプレートの中の 1 つのワークフロー テンプレート定義のみを開始できます。これは、アクティブなワークフロー テンプレート定義の中で最も有効的な定義だからです。詳細については、5-7 ページの「テンプレート定義に関する作業」を参照してください。

パラメータを引き渡す

サブワークフローに*入力*パラメータとして定義された変数がある場合、その変数には親ワークフローから受け取った値が入れられます。したがって、ワークフローを開始アクションのプロパティで、サブワークフローに入力パラメータとして渡す値を指定する必要があります。通常、値は親ワークフローに定義された、対応する変数から取られます。

同様に、サブワークフローに出力パラメータとして定義された変数がある場合、サブワークフローの取得または計算した値が親ワークフローに戻されます。この場合も、ワークフローを開始アクションのプロパティで、結果の値を受け取る親ワークフロー変数を指定する必要があります。

サブワークフローを非同期的または同期的に実行する

ワークフローを開始アクションは、同期モードおよび非同期モードのいずれでもワークフローを開始するために使用できます。同期モードでは、呼び出し側ワークフローは、サブワークフローの処理完了を待ってから次のノードに進みます。この方法でアクションを設定するには、開始ノードまたは分岐ノードでワークフ

ローを開始アクションを使用するか、またはタスク ノードにおいて、タスク ノードではなくワークフローを開始アクションのサブアクションとしてタスクに完了マークを付けます。

非同期モードでは、呼び出し側ワークフローは呼び出されたワークフローの完了を待たずに次のノードに続くため、両方のワークフローが並行実行されます。この方法でアクションを設定するには、タスク ノードでアクションを使用し、タスク ノード自体でタスクに完了マークを付けるか、または分岐、開始、イベントのような他のタイプのノードでアクションを使用します。

注意： 内部 JMS キューに XML メッセージをポストして、イベント トリガされるワークフローをトリガすることによっても、別のワークフローを非同期的または同期的に実行できます。パラメータの引き渡しなど、2 つのワークフロー間の相互作用は、XML/JMS メッセージングにより行われ、実行の制御は、親ワークフローのイベント ノードを使用して、サブワークフローから戻される応答を受け取ることにより行われます。実際、クラスタ化環境で WebLogic Integration を実行している場合、この方法の方が負荷バランスをよく制御できます。詳細については、6-88 ページの「JMS トピックまたはキューへの XML メッセージのポスト」を参照してください。

親ワークフローがサブワークフローの完了を待ってから先に進むかどうかにかかわらず、呼び出されたワークフローの完了時に実行する任意のサブアクションのセットを定義しておくこともできます。

サブワークフローをトラッキングする

呼び出し側ワークフローで定義された変数に、呼び出されたワークフロー インスタンスの ID を割り当てることができます。インスタンス ID は文字列として戻されるため、文字列型として定義した変数を作成してください（その手順については、5-28 ページの「変数に関する作業」参照）。呼び出し側ワークフローは、`WorkflowVariable()` を使用して式の中で参照変数を使用することにより、呼び出されたサブワークフローから変数を取り出すことができます。`WorkflowVariable()` 関数でのインスタンス ID の使い方の詳細については、8-12 ページの「実行時のワークフロー データを収集する」を参照してください。

図 6-19 [ワークフローを開始] ダイアログ ボックス

サブワークフローを呼び出す手順は、以下のとおりです。

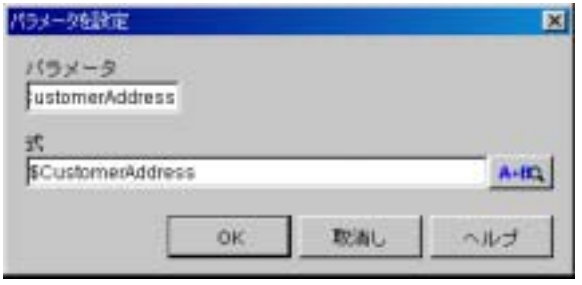
1. [アクションを追加] ダイアログ ボックスの [ワークフロー アクション] フォルダから [ワークフローを開始] を選択し、[OK] をクリックすると、[ワークフローを開始] ダイアログ ボックスが表示されます。
2. [開始するワークフロー] フィールドで、アクション実行時に開始するワークフロー テンプレートを選択します。このフィールドには、現在のオーガニゼーションに関連付けられたワークフロー テンプレートがすべて表示される。これらのワークフロー テンプレートは、*呼び出された側のワークフロー* の [開始]、現在有効な [開始] および [終了] の各日付を伴うテンプレート

定義を少なくとも 1 つ含み、[アクティブ] とマークされている。ワークフローを選択すると、ダイアログ ボックス下部の [パラメータ] タブと [結果] タブに、そのワークフロー内で定義された入力変数と出力変数が表示される。

3. [オーガニゼーション内での開始] フィールドで、以下のいずれかのオプションを選択します。
   - 現在のオーガニゼーション－現在のオーガニゼーション（フォルダ ツリー上の [オーガニゼーション] ドロップダウン リストから選択されたオーガニゼーション）で選択します。
   - 式－実行時に決定されるオーガニゼーションのためにワークフローの開始を選択します。オーガニゼーションは、ワークフロー式により、引用符で囲んだオーガニゼーションを指定する文字列の入力により、または実行時に評価され、オーガニゼーションの名前になる式の入力により決定されます。たとえば、前に別のノードから受け取った着信 XML メッセージからオーガニゼーション情報を格納している変数とすることができます。
4. （省略可能）[参照に使用する変数] フィールドで、現在のワークフローのために利用できる変数のリストから変数を選択することもできます。呼び出されたワークフローがインスタンス化されると、そのインスタンス ID が自動的に選択された変数に割り当てられます。

   注意 : この変数は、文字型でなければなりません。
5. [パラメータ] タブの [パラメータ] リストからパラメータを選択し、[更新] をクリックすると、[パラメータを設定] ダイアログ ボックスが表示されます。これを使用して、呼び出されたワークフローに渡す値を指定します。

図 6-20 [パラメータを設定] ダイアログ ボックス



6. [式] フィールドに、式としてサブワークフローに渡す値を入力します。この値は、一般的に親ワークフロー内の対応する変数の値からなる。
7. [OK] をクリックします。パラメータとその値が、[ワークフローを開始] ダイアログ ボックスの [パラメータ] タブのリストに表示されます。
8. リストのすべてのパラメータについて、手順 5 から 7 を繰り返します。
9. [結果] タブの [結果] リストから結果を選択し、[更新] をクリックすると、[結果から変数を設定] ダイアログ ボックスが表示されます。これを使用して、親ワークフローでサブワークフローにより戻された値を格納する変数を指定します。

図 6-21 [結果から変数を設定] ダイアログ ボックス

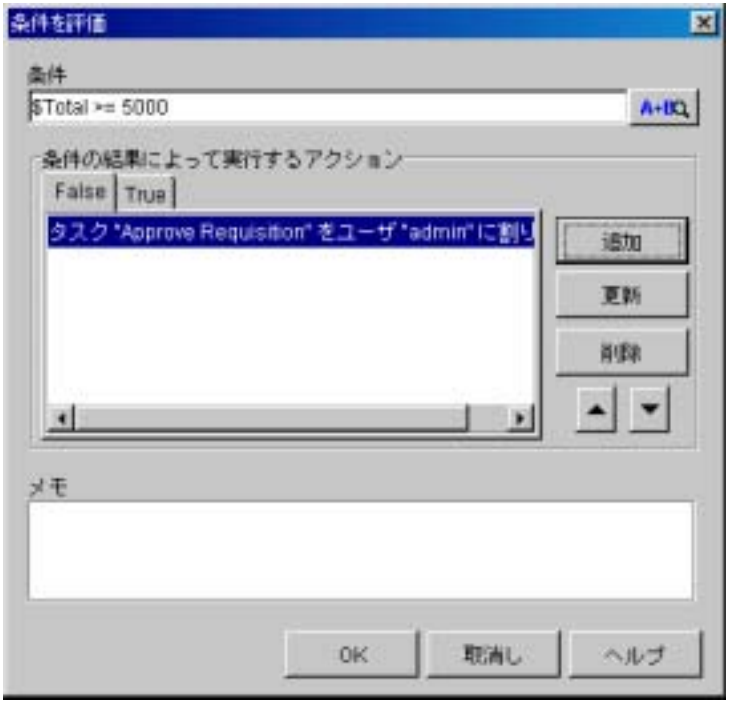
10. [変数] ドロップダウン リストから、サブワークフロー内で定義した対応する出力変数によって返される値を格納する変数を選択する。
11. [OK] をクリックします。結果とその値が、[ワークフローを開始] ダイアログ ボックスの [結果] タブのリストに表示されます。
12. リストのすべての結果について、手順 8 から 10 を繰り返します。
13. (省略可能) [アクション] タブを選択し、[追加] をクリックすると、[アクションを追加] ダイアログ ボックスが表示され、サブワークフロー完了時に実行するサブアクションを選択および定義できます。親ワークフロー内で実行される他のどのノードよりも先にサブワークフローを完了させたい場合は、このタブに [タスクに完了マークを付ける] アクションを必ず追加すること。[タスクに完了マークを付ける] ダイアログ ボックスで、ワークフローを開始アクションを指定したタスク ノードを選択します。
14. [OK] をクリックすると [ワークフローを開始] アクションが追加されます。

条件付きシーケンスを埋め込む

条件を評価アクションは、分岐ノードと同じ方法で機能します。つまり、実行時に条件式を評価し、その結果に応じてサブアクションの代替シーケンスを実行します。通常、このタイプのフローを実行するには、分岐ノードを使用してください。ただし、そのワークフローの独立したノードとしてではなく、別のノード内にアクションの条件付きセットを指定することにより行われる方法で、メインワークフロー内に条件付きサブワークフローを埋め込む場合もあります。これは、例外ハンドラの中など、条件を指定する他の方法がない場合に必要となります。

条件が True と評価された場合に実行する一連のアクション条件が False と評価された場合に実行する一連のアクション、またはその両方を定義します。

図 6-22 [条件を評価] ダイアログ ボックス



条件を評価する手順は、以下のとおりです。

1. [アクションを追加] ダイアログ ボックスの [その他のアクション] フォルダから [条件を評価] を選択し、[OK] をクリックすると、[条件を評価] ダイアログ ボックスが表示されます。
2. [条件] フィールドに、どのサブアクションのセットを実行するかを決定するために実行時に評価する有効なワークフロー条件を入力します。式の作成方法の詳細については、第 8 章「ワークフロー式の使用法」を参照してください。
3. [False] タブまたは [True] タブで、[追加] をクリックすると [アクションを追加] ダイアログ ボックスが表示され、次のボタンを使用して、それぞれ条件が False または True と評価された場合に実行するサブアクションを選択および定義します。
4. [OK] をクリックすると [条件を評価] アクションが追加されます。

実行時状態のモニタリング

ワークフロー モニタリングのために 2 つのアクションを使用できます。

- 監査エントリを作成－ワークフロー インスタンス実行中に記録される監査ログに対して、ユーザが指定した内容を持つエントリを追加します。6-45 ページの「監査エントリを作成する」で説明します。
- ワークフロー コメントを設定－ Studio で実行時にワークフロー インスタンスをモニタリングしている管理者に説明や指示のコメントを表示します。6-47 ページの「ワークフロー コメントを設定する」で説明します。

監査エントリを作成する

WebLogic Integration には、主要なすべてのユーザとのやりとりおよび変更を、JMS トピック、およびサーバ上のアクティブな WebLogic Integration ドメインのログ ディレクトリにある `myserver.log` に対して記録するデフォルト監査機能があります。

ワークフローのどのノードでも、監査エントリを作成アクションを使用して、ワークフローの実行中に監査ログに記録する実行時ワークフロー情報を追加定義できます。各エントリの日時が記録されるため、ユーザは実行時に記録したいデータを提供する式を定義します。それには、変数、関数、定数を含めることができます。たとえば、ロールに対して割り当てられたタスクで CurrentUser() 関数を使用することにより、それを実行したユーザを記録できます（実行時ワークフロー情報を戻す関数については、8-12 ページの「実行時のワークフロー データを収集する」を参照）。

注意： このアクションを有効にするには、ワークフローの [テンプレート定義のプロパティ] ダイアログ ボックスで監査をオンにする必要があります。詳細については、5-8 ページの「ワークフロー テンプレート定義を作成する」を参照してください。

図 6-23 [監査エントリを作成] ダイアログ ボックス

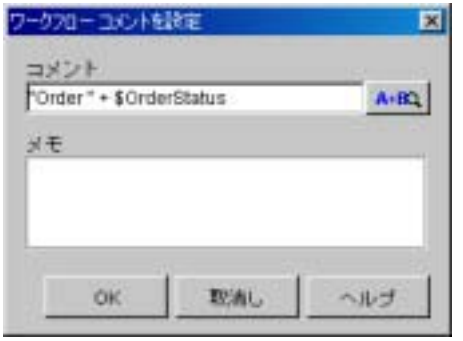
監査エントリを作成する手順は、以下のとおりです。

1. [アクションを追加] ダイアログ ボックスの [その他のアクション] フォルダで、[監査エントリを作成] を選択し [OK] をクリックして、[監査エントリを作成] ダイアログ ボックスを表示します。
2. [監査エントリ式] フィールドに、監査エントリ情報を生成するために実行時に評価する式を入力します。式の作成方法の詳細については、第 8 章「ワークフロー式の使用法」を参照してください。
3. [OK] をクリックするとアクションが追加されます。

ワークフロー コメントを設定する

ワークフロー コメントを設定アクションを使用すると、ワークフロー インスタンスにコメントを指定できます。コメントは、アクションが実行された際に、Studio の [ワークフロー インスタンス] ダイアログ ボックスの [コメント] カラムに表示されます（詳細については、10-2 ページの「ワークフロー インスタンスの操作」を参照）。一般的に、このコメントは情報の提供を目的とするもので、その時点でのワークフローのステータスを示します。

図 6-24 [ワークフロー コメントを設定] ダイアログ ボックス



ワークフローにコメントを設定する手順は、次のとおりです。

1. [アクションを追加] ダイアログ ボックスの [ワークフロー アクション] フォルダから [ワークフロー コメントを設定] を選択し、[OK] をクリックすると、[ワークフロー コメントを設定] ダイアログ ボックスが表示されます。
2. [コメント] フィールドに、コメントを生成するために実行時に評価する式を入力します。一般的に、この式は文字列と変数からなる。式とその構文の詳細については、第 8 章「ワークフロー式の使用法」を参照してください。

   注意 : ワークフローのコメントは最大 254 文字までです。コメントの長さは、式の長さが異なる場合があるため実行時まで決定されません。254 文字を超過するコメントは、実行時に警告なしに短縮されます。
3. [OK] をクリックすると [ワークフロー コメントを設定] アクションが追加されます。

手動タスクの設定

手動タスクを割り当てるため、または WebLogic Integration Worklist あるいはカスタム クライアント アプリケーションのユーザと対話するために、以下のタスク アクションを使用できます。

- [ユーザにタスクを割り当て] ―特定のユーザに対して、または負荷バランス調整に基づくユーザに対して、手動タスクを割り当てます。6-49 ページの「タスクをユーザに割り当てる」で説明します。
- [ロールにタスクを割り当て] ―ロールに対して手動タスクを割り当てます。6-51 ページの「ロールに対するタスクを割り当てる」で説明します。
- [ルーティング テーブルを使用してタスクを割り当て] ―指定した条件に従って、別のユーザまたはロールに同じタスクを割り当てます。6-53 ページの「ルーティング テーブルによりタスクを割り当てる」で説明します。
- [タスク期日を設定] ―割り当てられたユーザまたはロールがタスクを実行すべき期日を指定します。6-56 ページの「タスク期日を設定する」で説明します。
- [タスク コメントを設定] ―タスクを実行するユーザに対して実行時にコメントまたは指示を表示します。6-58 ページの「タスクのコメントを設定する」で説明します。
- [タスク優先度を設定] ―ユーザに対して実行時にタスクの優先順位レベルを表示します。6-60 ページの「タスク優先順位を設定する」で説明します。
- [タスクの割り当てを解除] ―タスクの割り当てを解除します。6-61 ページの「タスクの割り当てを解除する」で説明します。
- [XML をクライアントに送信] ― Worklist またはカスタム クライアント アプリケーションに対して XML メッセージを送信し、ユーザにフォームまたはプロンプトを表示する、あるいはクライアント上のプログラムまたはカスタム拡張を呼び出します。6-62 ページの「クライアント アプリケーションに対し XML メッセージを送信する」で説明します。

タスク アクションの配置に関するガイドライン

手動タスクを割り当てる場合、アクション配置について以下のガイドラインに従ってください。

- 同じタスクに関係するすべてのアクションは、同じタスク ノードに配置する必要があります。
- ユーザがタスクを実行する*前*に、タスクを閲覧するユーザに対して、必ずコメントまたは期日が表示されるようにするには、ユーザにタスクを割り当て / ロールにタスクを割り当て / ルーティング テーブルを使用してタスクを割り当てアクションの前に、タスク コメントを設定アクションとタスク期日を設定アクションを置きます。
- [タスクのプロパティ] ダイアログ ボックスの [実行時] タブでタスクに完了マークを付けます。
- クライアント アプリケーションに XML メッセージを送信し、プロンプト メッセージの表示、またはクライアント上の他のソフトウェア コンポーネントの呼び出しを行うには、まず、ユーザまたはロールにタスクを割り当て、その後、[タスクのプロパティ] ダイアログ ボックスの [実行時] タブに XML をクライアントに送信アクションを置きます。

タスクをユーザに割り当てる

個々のユーザに指定されたタスクを割り当てるにはユーザにタスクを割り当てアクションを使用します。ユーザにタスクを割り当てると、Worklist またはカスタム クライアント アプリケーションからタスクをビューおよび実行するユーザに通知が送信されます。Worklist アプリケーションの詳細については、『*WebLogic Integration Worklist ユーザーズ ガイド*』を参照してください。

タスクを割り当てる特定のユーザの指定、あるいはロール メンバーの指定ができます。その場合、システムにより、実行時に特定のロールに属するすべてのユーザの中で保留中タスク数が最小のユーザが決定されます。その後、そのユーザにタスクが割り当てられます。ユーザまたはロールは、定数として指定したり、実行時に提供されるワークフロー式から取得したりできます。

図 6-25 [ユーザにタスクを割り当て] ダイアログ ボックス

ユーザにタスクを割り当てる手順は、次のとおりです。

1. [アクションを追加] ダイアログ ボックスの [タスク アクション] フォルダから [ユーザにタスクを割り当て] を選択し、[OK] をクリックすると、[ユーザにタスクを割り当て] ダイアログ ボックスが表示されます。
2. [割り当てるタスク] フィールドで割り当てるタスクを選択します。一度に選択できるタスクは、1 つのみ。
3. ユーザを指定するには以下のオプションから選択します。
   - ユーザ名を指定するには、[ユーザ] ドロップダウン リストから名前を選択します。現在のオーガニゼーションと関連付けられているユーザのみが、このリストに表示されます。

- ロール メンバーにタスクを割り当てるには [ロール メンバーに割り当て] を選択し、表示された [ロール内のユーザ] ドロップダウン リストからロール名を選択します。現在のオーガニゼーションに対して定義されているロールのみが、このリストに表示されます。実行時に、このロールに属するユーザのうち、割り当てられているタスク数の最も少ないユーザにタスクが割り当てられます。
- 実行時に決定される ID を持つユーザを指定するには [ワークフロー式を使用] を選択し、[ユーザ] フィールドに実行時に適切なユーザを戻す式を入力します。

4. [OK] をクリックすると [ユーザにタスクを割り当て] アクションが追加されます。

ロールに対するタスクを割り当てる

特定のユーザに対するタスクの割り当てを希望しない場合は [ロールにタスクを割り当て] アクションが使用できます。ロールに対してタスクを割り当てることにより、そのロールに属するどのユーザでもタスクをビューおよび実行できますが、Worklist でのタスクの明示的通知はありません。

ロールに対してタスクを割り当てるには、ロール名を指定するか、実行時にロールを決定する式を指定します。

図 6-26 [ロールにタスクを割り当て] ダイアログ ボックス

ロールにタスクを割り当てる手順は、次のとおりです。

1. [アクションを追加] ダイアログ ボックスの [タスク アクション] フォルダから [ロールにタスクを割り当て] を選択し、[OK] をクリックすると、[ロールにタスクを割り当て] ダイアログ ボックスが表示されます。
2. [割り当てるタスク] フィールドで割り当てるタスクを選択します。一度に選択できるタスクは、1 つのみ。
3. ロールを指定するには以下のオプションから選択します。
   - ロール名を指定するには、[ロール] ドロップダウン リストから名前を選択します。現在のオーガニゼーションで定義されているロールのみが、このリストに表示されます。

- 実行時に決定される ID を持つロールを指定するには [ワークフロー式を使用] を選択し、[ロール] フィールドに実行時に適切なロールを戻す式を入力します。関数と式についての詳細については、第 8 章「ワークフロー式の使用法」を参照してください。

4. [OK] をクリックすると [ロールにタスクを割り当て] アクションが追加されます。

ルーティング テーブルによりタスクを割り当てる

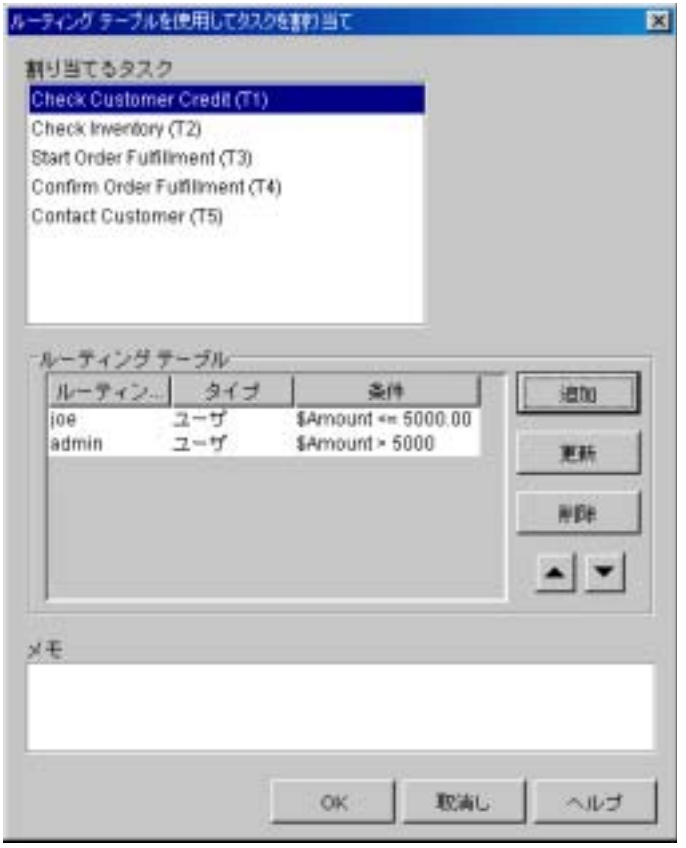
条件のセットに応じて、別のユーザまたはロールにタスクを割り当てることができます。ルーティング テーブルは、任意の数のルーティング条件のシーケンスからなっています。ルーティング条件には、タスクを割り当てるユーザまたはロールの候補、および割り当てが行われる条件を指定します。実行時にこのアクションが実行されると、True という結果が得られるまで条件が 1 つずつ評価されます。True という結果が得られると、該当するユーザまたはロールにタスクが割り当てられて、後続の条件は無視されます。

注意 : どのルーティング条件も満たされない場合は、実行時例外が発生します。

特定の条件によって行う代わりに、一定期間だけ特定のユーザまたはロールにすべてのタスクをルーティングする場合は、オーガニゼーションのためのルーティング仕様を作成します。詳細については、3-30 ページの「タスク ルーティングの管理」を参照してください。

注意 : ルーティング機能では、ユーザに割り当てられたタスクのみ再ルーティングされ、ロールに割り当てられたタスクは再ルーティングされません。再ルーティングされたタスクは、別のユーザ、ロールのユーザ、またはロールに送ることができます。ロールに割り当てられたタスクの再ルーティングについては 3-25 ページの「ロールに対するマッピングを変更する」を参照してください。

図 6-27 [ルーティング テーブルを使用してタスクを割り当て] ダイアログ ボックス



ルーティング テーブルを使ってタスクを割り当てる手順は、次のとおりです。

1. [アクションを追加] ダイアログ ボックスの [タスク アクション] フォルダから [ルーティング テーブルを使用してタスクを割り当て] を選択し、[OK] をクリックすると、[ルーティング テーブルを使用してタスクを割り当て] ダイアログ ボックスが表示されます。
2. [割り当てるタスク] フィールドで割り当てるタスクを選択します。一度に選択できるタスクは、1 つのみ。

3. [ルーティング テーブル] フィールドの横の [追加] をクリックすると、[ルーティング条件を定義] ダイアログ ボックスが表示されます。

図 6-28 [ルーティング条件を定義] ダイアログ ボックス

4. [条件] フィールドに、論理結果（True または False）を生成する有効なワークフロー条件式を入力します。式の作成方法については、第 8 章「ワークフロー式の使用法」を参照してください。
5. [割り当て先] フィールドで、[ユーザ]、[ロール内のユーザ]、または [ロール] のいずれかのオプションを選択します。
6. ドロップダウン リストで希望のユーザまたはロール名を選択します。
7. [OK] をクリックするとルーティング テーブルにルーティング条件が追加されます。
8. さらに必要な数のルーティング条件を追加します。ルーティングを削除するには [削除] ボタン、ルーティングを編集するには [更新] ボタン、テーブル内でルーティングの順序を変更するには矢印キーを使用してください。
9. [OK] をクリックすると [ルーティング テーブルを使用してタスクを割り当て] アクションが追加されます。

タスク期日を設定する

タスク期日を設定アクションを使用してタスクの実行期日を設定できます。期日は、タスクが割り当てられた Worklist またはカスタム クライアント ユーザに対して表示されます。期日は、分、時、日、週、または月の単位で表わすことができます。あるいは、実行時に日時が決定される式として表わすこともできます。

非手動タスクに対してこのアクションを使用することにより、ユーザにタスクを割り当てアクションを使用せずに、特定の日付以降に並行実行させるアクションを指定したり、ワークフローに時間遅延を導入することもできます。

期日超過アクションを非同期的および同期的に実行する

タスクが期日まで実行されない場合、期日後に実行すべきサブアクションを、同期モードまたは非同期モードで指定できます。非同期モードではサブアクションは並行実行され、その間にワークフローは次のノードに進みます。この方法でアクションを設定するには、タスク ノード自体で、アクションの入っているタスクに完了マークを付けます。これは、通常、アクションを手動でタスクに割り当てるために使用される方法です。

同期モードでは、このアクションを使用してワークフローで時間遅延を作成できます。ワークフローは、期日に達するまで待ってから期日超過アクションを実行し、その後、先に進みます。アクションをこの方法で設定するには、ユーザに割り当てられているタスクまたは割り当てられていないタスクまたは割り当てられていないタスクにアクションを置き、[タスク期日を設定] ダイアログ ボックスの [期日に実行するアクション] タブにタスクに完了マークを付けるアクションを追加することにより、アクションの入っているタスクに完了マークを付けます。

図 6-29 [タスク期日を設定] ダイアログ ボックス

タスクの期日を設定する手順は、次のとおりです。

1. [アクションを追加] ダイアログ ボックスの [タスク アクション] フォルダから [タスク期日を設定] を選択し、[OK] をクリックすると、[タスク期日を設定] ダイアログ ボックスが表示されます。
2. [期日を設定するタスク] フィールドで、アクション実行時に期日設定を持っている必要のあるタスクを選択します。一度に選択できるタスクは、1 つのみ。
3. [期日] タブの [式に設定] フィールドに、絶対日時または相対日時を指定する式を入力します。式は Java Date オブジェクトを戻さなければならないため、以下の関数を使用する必要があります。

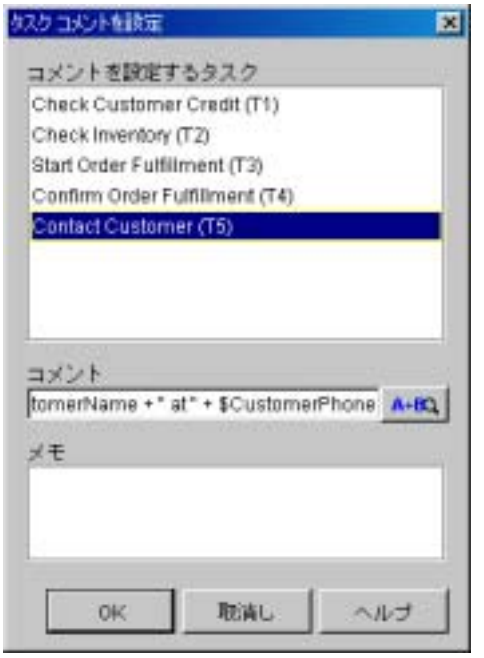
- 絶対日時を指定するには `StringToDate()` を使用します。詳細については、8-18 ページの「StringToDate()」を参照してください。
- 定数に対する相対的な値または変数ベースの日時を指定するには DateAdd() 関数を使用します。詳細については、8-20 ページの「DateAdd()」を参照してください。

4. （省略可能）DateAdd() 関数のパラメータとしてビジネス カレンダーを指定しない場合、以下のようにビジネス カレンダーを指定する方法もあります。
   - オーガニゼーション用－ワークフロー テンプレート定義インスタンスが実行されるオーガニゼーションに割り当てられたビジネス カレンダーを使用します。
   - 割り当て対象用－期日がユーザに対して手動タスクを割り当てるタスク ノードで定義されている場合に限り、タスク期日を設定アクションの入っているタスクを割り当てられたユーザまたはロールに属するカレンダーを使用します。
5. （省略可能）[期日に実行するアクション] タブを選択し、[追加] をクリックすると、[アクションを追加] ダイアログ ボックスが表示され、期日に達した際に実行するサブアクションを選択および定義できます。期日に達するまで先に進まずにワークフローを待たせたい場合、タスク ノードでこのアクションを使用し、このタブでタスクに完了マークを付けるアクションを必ず追加してください。[タスクに完了マークを付ける] ダイアログ ボックスで、タスク期日を設定アクションを指定したタスク ノードを選択してください。
6. [OK] をクリックすると [タスク期日を設定] アクションが追加されます。

タスクのコメントを設定する

タスク コメントを設定アクションを使用して、タスク インスタンスに関するコメントを設定できます。コメントは、アクションが実行される際に、Worklist または Studio でタスクをビューするユーザに対してテキスト メッセージとして表示されます。通常、このテキストは、ユーザに実行を求める手動での作業に関する情報または指示を提供します。テキスト メッセージは、Worklist アプリケーションのタスク リスト、または Studio の [ワークフロー インスタンス] ダイアログ ボックスまたは [Worklist] ダイアログ ボックスで、タスクの横の [コメント] カラムに表示されます（詳細については、第 10 章「ワークフローのモニタリング」を参照）。

図 6-30 [タスク コメントを設定] ダイアログ ボックス



タスクにコメントを設定する手順は、次のとおりです。

1. [アクションを追加] ダイアログ ボックスの [タスク アクション] フォルダから [タスク コメントを設定] を選択し、[] をクリックすると、[タスク コメントを設定] ダイアログ ボックスが表示されます。
2. [コメントを設定するタスク] で対象のタスクを選択します。一度に選択できるタスクは、1 つのみ。
3. [コメント] フィールドに、コメントを生成するために実行時に評価する式を入力します。一般的に、この式は文字列と変数からなる。式とその構文の詳細については、第 8 章「ワークフロー式の使用法」を参照してください。

   注意 : タスクのコメントは最大 254 文字までです。コメントの長さは、式の長さが異なる場合があるため実行時まで決定されません。254 文字を超過するコメントは、実行時に警告なしに短縮されます。
4. [OK] をクリックすると [タスク コメントを設定] アクションが追加されます。

タスク優先順位を設定する

タスク優先度を設定アクションを使用して、タスク優先順位を低、中、または高に設定できます。この優先度は、実行時におけるノードまたはノードのアクションの実行方法には影響を与えません。単に、Worklist に合わせてタスクを実行やソートを行うことができる Worklist ユーザに対して表示されるだけです。

[タスクのプロパティ] ダイアログ ボックスで現在のタスクの優先順位を設定できるため、ワークフローのどこかでタスクの優先順位を指定する条件の結果としてこのアクションを使用できます。

図 6-31 [タスク優先度を設定] ダイアログ ボックス

タスクの優先度を設定する手順は、次のとおりです。

1. [アクションを追加] ダイアログ ボックスの [タスク アクション] フォルダから [タスク優先度を設定] を選択し、[OK] をクリックすると、[タスク優先度を設定] ダイアログ ボックスが表示されます。

2. [優先度を設定するタスク] で対象のタスクを選択します。一度に選択できるタスクは、1 つのみ。
3. [優先度] ドロップダウン リストから、[低]、[中]、または [高] を選択します。
4. [OK] をクリックすると [タスク優先度を設定] アクションが追加されます。

タスクの割り当てを解除する

タスクの割り当てを解除アクションを使用して、現在のタスク割り当てを削除できます。タスクはユーザまたはロールに対する割り当てを持たなくなります。条件の結果としてタスクの割り当ての解除もできます。

図 6-32 [タスクの割り当てを解除] ダイアログ ボックス

タスクの割り当てを解除する手順は、次のとおりです。

1. [アクションを追加] ダイアログ ボックスの [タスク アクション] フォルダから [タスクの割り当てを解除] を選択し、[OK] をクリックすると、[タスクの割り当てを解除] ダイアログ ボックスが表示されます。
2. [割り当てを解除するタスク] フィールドで割り当て解除するタスクを選択します。一度に選択できるタスクは、1 つのみ。
3. [OK] をクリックすると [タスクの割り当てを解除] アクションが追加されます。

クライアント アプリケーションに対し XML メッセージを送信する

ユーザにタスクを割り当てた後、XML をクライアントに送信アクションを使用して XML ドキュメントをクライアントに送信することにより、ワークフローと、Worklist またはカスタム クライアント アプリケーションとの間の通信を行うことができます。この場合、クライアント アプリケーションが XML ドキュメントを識別し、該当するアクションを実行し、ワークフローに応答して XML ドキュメントを返すようにプログラムされている必要があります。Worklist アプリケーションの場合、XML メッセージを送信して、メッセージ プロンプトやフォームをユーザに表示したり、クライアント システム上でカスタム コンポーネントや実行可能プログラムを呼び出したりできます。カスタム クライアント アプリケーションの開発方法については、『*BPM クライアント アプリケーション プログラミング ガイド*』 を参照してください。

XML をクライアントに送信アクションは、XML の送信先のクライアント マシンまたはアプリケーションを実際に指定するわけではありません。したがって、まずタスクをユーザまたはロールに割り当てておく必要があります。すると、XML メッセージがタスクを実行するクライアントに送信されます。詳細については、6-48 ページの 「手動タスクの設定」 を参照してください。

メッセージを非同期的または同期的に送信する

XML メッセージは、クライアント アプリケーションに対して、非同期的または同期的に送信できます。同期モードでは、ワークフローはクライアントからの応答を待ってから次のノードに進みます。この方法でアクションを設定するには、

タスク ノードではなく、[XML をクライアントに送信] ダイアログ ボックスの [コールバック アクション] タブ上のサブアクションとして、アクションの入っているタスクにマークを付けます。

非同期モードでは、ワークフローはクライアントからの応答を待たずに次のノードに続くため、クライアント上のオペレーションは並行実行されます。この方法でアクションを設定するには、タスク ノード自体で、タスクに完了マークを付けます。

データを抽出する

XML メッセージを返信することにより、アプリケーションがワークフローに応答する場合、応答ドキュメントにより戻されるデータを格納するための変数を作成する必要があります（手順については 5-28 ページの「変数に関する作業」を参照）。また、応答 XML ドキュメントからデータ値を取り出すために、XPath 式（またはドット（）表記）を使用して変数を初期化しておく必要があります。そのメッセージの JMS プロパティがアプリケーションにより使用される場合、`EventAttribute()` 関数を使用してこのデータを検索することもできます（詳細については、8-6 ページの「実行時のイベント データを抽出する」を参照）。

XML をクライアントに送信アクションを定義する

以下のプロシージャが、非 Worklist アプリケーションのために生成されます。Worklist に XML メッセージを送信する方法の詳細については、6-66 ページの「Worklist アプリケーションに対する XML メッセージを送信する」を参照してください。タイプ指定された XML ドキュメントを使用する方法の詳細については、7-11 ページの「タイプ指定ドキュメントを操作する」を参照してください。

図 6-33 [XML をクライアントに送信] ダイアログ ボックス

クライアントに XML メッセージを送信する手順は、次のとおりです。

1. [アクションを追加] ダイアログ ボックスの [統合アクション] フォルダから [XML をクライアントに送信] を選択し、[OK] をクリックすると、[XML をクライアントに送信] ダイアログ ボックスが表示されます。ルート要素と `actionid` 要素を持つデフォルト XML ドキュメント構造が作成されます。

   注意 : システムにより生成される `actionid` 要素とその値は、実行時に XML をクライアントに送信アクションを識別するために使用されます。`actionid` 要素は、ルート要素の最初の子要素でなければなりま

せん。actionid 要素の移動、削除、編集は行ってはなりません。また、この要素に下位要素を追加しないでください。

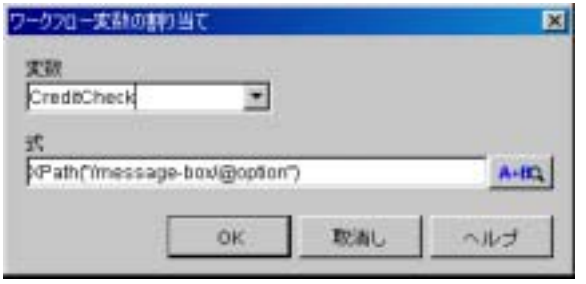
2. XML ドキュメント構造を指定するには以下のいずれかを行います。
   - 新たな自由形式ドキュメントを作成するには、[子を追加] ボタンをクリックし、ノードを追加することにより、ドキュメントの作成を開始します。
   - 既存の XML ドキュメントを指定するには、[インポート] ボタンをクリックしてドキュメントをロードします。
   - 新たなタイプ指定 XML ドキュメントを作成するには、[コンテンツ タイプを設定] ボタンをクリックし、適切なスキーマ ドキュメントをロードします。actionid 値は削除され、手順 9 でアクションを保存する際にリセットされます。

   上記のオプションの詳細な手順については、7-2 ページの「XML ドキュメントの作成と編集」を参照してください。

   ユーザ定義の XML ドキュメントは、ワークフロー テンプレート定義内に保存される。

3. クライアントからの応答を受け取る変数を指定するには、[コールバック変数] タブを選択し、[追加] をクリックすると、[ワークフロー変数の割り当て] ダイアログ ボックスが表示されます。

図 6-34 [ワークフロー変数の割り当て] ダイアログ ボックス



4. [式] フィールドに、実行時に評価される応答 XML ドキュメントからデータを抽出する式を入力します。そのために、次の操作のうちのどちらかを実行します。

- 受信 JMS ヘッダ データを取り込むために、EventAttribute() 関数を使用します詳細については、8-7 ページの「EventAttribute()」を参照してください。
- 着信 XML コンテンツを取り込むには XPath() 関数（8-8 ページの「XPath()」を参照）、または文字列を戻す XML 要素のドット（）表記（詳細については 8-11 ページの「XML 要素のドット表記」を参照）を使用します。また [式] ボタン  を使用して XPath Wizard を呼び出し、このウィザードを使ってサンプルの着信ドキュメントから XPath の式を自動的に生成できます。詳細については、8-31 ページの「XPath Wizard を使用する XPath 式の作成」を参照してください。

5. [OK] をクリックします。[XML をクライアントに送信] ダイアログ ボックスの [コールバック変数] タブのリストで変数初期化が行われます。
6. 応答ドキュメントから取り込む必要があるデータ項目すべてについて、手順 4 ～ 5 を繰り返します。
7. [コールバック アクション] タブを選択し、[追加] をクリックして [アクションを追加] ダイアログ ボックスを表示します。そこで、クライアントからの応答を受信したときに実行するサブアクションを指定します。クライアントからの応答を待ってからワークフロー内の他のノードに進みたい場合は、このタブのアクション リストの最後に必ず [タスクに完了マークを付ける] アクションを追加してください。[タスクに完了マークを付ける] ダイアログ ボックスでは、[XML をクライアントに送信] アクションを指定したタスク ノードを選択します。
8. タイプ指定ドキュメントを使用している場合、`actionid` 値の追加を要求されます。[はい] をクリックするとドキュメントが更新されます。
9. 再度 [OK] をクリックするとアクション定義が保存されます。

Worklist アプリケーションに対する XML メッセージを送信する

注意： Worklist クライアント アプリケーションは、WebLogic Integration のこのリリースでは廃止になっています。置き換えられる機能の詳細については『*BEA WebLogic Integration リリース ノート*』を参照してください。

Worklist クライアント アプリケーションは、事前定義された DTD (Document Type Definition: ドキュメント タイプ定義) ファイルの 4 つのペアに準拠する XML ドキュメントに対する応答によってアクションを実行するように設計されており、これらのファイルにより以下のことを行うことができます。

- ユーザに対するプロンプトの入ったメッセージ ボックスを表示します。
- ユーザに対する入力フィールドの入ったフォームを表示します。
- クライアント上のカスタム アドイン コンポーネントを呼び出します。
- クライアント上の実行可能プログラムを呼び出します。

事前定義された DTD ファイルの 4 つのペアは、WebLogic Integration サーバ インストレーションの次のディレクトリにあります。

WLI_HOME\docs\apidocs\com\bea\wlpi\common\doc-files

このパスでは、*WLI_HOME* は WebLogic Integration をインストールしたディレクトリを表します。多くの場合、ディレクトリは、c:\bea\weblogic700\integration となります。

各ペアは、Worklist クライアントに対する要求のための DTD ファイルと、Worklist クライアントからの応答のための DTD ファイルで構成されます。次の表に、事前定義された DTD ファイルと、要求 DTD ファイルの 1 つに準拠する XML ドキュメントを受け取った際に Worklist アプリケーションにより実行されるアクションをリストします。

表 6-2 Worklist DTD ファイル

DTD ペア		使用目的
要求	ClientMsgBoxReq.dtd	メッセージまたはクエリに対してユーザが応答するためのメッセージ ダイアログ ボックスを表示する。 応答 = ok/yes/no/cancel 詳細については、6-68 ページの「ユーザに対してメッセージ プロンプトを表示する」を参照。
応答	ClientMsgBoxResp.dtd	

表 6-2 Worklist DTD ファイル

DTD ペア		使用目的
要求	ClientSetVarsReq.dtd	ユーザが値を入力する入力フィールドの入ったプロンプト ダイアログ ボックスを表示する。 応答 = フィールド名と値のペア 詳細については、6-72 ページの「ユーザに対してフォームを表示する」を参照。
応答	ClientSetVarsResp.dtd	
要求	ClientCallPgmReq.dtd	クライアント マシンでプログラムを実行する。 応答 = プログラム出口コード 詳細については、6-74 ページの「クライアント上の実行可能プログラムを呼び出す」を参照。
応答	ClientCallPgmResp.dtd	
要求	ClientCallAddInReq.dtd	Worklist クライアント アプリケーションに対するカスタム拡張を呼び出す。 応答 = カスタム 詳細については、6-76 ページの「クライアント上でのカスタム Worklist 拡張を呼び出す」を参照。
応答	ClientCallAddInResp.dtd	

注意 : これらの DTD に頻繁にアクセスする場合、取り出しやすいようにリポジトリにインポートしておくこともできます。その手順については、4-27 ページの「リポジトリにあるエンティティの管理」を参照してください。

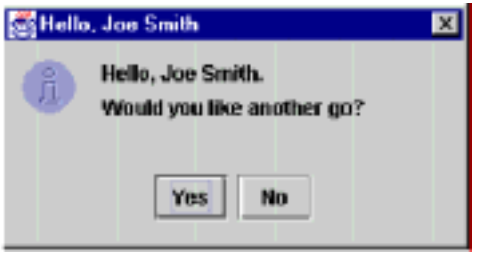
クライアントから受け取る各応答について、応答により戻される値を置くための変数を作成する必要があります。大部分は文字列型変数です。変数作成の手順については、5-28 ページの「変数に関する作業」を参照してください。

以下の節では、各 DTD ペアに必要なドキュメント構造について詳しく説明します。

ユーザに対してメッセージ プロンプトを表示する

次のサンプルのように、ClientMsgBox DTD を使用することにより、Worklist ユーザに対してメッセージ ボックスを表示できます。

図 6-35　メッセージ プロンプトのサンプル



要求 DTD では次の構造を持つドキュメントが必要です。

コード リスト 6-1　ClientMsgBoxReq XML ドキュメントの構造

```
<message-box title="text"
style="{plain|information|question|warning|error}"
options="{ok|ok_cancel|yes_no|yes_no_cancel}">
text
   <actionid>provided by default</actionid>
</message-box>
```

すべての要素と属性は必須です。これらについて以下で説明します。

表 6-3　ClientMsgBoxReq の要素と属性

要素または属性	説明	有効な値
message-box	ダイアログ ボックスのメッセージに表示するテキスト。	任意のテキスト文字列。
title	ダイアログ ボックスのタイトル バーに表示するテキスト。	任意のテキスト文字列。

表 6-3 ClientMsgBoxReq の要素と属性

要素または属性	説明	有効な値
style	ダイアログ ボックスの左上隅に表示する [Swing] アイコン。	デフォルトは plain。
	アイコンなし	plain
		information
		question
		warning
		error

表 6-3 ClientMsgBoxReq の要素と属性

要素または属性	説明	有効な値
options	ダイアログ ボックスの下部の選択ボタンと、その中のテキスト。	デフォルトは ok。
	[はい]、[いいえ]、[取消し] の 3 つのボタン	yes_no_cancel style 要素が plain または question に設定されている場合のみ有効。
	[OK] と [取消し] の 2 つのボタン	ok_cancel style 要素が error、plain、または warning に設定されている場合のみ有効。
	1 つの [OK] ボタン	ok style 要素が error、information、plain、または warning に設定されている場合のみ有効。
	[はい] と [いいえ] の 2 つのボタン	yes_no style 要素が plain または question に設定されている場合のみ有効。

応答ドキュメントは、選択されたボタンに従ってユーザの応答をメッセージ ボックスに表示します。応答 DTD には次の構造が必要です。

コード リスト 6-2 ClientMsgBoxReq XML ドキュメントの構造

```
<message-box option="{ok|yes|no|cancel}" />
```

要素と属性は必須です。それらについて、ワークフロー変数に値を戻すために必要な式と共に、以下で説明します。

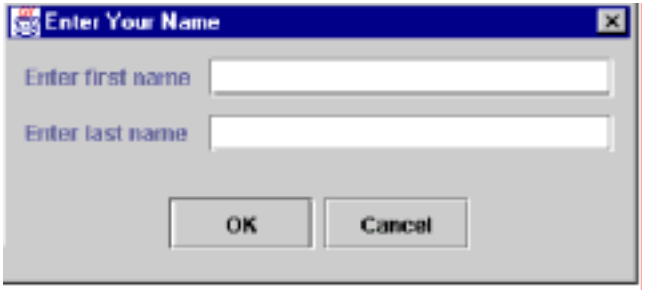
表 6-4 ClientMsgBoxResp の要素と属性

要素または属性	説明	有効な値	値を抽出するために必要な式
option	クライアント ユーザにより選択されたボタン。文字列として表わされる。	ok yes no cancel	XPath(“/message-box/@option/text()”)

ユーザに対してフォームを表示する

次のサンプルのように、ClientSetVars DTD を使用することにより Worklist ユーザに対してフォームを表示できます。

図 6-36 フォームのサンプル



要求 DTD では次の構造を持つドキュメントが必要です。

コード リスト 6-3 ClientSetVarsReq XML ドキュメントの構造

```
<set-variables title="text">
text
   <actionid>provided by default</actionid>
   <variable name="variable name" prompt="text" />
   [<variable name="variable name" prompt="text" />]
</set-variables>
```

すべての要素と属性は必須です。少なくとも 1 つの <variable> 要素が必要です。さらに、任意の数の <variable> 要素を指定できます。要素と属性について以下で説明します。

表 6-5 ClientSetVarsReq の要素と属性

要素または属性	説明	有効な値
set-variables	ダイアログ ボックスのメッセージに表示するテキスト。	任意のテキスト文字列。
title	ダイアログ ボックスのタイトル バーに表示するテキスト。	任意のテキスト文字列。
name	入力フィールドを識別する名前。	任意のテキスト文字列。
prompt	入力フィールドの前にプロンプトとして表示するテキスト。	任意のテキスト文字列。

応答ドキュメントは、ユーザの応答をそれぞれの入力フィールドに表示します。応答 DTD には次の構造が必要です。

コード リスト 6-4 ClientSetVarsResp XML ドキュメントの構造

```
<set-variables>
   <variable name="variable_name_1">response_1</variable>
   [<variable name="variable name_2">response_1</variable>]
   .
   .
   .
</set-variables>
```

すべての要素と属性は必須です。<variable> 要素の数は、要求ドキュメントで使用された数に対応させてください。要素と属性は必須です。ワークフロー変数に値を戻すために必要な式について以下で説明します。

表 6-6 ClientSetVarsResp の要素と属性

要素または属性	説明	有効な値	値を抽出するために必要な式
variable	クライアントにより与えられた応答。	任意のテキスト文字列。	XPath("/set-variables/variable[@name="*field_name*"]/text()")
name	入力フィールドを識別するために使用する名前。	要求ドキュメントで使用した名前に対応。	

クライアント上の実行可能プログラムを呼び出す

ClientCallProgram DTD を使用して、Worklist クライアント上のプログラムを呼び出すことができます。要求ドキュメントには次の構造が必要です。

コード リスト 6-5 ClientCallProgramReq XML ドキュメントの構造

```
<call-program name="name" mode="{sync|async}">
    <actionid>provided by default</actionid>
    [<parm>parameter_1</parm>]
    .
    .
    .
    [<env-var name="name">environment variable
      definition_1</env-var>]
    .
    .
    .
</call-program>
```

すべての要素と属性は必須です。ただし、<parm> 要素と <env-var> 要素は省略可能です。<parm> 要素と <env-var> 要素は、指定しないことも、あるいは複数指定することも可能です。

表 6-7 ClientCallProgramReq の要素と属性

要素または属性	説明	有効な値
name	プログラムの名前。	テキスト文字列。
mode	Worklist に関連して、実行可能プログラムを実行する際のモード。	デフォルトは、async。
	プログラムは同期的に実行される。呼び出されたプログラムが終了するまで、Worklist の続行はブロックされ、そのユーザ インタフェースにアクセスできない。	sync
	プログラムは非同期的に実行され、Worklist と並行して実行される。呼び出されたプログラムの実行中も、そのユーザ インタフェースにアクセスできる。	async
parm	プログラムに渡すパラメータ。	任意のテキスト文字列。
env-var	呼び出されるプログラムと関連付ける環境変数の定義。	テキスト文字列。
name	環境変数のシンボリック名。	テキスト文字列。

応答 DTD には次の構造が必要です。

コード リスト 6-6 ClientCallProgramResp XML ドキュメントの構造

```
<call-program exit-value="value" />
```

要素と属性は必須です。 これらについて以下で説明します。

表 6-8 ClientCallProgramResp の要素

要素または属性	説明	有効な値	値を抽出するために必要な式
exit-Value	呼び出されたプログラムの数値出口コード。オペレーティング システムにより検索される。	有効な終了コードに関する詳細については、該当するプログラム ドキュメントを参照。	XPath("/call-program/@exit-value")

クライアント上でのカスタム Worklist 拡張を呼び出す

ClientCallAddIn DTD を使用して、Worklist クライアントに対してカスタム拡張を呼び出すことができます。要求ドキュメントには次の構造が必要です。

コード リスト 6-7 ClientCallAddInReq XML ドキュメントの構造

```
<call-addin name="name" mode="{sync|async}">
    <actionid>provided by default</actionid>
    [<parm>parameter_1</parm>]
    .
    .
    .
</call-addin>
```

すべての要素と属性は必須です。ただし、<parm> 要素は省略可能です。<parm> 要素は、指定しないことも、あるいは複数指定することも可能です。

表 6-9 ClientCallAddInReq の要素と属性

要素または属性	説明	有効な値
name	com.bea.wlpi.client.worklist.WorklistAddIn インタフェースを実装するカスタム Java クラスの名前。	Java クラスの完全修飾 JNDI 名。

表 6-9 ClientCallAddInReq の要素と属性

要素または属性	説明	有効な値
mode	Worklist に関連して、プログラムを実行する際のモード。	デフォルトは、async。
	プログラムは同期的に実行される。呼び出されたプログラムが終了するまで Worklist の続行はブロックされ、そのユーザ インタフェースにアクセスできない。	sync
	プログラムは非同期的に実行され、Worklist と並行して実行される。プログラムの実行中も、そのユーザ インタフェースにアクセスできる。	async
parm	拡張に渡すパラメータ。	任意のテキスト文字列。

応答ドキュメントの指定は省略可能です。要素は省略可能で、任意の数のカスタム要素と属性を定義できます。

コード リスト 6-8 ClientCallAddInResp XML ドキュメントの構造

```
<call-addin>
       [<tag_name_1 attribute_name_1=”attribute_value”
         . . .>value</tag_name_1>]
       .
       .
       .
</call-addin>
```

これらの要素について以下で説明します。

表 6-10 ClientCallAddInResp の要素と属性

要素または属性	説明	有効な値	値を抽出するために必要な式
任意の要素名	任意の数の属性で構成できる。	カスタム定義。	XPath("/call-addin/*path*/text()")

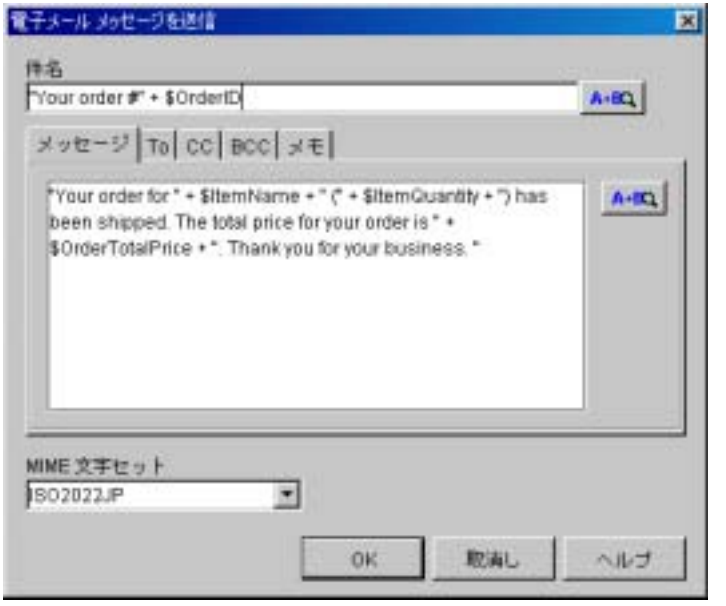
電子メール メッセージを送信

電子メール メッセージを送信アクションを使用して、WebLogic Integration システムのユーザまたは外部パーティのユーザにも電子メール メッセージを送信できます。メッセージの伝送には、インターネット標準の SMTP が使用されます。

設計時にオペレーティング システムがサポートしている文字セットであればどれでも電子メール メッセージを作成できます。また、メッセージを送信するために実行時にサーバにより使用される文字セットを指定できます。英語ロケールでは、サーバにより使用されるデフォルト文字セットは、Java Virtual Machine により使用されるデフォルト文字セットである cp1252 ですが、Java 言語によりサポートされるどの文字セットでも指定できます。文字セットのリストについては、http://java.sun.com/j2se/1.3/docs/guide/intl/encoding.doc.html を参照してください。

注意： このアクションには、WebLogic Integration サーバのインストレーション プロセス中またはその後で、電子メール サーバが正しくコンフィグレーションされている必要があります。インストレーション後にメール サーバ プロパティをコンフィグレーションする方法については、『*WebLogic Integration の起動、停止およびカスタマイズ*』の「WebLogic Integration のカスタマイズ」にある「メール セッション プロパティのカスタマイズ」を参照してください。

図 6-37 [電子メール メッセージを送信] ダイアログ ボックス



電子メール メッセージを送信する手順は、以下のとおりです。

1. [アクションを追加] ダイアログ ボックスの [その他のアクション] フォルダから [電子メール メッセージを送信] を選択し、[OK] をクリックすると、[電子メール メッセージを送信] ダイアログ ボックスが表示されます。
2. [件名] フィールドに、電子メールの件名を生成するために実行時に評価する有効なワークフロー式を入力します。この式は、リテラル、変数、演算子、および関数で構成できる。式の作成方法の詳細については、第 8 章「ワークフロー式の使用法」を参照してください。
3. [メッセージ] タブのテキスト ボックスに、電子メールのメッセージ テキストを生成するために実行時に評価する有効なワークフロー式を入力します。
4. (省略可能)[MIME 文字セット] フィールドで、Java 言語によりサポートされている MIME 文字セットを入力または選択し、サーバがメッセージを送信する際に使用させることができます。たとえば、2 バイト言語の文字を含むメッセージを送信する場合は、UTF-8 (Unicode 標準フォーマットの 1 つ) を

選択する。受取人がメッセージを正しく表示するためには、選択されたエンコード方法が受取人の使用する電子メール クライアント プログラムでもサポートされていなければなりません。

注意 : ユーザが文字入力に使用するエンコードは、[MIME 文字セット] フィールドでの設定とは関係なく、OS およびロケールによって決まります。

5. [To] タブを選択すると受取人に関する次の情報が表示されます。

Addressee	電子メール アドレス、ユーザ名またはロール名、あるいは実行時に電子メール アドレスを生成するために定義された式。
Via	アドレスが定数として指定されているか、式として指定されているかを示す。
Type	アドレス、ユーザ、またはロール。

図 6-38 [電子メール メッセージを送信] ダイアログ ボックスの [To] タブ

6. [追加] をクリックすると受取人が追加されます。[メールの宛先] ダイアログ ボックスが表示されます。

図 6-39 [メールの宛先] ダイアログ ボックス

7. 以下のオプションの 1 つを選択することにより、アドレスを指定します。
   - アドレス－フィールドに電子メール アドレスを入力するか、[式] チェック ボックスをオンにし、実行時に電子メール アドレスを生成するワークフロー式を入力します。
   - ユーザ－現在のオーガニゼーションと関連付けられたすべてのユーザの中から適切なユーザを選択するためのドロップダウン リストを表示するには、このオプションを選択します。電子メール アドレスは、3-15 ページの「ユーザを作成する」に説明されているように、ユーザのプロパティから取得されます。あるいは、[式] チェック ボックスをオンにし、`CurrentUser()` 関数、`WorkflowAttribute("Initiator")` 関数、`TaskAttribute("Assignee")` 関数のようなワークフロー式を入力します。この式は、実行時にユーザの電子メール アドレス、または引用符で囲まれた特定の電子メール アドレスを生成します。
   - ロール－現在のオーガニゼーションと関連付けられたすべてのロールの中から適切なロールを選択するためのドロップダウン リストを表示するには、このオプションを選択します。メッセージは、そのロールに属するすべてのユーザに送信されます。あるいは、[式] チェック ボックスをオンにし、実行時にロール名を戻すことのできる `TaskAttribute()` 関数など、実行時にロール名を生成するワークフロー式を入力します。

   詳細については第 8 章「ワークフロー式の使用法」を参照してください。
8. [OK] をクリックします。新しい受取人がリストに追加されます。

9. さらに受取人を追加するには、手順 6 から 8 を繰り返します。受取人を更新するには、リストでそれを選択し、[更新] をクリックします。受取人を削除するには、リストでそれを選択し、[削除] をクリックします。メッセージが表示されたら削除を確認します。
10. (省略可能) [CC] タブまたは [BCC] タブで、手順 5 から 8 を繰り返し、明示する同報受取人または隠す同報受取人を指定してください。
11. [OK] をクリックすると [電子メール メッセージを送信] アクションが追加されます。

コンポーネントの呼び出し

以下のアクションを使用して、EJB、Java クラス、実行可能プログラムなどのソフトウェア コンポーネントを呼び出したり、ワークフローとコンポーネントの間で入力パラメータや出力パラメータを直接受け渡すことができます。

- プログラムの呼び出し－サーバ上の実行可能プログラムを呼び出します。プログラムはワークフローと並行実行されます。6-82 ページの「サーバ上の実行可能なプログラムを呼び出す」で説明します。
- ビジネス オペレーションを実行－ EJB または Java クラスのメソッドを表わす、事前コンフィグレーションされたビジネス オペレーションを呼び出します。6-84 ページの「ビジネス オペレーションを呼び出す」で説明します。

サーバ上の実行可能なプログラムを呼び出す

プログラムの呼び出しアクションを使用して、WebLogic Integration サーバ上の実行可能プログラムを呼び出すことができます。このアクションは、常に非同期的に実行されます。つまり、ワークフロー内でこのアクションの後に続くアクションは、呼ばれた側のプログラムが完了するのを待つことなく同時に実行されます。

注意： XML をクライアントに送信アクション内でプログラムの呼び出しアクションを使用して、cmd.exec などのシェル プログラムへのアクセスを許可する場合は注意してください。これによりクライアント コンピュータへの完全なアクセスが可能になるため、アプリケーションのセキュリティが低減します。

図 6-40 [プログラムの呼び出し] ダイアログ ボックス

サーバ上の実行可能プログラムを呼び出す手順は、以下のとおりです。

1. [アクションを追加] ダイアログ ボックスで [統合アクション] フォルダを展開し、[プログラムの呼び出し] を選択して、[OK] をクリックすると、[プログラムの呼び出し] ダイアログ ボックスが表示されます。
2. [プログラム] フィールドに、実行可能ファイルの名前を拡張子も含めて入力します。

   DOS シェル固有のコマンド（echo など）を含まない .cmd ファイルや .bat ファイルのような DOS スクリプト ファイルを実行する場合、以下のいずれかを行います。

   - 環境変数で設定した WebLogic Integration サーバ パスにこのファイルのパスが含まれている場合は、次のように拡張子を含む完全なファイル名を入力する。testscript.bat
   - 環境変数で設定した WebLogic Integration サーバ パスにこのファイルのパスが含まれていない場合には、次のようにパスおよびファイルの完全修飾名を入力する。c:\mydirectory\myfiles\testscript.bat

DOS シェル固有のコマンド（echo など）を含む .cmd ファイルや .bat ファイルのような DOS スクリプト ファイルを実行する場合、次を入力します。

```
c:\winnt\system32\cmd
```

3. [引数] フィールドに、変数名などプログラムに渡す引数を生成するために実行時に評価する有効なワークフロー式を入力します。

   DOS シェル固有のコマンド（echo など）を含む .cmd ファイルや .bat ファイルのような DOS スクリプト ファイルを実行する場合、次を入力します。

   "/c start c:*path**filename*.*extension expression*"

4. [OK] をクリックすると [プログラムの呼び出し] アクションが追加されます。

ビジネス オペレーションを呼び出す

ビジネス オペレーションを実行アクションを使用して、EJB や Java クラスなどの Java コンポーネントで、ビジネス アクティビティを実行するメソッドを呼び出すことができます。

開始したいビジネス オペレーションは、あらかじめ定義しておく必要があります。さらに、Java オブジェクト、セッション EJB、エンティティ EJB の各変数が、ビジネス オペレーションによって呼び出されるメソッドを持つ Java クラスまたは EJB インスタンスへの参照を保存するように、あらかじめ定義しておく必要があります。ビジネス オペレーションを定義する方法の詳細については、4-9 ページの「ビジネス オペレーションのコンフィグレーション」を参照してください。変数の定義方法の詳細は、5-28 ページの「変数に関する作業」を参照してください。

また、EJB 上のメソッドまたは Java クラス上の非静的メソッドを呼び出すビジネス オペレーションを実行できるようにするには、その前に、以下のルールに従って、WebLogic Integration サーバで Java クラスまたは EJB のインスタンスを作成するビジネス オペレーションを呼び出す必要があります。

- Java クラス上の静的メソッドを表わすビジネス オペレーションを呼び出す場合、コンストラクタ メソッドを呼び出す必要はありません。
- Java クラス上の非静的メソッドを表わすビジネス オペレーションを呼び出すには、まず、クラスのコンストラクタ メソッドを表わすビジネス オペレー

ションを呼び出す必要があります。このビジネス オペレーションは、ワークフローで 1 回呼び出せば十分です。

- エンティティ EJB 上のメソッド呼び出しを表わすビジネス オペレーションを呼び出すには、まず、EJB インスタンスを作成するビジネス オペレーションを呼び出す必要があります。このビジネス オペレーションは、ワークフローで 1 回呼び出せば十分です。
- セッション EJB (ステートレスまたはステートフル) でメソッド呼び出しを表わすビジネス オペレーションを呼び出すには、まず、EJB インスタンスを作成するビジネス オペレーションを呼び出す必要があります。このビジネス オペレーションは、その EJB で他のメソッドを呼び出すトランザクションごとに 1 回呼び出す必要があります。ワークフローでのトランザクションと境界については、『*BPM クライアント アプリケーション プログラミング ガイド*』の「BPM トランザクション モデルの理解」を参照してください。

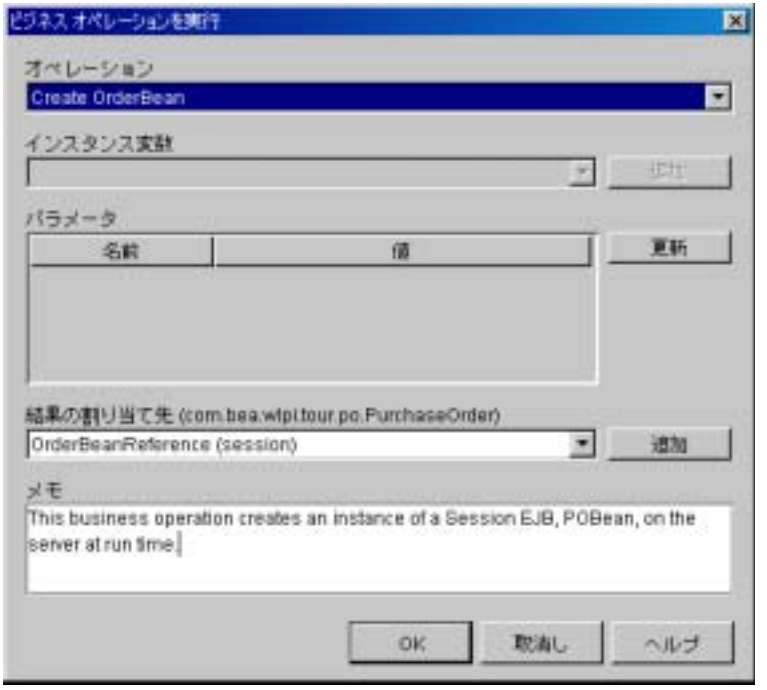
インスタンスを作成するビジネス オペレーションを実行する場合、*インスタンス変数*と呼ばれる変数に対してインスタンスへの参照を割り当てます。その後、同じ EJB またはクラスに入っているメソッドを表わす他のビジネス オペレーションを呼び出す際に、EJB または Java クラス インスタンスを参照するインスタンス変数を識別します。詳細については以下の節で説明します。

EJB または Java クラス インスタンスを作成するためのビジネス オペレーションを呼び出す

EJB または Java クラス インスタンスを作成するビジネス オペレーションを呼び出す場合、以下のように、インスタンスに対する参照を同じデータ型の変数に対して割り当てる必要があります。

- Java クラスの場合、インスタンス変数は Java オブジェクト タイプ
- セッション EJB の場合、インスタンス変数はセッション EJB タイプ
- エンティティ EJB の場合、インスタンス変数はエンティティ EJB タイプ

図 6-41 [ビジネス オペレーションを実行] ダイアログ ボックス



EJB または Java クラス インスタンスを作成するためにビジネス オペレーションを呼び出す手順は、以下のとおりです。

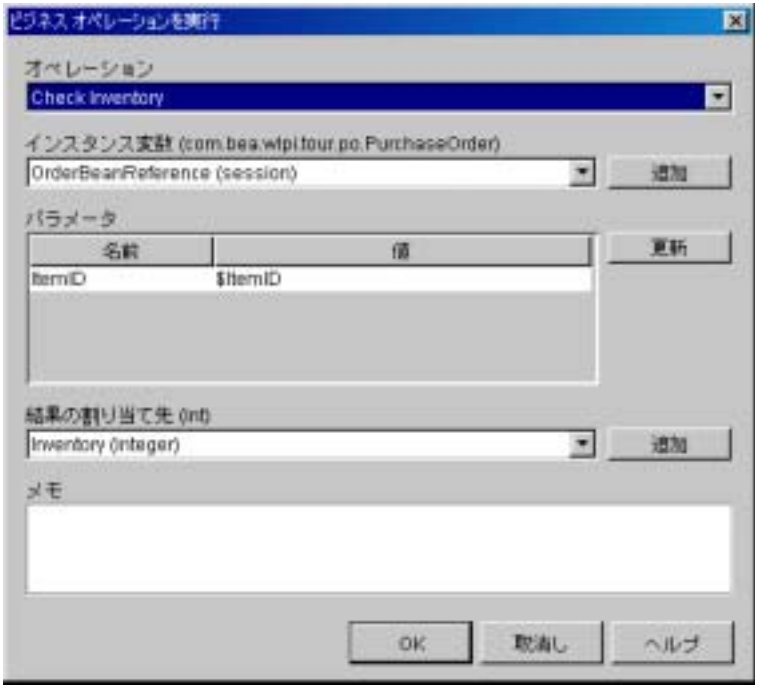
1. [アクションを追加] ダイアログ ボックスの [統合アクション] フォルダから [ビジネス オペレーションを実行] を選択し、[OK] をクリックすると、[ビジネス オペレーションを実行] ダイアログ ボックスが表示されます。
2. [オペレーション] ドロップダウン リストから、Java クラスまたは EJB インスタンスを作成するビジネス オペレーションを選択します。
3. [結果の割り当て先] ドロップダウン リストから、このビジネス オペレーションのためのインスタンス変数を選択するか、[追加] をクリックして [変数を作成] ダイアログ ボックスを呼び出し、変数を作成します。変数の型は、作成されるコンポーネントのタイプに対応している必要があります。
4. [OK] をクリックすると [ビジネス オペレーションを実行] アクションが追加されます。

他のビジネス オペレーションを呼び出す

EJB または Java クラスの create() またはコンストラクタ メソッドのためのビジネス オペレーションを実行をノードに追加した後、同じクラスまたは EJB 上のメソッドを呼び出す他のビジネス オペレーションを追加できます。

合計価格の計算のような、パラメータを取って結果を戻すメソッドの場合、ビジネス オペレーションにより戻される値を格納する変数も作成する必要があります。この変数は、メソッドにより指定された変数と同じタイプにしてください。

図 6-42 [ビジネス オペレーションを実行] ダイアログ ボックス



他のビジネス オペレーションを呼び出す手順は、以下のとおりです。

1. [アクションを追加] ダイアログ ボックスの [統合アクション] フォルダから [ビジネス オペレーションを実行] を選択し、[OK] をクリックすると、[ビジネス オペレーションを実行] ダイアログ ボックスが表示されます。

2. [オペレーション] ドロップダウン リストから、実行するビジネス ロジックを表わすビジネス オペレーションを選択します。EJB 上のメソッド、または非静的 Java クラス メソッドを呼び出すビジネス オペレーションの場合、[インスタンス変数] ドロップダウン リストに、Java オブジェクト 変数、セッション EJB 変数、またはエンティティ EJB 変数が表示されます。
3. [インスタンス変数] ドロップダウン リストから、EJB または Java クラスに対する参照を格納する変数を選択します。詳細については、6-85 ページの「EJB または Java クラス インスタンスを作成するためのビジネス オペレーションを呼び出す」を参照してください。
4. ビジネス オペレーションにより [パラメータ] リストにパラメータが表示された場合、そのリストからパラメータを選択し、[更新] をクリックし、表示された [Expression Builder] ダイアログ ボックスを使用して、この値を定義します。一般に、この値は、既に定義されたワークフロー変数により提供されます。
5. ビジネス オペレーションから結果が戻される場合、[結果の割り当て先] ドロップダウン リストから結果を割り当てる変数を選択するか、あるいは、[追加] をクリックして、[変数を作成] ダイアログ ボックスを呼び出し、変数を定義します。変数の型は、メソッドにより指定された型と一致している必要があります。
6. [OK] をクリックすると [ビジネス オペレーションを実行] アクションが追加されます。

JMS トピックまたはキューへの XML メッセージのポスト

イベントをトリガする目的で、指定された宛先に XML メッセージを送信するには、XML イベントをポストアクションを使用します。このアクションにより、新しい XML 文書を作成またはワークフローの既存の XML 型変数の内容を使用します。いずれの場合でも JMS メッセージ内に XML コンテンツを埋め込みます。この XML コンテンツは、外部アプリケーションで処理するために外部 JMS キューまたはトピックにポストすることも、あるいは、別のワークフローで処理するために内部キューにポストすることもできます。

注意： クラスタ化環境で WebLogic Integration を実行している場合、ワークフローを開始アクションで直接ワークフローを呼び出すより、別のワークフローを開始する XML イベントをポストするほうが、負荷バランスをよりよく調整できます。

送信する XML ドキュメントの作成もできれば、XML リポジトリ、ディスク上のファイル、または URL から既存の XML ドキュメントのインポートも可能です。また、XML コンテンツを実行時に指定できる変数として送信されるドキュメントを指定することもできます。

XML イベントをポストアクションは、XML メッセージ コンテンツだけでなく、アクションのプロパティに指定されたオプションに従って、JMS ヘッダと値をメッセージに挿入します。WebLogic Server サーバの標準 JMS ヘッダ フィ - ルドの詳細については、次の URL にある『*WebLogic JMS プログラマーズ ガイド*』の「WebLogic JMS の基礎」を参照してください。
http://edocs.beasys.co.jp/e-docs/wls/docs70/jms/fund.html

JMS メッセージング オプションについて以下の節で説明します。

イベントを非同期的または同期的にポストする

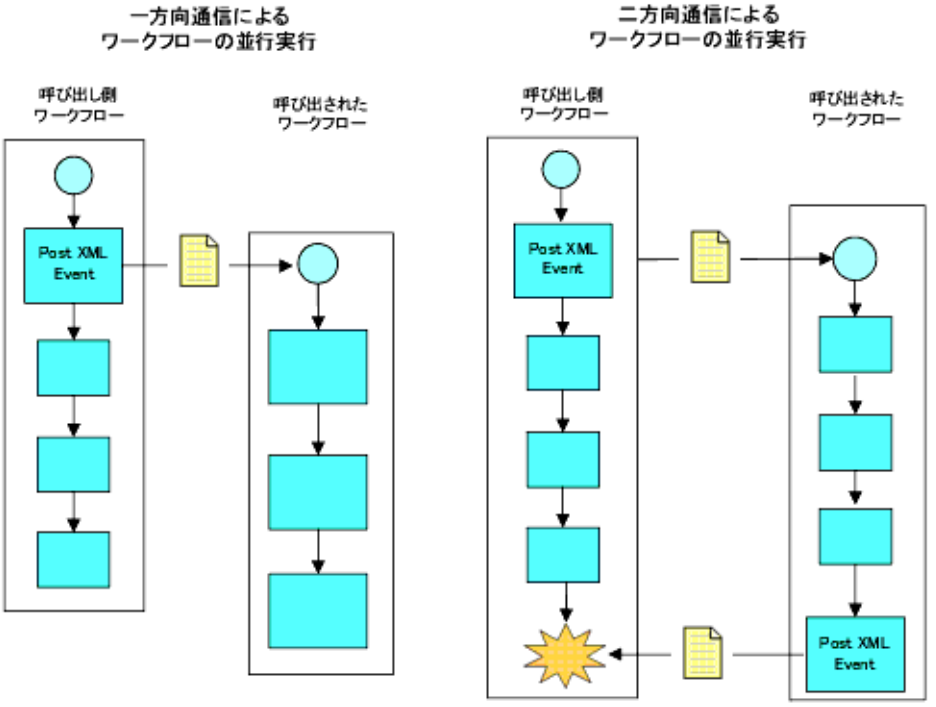
発信メッセージを作成するようにコンフィグレーションされたワークフローまたは他のコンポーネントに関連して、XML イベントを、非同期的にも、あるいは同期的にも機能するよう設定できます。そのためには、受取側ワークフローから戻される確認メッセージを受け取るための通信を開始するワークフローでイベント ノードを使用します。

デフォルトでは、XML イベントをポストアクションは「発信後削除（fire and forget）」方式でメッセージをポストするだけの非同期的なものであり、その間にワークフローは次のアクションに進みます。したがって、並行して実行される他のワークフローまたはアプリケーションをトリガする場合、および呼び出し側ワークフローは、呼び出されたワークフローまたはアプリケーションからの返信を受け取る必要がない場合には、そのままそのアクションを使用してください。

呼び出し側ワークフローと呼び出されたワークフローまたはアプリケーションとを並行して実行し、呼び出し側ワークフローが呼び出されたワークフローまたはアプリケーションからの*返信を期待*している場合、「適時（just in time）」方式でメッセージを受け取るように、呼び出し側ワークフローでイベント ノードを設

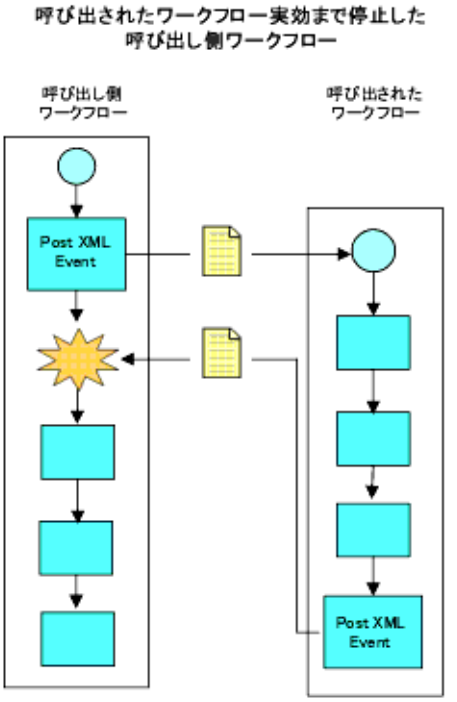
定できます。つまり、ワークフローの中で、呼び出されたワークフローまたはアプリケーションにより戻されるデータを必要とするポイントにのみイベント ノードを設定できます。次の図に 2 つのシナリオを示します。

図 6-43 XML イベントの非同期的ポスト



XML イベントを純粋に同期的方法でポストする場合、つまり、呼び出されたワークフローまたはアプリケーションが実行を完了するまで、呼び出し側ワークフローが先に進まずに待つようにする場合、呼び出されたワークフローまたはアプリケーションが実行を終了した際にメッセージを送信するように設定し、呼び出し側ワークフローの XML イベントをポストアクションの直後にこのメッセージをリスンするイベント ノードを置きます。この設計を次の図に示します。

図 6-44　イベントの同期的ポスト



JMS メッセージング オプション

以下の節では、[XML イベントをポスト] ダイアログ ボックス内から利用できる各種 JMS メッセージング オプションについて説明します。

送り先

内部 JMS キューを指定することにより、現在のワークフローまたは別のワークフローでイベント ノードをトリガしたり、イベントによりトリガされる開始を定義した別のワークフローを開始したりできます（開始ノードとイベント ノードでイベントをトリガする方法の詳細については、5-39 ページの「イベントおよびイベントトリガ型開始のプロパティを定義する」を参照）。デフォルトの

メッセージ送信先である内部 JMS キューの JNDI 名は、`com.bea.wlpiEventQueue` です。WebLogic Server で他にもキューがコンフィグレーションされている場合、代替キュー名を指定できます（WebLogic Integration のための代替メッセージ キューを設定する方法の詳細については、『*WebLogic Integration の起動、停止およびカスタマイズ*』の「WebLogic Integration のカスタマイズ」にある「カスタム Java Message Service キューのコンフィグレーション」を参照）。

また、指定した JMS トピックにサブスクライブする外部アプリケーションや、JMS キュー受信者である特定のアプリケーションと通信するために、外部 JMS トピックやキューに XML メッセージを送信することもできます。

ヘッダ

JMS メッセージは、メッセージと共に常に伝送されるヘッダ フィールドの標準セットを含んでいます。[XML イベントをポスト] ダイアログ ボックスで利用できるオプションにより自動的に挿入される `JMSDeliveryMode`、`JMSDestination`、`JMSPriority`、`JMSExpiration` (存続時間) の各ヘッダに加え、アプリケーション固有の情報のプロパティ フィールドや値を発信メッセージに追加し、メッセージ本文には適していない情報の指定もできます。たとえば、別のワークフローをトリガするために XML メッセージングを使用する場合、ワークフローを開始するオーガニゼーションの名前を指定するためにプロパティ フィールドを使用できます。

JMS メッセージ プロパティは、名前と値の組み合わせです。名前には、Java 言語で有効な識別子ならば、ほとんどの文字列を使用できます。値は、ブール、バイト、ショート、整数、ロング、フロート、倍精度、または文字列にできます。

JMS ヘッダおよびプロパティ フィールドの詳細については、次の URL にある『*WebLogic JMS プログラマーズ ガイド*』の「WebLogic JMS の基礎」を参照。
http://edocs.beasys.co.jp/e-docs/wls/docs70/jms/fund.html

順序指定メッセージまたはアドレス指定メッセージが使用された場合、[XML イベントをポスト] ダイアログ ボックス オプションの [アドレス指定] タブに入力された値に基づいて、順序指定メッセージの場合は指定された順序キーのためのプロパティ フィールドが、アドレス指定メッセージの場合は指定されたワークフロー インスタンス ID のためのプロパティ フィールドが、WebLogic Integration により挿入されます。ただし、さらに 2 つのプロパティがサポートされており、手動で挿入できます。

- WLPIInstanceID －実行時に決定されるワークフロー インスタンス ID の入った単一の変数、またはカンマで区切られた変数のリスト。たとえば、次のいずれかです。
  - `$ParentID`
  - `$Child1ID + “,” + $Child2ID + “,” + $Child3ID`
- WLPI テンプレート名－実行時に決定されるテンプレート名の入った単一のテンプレート名、またはカンマで区切られたテンプレート名のリスト。

この機能は、現在のワークフローとの会話に入力されたワークフロー インスタンスまたはテンプレート名のリストを統合して、単一のメッセージにより外部アプリケーションに渡す必要がある場合に便利です。

メッセージが別のワークフローにより受信されるようにする場合、受信側ワークフローは、開始ノードまたはイベント ノードのイベント キー式または変数初期化で `EventAttribute()` 関数を使用することにより、プロパティ フィールドに指定された情報を検索できます。複数のインスタンス ID またはテンプレート名を指定する場合、Java オブジェクト データ型として定義された変数に関数の結果を割り当てる必要があります。

イベント キー式の詳細については、4-21 ページの「イベント キーのコンフィグレーション」を参照してください。イベント データから変数を初期化する方法の詳細については、5-46 ページの「イベント データからの変数を初期化する」を参照してください。`EventAttribute()` 関数の詳細については、8-6 ページの「実行時のイベント データを抽出する」を参照してください。

配信モード

メッセージは、永続または非永続のいずれかとして指定できます。永続メッセージは、データベース表に書き込まれ、JMS サーバが失敗した場合でも失われません。このメッセージは、サーバが回復した後に再度配信される。非永続メッセージは、JMS サーバが失敗すると失われることがあります。このメッセージは、サーバが回復しても再配信されない。デフォルトの配信モードは永続です。

存続時間

メッセージを永続とするか、非永続とするかは、メッセージの期限により指定できます。消滅時間までに配信されない場合、メッセージは破棄されます。このオプションは、株の指し値のように、ある時刻以降には配信してはならないメッセージの場合に便利です。存続時間はミリ秒単位で表わされます。たとえば、

メッセージを 1 時間有効にする場合は、3600000 (60 分 × 60 秒 × 1000 ミリ秒) という値を指定する。デフォルト値の 0 (ゼロ) は、メッセージに期限がないことを示す。

アドレス指定されたメッセージに対して消滅時間を指定した場合、指定されたすべての受取人にメッセージが正常に配信されるか、あるいは消滅するかのいずれか早い方のイベントが発生するまでメッセージは存続します。

優先度

0 から 9 のレベルの優先順位を割り当てることができます。レベル 0 から 4 は通常の優先順位です。レベル 5 から 9 は迅速を求める優先順位です。優先度が至急のメッセージは、通常の優先度のメッセージよりも先に配信される。一般に、アラーム メッセージやシャットダウン メッセージに対して迅速順位を使用します。デフォルト優先順位はレベル 4 です。

優先順位は順位付きメッセージングをオーバーライドするため、同じ順位キーを持つメッセージに対しては、すべて同じレベルの優先順位を指定する必要があります。推奨設定は、デフォルトの 4 のままにすることです。

他の優先順位レベルを使用するには、まず、WebLogic Server で宛先キーをコンフィグレーションする必要があります。詳細については、次の URL にある『*WebLogic Server 管理者ガイド*』の「JMS の管理」を参照してください。
http://edocs.beasys.co.jp/e-docs/wls/docs70/adminguide/jms.html

トランザクション モード

メッセージを直ちに送信するか、XML イベントをポストアクションの入っている現在のトランザクションがコミットする際に送信するかを指定できます。メッセージを直ちに送信すると、トランザクションが完了したかどうかにかかわらず、メッセージは送信されます。コミット時にメッセージを送信すると、トランザクションが正常に完了し、コミットが発行された場合に限り、メッセージは送信されます。トランザクションが正常に完了せず、ロールバックされた場合、メッセージは送信されません。デフォルトは、トランザクションのコミット時です。ワークフロー トランザクション境界の詳細については、『*BPM クライアント アプリケーション プログラミング ガイド*』の「BPM トランザクション モデル」を参照してください。

アドレス指定メッセージング

インスタンス化されたワークフローの受信側イベント ノードがフローでアクティブ化されていない場合でも、現在のワークフローとの会話を開始した（ワークフローを開始アクションでワークフローを呼び出すことにより、または以前に送信された XML メッセージにより、ワークフロー内の開始ノードまたはイベント ノードをトリガすることにより）特定のワークフロー インスタンスに確実に応答メッセージが配信されるようにするには、アドレス指定メッセージングを使用できますノードのアクティブ化の詳細については、『*BPM クライアント アプリケーション プログラミング ガイド*』の「BPM トランザクション モデル」を参照してください。

アドレス指定メッセージングを使用する場合、通常は、メッセージの配信先のワークフロー インスタンス ID のリストを用意します。このリストは、XML メッセージに埋め込まれた `WorkflowAttribute("InstanceID")` 式により、またはワークフローを開始アクションによりワークフローに渡されるパラメータとして、発信元ワークフローから送信され、現在のワークフローの前のノードにより抽出され、変数に格納されます。インスタンス ID のリストは、通常、対象の ID の入ったカンマ区切りの変数のリストです。メッセージは、このリストに指定されたインスタンスに対してのみ配信されます。

注意： ワークフロー インスタンス ID は文字列として格納されるため、インスタンス ID 値を入れる変数を作成する場合は、文字列型として作成してください。ワークフロー属性関数の詳細については、8-12 ページの「実行時のワークフロー データを収集する」を参照してください。

存続期間を指定した場合、メッセージが指定されたすべての受取人に配信されるか、消滅するかのいずれか早い方のイベントが発生するまで、メッセージは存続します。

順序指定メッセージング

同じイベントのリスナに、メッセージを受け取った順序どおりに処理させるには、順序キーを指定します。たとえば、注文処理システムが注文を作成する要求と、注文の更新またはキャンセルを行う要求を受け取った場合、作成要求メッセージを先に処理させることができます。

順序キーは整数値でなければならず、その値は受け取った順序で処理させたいすべてのイベントで同じでなければなりません。たとえば、2 つのイベントが同時にポストされ、受け取った順序で処理されるようにする場合、各イベントに対して、整数値の 8 のような同じ値の順序キーを入力します。また、順序指定メッセージは、同じ JMS キューに送信する必要があります。

順序指定メッセージングは、メッセージ優先順位と互換性がないため、順序キーを使用する場合は、同じ順序キーを持つすべてのメッセージに対して同じレベルの優先順位を設定する必要があります。推奨設定は、デフォルトの 4 のままにすることです。

XML イベントをポストアクションを定義する

XML イベントを定義する手順は、以下のとおりです。

1. [アクションを追加] ダイアログ ボックスの [統合アクション] フォルダから [XML イベントをポスト] を選択し、[OK] をクリックすると、[XML イベントをポスト] ダイアログ ボックスが表示されます。

図 6-45 [XML イベントをポスト] ダイアログ ボックスの [XML メッセージ] タブ

2. [XML メッセージ] タブを選択し、以下のいずれかを行うことにより、このアクションを送信する XML メッセージを定義します。

   - XML ドキュメントが XML タイプ変数に格納されている場合は、[XML 変数から] ドロップダウン リストから変数を選択します。XML タイプ変数についての詳細については、5-28 ページの「変数に関する作業」を参照してください。
   - 既存の XML ドキュメントをロードまたは編集する場合は [構造] オプションを選択します。新たに自由形式ドキュメントを作成するには [子を追加] ボタンをクリックし、ノードの追加を開始します。既存の XML ド

キュメントを指定するには [インポート] ボタンをクリックし、ドキュメントをロードし、必要に応じて編集します。新たなタイプ指定 XML ドキュメントを作成するには [コンテンツ タイプを設定] ボタンをクリックし、スキーマ ドキュメントをロードします。上記のオプションの詳細な手順については、7-2 ページの「XML ドキュメントの作成と編集」を参照してください。

ユーザ定義の XML ドキュメントは、ワークフロー テンプレート定義内に保存される。

3. [送り先] タブの [送り先] オプションで、以下のいずれかを選択します。
   - [内部] ―デフォルト内部キューである `wlpiEventQueue` にメッセージを送信します。
   - [JMS トピック] ―外部トピックにメッセージをポストします。フィールドに、トピックの JNDI 名を引用符で囲んで入力するか、実行時にトピック名を決定する式を入力します。
   - [JMS キュー] ―外部キュー、または WebLogic Server でコンフィグレーションされている代替内部キューにメッセージをポストします。フィールドに、キューの JNDI 名を引用符で囲んで入力するか、実行時にキュー名を決定する式を入力します。

   注意 : 式の作成方法の詳細については、第 8 章「ワークフロー式の使用法」を参照してください。

図 6-46 [XML イベントをポスト] ダイアログ ボックスの [送り先] タブ

4. (省略可能) [トランザクション モード] オプションから以下のいずれかを選択します。
   - [トランザクションのコミット時] ―トランザクションが正常に完了し、コミットが発行された場合に限りメッセージを送信します。トランザクションが正常に完了せずにロールバックされた場合は、メッセージは送信されない。
   - [即時] ―トランザクションの完了を待たずに直ちにメッセージを送信します。

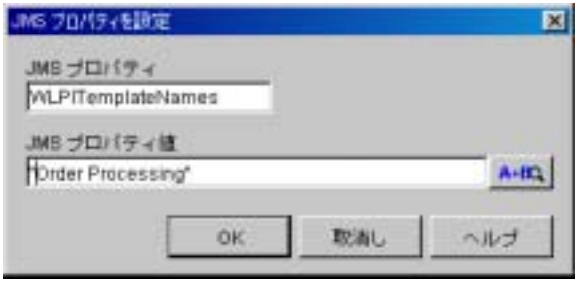
5. (省略可能) [メッセージ ヘッダ] タブを選択し、メッセージに追加する JMS メッセージ プロパティを指定します (利用可能なワークフロー固有のプロパティについては、6-92 ページの「ヘッダ」を参照)。[追加] をクリックすると、[JMS プロパティを設定] ダイアログ ボックスが表示される。このダイアログ ボックスの設定手順は、次のとおり。

   a. [JMS プロパティ] フィールドに、プロパティ名を指定する。

   b. [JMS プロパティ値] フィールドに、プロパティの値を入力する。引用符で囲んだ値を直接入力する、または実行時に評価されて値を生成する式を入力する。

   c. [OK] をクリックします。

図 6-47 [JMS プロパティを設定] ダイアログ ボックス



6. (省略可能) [メッセージ ヘッダ] タブで以下のオプションから選択することにより、配信モードを指定します。

   - [永続] ― メッセージをデータベース表に書き込み、JMS サーバ障害の場合でもメッセージが残るようにします。
   - [非永続] ― メッセージを残しません。JMS サーバが失敗すると、メッセージが失われることがあります。

図 6-48 [XML イベントをポスト] ダイアログ ボックスの [メッセージ ヘッダ] タブ

7. (省略可能) [メッセージ ヘッダ] タブで、[存続期間] フィールドに存続時間をミリ秒単位で指定するか、[式を使用] チェック ボックスにチェックして、実行時に評価されて値を生成する式をフィールドに入力します。値 0 はメッセージが消滅しないことがわかります。

8. (省略可能) ドロップダウン リストから 0 (最低の優先順位) から 9 (最高の優先順位) の値を選択することにより、メッセージ優先順位を指定します。あるいは、[式を使用] チェック ボックスをオンにし、実行時に評価されて値を生成する式をフィールドに入力します。

注意 : メッセージを並べるための順序キーを指定する場合は、デフォルト値 4 のままにします。

9. (省略可能) *アドレス指定*メッセージを送信する場合は、[アドレス指定] タブを選択し、1 つまたは複数のワークフロー インスタンスのために XML メッセージを残しておくことを指定します。[アドレス指定されたメッセージ] チェック ボックスにチェックし、[インスタンス ID] フィールドに 1 つの変数名、またはカンマ区切りの変数のリストを入力します。変数には、現在のワークフローであらかじめワークフロー インスタンス ID を格納しておきます (詳細については、6-95 ページの「アドレス指定メッセージング」を参照)。

図 6-49 [XML イベントをポスト] ダイアログ ボックスの [アドレス指定] タブ

注意 : [メッセージ ヘッダ] 領域で [存続期間] を指定すると、指定した期間だけメッセージが保持されます。

10. (省略可能) *順序指定*メッセージを送信する場合、[アドレス指定] タブでメッセージの順序キーを入力します。これは、順に処理されるようにするすべてのメッセージで同じ値にします。この値は、整数、または実行時に整数値を決定する式でなければなりません。

    注意 : 順序キーを指定する場合は、メッセージ優先順位をデフォルトの 4 のままにしてください。

11. [OK] をクリックすると [XML イベントをポスト] アクションが追加されます。

XML ドキュメントの変換

XSLT (Extensible Stylesheet Language Transformations: 拡張可能スタイルシート言語変換) は、XML ドキュメントを他の XML ドキュメントまたは非 XML ドキュメントに変換するためのルールを定義します。XSL テンプレート ドキュメントは、変換する入力 XML ドキュメントの要素と、その要素をどのように変換するかを指定します。

XSL 変換アクションにより、変換する入力 XML ドキュメント、変換の詳細を指定する XSL テンプレート ドキュメント、変換されたドキュメントを入れる出力変数を指定できます。実際の変換は実行時に行われ、WebLogic Server 付属の XSL 変換エンジンにより実行されます。

入力ドキュメントにはワークフロー式を入れることができます。その場合、式は変換を行う前にプロセス エンジンにより解決され、結果の入った式に置き換えられます。同様に、XSL テンプレート ドキュメントにはワークフロー変数に対する参照を入れることができます。その場合、参照は変換を行う前にプロセス エンジンにより解決され、適切な値の入った参照に置き換えられます。

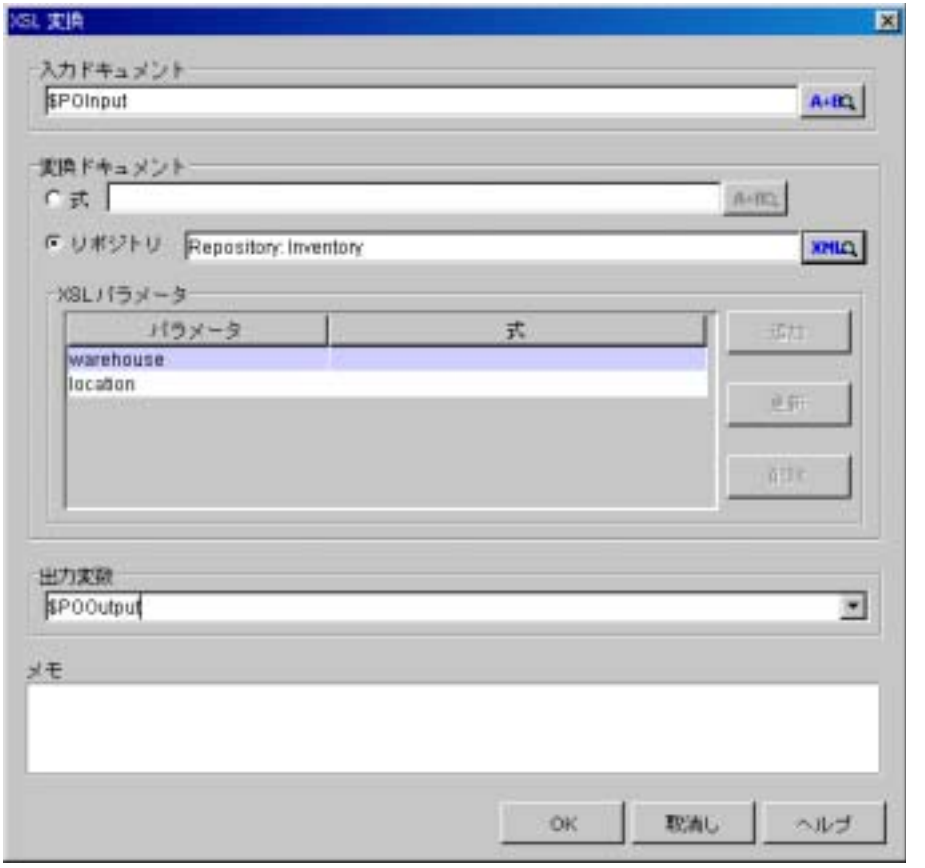
入力ドキュメントは、実行時にドキュメントの場所を指示する式として指定されることに注意してください。この式には、XML ドキュメントを格納したワークフロー変数の名前を含めることができます。この場合、XML タイプ変数を作成し (5-28 ページの「変数に関する作業」参照)、あらかじめワークフローで既存

または着信 XML ドキュメントを割り当てておく必要があります。これを行う方法の 1 つは、ワークフロー変数を設定アクションです。その手順については、6-22 ページの「変数値の設定」を参照してください。

XSL テンプレート ドキュメント（変換ドキュメント）は、リポジトリに格納されたエンティティにできます（詳細については、4-27 ページの「リポジトリにあるエンティティの管理」を参照）。あるいは、実行時にドキュメントの場所を示す式を使用することもできます。リポジトリにある XSL エンティティまたは実行時に場所指定される XSL ドキュメントがパラメータを取る場合、実行時にパラメータの値を提供する式を指定することもできます。

出力ドキュメントは、事前に作成できる XML または文字変数に格納する必要があります。その手順については、5-31 ページの「変数を作成する」を参照してください。

図 6-50 [XSL 変換] ダイアログ ボックス



XML ドキュメントを変換する手順は、以下のとおりです。

1. [アクションを追加] ダイアログ ボックスの [統合アクション] フォルダから [XSL 変換] を選択し、[OK] をクリックすると、[XSL 変換] ダイアログ ボックスが表示されます。
2. [入力ドキュメント] フィールドに、変換する XML ドキュメントを表わす式を入力します。この式は実行時に評価され、XML ドキュメントを得ることができる。また、この式には、以前に入力ドキュメントを格納した変数名を含めることもできる。

注意： 式の作成方法の詳細については、第 8 章「ワークフロー式の使用法」を参照してください。

3. [変換ドキュメント] フィールドの 1 つに、入力ドキュメントを変換するために使用する XSL テンプレート ドキュメントを指定します。以下のオプションからいずれか 1 つを選択します。
   - [式] ― XSL テンプレート ドキュメントを取得するために実行時に評価する式を入力します。
   - [リポジトリ] ― XML リポジトリに格納された XSL テンプレート ドキュメントの名前を入力するか、[XML] ボタンをクリックして [XML ファインダ] ダイアログ ボックスを表示させ、リポジトリから、または URL により指定された場所から、希望する XSL ファイルを選択します。XML ファインダを使用して XML エンティティを検索する手順については、7-19 ページの「XML エンティティを取り出す」 を参照してください。
4. XSL ドキュメントを表わす式を入力した場合、またはパラメータを取る XSL ドキュメントを指定した場合、必要に応じて [追加] をクリックして [XSL パラメータ] ダイアログ ボックスを表示させることにより、実行時に変換ドキュメントに渡すパラメータを追加できます。

図 6-51 [XSL パラメータ] ダイアログ ボックス

5. [パラメータ] フィールドにパラメータ名を入力する。
6. [式] フィールドに、実行時に評価されてパラメータの値を生成する式を入力する。
7. [OK] をクリックしてパラメータを追加する。新しいパラメータが [XSL パラメータ] リストに表示される。

8. パラメータをさらに追加する場合は、手順 4 ～ 7 を繰り返す。あるいは、[更新] または [削除] をクリックして既存のパラメータを更新または削除する。
9. [出力変数] ドロップダウン リストから、変換された XML ドキュメントを入れる変数を選択します。この変数は XML 型でなければならない。変数の名前の入力もできます。変数が存在しない場合には、変数を作成するように要求される。変数の定義方法の詳細は、5-28 ページの 「変数に関する作業」を参照してください。
10. [OK] をクリックするとアクションが追加されます。

例外処理

例外の処理に関係するアクションは、すべて第 9 章 「ワークフロー例外の処理」で説明します。

7　XML エンティティを操作

この章では、Studio を使用して XML エンティティの取り出し、構成、および保存を行う方法を説明します。

- XML ドキュメント管理タスクの概要
- XML ドキュメントの作成と編集
- XML ファインダによる XML エンティティの取り出しとエクスポート

XML ドキュメント管理タスクの概要

いくつかのワークフロー アクションには、ワークフローに埋め込まれているフリーフォーム XML ドキュメントやタイプ指定 XML ドキュメントの作成、編集、およびエクスポートに使用できる XML エディタが内蔵されています。XML ファインダを使用すると、さまざまなソース内の XML エンティティを検索し、そのエンティティを別のタイプのストレージで保存できます。XML ドキュメントのコンテンツとストレージの管理は次のように行います。

- （省略可能）既存の XML エンティティをファイル システムからリポジトリにインポートします。このエンティティには、XML ドキュメント テンプレートやインスタンス、XML スキーマ ドキュメントなどが含まれます。その手順は、4-27 ページの「リポジトリにあるエンティティの管理」で説明されています。

  あるいは、前にエクスポートした XML エンティティを既存のワークフロー パッケージからインポートします。詳細については、11-5 ページの「ワークフロー パッケージのインポート」の手順を参照してください。

- アクション ワークフロー変数を設定、XML イベントをポスト、XML をクライアントに送信、XSL 変換、および例外ハンドラの呼び出しを既存のノードまたはアクションに追加します。最初の 3 つのアクションについては、第 6 章「アクションの定義」を参照してください。例外ハンドラの呼び出しアク

ションについては、9-13 ページの「例外ハンドラを呼び出す」を参照してください。

- アクション ダイアログ ボックスで XML ドキュメントを構成、インポート、および編集します。手順の詳細は、7-2 ページの「XML ドキュメントの作成と編集」を参照してください。
- （省略可能）XML ファインダを使用して XML エンティティをファイル システムまたは XML リポジトリ データベースから取り出してワークフローにインポートします。手順の詳細は、7-19 ページの「XML エンティティを取り出す」を参照してください。
- （省略可能）XML ファインダを使用して、ワークフローに埋め込まれた XML ドキュメントをファイル システムまたは XML リポジトリ データベースにエクスポートします。手順の詳細は、7-26 ページの「XML エンティティをエクスポートする」を参照してください。

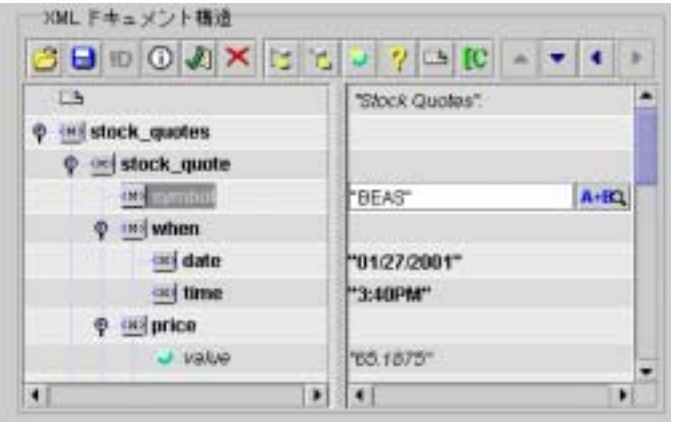
注意： XML ドキュメントを作成するダイアログ ボックスでは XML 要素値を式として入力する必要があるため、XML ドキュメントの定義を始める前に、ワークフロー式言語と Studio の Expression Builder ツールの学習も必要です。ワークフロー式に関する詳細については、第 8 章「ワークフロー式の使用法」を参照してください。

XML ドキュメントの作成と編集

次のアクション ダイアログ ボックスには、ワークフロー テンプレート定義内に保存された整形式の XML ドキュメントの作成、インポート、および編集に使用する、組み込み XML エディタが含まれています。

- [ワークフロー変数を設定]
- [XML イベントをポスト]
- [XML をクライアントに送信]
- [例外ハンドラの呼び出し]

図 7-1 アクション ダイアログ ボックスの XML エディタ



XML エディタには、次の標準 XML マークアップから成るノードで構成される XML ドキュメントが、ツリー構造で表示されます。

- Elements (要素)
- Attributes （属性）
- Comments (コメント)
- CDATA sections (CDATA セクション)
- Processing instructions (処理手順)

左ペインには要素および属性のタグなどのメタデータが表示され、右ペインには各タグに対する実際のデータ値が表示されます。

Studio のダイアログ ボックスで作成および編集するドキュメントは、実際には、実行時に XML ドキュメントの *インスタンス* を生成するために使われる XML ドキュメントの *テンプレート* です。つまり、要素および属性に対してワークフローの式を使用して実行時に値を生成し、これらのノードのうちの 1 つから Expression Builder を起動してユーザ指定の式を作成できます。要素と属性のデータはワークフローの式構文に従う必要があるため、文字列をすべて引用符で囲みます。入力した値が有効なワークフロー式でない場合、たとえば文字列が引用符で囲まれていない場合やバックスラッシュの前にエスケープ文字がない場合

は、  アイコンがその前に表示されます。ワークフロー式の言語、および Expression Builder の使い方については、第 8 章「ワークフロー式の使用法」を参照してください。

ドキュメントの新規作成、および既存のドキュメントの編集には、フリーフォーム モードとコンテンツ型モードの 2 つのモードがあります。フリーフォーム モードでは、コンテンツ タイプの検証を行わずにドキュメントを作成および編集します。特定のコンテンツ タイプに従う必要のない整形式の XML ドキュメントを生成するときには、フリーフォーム モードを使用します。フリーフォーム モードでの XML ドキュメント作成の詳細は、7-6 ページの「フリーフォーム ドキュメントを作成する」を参照してください。

コンテンツ タイプ モードでは、ドキュメントを検証する基準となる既存の外部スキーマ ドキュメントをロードすることで新規または既存ドキュメントのコンテンツ タイプを指定します。ドキュメントの作成または編集処理中に、ドキュメントの有効性を何度でも検証できます。コンテンツ型の指定されたドキュメントでの作業については、7-11 ページの「タイプ指定ドキュメントを操作する」を参照してください。

既存のフリーフォームおよびタイプ指定ドキュメントをリポジトリまたはディスク上のファイルからインポートすること（7-7 ページの「既存のドキュメントをインポートする」参照）や、コンテンツ タイプを既存ドキュメントに追加すること（7-15 ページの「既存のドキュメントへの新しいコンテンツ タイプを設定する」参照）も可能です。

さらに、Studio で作成または編集されたドキュメント テンプレートをリポジトリまたはディスク上のファイルにエクスポートすることが可能です。アクション ダイアログ ボックスから XML ドキュメントをエクスポートする方法については、7-26 ページの「XML エンティティをエクスポートする」を参照してください。

XML ドキュメントの作業に使用するダイアログ ボックス内には、次の表に示すツールバーが表示されます。XML ドキュメントの作成、インポート、編集、およびコンテンツ タイプの設定方法については、後の節で説明します。

表 7-1 XML エディタのツールバー ボタンとキーボード ショートカット

ボタン	キーボード ショートカット	目的
	〔Ctrl〕 + 〔q〕	リポジトリ、ディスク上のファイル、または URL 位置から、編集対象の既存の XML ドキュメントを取り出す。詳細については、7-19 ページの「XML エンティティを取り出す」を参照。
	〔Ctrl〕 + 〔w〕	ワークフロー内の XML メッセージをリポジトリまたはディスク上のファイルに保存する。詳細については、7-26 ページの「XML エンティティをエクスポートする」を参照。
	〔Ctrl〕 + 〔t〕	XML ドキュメントのコンテンツ タイプとして設定するスキーマ ドキュメントを取り出す。詳細については、7-11 ページの「タイプ指定ドキュメントを操作する」を参照。
	〔Ctrl〕 + 〔k〕	コンテンツ タイプとして現在設定されているスキーマ ファイルのコンテンツを表示する。詳細については、7-17 ページの「タイプ指定ドキュメントを検証する」を参照。
	〔Ctrl〕 + 〔l〕	ドキュメントが有効な XML で、現在のコンテンツ タイプに従っているかどうかを調べる。詳細については、7-17 ページの「タイプ指定ドキュメントを検証する」を参照。
	〔Delete〕	選択したノードを削除する、または文書型宣言が選択されている場合は、現在のドキュメントからコンテンツ タイプ定義を取り除く。
	〔Ctrl〕 + 〔Insert〕	選択した要素と同じレベルに、要素ノードを追加する。
	〔Insert〕	ドキュメントのルート要素ノードを追加する、または選択した要素の下位レベルの要素を追加する。
	〔Ctrl〕 + 〔a〕	選択した要素に属性ノードを追加する。

表 7-1 XML エディタのツールバー ボタンとキーボード ショートカット

ボタン	キーボード ショートカット	目的
	〔Ctrl〕 + 〔p〕	処理手順ノードを追加する。
	〔Ctrl〕 + 〔m〕	コメント ノードを追加する。
	〔Ctrl〕 + 〔n〕	CDATA セクション ノードを追加する。
	〔Ctrl〕 + 〔↑〕	選択したノードを、同じレベル内で上に移動する。
	〔Ctrl〕 + 〔↓〕	選択したノードを、同じレベル内で下に移動する。
	〔Ctrl〕 + 〔→〕	選択したノードを、下位レベルに移動する。
	〔Ctrl〕 + 〔←〕	選択したノードを、上位レベルに移動する。

フリーフォーム ドキュメントを作成する

フリーフォーム ドキュメントを作成する手順は、次のとおりです。

1. XML エディタを含むアクション ダイアログ ボックスで、[子を追加] ボタン  をクリックしてルート要素をドキュメントに追加します。デフォルトの要素名が選択された状態で表示されます。
2. デフォルト名 `element` を変更するには、新しい名前を入力して上書きします。
3. 要素に値を追加するには、右ペインで要素の横にあるフィールドをダブルクリックし、ワークフローの式構文を使って値を入力します。

4. 表 7-1 に示すツールバー ボタンを使用する、または 7-9 ページの「XML ドキュメントを編集する」に示す手順に従って、ドキュメントにノードと値をさらに追加します。
5. ドキュメントの作成が完了したときに、次の操作のうち 1 つを行います。
   - (省略可能) ドキュメントをリポジトリまたはファイルにエクスポートします。エクスポートの手順は、7-26 ページの「XML エンティティをエクスポートする」を参照してください。
   - アクション ダイアログ ボックスで [OK] をクリックし、ドキュメントをワークフローに保存します。要素または属性に無効なワークフローの式が含まれている場合、ドキュメントを保存する前に訂正するように求められます。
   - ワークフローからドキュメントを破棄するには、アクション ダイアログ ボックス内の [取消し] をクリックします。

既存のドキュメントをインポートする

Studio の XML エディタで作成後にエクスポートした、またはその他のメソッドで作成されたフリーフォームおよびコンテンツ型の XML ドキュメントは、両方ともインポートできます。コンテンツ型のドキュメントのインポートについては、7-12 ページの「タイプ指定ドキュメントのインポートについて」を参照してください。

既存 XML ドキュメントをワークフローにインポートすると、XML エディタによって XML ドキュメントが次の 2 種類に区別されます。

- Studio の XML エディタで前にエクスポートされた XML ドキュメント。つまり、ワークフロー式構文に即した要素値と属性値を含むワークフロー XML ドキュメント *テンプレート*。
- 別の方法で作成された XML ドキュメント。つまり、標準 XML 値を含む XML ドキュメント *インスタンス*。

Studio から作成およびエクスポートされたワークフロー XML ドキュメント テンプレートでは、要素および属性の文字列の値が引用符で囲まれていますが、その他のドキュメントのインスタンスでは、通常、囲まれていません。そのため、Studio のアクション ダイアログ ボックス以外で作成された XML ドキュメント

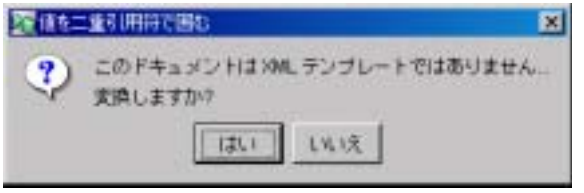
では、ドキュメントを XML テンプレートに変換するよう求められます。変換処理を行うと、必要に応じてすべての要素および属性の値が引用符で囲まれるため、手動で挿入する必要はありません。

注意： Studio では、前のバージョンの WebLogic Process Integrator からエクスポートされた XML ドキュメントはドキュメント テンプレートとして認識されないため、この場合も、これらのドキュメントを XML ドキュメントにコンバートするよう要求されます。XML エディタが余分な引用符を挿入しないように、インポートしたドキュメントを変換しないでください。

既存のフリーフォームまたはコンテンツ型 XML ドキュメントをインポートする手順は、次のとおりです。

1. アクション ダイアログ ボックスで [インポート] ボタン  をクリックし、XML ファインダを表示します。
2. 7-19 ページの「XML エンティティを取り出す」の手順に従って、XML ファインダを使用してさまざまなソースからドキュメントを選択します。ドキュメントがドキュメント テンプレートとして検出されなかった場合、「値を二重引用符で囲む」というメッセージが表示されます。

図 7-2 [値を二重引用符で囲む] ダイアログ ボックス



3. 次の操作を行います。
   - 旧バージョンの WebLogic Process Integrator から作成およびエクスポートされたドキュメントの場合は、[いいえ] をクリックします。
   - その他のメソッドによって作成されたドキュメントの場合は、[はい] をクリックします。

   ドキュメントが XML エディタにインポートされます。

4. 必要に応じて、表 7-1 に示すツールバー ボタンを使用するか、または 7-9 ページの「XML ドキュメントを編集する」に示す手順に従って、ノードと値を追加および編集します。
5. （省略可能）コンテンツ型のドキュメントでは、7-17 ページの「タイプ指定ドキュメントを検証する」の手順に従って、ドキュメントの検証を行います。
6. （省略可能）別のドキュメントをインポートして現在のドキュメントと置き換えるには、[インポート] ボタンをクリックします。現在のドキュメントを上書きするプロンプトが表示されたら、[はい] をクリックしてインポートを続行します。または [いいえ] をクリックして取り消します。
7. ドキュメントの編集が完了したときに、次の操作のうち 1 つを行います。
   - （省略可能）ドキュメントを XML リポジトリまたはディスク上のファイルにエクスポートします。エクスポートの手順は、7-26 ページの「XML エンティティをエクスポートする」を参照してください。
   - アクション ダイアログ ボックス内の [OK] をクリックして、ワークフロー内にドキュメントを保存します。要素または属性に無効なワークフローの式が含まれている場合、ドキュメントを保存する前に訂正するように求められます。
   - ワークフローからドキュメントを破棄するには、アクション ダイアログ ボックス内の [取消し] をクリックします。

XML ドキュメントを編集する

新規またはインポートされた、フリーフォームまたはコンテンツ型のドキュメントの編集手順は、一般的に次のとおりです。コンテンツ型のドキュメントを編集する際の詳細は、7-11 ページの「タイプ指定ドキュメントを操作する」を参照してください。

表 7-2 XML ドキュメントの編集

目的	操作
ドキュメント内を移動する	左ペインですべてのノードを展開する。右ペインの右側にあるスクロール バーを使用する、または〔Tab〕、〔↑〕、〔↓〕を押して、ドキュメント内を上下に移動する。

表 7-2 XML ドキュメントの編集

目的	操作
フリーフォーム ドキュメントの最初の要素を追加する	[子を追加] ボタンをクリックする、または〔Insert〕を押す。
要素にサブ要素を追加する	左ペイン内で要素を選択し、[子を追加] ボタンをクリックする、または〔Insert〕を押す。
別の要素と同じレベルに要素を追加する	左ペイン内で要素を選択し、[兄弟を追加] ボタンをクリックする、または〔Ctrl〕+〔Insert〕を押す。
要素名を編集する	左ペインで要素をダブルクリックしてエントリ フィールドを表示する。必要に応じて要素名を変更する。
要素に属性を追加する	左ペイン内で要素を選択し、[属性を追加] ボタンをクリックする、または〔Ctrl〕+〔a〕を押す。
属性名を編集する	左ペインで属性をダブルクリックしてエントリ フィールドを表示する。必要に応じて属性名を変更する。
要素または属性の値を編集する	右ペイン内で、値を追加または編集する要素または属性の横にある線をダブルクリックする。エントリ フィールドが表示されるので値を入力する。または [式] ボタンをクリックして [Expression Builder] を呼び出し、有効なワークフロー式を構築する。文字列は二重引用符で囲む必要がある。構文が無効な場合、エントリの横に赤い X が表示される。ワークフローの式の定義については、第 8 章「ワークフロー式の使用法」を参照。
コメント ノードを追加する	左ペイン内でコメントを追加する要素を選択し、[コメントを追加] ボタンをクリックする、または〔Ctrl〕+〔m〕を押す。
処理手順を追加する	左ペイン内で、処理手順を追加する要素を選択し、[処理手順を追加] ボタンをクリックする、または〔Ctrl〕+〔p〕を押す。
処理手順のターゲットを変更する	左ペインで処理手順をダブルクリックしてエントリ フィールドを表示する。必要に応じて処理手順対象を変更する。

表 7-2 XML ドキュメントの編集

目的	操作
CDATA セクションを追加する	左ペイン内で、CDATA セクションを追加する要素を選択し、[CDATA を追加] ボタンをクリックする、または〔Ctrl〕 + 〔n〕 を押す。
コメント、処理手順、または CDATA セクションの値の編集	右ペインで、値を編集するコメント、処理手順、または [CDATA] アイコンの横にあるラインをダブルクリックする。エントリ フィールドが表示されるので値を入力する。
ノードの順序を変更する	移動する要素、属性、コメント、処理手順、または CDATA セクションを選択し、[ノードを上に移動] または [ノードを下に移動] ボタンをクリックする、または〔Ctrl〕 +〔↑〕 または 〔Ctrl〕 +〔↓〕 を押す。
ノードの階層的レベルを変更する	レベルを変更する要素、コメント、処理手順、または CDATA セクションを選択し、[ノードを右に移動] または [ノードを左に移動] ボタンをクリックする、または〔Ctrl〕 +〔→〕 または 〔Ctrl〕 +〔←〕 を押す。
ノードを削除する	左右いずれかのペイン内で、削除する要素または属性を選択して、[削除] ボタンをクリックする、または〔Delete〕 を押す。

タイプ指定ドキュメントを操作する

既存の外部スキーマに基づいて XML ドキュメントを作成することができます。内部 DTD 宣言はサポートされていません。

また、既存ドキュメントをスキーマに基づいてインポートすることや、XML エディタに既にロードされているドキュメントのコンテンツ タイプを設定することも可能です。

参照スキーマの格納について

WebLogic Integration サーバは、XML ドキュメント テンプレートに関連付けられているスキーマ ドキュメントに実行時にアクセスできなければならないため、Studio XML ドキュメントによって参照されるスキーマは URL でアクセスできる場所にある XSD ファイルとして、またはリポジトリ内のエンティティとして存在する必要があります。

注意 : URL が WebLogic Integration サーバと異なるマシン上の場所を指す場合、実行時に処理エンジンが XML ドキュメントのインスタンスを生成しても、URL で参照するシステムが非アクティブな状態である危険性があります。このような場合、サーバ例外が発生します。XSD ファイルは、実行時に確実にアクセスできる場所（WebLogic Integration サーバのファイル システム上など）に格納するようにしてください。また、使用する URL が実行時において有効で、必要なリソースの正しい場所を指すことも確認してください。

スキーマ リソースをリポジトリに格納しておくと、頻繁にアクセスされるドキュメントの取り出しが容易になる場合があります。リソースをリポジトリにインポートする手順については、4-27 ページの「リポジトリにあるエンティティの管理」を参照してください。

一方で、新規作成されたドキュメントをエクスポートする場合や、インポート済みのドキュメントを再エクスポートする場合は、リポジトリに保持されるスキーマへの参照はコンテンツ タイプ宣言内のエンティティ名でのみ識別され、サードパーティ製の XML パーサでは解決されないこともあります。したがって、エクスポートされた XML ドキュメントが WebLogic Integration の外と内のどちらで使用されるかに応じて、XSD を適切な場所に配置することが重要です。XML ドキュメントを WebLogic Integration の外部で使用する場合、ディスク ファイル上にリソースを格納し、URL でアクセスすることをお勧めします。XML ドキュメントを WebLogic Integration 内で使用する場合、XML リポジトリにリソースを格納することをお勧めします。

タイプ指定ドキュメントのインポートについて

文書型宣言を含む既存ドキュメントを外部スキーマ ドキュメントにインポートすることは可能ですが、内部ドキュメントにインポートすることはできません。設計時に WebLogic Integration サーバが参照スキーマにアクセスできること、および文書型宣言でドキュメントの場所を指定するために有効な URL が使用され

ていることを確認してください。宣言にドキュメント名のみが含まれていて、完全な場所は含まれていない場合は、WebLogic Integration サーバは参照を解決できず、ドキュメントのインポートは不可能です。

別の方法として、参照スキーマ リソースを XML リポジトリに格納することも可能です。この方法では、リポジトリ エンティティの名前を文書型宣言に単に指定すれば、WebLogic Integration サーバが参照を解決します。リソースをリポジトリにインポートする手順については、4-27 ページの「リポジトリにあるエンティティの管理」を参照してください。

インポートしたドキュメントを再度エクスポートする場合は、前述の実行時の注意事項をご覧ください。

コンテンツ型ドキュメントのインポート手順については、7-7 ページの「既存のドキュメントをインポートする」を参照してください。

タイプ指定ドキュメントを作成する

XML ドキュメントをスキーマに基づいて作成すると、XML エディタによって以下のコンポーネントから構成されるデフォルト ドキュメントが作成されます。

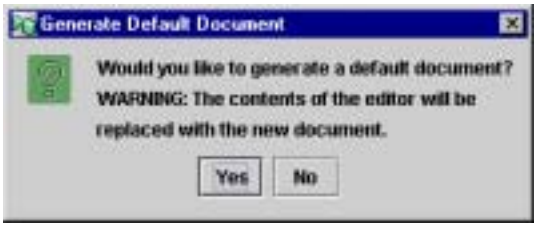
- スキーマ定義ドキュメントの場合は、参照スキーマを指定するルート要素内の属性。
- すべての必須要素および属性、任意の要素および属性の 1 つのインスタンス。コンテンツ タイプ定義によってサポートされる、任意の要素および属性をさらに追加できます。
- 要素または属性のデフォルト値。定義済みの許容値のセットからデフォルト値を指定します。値を編集して、コンテンツ タイプ定義によってサポートされる別の値を指定できます。
- 処理手順。ドキュメントがワークフロー XML テンプレートであるかどうかを示します。処理手順は見えませんが、エクスポートされたドキュメント内に生成されます。

新しいコンテンツ型ドキュメントを作成する手順は、次のとおりです。

1. XML エディタを含むアクション ダイアログ ボックスで、[コンテンツ タイプを設定] ボタン  をクリックします。[XML ファインダ] が表示されます。

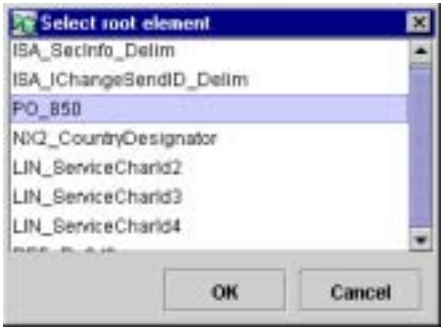
2. 7-19 ページの「XML エンティティを取り出す」の手順に従い、[XML ファインダ] を使用して適切なソースからスキーマ ドキュメントを選択します。コンテンツ型ドキュメントが取得されると、「デフォルトのドキュメントを生成」というメッセージが表示されます。

図 7-3 [デフォルトのドキュメントを生成] ダイアログ ボックス



3. [はい] をクリックして、デフォルト ドキュメントを生成します。[ルート要素を選択してください] ダイアログ ボックスに、定義したすべての要素が表示されます。

図 7-4 [ルート要素を選択してください] ダイアログ ボックス



4. ドキュメントのルート要素となる要素を選択し、[OK] をクリックします。デフォルト ドキュメントが作成されます。DTD ベースのドキュメントでは、ルート要素の上位にプロログが文書型宣言と一緒に挿入されます。スキーマに基づくドキュメントでは、ルート要素内にスキーマを参照する属性が挿入されます。

5. 表 7-1 のツールバー ボタンを使用、または 7-9 ページの「XML ドキュメントを編集する」の手順に従って、ノードと値を必要に応じて追加および編集します。

6. 7-17 ページの「タイプ指定ドキュメントを検証する」の手順に従って、ドキュメントを検証します。
7. ドキュメントの作成および検証が完了したときに、次の操作のうち 1 つを行います。
   - （省略可能）ドキュメントを XML リポジトリまたはディスク上のファイルにエクスポートします。エクスポートの手順は、7-26 ページの「XML エンティティをエクスポートする」を参照してください。
   - アクション ダイアログ ボックス内の [OK] をクリックして、ワークフロー内にドキュメントを保存します。要素または属性に無効なワークフローの式が含まれている場合、ドキュメントを保存する前に訂正するように求められます。
   - ワークフローからドキュメントを破棄するには、アクション ダイアログ ボックス内の [取消し] をクリックします。

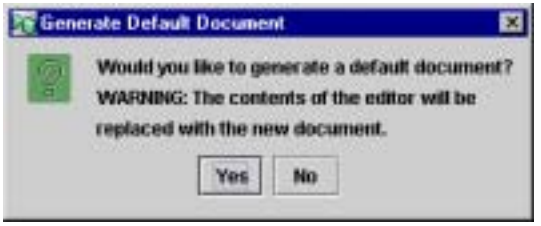
既存のドキュメントへの新しいコンテンツ タイプを設定する

作成したドキュメントまたはインポートした既存のドキュメントに、コンテンツ タイプ定義を適用できます。また、コンテンツ タイプ ドキュメントからコンテンツ タイプを削除したり、新規または既存のコンテンツ型ドキュメントのコンテンツ タイプを変更することもできます。

フリーフォーム ドキュメントにコンテンツ タイプ定義を適用する手順は、次のとおりです。

1. 以下のいずれか 1 つを実行します。
   - 7-6 ページの「フリーフォーム ドキュメントを作成する」の手順に従って、新しいフリーフォーム ドキュメントを作成します。
   - 7-7 ページの「既存のドキュメントをインポートする」に記載の手順 1 ～ 3 に従い、既存のフリーフォーム ドキュメントをインポートします。
2. [コンテンツ タイプを設定] ボタン  をクリックします。[XML ファインダ] が表示されます。
3. 7-19 ページの「XML エンティティを取り出す」の手順に従い、[XML ファインダ] を使用して適切なソースからスキーマ ドキュメントを選択します。コンテンツ型ドキュメントが取得されると、「デフォルトのドキュメントを生成」というメッセージが表示されます。

図 7-5 [デフォルトのドキュメントを生成] ダイアログ ボックス



4. [いいえ] をクリックして、既存のドキュメントを保持します。
5. 7-9 ページの「XML ドキュメントを編集する」の説明に従って、ドキュメントを定義します。
6. 7-17 ページの「タイプ指定ドキュメントを検証する」の説明に従って、ドキュメントを検証します。

コンテンツ型ドキュメントからコンテンツ タイプ定義を削除する手順は、次のとおりです。

1. スキーマ型ドキュメントでは、スキーマを参照するルート要素の属性ノードを選択します。
2. [削除] ボタン をクリックします。コンテンツ タイプ定義が削除され、ドキュメントがフリーフォーム ドキュメントに変わります。

コンテンツ型ドキュメントのコンテンツ タイプを変更する手順は、次のとおりです。

1. スキーマ型ドキュメントでは、スキーマを参照するルート要素の属性ノードを選択します。
2. ルート要素の上位にある、コンテンツ タイプ定義を含むノードを選択します。
3. [削除] ボタン をクリックします。コンテンツ タイプ定義が削除されます。
4. [コンテンツ タイプを設定] ボタン をクリックします。[XML ファインダ] が表示されます。

5. 7-19 ページの「XML エンティティを取り出す」の手順に従い、[XML ファインダ] を使用して適切なソースからスキーマ ドキュメントを選択します。コンテンツ型ドキュメントが取得されると、「デフォルトのドキュメントを生成」というメッセージが表示されます。
6. [いいえ] をクリックして、既存のドキュメントを保持します。
7. 7-9 ページの「XML ドキュメントを編集する」の説明に従って、ドキュメントを定義します。
8. 7-17 ページの「タイプ指定ドキュメントを検証する」の説明に従って、ドキュメントを検証します。

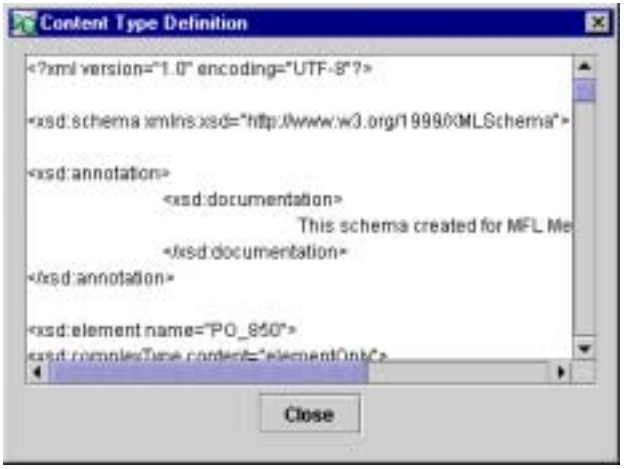
タイプ指定ドキュメントを検証する

新規作成またはインポートされたタイプ指定ドキュメントを構成または編集している間は、現在のコンテンツ タイプとして設定されているスキーマ ドキュメントのコンテンツを参照でき、エラーの原因の参照とその修正を同時に可能にする Studio 検証機能を利用できます。

関連付けられているスキーマのコンテンツを参照する手順は、次のとおりです。

1. [コンテンツ タイプ定義を表示] ボタン  をクリックします。[コンテンツ タイプ定義] ウィンドウが開き、現在のコンテンツ タイプ定義ドキュメントのコンテンツが表示されます。

図 7-6 [コンテンツ タイプ定義] ウィンドウ

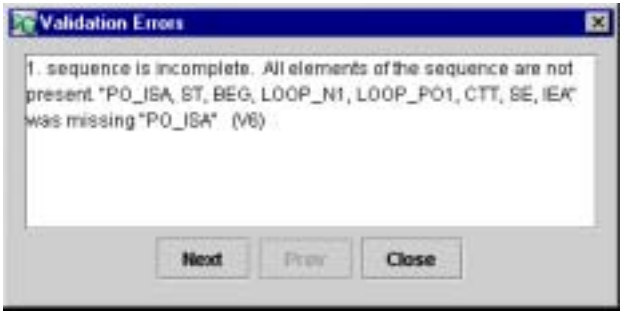


2. [閉じる] をクリックして、ウィンドウを終了します。

新規またはインポートしたコンテンツ型ドキュメントをコンテンツ タイプ定義に対して検証する手順は、次のとおりです。

1. [ドキュメント構造を検証] ボタン  をクリックします。ドキュメントが有効かどうかを示すメッセージが表示されます。ドキュメントにエラーが含まれている場合、[検証エラー] ダイアログ ボックスが開き、エラーの性質および場所が表示されます。

図 7-7 [検証エラー] ダイアログ ボックス



2. [次へ] または [前へ] をクリックしてエラーのリストをスクロールし、エラーのあるノードを編集します。
3. [閉じる] をクリックして、ダイアログ ボックスを閉じます。
4. [ドキュメント構造を検証] ボタンを再度クリックし、ドキュメントを再検証します。「有効なドキュメント」というメッセージが表示されるまで、手順 1 から 3 を繰り返します。
5. [OK] をクリックして、[有効なドキュメント] メッセージ ボックスを閉じます。

XML ファインダによる XML エンティティの取り出しとエクスポート

[XML ファインダ] ダイアログ ボックスを使用すると、XML エンティティを取り出してリポジトリ、ローカル ファイル システム、または URL に保存できます。また、このダイアログ ボックスには、最近使用した XML エンティティのリストが保持されているため、これらのエンティティを再度使用するときに検索する必要がありません。

XML エンティティとしては、XML ドキュメント、文書型定義（DTD）、スキーマ ドキュメント（XSD）、MFL (Message Format Language: メッセージ フォーマット言語) ファイル、XSL (Extensible Stylesheet Language: 拡張スタイルシート言語) テンプレート ドキュメントなどがあります。

XML ファインダにアクセスするには、いくつかの方法があります。XML ファインダにアクセスする方法によって、また、ドキュメントを検索するか保存するかによって、操作方法は異なります。この方法について、以下の節で説明します。

XML エンティティを取り出す

XML エンティティを検索して、次のダイアログ ボックスにロードできます。

- [ワークフロー変数を設定]、[XML をクライアントに送信]、[XML をクライアントに送信] および [例外ハンドラの呼び出し] の各アクション ダイアログ ボックスでは、リポジトリ、ローカル ファイル システム、または URL から既存の XML ドキュメントを検索できます。取り出した XML ドキュメントは、ダイアログ ボックス内で編集できます。リポジトリまたは URL から既存のスキーマ ドキュメントを取り出して、構成または編集するドキュメントのコンテンツ タイプを設定することも可能です。
- [XSL 変換] ダイアログ ボックスでは、リポジトリまたは URL から XSL ドキュメントの名前を検索して、変換ドキュメントとして使用できます。
- [XPath Wizard] では、リポジトリ、ローカル ファイル システム、または URL から XML エンティティを開いて、XPath 式を生成またはテストできます。
- [_ ツール | XML ファインダを表示] を選択すると、XML エンティティをローカル ファイル システムまたは URL からリポジトリにインポートできます。詳細については、4-27 ページの「リポジトリにあるエンティティの管理」を参照してください。

最近使用した XML エンティティを取り出す

XML ファインダにより、ワークステーションで最近使用した XML エンティティのリストが保持されます。リスト内のエンティティは、再検索しなくても再使用できます。

最近使用したエンティティのリストからエンティティを取り出す手順は、次のとおりです。

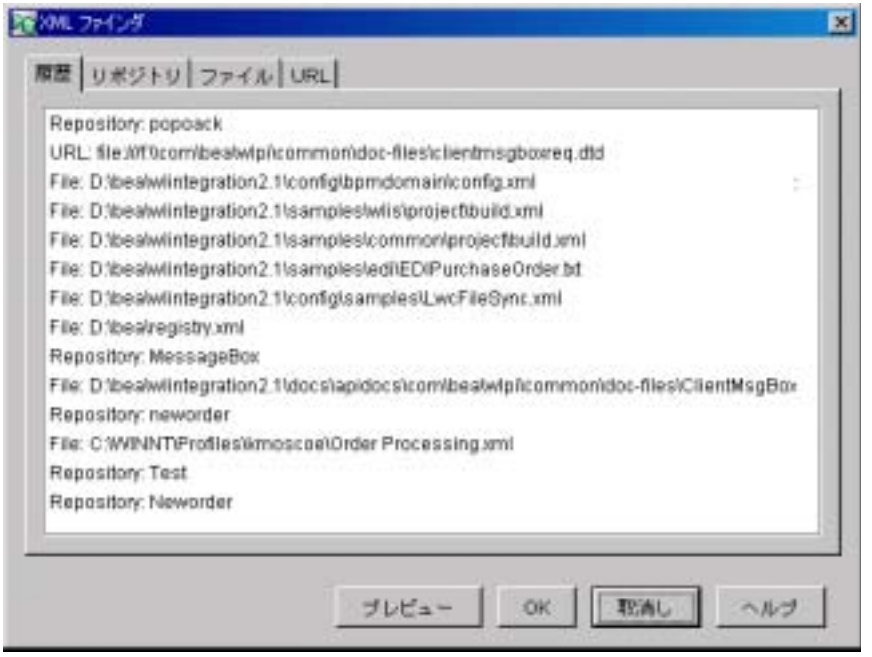
1. 以下のいずれかの操作を行って [XML ファインダ] を開きます。
   - [ワークフロー変数を設定]、[XML をクライアントに送信]、[XML イベントをポスト]、または [例外ハンドラの呼び出し] アクション ダイアログ ボックスで、[インポート]  ボタンをクリックして XML ドキュメントを取り出します。
   - [XPath Wizard] で [開く]  ボタンをクリックします。

   注意 : [履歴] タブは、[XSL 変換] ダイアログ ボックスや、スキーマ ドキュメントを開くことのできるダイアログ ボックスでは使用できません。これらの操作により、実行時に XML ドキュメント インスタン

スが作成されるからです。この場合、実行時に処理エンジンがファイルにアクセスできないため、最近使用したエンティティを指定するのは適切ではありません。

2. [XML ファインダ] ダイアログ ボックス内の [履歴] タブを選択します。ディスク ファイルはファイル名と場所で識別され、リポジトリ エンティティはエンティティ名で識別されます。

図 7-8 [XML ファインダ] : [履歴] タブ



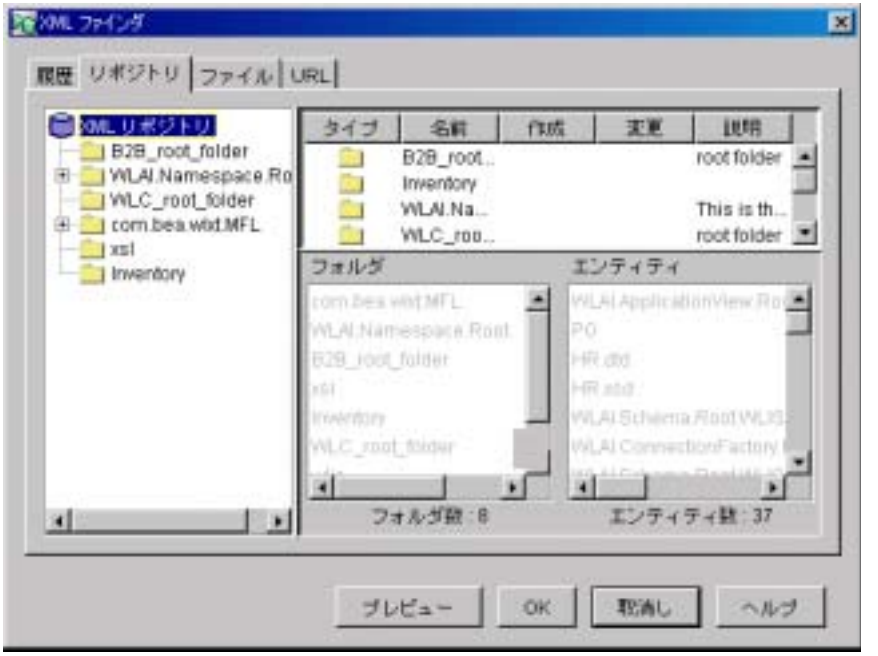
3. 使用するエンティティを選択します。
4. (省略可能) エンティティのコンテンツを確認するには、[プレビュー] をクリックして [ドキュメントのプレビュー] ウィンドウを開きます。[OK] をクリックしてウィンドウを閉じます。
5. エンティティを元のダイアログ ボックスに戻すには、[OK] をクリックします。

リポジトリから取り出す

リポジトリからエンティティを検索する手順は、次のとおりです。

1. 以下のいずれかの操作を行って [XML ファインダ] を開きます。
   - [ワークフロー変数を設定]、[XML をクライアントに送信]、[XML イベントをポスト]、または [例外ハンドラの呼び出し] アクション ダイアログ ボックスで、[インポート]  ボタンをクリックして XML ドキュメントを取り出します。または、[コンテンツ タイプを設定]  ボタンをクリックしてスキーマ ドキュメントを取り出します。
   - [XSL 変換] ダイアログ ボックスで [XML] ボタンをクリックします。
   - [XPath Wizard] で [開く]  ボタンをクリックします。
2. [XML ファインダ] ダイアログ ボックス内の [リポジトリ] タブを選択します。
3. XML エンティティを検索するフォルダを選択します。

図 7-9 [XML ファインダ] : [リポジトリ] タブ



4. 右側の一番上のパネルで、取り出すエンティティを選択します。
5. （省略可能）エンティティのコンテンツを確認するには、[プレビュー] をクリックして [ドキュメントのプレビュー] ウィンドウを開きます。[OK] をクリックしてウィンドウを閉じます。
6. エンティティを元のダイアログ ボックスに戻すには、[OK] をクリックします。

ファイル システムから取り出す

ローカル ディスク ドライブ上のファイル、またはローカル マシンにマップされたあらゆるネットワーク ドライブから、コンテンツを検索できます。

ローカル ファイル システムからエンティティを取り出す手順は、次のとおりです。

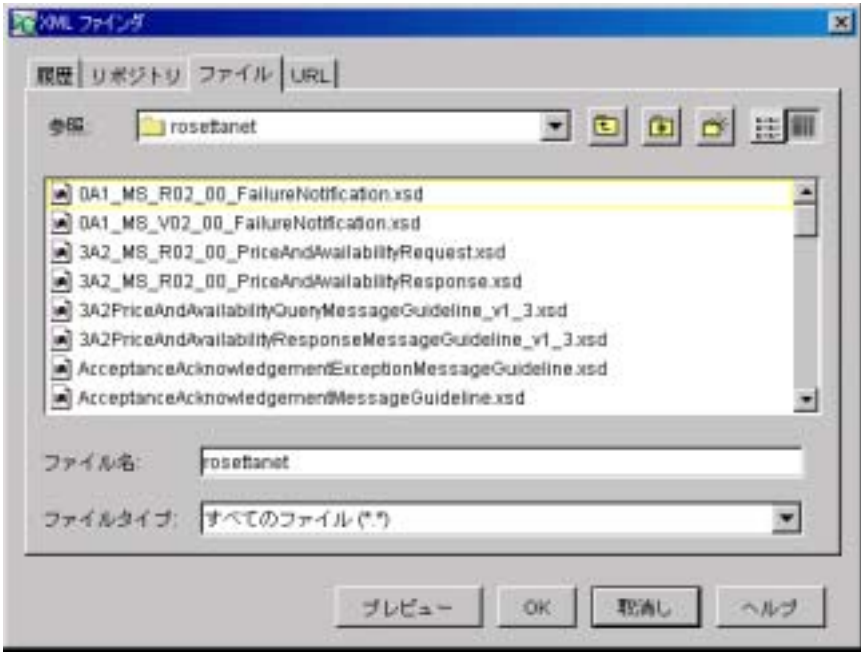
1. 以下のいずれかの操作を行って [XML ファインダ] を開きます。

- [ワークフロー変数を設定]、[XML をクライアントに送信]、[XML イベントをポスト]、または [例外ハンドラの呼び出し] アクション ダイアログ ボックスで、[インポート] ボタンをクリックして XML ドキュメントを取り出します。
- [XPath Wizard] で [開く] ボタンをクリックします。

注意 : [ファイル] タブは、[XSL 変換] ダイアログ ボックスや、スキーマ ドキュメントを開くことのできるダイアログ ボックスでは使用できません。これらの操作により、実行時に XML ドキュメント インスタンスが作成されるからです。この場合、実行時に処理エンジンがローカル ファイル システムにアクセスできないため、ローカル ファイル システム上にファイルを指定するのは適切ではありません。

2. [XML ファインダ] ダイアログ ボックス内の [ファイル] タブを選択します。

図 7-10 [XML ファインダ] : [ファイル] タブ



3. [参照] フィールドに、XML エンティティを検索するドライブおよびフォルダを指定します。必要に応じて、[参照] フィールドの右にあるボタンを使用します。
4. ファイルのリストから、検索するエンティティを選択します。
5. (省略可能) エンティティのコンテンツを確認するには、[プレビュー] をクリックして [ドキュメントのプレビュー] ウィンドウを開きます。[OK] をクリックしてウィンドウを閉じます。
6. エンティティを元のダイアログ ボックスに戻すには、[OK] をクリックします。

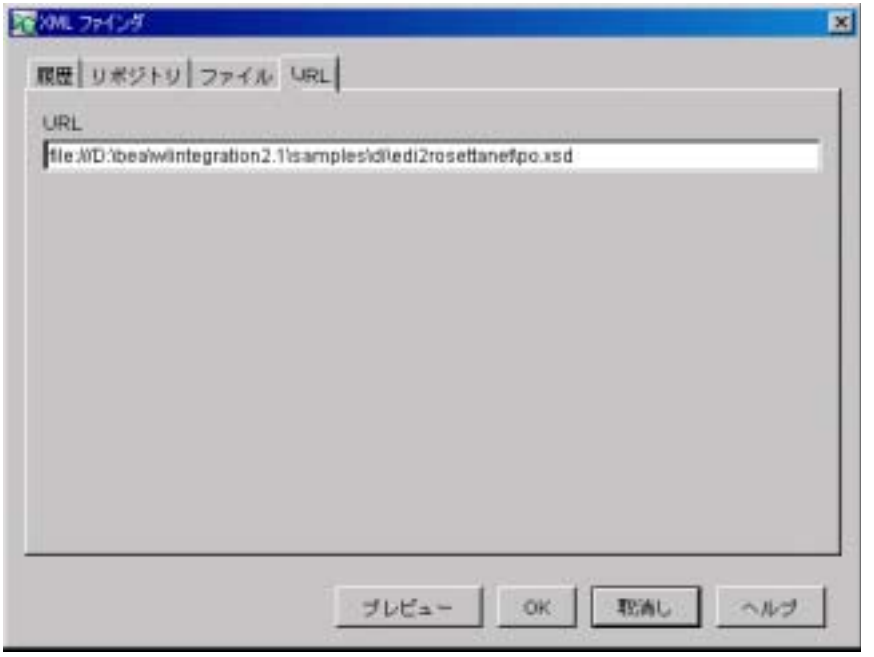
URL から取り出す

URL を使って、ローカル マシン、ネットワーク上のリモート マシン、またはその他の外部システムの場所を指定できます。リモート マシンの場所は、ローカル マシン上のドライブにマップされている必要があります。

URL からエンティティを取り出す手順は、次のとおりです。

1. 以下のいずれかの操作を行って [XML ファインダ] を開きます。
   - [ワークフロー変数を設定]、[XML をクライアントに送信]、[XML イベントをポスト]、または [例外ハンドラの呼び出し] アクション ダイアログ ボックスで、[インポート]  ボタンをクリックして XML ドキュメントを取り出します。または、[コンテンツ タイプを設定]  ボタンをクリックしてスキーマ ドキュメントを取り出します。
   - [XSL 変換] ダイアログ ボックスで [XML] ボタンをクリックします。
   - [XPath Wizard] で [開く]  ボタンをクリックします。
2. [XML ファインダ] ダイアログ ボックス内の [URL] タブを選択します。

図 7-11 [XML ファインダ] : [URL] タブ



3. [URL] フィールドに、検索する XML エンティティのプロトコル、サーバ、パス、およびファイル名を含む完全な URL を入力します。
4. （省略可能）エンティティのコンテンツを確認するには、[プレビュー] をクリックして [ドキュメントのプレビュー] ウィンドウを開きます。[OK] をクリックしてウィンドウを閉じます。
5. エンティティを元のダイアログ ボックスに戻すには、[OK] をクリックします。

XML エンティティをエクスポートする

Studio で作成された XML ドキュメントは以下のダイアログ ボックスで保存できます。

- [ワークフロー変数を設定]、[XML をクライアントに送信]、[XML イベントをポスト] および [例外ハンドラの呼び出し] の各アクション ダイアログ ボックスでは、アクション内に作成した XML ドキュメントを、リポジトリまたはローカル ファイル システムに保存できます。保存したドキュメントは、別のワークフロー アクション、ノード、またはテンプレート定義で再使用できます。
- [ツール | XML ファインダを表示] を選択すると、XML リポジトリに格納されている XML エンティティをファイル システムにエクスポートできます。詳細については、4-27 ページの「リポジトリにあるエンティティの管理」を参照してください。

アクション ダイアログ ボックスからエクスポートされたドキュメントは、標準 XML エスケープ シーケンス（"）でフォーマットされ、要素と属性の値は引用符で囲まれます。コンテンツ型ドキュメントでは、DOCTYPE を含むプロローグ、および有効性を示すエピログも挿入されます。

リポジトリへエクスポートする

XML ドキュメント テンプレートをアクション ダイアログ ボックスからリポジトリにエクスポートすると、エントリが 2 段階の手順で作成されます。最初に、空のエンティティが作成されます。次に、エンティティにドキュメント テンプレートのコンテンツが入力されます。このため、コンテンツが実際に入力されるまで、ドキュメントのプレビューは不可能です。コンテンツの入力は、XML ファインダの終了時に行われます。

アクション ダイアログ ボックスのエンティティをリポジトリに保存する手順は、次のとおりです。

1. [ワークフロー変数を設定]、[XML をクライアントに送信]、[XML イベントをポスト]、または [例外ハンドラの呼び出し] アクション ダイアログ ボックスで、[エクスポート] ボタン  をクリックします。XML ファインダが表示されます。
2. [リポジトリ] タブを選択します。
3. 左ペイン内で、エンティティを保存するフォルダを右クリックし、表示されたメニューから [エンティティを追加] を選択します。[エンティティを追加] ダイアログ ボックスが表示されます。

4. [名前] フィールドに、追加するエンティティに固有な名前を入力します。

図 7-12 [エンティティを追加] ダイアログ ボックス

5. [タイプ] ドロップダウン リストから、追加するエンティティの型を選択します。
6. (省略可能) [説明] フィールドおよび [メモ] フィールドに、そのエンティティに関する説明とメモをそれぞれ入力します。
7. [OK] をクリックします。エンティティが [XML ファインダ] の右上にあるウィンドウに表示されます。
8. [OK] をクリックし、XML ファインダを終了します。リポジトリ内にエンティティが作成されます。

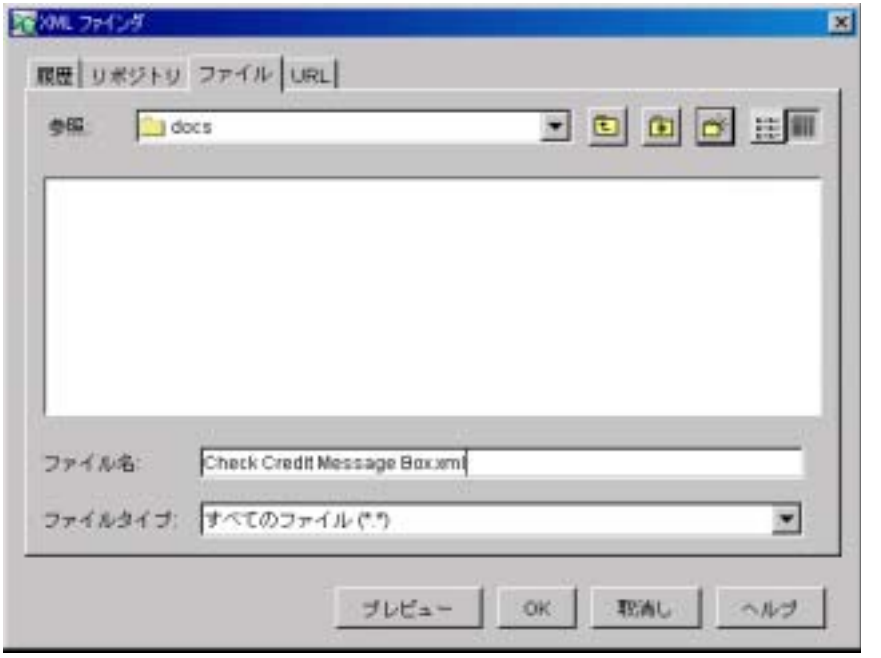
ファイル システムへエクスポートする

アクション ダイアログ ボックスから、ローカル ディスク ドライブまたはローカル マシンにマップされた任意のリモート ドライブ上に、作成または編集した XML ドキュメントを保存できます。ドキュメントは .xml ファイルとして保存されます。

ディスク上のファイルにエンティティを保存する手順は、次のとおりです。

1. [ワークフロー変数を設定]、[XML をクライアントに送信]、[XML イベントをポスト]、または [例外ハンドラの呼び出し] アクション ダイアログ ボックスで、[エクスポート] ボタン  をクリックします。XML ファインダが表示されます。
2. [XML ファインダ] ダイアログ ボックス内の [ファイル] タブを選択します。

図 7-13 [XML ファインダ] : [ファイル] タブ



3. [参照] フィールドに、XML エンティティを保存するドライブおよびフォルダを指定します。必要に応じて、[参照] フィールドの右にあるボタンを使用します。
4. [ファイル名] フィールドに、作成するファイルの名前を入力し、拡張子 `.xml` を追加します。ファイルが既に存在する場合、警告メッセージが表示されます。[はい] をクリックしてファイルを上書きする、[いいえ] をクリックして新しいファイル名を入力する、または [取消し] をクリックしてエクスポートを取り消します。
5. [OK] をクリックし、ファイルを保存して XML ファインダを終了します。

最近アクセスしたファイルへエクスポートする

[履歴] タブを使用して、既存のファイルにエクスポートしたり、既存のファイルを上書きできます。

最近アクセスしたファイルにエンティティを保存する手順は、次のとおりです。

1. [ワークフロー変数を設定]、[XML をクライアントに送信]、[XML イベントをポスト]、または [例外ハンドラの呼び出し] アクション ダイアログ ボックスで、[エクスポート] ボタン  をクリックします。[XML ファインダ] が表示されます。
2. [XML ファインダ] ダイアログ ボックス内の [履歴] タブを選択します。
3. エンティティのリストから、上書きする XML ファイルまたはリポジトリのエンティティを選択します。
4. [OK] をクリックしてファイルを保存し、[XML ファインダ] を終了します。

URL で指定したファイルへエクスポートする

[URL] タブは、URL で指定可能な既存ファイルへのエクスポートおよび上書きのみに使用できます。

URL で場所を指定した既存のファイルにエンティティを保存する手順は、次のとおりです。

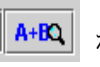
1. [ワークフロー変数を設定]、[XML をクライアントに送信]、[XML イベントをポスト]、または [例外ハンドラの呼び出し] アクション ダイアログ ボックスで、[エクスポート] ボタン  をクリックします。[XML ファインダ] が表示されます。
2. [XML ファインダ] ダイアログ ボックス内の [URL] タブを選択します。
3. [URL] フィールドに、上書きする XML ファイルのプロトコル、サーバ、パス、およびファイル名を含む完全な URL を入力します。
4. [OK] をクリックしてファイルを保存し、[XML ファインダ] を終了します。

8　ワークフロー式の使用法

この章ではワークフローを表現する式の言語を説明し、またワークフロー式を生成する場合の Expression Builder と XPath Wizard の使用法について説明します。

- ワークフロー式の概要
- リテラルの使い方
- 変数の使い方
- 演算子の使い方
- 関数の使用法
- 変数を代入した場合の日付の型変換
- Expression Builder の使い方
- XPath Wizard を使用する XPath 式の作成

ワークフロー式の概要

Studio のダイアログ ボックス内でフィールドの横に [式] ボタン  が表示される場合、そのフィールドにはワークフロー式の言語で定式化したエントリを入力する必要があります。式は、テンプレート定義ラベルの作成、分岐ノードやイベントでの条件の定義、イベント キーの設定、および実行時に提供する情報の指定を行うために、Studio 全体で使用されます。

ワークフロー式は、実行時にシステムによって実行される計算を定義する数式で、文字列、整数、およびその他の定数などのリテラルや、ワークフロー変数、演算子、ワークフロー関数で構成されます。ワークフロー式構文を使って、文字列の操作、関係と条件のテスト、算術計算の実行、実行時にワークフローまたは XML メッセージからの情報を取得する関数の使用などを行うことができます。

式の結果は、文字列、整数、倍精度の値、日付 / 時間の値、またはブール値になります。結果としてブール値を返す式は、条件式または条件と呼びます。

また Studio では、式の作成に役立つ次の 2 つのツールが用意されています。1 つは Expression Builder で、これは構文の検査とエラー情報を提供します（8-28 ページの「Expression Builder の使い方」を参照）。もう 1 つは XPath Wizard で、これはサンプルの XML ドキュメントから XPath 式を生成する場合に使います（8-31 ページの「XPath Wizard を使用する XPath 式の作成」を参照）。

以下の節では、リテラル、演算子、およびワークフローの各変数の構文を示し、また組み込み関数の使用法と構文について説明します。

リテラルの使い方

式には、リテラルの値または定数が含まれています。使用可能なリテラルは次の表のとおりです。

表 8-1 リテラルの使用法

リテラル型	フォーマット	説明	サンプル
文字列	*“string”* または *‘string’*	文字列は一重または二重引用符で囲まれる。文字列に特殊文字を埋め込むには、以下のエスケープ シーケンスを使用する。 \r キャリッジ リターン \n 改行文字 \' 一重引用符 \" 二重引用符 \f 用紙送り文字 \t タブ文字 \\ バックスラッシュ文字 \0 ASCII ヌル文字	“cancelled”

表 8-1 リテラルの使用法

リテラル型	フォーマット	説明	サンプル
整数	[+|-] *digits*	-2,147,483,647 ~ +2,147,483,648 の範囲内の 32 ビット符号付き整数（約 9 桁または精度）。	5000
倍精度の値	[+|-]*digits*[.[*digits*]]	64 ビット IEEE 標準の倍精度浮動小数点数。範囲は $-2^{53} \times 10^{104}$ ～ $+2^{53} \times 10^{104}$ まで (約 15 桁の精度)。	5000.00
日付	“*MM/dd/yyyy hh*:*mm*:*ss* AM|PM GMT[+|-]*hh*:*mm*:*ss*”		"10/01/2001 12:11:11 AM GMT-04:00"

注意 : 日付のリテラルは、日付型の変数を設定するためにのみ使用できます。他のコンテキストで使用する関数や式には、日付のリテラルを使用できません。

変数の使い方

式には、カレント ワークフローで定義したワークフロー変数の参照が含まれます。

ワークフロー変数の参照には、以下の構文のいずれか 1 つを使用できます。

- :*variable*
- :'*variable*'
- :"*variable*"
- $*variable*
- $'*variable*'
- $"*variable*"

演算子の使い方

次の表は、利用可能な演算子の値を示します。

表 8-2 演算子の使用法

演算子	記号	構文	オペランド	結果
AND	AND	*expr1* AND *expr2*	論理オペランド	expr1 と expr2 の両方が True の場合は、True。それ以外の場合は False。
OR	OR	*expr1* OR *expr2*	論理オペランド	expr1 または expr1 と expr2 の両方が True の場合は、True。それ以外の場合は False。
XOR	XOR	*expr1* XOR *expr2*	論理オペランド	expr1 または expr2 の一方のみが True の場合は、True。それ以外の場合は False。
NOT	NOT	NOT *expr*	論理オペランド	expr が False の場合は True。それ以外の場合は False。
（括弧）	()	(*expr*)	任意の式	expr を最初に評価する。
乗算	*	*expr1* * *expr2*	数値オペランド	数値の積。
除算	/	*expr1* / *expr2*	数値オペランド	数値の商。
モジュロ	%	*expr1* % *expr2*	数値オペランド	数値のモジュロ（expr1 を expr2 で除算した時の残り）。
加算	+	+ *expr* *expr1* + *expr2*	数値オペランド 文字列または数値	単項プラス。 文字列の連結または数値の加算。
減算	-	- *expr* *expr1* - *expr2*	数値オペランド 数値オペランド	マイナスの単項演算子（expr を減算する）。 減算。

表 8-2 演算子の使用法

演算子	記号	構文	オペランド	結果
より小さい	<	*expr1 < expr2*	文字列または数値オペランド	expr1 が expr2 より小さい場合は、論理的に True。それ以外の場合は False。
より小さいまたは等しい（以下）	<=	*expr1 <= expr2*	文字列または数値オペランド	expr1 が expr2 以下の場合は、論理的に True。それ以外の場合は False。
等しい	=	*expr1 = expr2*	文字列または数値オペランド	expr1 が expr2 と等しい場合は、論理的に True。それ以外の場合は False。
等しくない	<>	*expr1 <> expr2*	文字列または数値オペランド	expr1 が expr2 に等しくない場合は、論理的に True。それ以外の場合は False。
より大きい	>	*expr1 > expr2*	文字列または数値オペランド	expr1 が expr2 より大きい場合は、論理的に True。それ以外の場合は False。
より大きいまたは等しい（以上）	>=	*expr1 >= expr2*	文字列または数値オペランド	expr1 が expr2 以上の場合は、論理的に True。それ以外の場合は False。

関数の使用法

関数とは組み込みの式のことで、これを使用して実行時にデータを取得できます。関数は、変数の型キャスト、ワークフロー情報の識別、データ演算の実行など、さまざまな目的で使用します。また多くの関数では、特定のワークフローやシステム データを実行時に返す組み込み属性を指定できます。

以下の節では、ワークフロー関数をカテゴリ別にグループ化して示します。それぞれのリスト内では、関数の代表的な属性を丸かっこで囲んで示しています。ただし、関数に他の関数や式を組み込んだり、属性を埋め込み式として表現することもできます。

注意： この節で示すデフォルトの関数の他に、プラグイン関数も使用することができます。

関数の結果を変数に代入するか、または 2 つの関数の結果を比較したい場合は、それぞれのデータの型が一致することを確認する必要があります。したがって、返却値の型も以下の関数の説明に一覧で示します。

実行時のシステム データを収集する

システム情報を実行時に収集するには、次の関数を使用します。

- Date()

Date()

説明	ノードのアクティブ化、またはアクションの実行時など、式が評価された瞬間のシステム日時を返す。
フォーマット	`Date()`
戻り値の型	Java の Date オブジェクト。 戻り値を文字列としてフォーマットし、`Date()` 関数を `DateToString()` 関数に組み込み、フォーマットを指定する。詳細については、8-17 ページの「DateToString()」を参照。

実行時のイベント データを抽出する

ワークフローの JMS メッセージのヘッダまたは XML 本文から内容を抽出する場合、あるいはプロパティを調べる場合に以下の関数を使うことができます。

- EventAttribute()
- EventData()
- XPath()

EventAttribute()

説明	受信する JMS メッセージまたはプラグイン定義イベントから、イベント プロパティを取得する。
フォーマット	`EventAttribute(`*`expression`*`)` *`expression`* は、JMS ヘッダまたはプロパティ フィールドの名前を返す式。JMS ヘッダおよびプロパティ フィールドについては、次の URL にある『*WebLogic JMS プログラマーズ ガイド*』の「WebLogic JMS の基礎」の「Message」を参照。 http://edocs.beasys.co.jp/e-docs/wls/docs70/jms/fund.html **注意：** `EventAttribute()` は、受信した JMS メッセージが消費される、以下のコンテキストのみで使用します。 ■ イベント ノード ■ イベントによってトリガされる開始ノード ■ [イベント キー] コンフィグレーション
サンプル	`EventAttribute("JMSDestination")`
戻り値の型	Java オブジェクト

EventData()

説明	受信 JMS メッセージの実際の内容、またはプラグインが定義されたイベントを検索する。内容は XML ドキュメントの場合もある。 **注意：** `EventData()` は、受信 JMS メッセージが使用される、以下のコンテキストのみで使用。 ■ イベント ノード ■ イベントによってトリガされる開始ノード ■ [イベント キー] コンフィグレーション
フォーマット	`EventData()`
戻り値の型	Java オブジェクト

XPath()

説明	XML ドキュメントから内容を抽出する。

フォーマット	XPath(“*xpathstring*”, [,*xmldocument*]) *xpathstring* は、XPath 言語の式。 *xmldocument* は、XPath 式の評価対象となる XML ドキュメントを取得する式。これは、有効な XML ドキュメントのソース テキストを含む文字列、または XML テキストを含む XML または文字列変数への参照（この方が一般的）を指す。このパラメータは省略できる。これを指定しない場合、XML ドキュメントが着信 XML イベントと見なされる。 **注意：** XML ドキュメントを含む変数を指定しなかった場合、XPath() 関数は、受信する XML ドキュメントのアイデンティティが判明している以下のコンテキストの中でのみ使用できる。 ■ イベント ノード ■ イベントによってトリガされる開始ノード ■ イベント キー コンフィグレーション ■ XML をクライアントに送信アクション ■ 例外ハンドラの呼び出しアクションから XML ドキュメントを受け取る例外ハンドラ。
サンプル	以下を参照。
戻り値の型	DOM オブジェクト 通常は Node List を返すワークフローの XPath 関数式の中で、XPath 言語の text() 関数を使用する（詳細は以下を参照）。ただし、他の XPath の関数は、倍精度の値、ブール値、文字列、または整数の型を返すことがある。

XPath は、XML ドキュメントの部分をアドレッシングする言語です。これは文字列、数字、およびブール値を扱う基本的な機能を提供します。XPath は、XML ドキュメントの階層構造でナビゲートするための URL のパス名から名前を取得します。XPath 言語の公式な仕様を入手するには、次のインターネット サイトを参照してください。http://www.w3.org/TR/xpath.html

XPath の表記法や関数のクイック リファレンス ガイドを入手するには、次のインターネット サイトを参照してください。
http://www.mulberrytech.com/quickref/XSLTquickref.pdf

各ノードからのテキスト値の取り出し、属性値の取り出し、サブツリーの選択、属性値などによる XML ドキュメントからのノードの選択など、XPath の最も一般的な使用サンプルを次に示します。リスト 8-1 の XML ドキュメントを検証してください。

コード リスト 8-1 XML サンプル ドキュメント

```
<?xml version="1.0"?>
<a>
  <b name="bill">This is the first value</b>
  <c>
    <d id="d1">This is the second value</d>
    <d id="d2">This is the third value</d>
    <d id="d3">This is the fourth value</d>
    <d id="d4">This is the fifth value</d>
  </c>
</a>
```

<b> 要素のテキスト値（"This is the first value" など）を選択する場合 :

```
XPath("/a/b/text()")
```

ID 属性が "d3" の値を持つ <d> 要素（"This is the fourth value" など）を選択するには次のコードを使います。

```
XPath("/a/c/d[@id=\"d3\"]/text()")
```

2 番目の <d> 要素のテキスト値（"This is the third value" など）を選択するには次のコードを使います。

```
XPath("/a/c/d[2]/text()")
```

<b> 要素の名前属性の値（"bill" など）を選択するには次のコードを使います。

```
XPath("/a/b/@name")
```

<c> サブツリー全体を選択するには次のコードを使います。

```
   <c>
       <d id="d1">This is the second value</d>
       <d id="d2">This is the third value</d>
       <d id="d3">This is the fourth value</d>
```

```
        <d id="d4">This is the fifth value</d>
    </c>)
    XPath("/a/c")
```

注意： XPath 式は、ユーザ自身がキー入力するか、または XPath Wizard を使って作成できます。Wizard の使用法の詳細は、8-31 ページの「XPath Wizard を使用する XPath 式の作成」を参照してください。

XML 要素のドット表記

変数の初期化の際に、[開始のプロパティ]、[イベントのプロパティ]、[例外ハンドラのプロパティ]、および [XML をクライアントに送信] でより簡素なドット（.）表記を使用して XML ドキュメントから要素データを取得することができます。それには、次の構文を使います。

root_element.subelement1.subelement2.subelement3 . . .

XML ドキュメントの例としては次のようなものがあります。

コード リスト 8-2 XML ドキュメントの例

```
<account>
       <number>847365</number>
       <customer>John Doe</customer>
       <balance>
              <status>past due</status>
              <date_due>7-11-2001</date_due>
              <amount_due>5670.85</amount_due>
       </balance>
       <credit_limit>7500.00</credit_limit>
</account>
```

この例では、期限切れの値を検索するのに、次の式を使用します。

```
account.balance.status
```

この表記を使って属性値を取得することはできません。また、式の先頭がルート要素である必要があります。サブ要素で式を始めることはできません。

この表記では常に文字列が返されます。型キャストを実行する関数と併用したとしてもその他のデータ型を返すことはできません。

実行時のワークフロー データを収集する

実行時のデータをワークフローから収集するには、以下に一覧で示す関数を使用します。

- CurrentUser()
- TaskAttribute()
- WorkflowAttribute()
- WorkflowVariable()

CurrentUser()

説明	現在タスクを実行しているユーザの ID (ユーザ名) を返す。
フォーマット	`CurrentUser()`
戻り値の型	文字列

TaskAttribute()

説明	ワークフローのタスクに関する情報を提供する。
フォーマット	`TaskAttribute(“attribute” [,”taskname”])` *attribute* は、タスクの属性を示す。以下の属性が使用できる。

関数の属性	情報	戻り値の型
`TaskId`	システム定義のタスク インスタンス ID	文字列

Assignee	ユーザまたは assignee (受託者) の役割 ID	文字列
Priority	ユーザが設定した優先順位	整数
Due	タスクの期限経過日	日付
Name	タスクの名前	文字列
Started	タスクの開始日 / 時間	日付
Completed	完了日 / 時間	日付
Comment	ユーザが設定したタスクのコメント	文字列
	taskname を使って、式が定義された現在のタスク以外のタスクを指定できる。 **注意 :** 指定できるのは同じワークフロー内のタスクのみです。	

注意 : 上記の表に示す属性以外のものを使用すると、サーバで例外が発生します。

WorkflowAttribute()

説明	現在のワークフローのみの情報を表示する。
フォーマット	WorkflowAttribute(“*attribute*”) *attribute* はワークフローの属性である。以下の属性が使用できる。

関数の属性	**情報**	**戻り値の型**
InstanceId	システム定義のワークフロー インスタンス ID	文字列
TemplateId	システム定義のワークフロー テンプレート ID	文字列
TemplateDefinitionId	システム定義のワークフローのテンプレート定義 ID	文字列

Initiator	ワークフロー イニシエータ	文字列
ParentId	呼び出されたワークフロー インスタンスの場合、親ワークフロー インスタンスのシステム定義のワークフロー インスタンス ID	文字列
Name	ユーザが設定したテンプレート名	文字列
Started	開始日 / 時間	日付
Completed	完了日 / 時間	日付
Label Id	ユーザが設定したテンプレート定義ラベル。	文字列
Comment	ユーザが設定したワークフローのコメント	文字列
ExceptionType ErrorType	エラーを発生した Java 例外クラスの名前。	文字列
ExceptionNumber ErrorNumber	例外ハンドラで処理された、エラーのメッセージ番号。 ワークフローのエラー メッセージをエラー番号やテキストで分類したリストについては、9-15 ページの「システム エラー メッセージ」 を参照。	整数

ExceptionSeverity ErrorSeverity	重大度は、以下の 5 つの値のいずれかで示される。 ■ 0: エラー タイプは不明。内部使用専用。 ■ 1: ユーザ要求の処理時に致命的な例外が発生した。 ■ 2: ワークフロー ステータスの不一致など、致命的で違法な条件が発生した。 ■ 3: ユーザが手作業で修正できる、致命的でないワークフロー条件が発生した。 ■ 4: WorkflowProcessor.invokeWorkfowErrorHandler を呼び出すアプリケーション、または例外ハンドラを呼び出しアクションを実行するワークフローでカスタム エラーが発生した。 **注意：** ワークフローが例外ハンドラを呼び出しアクションを介すか、または API を介して例外ハンドラを呼び出した場合、この値は 4 に設定されます。	整数
ExceptionText エラー メッセージ本文	例外ハンドラで処理された、エラーのメッセージ テキスト。 ワークフローのエラー メッセージをエラー番号やテキストで分類したリストについては、9-15 ページの「システム エラー メッセージ」を参照。	文字列
ExceptionObject ErrorObject	例外ハンドラで処理される、例外オブジェクト。	例外オブジェクト

注意： 上記の表に示す属性以外のものを使用すると、サーバで例外が発生します。

WorkflowVariable()

説明	特定のワークフロー インスタンスに関するワークフロー変数の値を返す。

フォーマット	`workflowVariable(`*`instanceid`*`, `*`variable`*`)` *`instanceid`* は、ワークフロー インスタンスの ID で、通常は別のワークフローから送信された `WorkflowAttribute ("InstanceID")` 関数によって値が設定される、文字列型の変数として表現される。 *`variable`* は、ワークフロー変数。フォーマットの詳細は、8-3 ページの「変数の使い方」を参照。
サンプル	`workflowVariable($InstanceID), $ItemQuantity)`
戻り値の型	オブジェクト

データ型を変換する

以下の関数を使用して、1 つの式の戻り値の型を別のものに変換できます。ワークフローの型の変換ルールについては、8-24 ページの「変数を代入した場合の日付の型変換」を参照してください。

- DateToString()
- StringToDate()
- ToInteger()
- ToString()

DateToString()

説明 日付を文字列に変換する。

フォーマット *dateToString(“date”, “format”)*

date は文字列の値に変換する日付を表す。日付は Java の日付オブジェクトとして表現しなければならず、したがって日付を指定する場合は以下の関数を組み込む。

- 式を評価する際に、現在の実行時の日付および時間を指定する場合は、`Date()` を使用する。8-6 ページの 「Date()」 を参照。
- カレント ワークフローの開始または終了時に、実行時の日付および時間を指定する場合は、`WorkflowAttribute(“Started”)` または `WorkflowAttribute(“Completed”)` を使用する。
- カレント タスクが呼び出されるか終了する実行時の日付および時間を指定する場合は、`TaskAttribute(“Started”)` または `TaskAttribute(“Completed”)` を使用する。

format は文字列のフォーマットを指定する文字列を表す。以下は指定可能な値である。

注意： フォーマットで指定する文字は、大文字と小文字が区別されます。

フォーマット	説明	サンプル
yyyy	年	2000
MM	月	1 月 = 01、2 月 = 02、
dd	日	02 または 28
DD	1 年の日数（365 日）をベースにした、3 桁の日付	2000 年 10 月 18 日 = 292（1 月 1 日から数えて 292 日目）
hh	時間（12 時間制）	午後 1 時 = 01、午後 2 時 = 02
HH	時間（24 時間制）	午後 1 時 = 13、午後 2 時 = 14
mm	分	02
ss	秒	35
SSS	ミリ秒	370

次のセパレータ文字、`-`（ハイフン）、`/`（スラッシュ）、`:`（コロン）、`.`（ピリオド）、スペースが有効です。

サンプル	DateToString(Date(), “yyyy-MM-dd HH:mm:ss.SSS”) は次のようになる 2000-10-18 14:30:35.370
戻り値の型	文字列

StringToDate()

説明	文字列を日付に変換する。
フォーマット	StringToDate(“*string*”, “*format*”) *string* は、日付の値に変換する文字列を表す。文字列のフォーマットは、2 番目の引数で指定したフォーマットに従う必要がある。 *format* は、日付のフォーマットを指定する文字列を表す。利用可能なフォーマットについては、8-22 ページの「日付関数のフォーマット」 を参照。
サンプル	StringToDate(“2001.09.10”, “yyyy.MM.dd”)
戻り値の型	日付

ToInteger()

説明	文字列値を整数に変換する。 **注意：** 文字列は、有効な整数値を表す必要がある。
フォーマット	ToInteger(*expression*) *expression* は、整数に変換する式（二重引用符で囲んだ文字列）。
サンプル	ToInteger(ToString(XPath(“/item/quantity/text()”)))
戻り値の型	整数

ToString()

説明	任意のデータ型を文字列に変換する。
フォーマット	`ToString(`*`expression`*`)` *`expression`* は、文字列に変換する式。
サンプル	`ToString($TotalPrice)`
戻り値の型	文字列

データを処理する

データに対して各種のオペレーションを行う場合、以下に示す関数を使用できます。

- Abs()
- DateAdd()
- StringLen()
- SubString()

Abs()

説明	式の絶対値を返す。
フォーマット	`Abs(`*`expression`*`)` *`expression`* は、絶対値を計算するワークフロー式を表す。
サンプル	`Abs($ItemPrice * $ItemQuantity)`
戻り値の型	整数、倍精度の値、または文字列（入力値による）

DateAdd()

説明 日付計算を実行する。

フォーマット DateAdd(*date, "interval", number* [,*business calendar name*])

date は、基本の日付を表す。日付は Java の日付オブジェクトを返却しなければならず、したがって日付を指定する場合は以下の関数を組み込む。

- 式を評価する際に、現在の実行時の日付および時間を指定する場合は、Date() を使用する。8-6 ページの「Date()」を参照。
- カレント ワークフローの開始または終了時に、実行時の日付および時間を指定する場合は、WorkflowAttribute("Started") または WorkflowAttribute("Completed") を使用する。
- カレント タスクが呼び出されるか終了する実行時の日付および時間を指定する場合は、TaskAttribute("Started") または TaskAttribute("Completed") を使用する。
- 定数の基本日付を指定する場合は、適切なフォーマットで StringToDate() 関数を使用する。8-18 ページの「StringToDate()」を参照。フォーマットの詳細は、8-22 ページの「日付関数のフォーマット」を参照。

interval は、使用する時間の単位を表す。以下は、指定可能な値である。

注意： 時間の単位は、m と M 以外は大文字と小文字を区別しません。

インターバル	説明
S	秒
m	分
H	時間
D	日
W	週
M	月
BH	ビジネス時間
BD	ビジネス日

number は、加算または減算を行う単位の数を示す整数である（符号付も可能）。

business calendar name は、ビジネス時間およびビジネス日の計算で使用するビジネス カレンダーの名前である。この値を指定しないと、デフォルトのカレンダーが使用される。

サンプル	`DateAdd(Date(), D, 7, mycalendar)`
戻り値の型	日付

StringLen()

説明	文字列の長さの値を返す。
フォーマット	`stringlen(`*`expression`*`)` *`string`* は、長さを示す文字列または式を表す。
サンプル	`stringlen(TaskAttribute("Comment"))`
戻り値の型	整数

SubString()

説明	文字列からサブ文字列を抽出する。
フォーマット	`SubString(`*`expression, start`* `[,`*`length`*`])` *`string`* は、下位文字列を抽出する文字列を表す。 *`start`* は、文字列内の開始位置を表す（最初の位置は、0）。 *`length`* は下位文字列の長さを表す。このパラメータは省略できる。サブ文字列が文字列に収まらない場合、開始文字から最後の文字まで抽出される。
サンプル	`SubString(TaskAttribute("Comment"), 0, 50)`
戻り値の型	文字列

日付関数のフォーマット

日付関数の日付フォーマットを指定するには、*タイム パターン* 文字列を使用して日付と時間を記述します。たとえば、DateToString() 関数では、次のようなタイム パターン文字列が使用されています。

```
"yyyy.MM.dd G 'at' hh:mm:ss z"
```

結果は、次のようなフォーマットになります。

```
2000.07.31 AD at 13:10:35 PDT
```

同様に、StringToDate() 関数を使用する場合は、ユーザが指定するフォーマットに応じて日付のリテラルを書式指定しなければなりません。

ここではパターンの文字を説明し、書式指定の指針を示し、また指定サンプルを示します。

表 8-3 パターンの文字の定義

記号	説明	フォーマット	サンプル
G	紀元前 / 後の表示	*Text*	AD
y	年	*number*	1996
M	月	*月 & 01 - 12*	July & 07
d	日	*01 - 31*	10
h	午前 / 午後の時間	*1 - 12*	12
H	時	*0 - 23*	0
m	分	*number*	30
s	秒	number	55
S	ミリ秒	*number*	978
E	曜日	*曜日*	Tuesday
D	1 月 1 日からの積算日数	*number*	189
F	1 か月の中の何週目	*number*	2 (7 月の 2 週目の水曜日)

表 8-3 パターンの文字の定義

記号	説明	フォーマット	サンプル
w	1 年の中の何週目	*number*	27
W	1 か月の中の何週目	*number*	2
a	午前 / 午後（am/pm）マーカー	AM または PM	PM
k	時	*1 - 24*	24
K	午前 / 午後の時間	*0 - 11*	0
z	時間帯	*短縮形のテキスト*	PST
'	テキストのエスケープ文字		'at'
''	一重引用符	'	'Wednesday' 's'

フォーマットの指針

フォーマットはパターン文字の数によって決定されます。

- テキストの場合－ 4 またはそれ以上のパターン文字の場合は、フル フォームを使用。3 文字以下の場合で、0 でない場合は、ショート フォームまたは短縮形フォームを使用します。
- 数字の場合－最小桁数。桁数が少ない場合は、ゼロが埋め込まれます。年は特別な処理が行われます。つまり、'y' の数値が 2 の場合、年は 2 桁に切り捨てられます。
- *テキストまたは数字*の場合－パターン文字の数が 3 文字以上の場合は、テキストを使用します。それ以外の場合は、数字を使用します。

パターン内のいずれかの文字が ['a'..'z'] と ['A'..'Z'] の範囲外の場合、その文字は引用符で囲まれたテキストとして扱われます。たとえば、':', '.', ' ', '#' および '@' などの文字は、一重引用符で囲まれていなくても、結果は時間のテキストで示されます。

無効なパターン文字を含むパターンは、フォーマット処理またはパーシングの結果例外が発生します。

時間パターンのサンプル

表 8-4 米国のロケールを使用したサンプル

フォーマット パターン	結果
`"yyyy.MM.dd G 'at' hh:mm:ss z"`	1996.07.10 AD at 15:08:56 PDT
`"EEE, MMM d, ''yy"`	Wed, July 10, '96
`"h:mm a"`	12:08 PM
`"hh 'o''''clock' a, zzzz"`	12 o'clock PM, Pacific Daylight Time
`"K:mm a, z"`	0:00 PM, PST
`"yyyyy.MMMMM.dd GGG hh:mm aaa"`	1996.July.10 AD 12:08 PM

変数を代入した場合の日付の型変換

式を実行した結果の値をワークフローの変数に代入する場合があります。たとえば、XML ドキュメントから XPath 式で返されたデータを使って、ワークフローの起動時に変数を初期設定する場合です。また、1 つの変数の型の値を別のものに割り当てたい場合があります。

式の戻り値の型が、式の結果の値を割り当てたい変数の型と異なる場合、サーバは自動的に型の変換を行います。ただし、型の変換を行うことができない場合もあり、その場合は例外が発生します。次の表は、型の変換を行う方法を示します。表中の * の数は、表の後の「メモ」の項目番号に対応しています。

注意： 以下の表で示す型の変換ルールは、データ型のデフォルト設定値に適用されます。ヌル値のサポートに用いる変換ルールについては、『*WebLogic Integration の起動、停止およびカスタマイズ*』の「WebLogic Integration のカスタマイズ」にある「Null 変数をサポートする BPM のコンフィグレーション」を参照してください。

表 8-5 データの型の変換ルール

評価する元の式	ターゲットの変数の型					
	文字列	整数	倍精度の値	日付	Boolean	XML
文字列	文字列	整数 *	倍精度の値 *	日付 *	ブール値 **	メモの ***** を参照。
整数	文字列	整数	整数の倍精度値	NS	整数の値が 0 の場合は False、それ以外の値の場合は True。	NS
倍精度の値	文字列	倍精度の整数の値	倍精度の値	NS	倍精度の値が 0.0 の場合は False、それ以外の値の場合は True。	NS
日付	文字列	NS	NS	日付	NS	NS
Boolean	「True」または「False」	True の場合は 1、False の場合は 0。	True の場合は 1.0、False の場合は 0.0。	NS	ブール値の True または False。	NS
XML	文字列にシリアライズされた DOM	整数 ***	倍精度の値 ***	日付 ***	ブール値 ****	XML
ノードリスト	文字列にシリアライズされた DOM	整数 ***	倍精度の値 ***	日付 ***	ブール値 ****	XML ノード ***

NS = サポートされません（Not supported）。変換を行おうとすると、型の変換が違法なことを示すエラー メッセージが表示されます。

ノード リスト = XPath の text() 関数で返されるオブジェクトです。

* ソースの値が、ターゲットのデータの型に対して有効な入力でない場合、型の変換が違法なことを示すエラー メッセージが表示されます。

** 文字列の値が「True」、「t」、または「1」の場合は True です。文字列の値が「False」、「f」、「0」の場合は False です。文字列の値がそれ以外の場合は、型の変換が違法なことを示すエラー メッセージが表示されます。

*** 文字列にシリアライズされ、その後でターゲットのデータの型に変換される DOM オブジェクトです。ソースの値が、ターゲットのデータの型に対して有効な入力でない場合、型の変換が違法なことを示すエラー メッセージが表示されます。

**** 文字列にシリアライズされる DOM オブジェクトです。文字列の値が「True」、「t」、または「1」の場合は True です。文字列の値が「False」、「f」、「0」の場合は False です。文字列の値がそれ以外の場合は、型の変換が違法なことを示すエラー メッセージが表示されます。

***** XML ドキュメントの要素に解析する DOM で、単なるノードではありません。これは、文字列に再変換した場合に、文字列に以下のような XML のヘッダが含まれることを意味します。`<?xml version=”1.0” encoding=”UTF-8”?>`. ソースの値が、DOM の解析に有効でない場合、型の変換が違法なことを示すエラー メッセージが表示されます。

これらの変換ルールは式の計算を完全にサポートしていないことに注意してください。たとえば、次の式を仮定します。

```
XPath(“your_expression”) + 2
```

your_expression のデータの型が不明なため、この文の結果は計算できません。この文を有効にするには、型キャスト関数を使用して、ユーザが希望するデータの型に結果を変換する必要があります。

たとえば、*your_expression* の評価が 3 の値になると仮定し、また結果を文字列のデータ型にすると仮定します。その場合は、次の文を使用する必要があります。

```
ToString(XPath(“your_expression”) + 2
```

この場合、式の評価は `52` の文字列の値になります。

たとえば、*your_expression* の評価が 3 の値になると仮定し、また結果を整数にすると仮定します。その場合は、次の文を使用する必要があります。

```
ToInteger(ToString(XPath("your_expression")) + 2
```

この場合、式の評価は 7 の整数値になります。

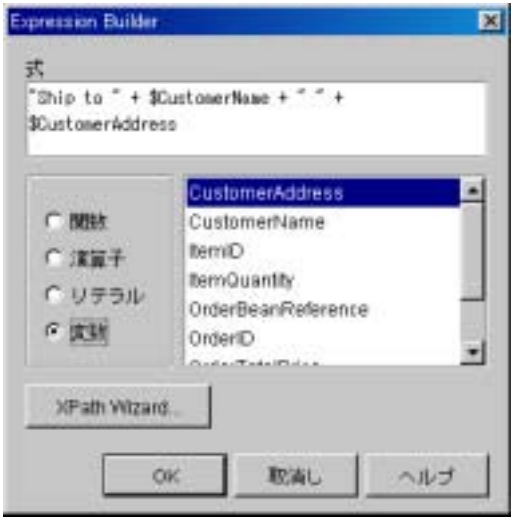
Expression Builder の使い方

Expression Builder は、関数、演算子、リテラルおよび変数から成る式の作成に役立ちます。Studio 全体を通して、次の図に示す [Expression Builder] ボタンがダイアログ ボックスの各フィールドに表示されます。

図 8-1 [Expression Builder] ボタン

[Expression Builder] ダイアログ ボックスを表示するには、このボタンをクリックします。

図 8-2 [Expression Builder] ダイアログ ボックス



注意： 式の一部になる関数に対してプラグインが定義されている場合、このダイアログ ボックスには新しい関数が含まれます。

式を作成する手順は次のとおりです。

1. 次の操作を行います。
   - 式または式のコンポーネントを、[式] フィールドに直接キー入力します。
   - [関数]、[演算子]、[リテラル] または [変数] オプションから選択し、スクロール可能リストの右のコンポーネントをダブルクリックして、選択した式のテキストを [式] フィールドに入れます。場合によっては、挿入された項目にプレースホルダが含まれることもあります。正しい情報を入力する、またはコンポーネントのリストから追加コンポーネントを挿入して、プレースホルダを置き換えます。
   - （省略可能）XPath の関数の場合、XPath Wizard をクリックして XPath Wizard を開き、XML のサンプル ドキュメントから XPath 関数の式を自動的に生成することができます。Wizard の使用法の詳細は、8-31 ページの「XPath Wizard を使用する XPath 式の作成」を参照してください。
2. [式] フィールド内の式が完成した後、[OK] をクリックして式をダイアログ ボックスの元のフィールドに戻します。

無効な式を入力した場合、式が無効な理由を説明するメッセージが表示されます。表示されることのあるメッセージについて、考えられる原因を以下の表に示します。

表 8-6 無効な式に対するメッセージ

メッセージ	主な原因
無効な *operator* オペランドです	AND、OR、XOR、NOT、*、/、%、+、-、<、<=、=、<>、>=、> 演算子に対する一方または両方のオペランドが有効でない。たとえば、次のような非数値の文字列に対して算術演算を行おうとした場合 :`"Name: " * 25.4`
比較オペランド *operator*、*operator* が一致しません	異なる型の値の比較を試みた。たとえば、次のように文字列と数値を比較した場合。`“mystring” <= 56.9`
NumberFormatException	数字の文字列表現で、数字の型に必要なフォーマットを使っていない。たとえば、`1.23ZX`
関数名 "function" が認識されません	未定義の関数を式で呼び出している。たとえば、`"Name: " + somefunc()` は、有効な関数名ではない。
不完全なエスケープ シーケンス	引用符で囲んだ文字列内の最後の文字がバックスラッシュ文字で、その後に続く文字が次のどれにも該当しない。r、n、f、t、\、'、"、0
無効なエスケープ シーケンス	引用符で囲んだ文字列にバックスラッシュ文字が含まれ、その後に続く文字が次のどれにも該当しない。r、n、f、t、\、'、"、0
Fatal error	式の評価子で内部エラーが発生した。式には構文エラーが含まれると考えられる。
文字列 "string" が閉じていません	文字列のリテラルで、終りの引用符が一致していない。次のサンプルでは、前が二重引用符で後が一重引用符になっており、引用符が一致していない。`"This is an unclosed string'`
第 n 行に不正な文字「x」があります	式に、式言語の構文に違反する文字が含まれる。次の例では、!が無効な文字。`7 * $wks + " days" !`

表 8-6 無効な式に対するメッセージ

メッセージ	主な原因
Error:unmatched input	評価子が式を解釈できない。式には構文エラーが含まれると考えられる。
構文エラー	式に、無効なトークンが含まれている。次の例では、+ 演算子が省略されているため、名前は無効。`"Name: " $name` 正しい式は、次のとおり。`"Name: " + $name`

Expression Builder は、次のそれぞれについて有効性は調べません。

- 式で参照するワークフロー変数のデータの型。
- 関数に渡されるかっこの正しい数。
- 型の一致。式の中で型の不一致がある場合（たとえば、数値でない文字列に対して数値演算を行った場合など）、式を評価する際に実行時エラーが発生します。
- 関数の属性。ある関数に対して有効でない属性を使用した場合、実行時の例外が発生します。

XPath Wizard を使用する XPath 式の作成

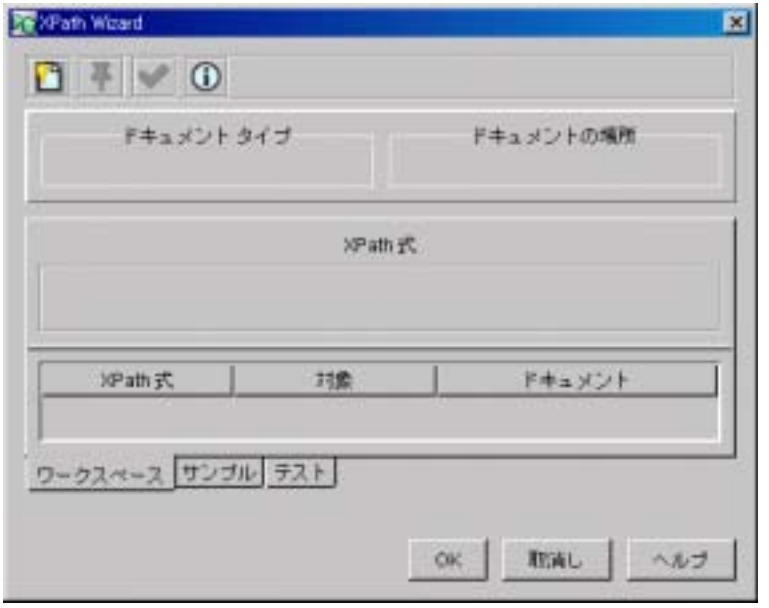
XPath Wizard のグラフィカル インタフェースを使って、実際の XML エンティティから自動的に XPath 式を生成したり、XML エンティティの定義と検証を行う XPath 式を定義およびテストできます。XPath 式を生成してテストした後でその式を Expression Builder に返すことにより、ワークフロー構文に準拠するために追加する必要がある引数を Expression Builder に追加できます。

XPath Wizard は、サンプルの XML ドキュメント、つまり Document Type Definition (DTD)、XML Schema Document (XSD)、Extensible Stylesheet Language (XSL) または Message Format Language (MFL) ファイルをサンプリングできます。ユーザが DTD、Schema、XSL、または MFL ファイルを Wizard に

ロードすると、ユーザが選択したファイル内の仕様に基づいて、サンプルの XML ドキュメントが自動的に生成されます。生成された XML ドキュメントを使用して、XPath 式を作成します。

[XPath Wizard] を開くには、[Expression Builder] から XPath Wizard をクリックします。

図 8-3 XPath Wizard



[XPath Wizard] には、[ワークスペース] タブ、[サンプル] タブ、および [テスト] タブの 3 つのタブが含まれています。これらは次を行うために使用します。

- [ワークスペース] タブー Studio セッション中に作成したすべての XPath ロケーション式と関数を表示します。これらの式と関数は、[テスト] タブで 1 つずつ選択してテストする、または [Expression Builder] に返します。8-36 ページの「XPath 式を表示する」を参照。
- [サンプル] タブーサンプルの XML、DTD、MFL、および Schema ドキュメントに対して XPath 式を生成します。手順の詳細は、8-34 ページの「XPath ロケーション式を XML エンティティから生成する」を参照してください。

- [テスト] タブ― XPath 式を XML ドキュメントに適用してテストします。8-38 ページの 「XPath 式をテストする」 を参照。

これらの 3 つのタブには、ツールバーと以下の 2 つの読み取り専用フィールドが表示されます。

- ドキュメント タイプ―現在ロードされているドキュメントが型定義と関連付けられているかどうかを示します。ロードされたファイルが DTD または XSD の場合、パスとファイル名が与えられます。その他のドキュメントの場合は、ファイルの型や ID が表示されません。
- ドキュメントの場所―現在ロードされているドキュメントがシステムで生成されたサンプル ドキュメントかどうかを示します。ロードされたファイルが XML、MFL または XSL ドキュメントの場合、パスとファイル名が与えられます。

ツールバーのボタンについて以下の表で説明します。

表 8-7 [XPath Wizard] ツールバー ボタン

ボタン	説明
	XML エンティティを XPath Wizard にロードする。詳細については、8-34 ページの 「XPath ロケーション式を XML エンティティから生成する」 を参照。
	XML 式を [ワークスペース] に固定する。詳細については、8-34 ページの 「XPath ロケーション式を XML エンティティから生成する」 を参照。
	選択した XML 式を、XML エンティティに対してテストする。詳細については、8-38 ページの 「XPath 式をテストする」 を参照。
	XPath Wizard に関する情報を表示する。

XPath ロケーション式を XML エンティティから生成する

Xpath Wizard を使用して、以下をターゲットとする XPath ロケーション式を生成できます。

- 要素コンテンツ
- 属性コンテンツ
- ノード

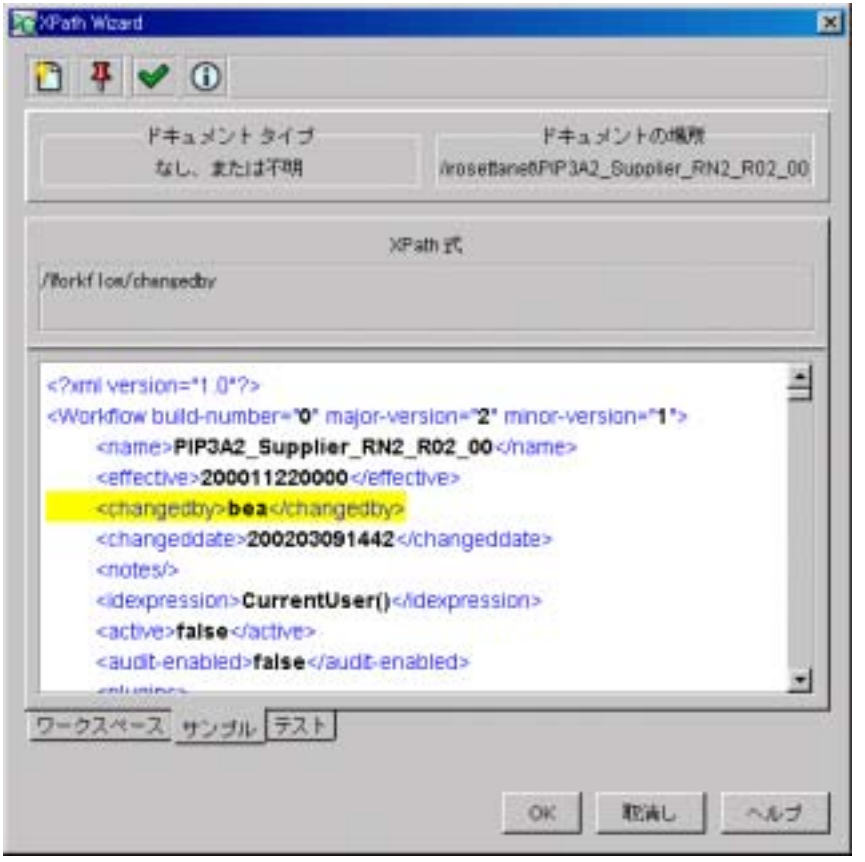
XML§MFL、XSL、DTD または XSD ドキュメントをサンプリング用にロードできます。ユーザが DTD または XSD ドキュメントを開くと、Wizard はサンプルの XML ドキュメントを、ユーザが選択した DTD または Schema から生成します。生成された XML ドキュメントを使用して、XPath 式を作成します。

XPath ロケーション式を生成する手順は、次のとおりです。

1. [Expression Builder] ダイアログ ボックスで [XPath Wizard] をクリックし、[XPath Wizard] を表示します。
2. [XPath Wizard] の [サンプル] タブを選択し、XPath 式を作成する [サンプル] 領域を表示します。
3.  **ボタンをクリックして、**[XML ファインダ] ダイアログ ボックスを開きます。[XML ファインダ] ダイアログ ボックスを使って、XPath 式の作成に使用する XML、DTD、XSD、MFL または XSL ドキュメントを検索します。詳細については、7-19 ページの「XML エンティティを取り出す」 を参照してください。

   XML ファインダによってドキュメントが検出されると、ドキュメントのコンテンツが Xpath Wizard に表示されます。ユーザが DTD または XSD ドキュメントを開くと、Wizard はサンプルの XML ドキュメントを、ユーザが選択した DTD または Schema から生成します。

図 8-4 [XPath Wizard] : サンプルの XML ドキュメントを持つ [サンプル] タブ



4. XPath 式のターゲットにする、要素、属性、またはノードのマークアップまたはコンテンツを選択します。選択したターゲットに基づいて、XPath Wizard によって XPath 式が生成され、[XPath 式] フィールドに表示されます。

5.  ボタンをクリックして、ユーザが作成した XPath 式を [ワークスペース] に入れます。

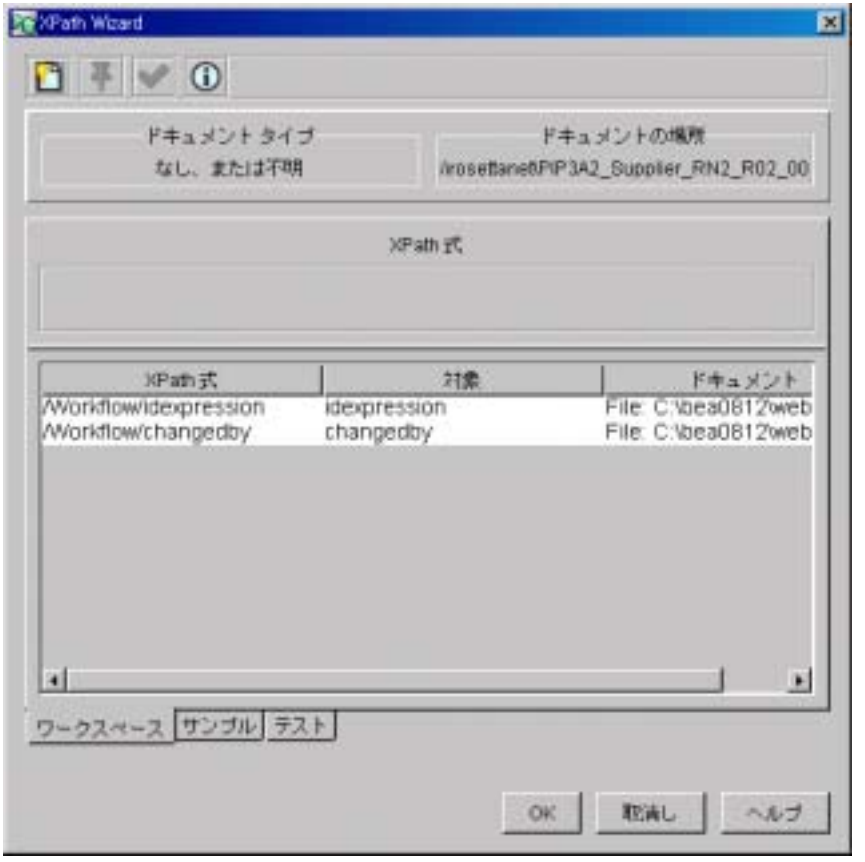
6. 作成した式を表示するには、[ワークスペース] タブをクリックします。詳細については、8-36 ページの「XPath 式を表示する」 を参照してください。

7. 作成した式をテストするには、[テスト] タブをクリックします。詳細については、8-38 ページの「XPath 式をテストする」 を参照してください。
8. [OK] をクリックして、式を [Expression Builder] に返します。

XPath 式を表示する

Studio セッション中に XPath Wizard で作成したすべての XPath 式は、[ワークスペース] に配置できます。XPath 式を作成するごとに、式を [ワークスペース] に固定できます。このようにして、1 つの XML ドキュメントに対して複数の式を作成し、これらの式が正しい結果を返すことを後でテストできます。

図 8-5 [XPath Wizard] : [ワークスペース] タブ



ワークスペースには、各 XPath 式について以下の情報が表示されます。

XPath 式	[サンプル] タブで生成した、または [テスト] タブに入力した XPath ロケーション式。
ターゲット	XPath 式のターゲットである、XML ドキュメント内の要素または属性。
ドキュメント	XPath 式が生成される元となったソース ドキュメント。

式をテストするには、リスト内の式を選択して [テスト] タブをクリックします。テストの手順については、8-38 ページの「XPath 式をテストする」 を参照してください。

[Expression Builder] に式を返すには、リスト内の式を選択して [OK] をクリックします。[XPath Wizard] ウィンドウが閉じ、選択した式が [Expression Builder] に配置されます。

XPath 式をテストする

XPath Wizard のテスト機能は、次の目的で使用します。

- Expression Builder で構築した、またはサンプル ドキュメントから生成したロケーション式をテストします。1 つのドキュメントから XPath ロケーション式を生成した後、他のサンプル ドキュメントに対してこれらの式をテストして、正しいコンテンツが返されることを確認したほうがよい場合があります。
- XPath 言語関数など、ユーザが作成した別の型の XPath 式を、サンプル ドキュメントに対して検証します。Expression Builder で構築した式を直接 XPath Wizard に渡す、Sample 機能によって生成済みの場所に XPath 関数を追加する、または XPath Wizard で式を新しく作成できます。XPath Wizard の以下のテスト機能は、XPath 言語のすべての関数をサポートしています。
  - 文字列関数－ノードおよび属性の文字列のコンテンツに対して、アクセス、抽出、結合などの処理を行います。
  - ブール関数－引数に対して Boolean テストまたは演算を行い、それによって得られた値を返します。
  - 数値関数－ドキュメント内のノード数や数値の合計など、数値で表される結果を返します。
  - ノード設定関数－個々のノードまたはノード セットについての情報を返します。

  XPath 言語によって指定される関数の一覧については、http://www.w3.org/TR/xpath.html を参照してください。

  式を構築した後、計算値が XPath Wizard によって返されます。

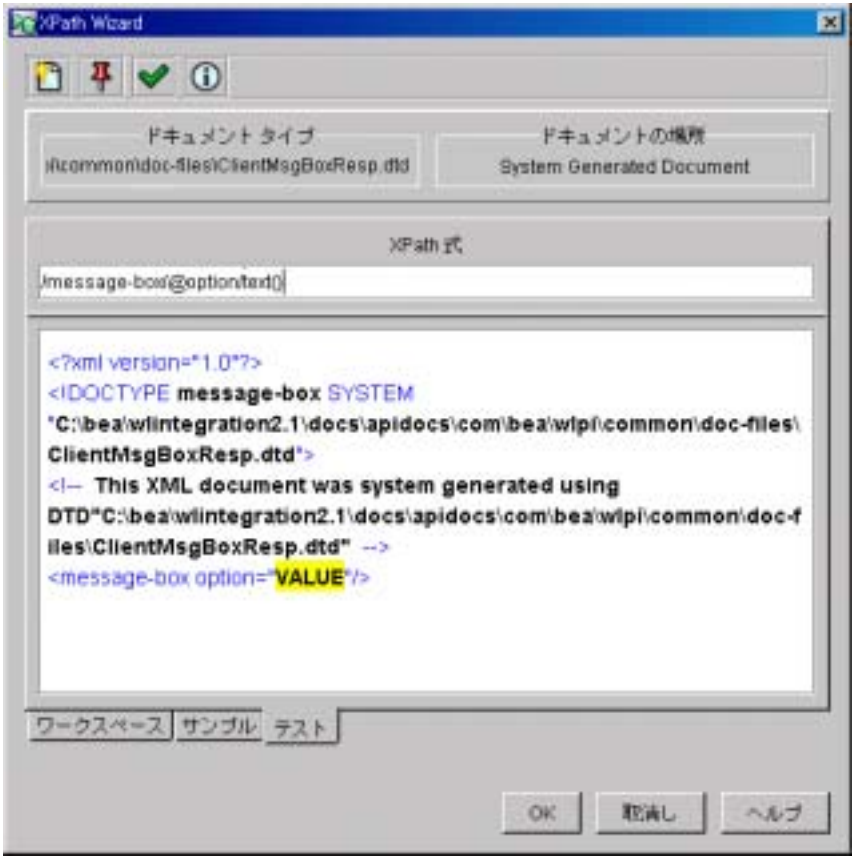
ロケーション式をテストする

XPath のロケーション式をテストする手順は、次のとおりです。

1. （省略可能）テストする XPath 式を [Expression Builder] に入力して、[XPath Wizard] をクリックします。XPath 式が、[ワークスペース] 内の [XPath 式] リストの最初の項目として表示されます。
2. [サンプル] タブをクリックして [開く] ボタンをクリックし、[XML ファインダ] ダイアログ ボックスを表示します。[XML ファインダ] ダイアログ ボックスを使って、XPath 式のテストに使用する XML、DTD、XSL、XSD、または MFL ドキュメントを検索します。詳細については、7-19 ページの「XML エンティティを取り出す」 を参照してください。

   XML ファインダによってドキュメントが検出されると、ドキュメントのコンテンツが XPath Wizard に表示されます。ユーザが DTD または XSD ドキュメントを開くと、Wizard はサンプルの XML ドキュメントを、ユーザが選択した DTD または Schema から生成します。
3. 手順 1 で Expression Builder から式を渡さなかった場合、サンプル ドキュメントから XPath ロケーション式を生成します。手順については、8-34 ページの「XPath ロケーション式を XML エンティティから生成する」 を参照してください。
4. [ワークスペース] 内で、テストする式を選択します。
5. [テスト] タブを選択します。[テスト] 領域に XML ドキュメントが表示され、XPath 式のターゲットが強調表示されます。

図 8-6 [XPath Wizard] : [テスト] タブ



6. 必要に応じて、[XPath 式] フィールドで XPath 式を編集して修正します。
7. 式を再びテストして、その値をドキュメント内でハイライトさせるには、 ボタンをクリックします。式の新しい結果が XML ドキュメント内で強調表示されます。
8. [OK] をクリックして、式を [Expression Builder] に返します。

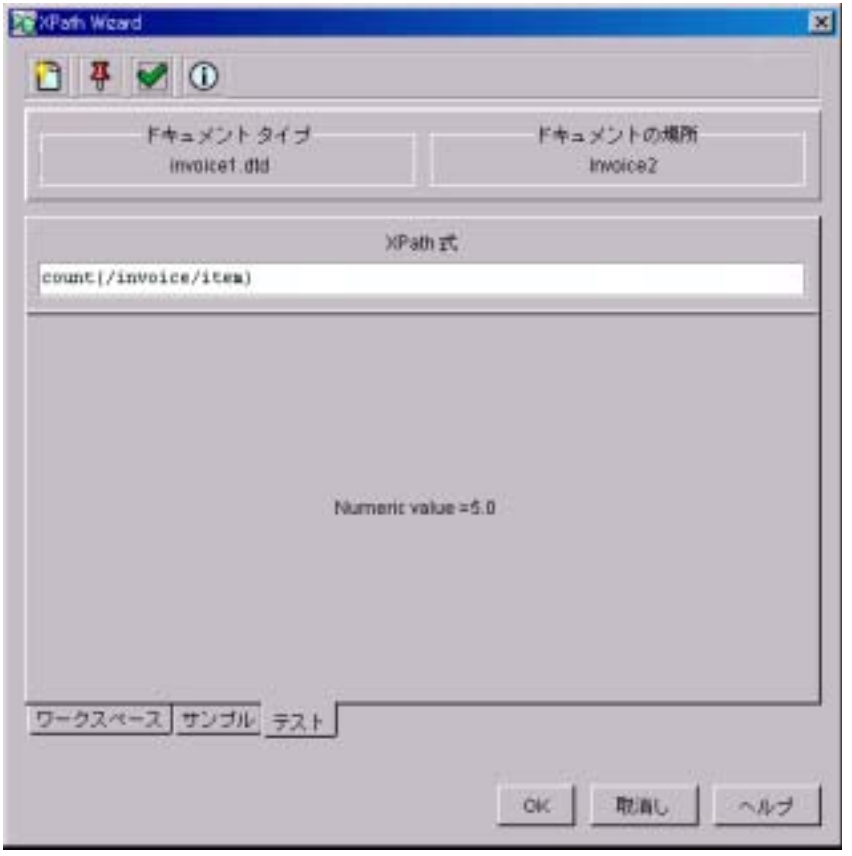
関数を含む式をテストする

関数を含む XPath 式をテストする手順は、次のとおりです。

1. （省略可能）テストする XPath 式を [Expression Builder] に入力して、[XPath Wizard] をクリックします。XPath 式が、[ワークスペース] 内の [XPath 式] リストの最初の項目として表示されます。
2. [サンプル] タブをクリックして [開く] ボタンをクリックし、[XML ファインダ] ダイアログ ボックスを表示します。[XML ファインダ] ダイアログ ボックスを使用して、XPath 式のテストに使用する XML、DTD、または Schema (XSD) ドキュメントを取り出します。詳細については、7-19 ページの「XML エンティティを取り出す」 を参照してください。

   ユーザが DTD または XSD ドキュメントを開くと、Wizard はサンプルの XML ドキュメントを、ユーザが選択した DTD または Schema から生成します。生成された XML ドキュメントを使用して、XPath 式をテストします。
3. （省略可能）サンプル ドキュメントから XPath ロケーション式を生成します。手順については、8-34 ページの「XPath ロケーション式を XML エンティティから生成する」 を参照してください。
4. [テスト] タブを選択します。
5. [XPath 式] フィールドに式を入力するか、[ワークスペース] から配置した式を編集（関数を追加するなど）します。
6.  ボタンをクリックします。[テスト] 領域に関数の戻り値が表示されます。

図 8-7 XPath の関数の演算結果



7. 必要に応じて、[XPath 式] フィールドで XPath 関数を編集して修正します。
8. 関数を再びテストする場合は、 ボタンをクリックします。[テスト] 領域に関数の戻り値が表示されます。
9. [OK] をクリックして、関数を [Expression Builder] に返します。

9　ワークフロー例外の処理

この章では、内部および外部的に発生したワークフロー例外条件を処理する方法について説明します。

- ワークフローの例外処理
- 例外ハンドラ定義タスクの概要
- 例外ハンドラの定義
- ワークフローからの例外ハンドラの呼び出し
- システム エラー メッセージ

ワークフローの例外処理

ワークフローの例外処理機能により、実行時に内部的および外部的に発生する例外条件に対し、定義、トラップ、および対処ができます。例外には、ユーザがワークフロー内で対象とする特定の異常な条件、またはトラップし対処する代表的な実行時のサーバ例外があります。

WebLogic Integration 内の例外処理は、タスク レベルではなく、ワークフロー レベルで行われます。すべてのワークフロー テンプレート定義には、少なくとも 1 つの例外ハンドラ、すなわちシステム例外ハンドラがあります。システム例外ハンドラは、デフォルトによる初期例外ハンドラで、例外が発生するたびに呼び出されます。例外は、ワークフロー内から特定の例外ハンドラが呼び出される前に発生する可能性があるため、*初期*例外ハンドラのコンセプトが必要になります。たとえば、開始ノードで変数を初期化する間に例外が発生することがあります。この場合、システム例外ハンドラは、アクティブなトランザクションにロールバックのみというマークを付けること、およびクライアントに対して例外を再送出することで、例外に応答します。

ワークフローの例外処理機能により、例外に対応して実行するアクションを指定するカスタマイズされた例外ハンドラを、定義できます。例外ハンドラは、コミットとロールバック処理のパスを持ちます。ユーザは、これらのパスのそれぞれで実行するワークフロー内の所定のアクションを指定します。例外が発生した

際に、現在アクティブなトランザクションのロールバックのみがマークされた場合、例外ハンドラはそのトランザクションに対するロールバック処理パスを実行します。トランザクションに対してロールバックのみのマークが付けられていない場合、例外ハンドラはそのコミット パスを実行します。ワークフローのトランザクション モデルについては、『*BPM クライアント アプリケーション プログラミング ガイド*』の「BPM トランザクション プログラミング モデル」を参照してください。

また、キャッチしたい特定の例外を、型、重大度、テキスト、その他の基準別に識別するために、ワークフローの関数を条件付きの式で使用することができます。詳細については、8-12 ページの「実行時のワークフロー データを収集する」を参照してください。

カスタム例外ハンドラとそのアクションは、一度定義すれば、以下の 3 つの方法で呼び出すことができます。

- 初期例外ハンドラとして動作するように設定する方法。これにより、ワークフローが示された時点で直ちに、指定されたカスタム例外ハンドラは例外をキャッチします。
- 任意のワークフロー ノードで、ワークフロー例外ハンドラを設定アクションを使用する方法。これによりカスタム例外ハンドラは、それ以降から、ワークフロー内で次のワークフロー例外ハンドラを設定アクションになるか、ワークフローが終了するかのどちらか最初に起こるまで、アクティブ状態になっています。
- 任意のワークフロー ノードで例外ハンドラの呼び出しアクションを使用する方法。これにより、指定された例外ハンドラ内で定義されたアクションは、例外の発生の有無にかかわらず、そのポイントで実行されます。必要に応じて、XML ドキュメントを例外プロセッサに送信し、この XML ドキュメントからの値を用いて、ワークフローの変数を作成します。このアクションは一般的に、ワークフローの定義で特に指定された例外をキャッチするような条件付の状況で使用されます。

例外ハンドラ定義タスクの概要

例外ハンドラは、例外が発生したトランザクションのコミット / ロールバック パスで各種のアクションを指定できるワークフロー内のサブワークフローのようなものです。例外ハンドラは一般的な例外の発生に対応でき、また例外ハンドラ

は、トラップする特定の例外を指定するために条件を用いることもできます。ワークフロー内でカスタム例外ハンドラを使用するには、以下を行う必要があります。

注意： 例外ハンドラやそれを参照するアクションの定義を始める前に、ワークフロー式の言語や Studio の Expression Builder や XPath Wizard についての学習も必要です。例外ハンドラの呼び出しアクションでの XML ドキュメントの構成や、例外ハンドラでの変数の初期化など、この節で説明するタスクの多くを行うには、ダイアログ ボックスのフィールドに式を入力する必要があります。ワークフロー式の詳細については、第 8 章「ワークフロー式の使用法」を参照してください。

1. 例外ハンドラを作成し、次に必要に応じて、それを初期例外ハンドラとして設定します。手順の詳細は、9-5 ページの「カスタム例外ハンドラを作成する」を参照してください。
2. 例外ハンドラにアクションを追加します。アクションの詳細については、第 6 章「アクションの定義」を参照してください。
3. 終了メソッドを例外ハンドラに追加します。手順の詳細は、9-8 ページの「例外ハンドラを終了する」を参照してください。
4. 例外ハンドラを呼び出すワークフロー ノードに、アクションを追加します。手順の詳細は、9-11 ページの「ワークフローからの例外ハンドラの呼び出し」を参照してください。

例外ハンドラの定義

例外ハンドラ定義機能を使用して、カスタム例外ハンドラを定義します。例外ハンドラを定義しない場合は、デフォルトでシステム例外ハンドラが使用されます。

カスタム例外ハンドラは、初期例外ハンドラとして設定できます。つまり、ワークフローがインスタンス化されるとその例外ハンドラがすぐにアクティブになります。テンプレート定義では、例外ハンドラを設定アクションを用いて、ワークフローの開設時にアクティブな例外ハンドラを変更できます（9-12 ページの「ワークフロー例外ハンドラを設定する」を参照）。

現在のテンプレート定義に対して定義されたカスタム例外ハンドラは、[例外ハンドラ] フォルダの下のフォルダ ツリーに表示され、また [テンプレート定義] のプロパティ ダイアログ ボックスに表示されます。

図 9-1 [テンプレート定義のプロパティ] ダイアログ ボックス : [例外ハンドラ] タブ

カスタム例外ハンドラを定義する際、コミットおよびロールバックの両方の例外条件で実行するアクションを指定できます。コミット パスの場合、すべてのワークフロー アクションを実行できます。ロールバック パスの場合は、次のアクションのみを実行できます。XML イベントをポスト、プログラムの呼び出し、ビジネス オペレーションを実行、例外ハンドラを編集、処理なし、および監査エントリを作成。アクションの詳細については、第 6 章「アクションの定義」を参照してください。

注意： 一部の Enterprise Java Beans (EJB) は、UserTransaction メソッドを呼び出してトランザクションにロールバックのみのマークを付けることができ、またコンテナ境界で非検査の例外を返すことができます。これら 2 つのケースのいずれの場合も、WebLogic Server はトランザクションをロールバックします。

また、ワークフロー変数を定義することによって、例外ハンドラの呼び出しアクション内で定義された XML メッセージの値を取り込むこともできます。詳細については、9-13 ページの「例外ハンドラを呼び出す」を参照してください。

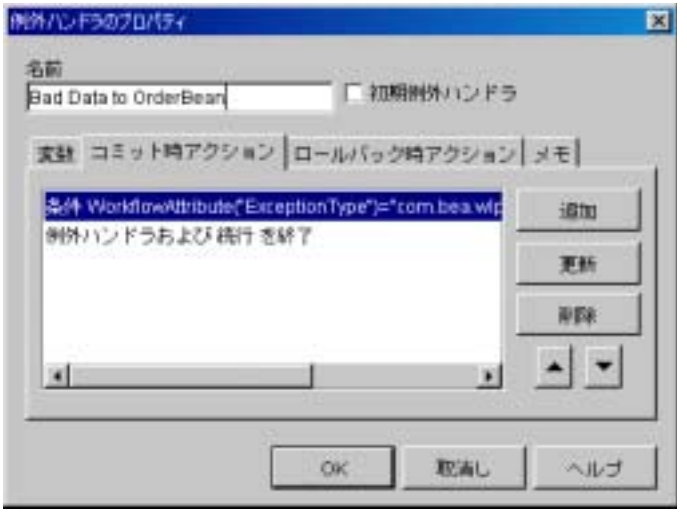
カスタム例外ハンドラを作成する

ワークフロー テンプレートの定義用に例外ハンドラを作成する手順は、次のとおりです。

1. 以下のいずれか 1 つを実行します。
   - フォルダ ツリー内で、テンプレート定義のフォルダを展開します。[例外ハンドラ] フォルダを右クリックして [例外ハンドラを作成] を選択します。
   - テンプレート定義のプロパティ ダイアログ ボックスの [例外ハンドラ] タブで、[追加] をクリックします。

   [例外ハンドラのプロパティ] ダイアログ ボックスが表示されます。

図 9-2 [例外ハンドラのプロパティ] ダイアログ ボックス



2. [名前] フィールドに、新しい例外ハンドラ用に有意で簡単に識別可能な名前を入力します。
3. (省略可能) [初期例外ハンドラ] チェックボックスを選択して、例外ハンドラをワークフロー内でデフォルトの初期例外ハンドラとしてマークを付けます。ワークフローが例外を検出すると、この例外ハンドラが最初に呼び出される。
4. (省略可能) [変数] タブで [追加] をクリックして、[ワークフロー変数の割り当て] ダイアログ ボックスを表示します。このダイアログ ボックスは、実行時に評価する式を変数の値として指定することにより、ワークフロー内で作成した変数を初期化するために使用できます。
5. [変数] ドロップダウン リストで、受信するデータを格納する変数を選択します。
6. 以下のいずれかを実行して、実行時に評価して変数の値を生成する式を [式] フィールドに入力します。
   - 定数の指定に使用する構文については、8-2 ページの「リテラルの使い方」を参照してください。
   - 変数を渡すための値を含む XML メッセージを送信するために、例外ハンドラの呼び出しアクションを使用する場合は (9-13 ページの「例外ハン

ドラを呼び出す」を参照)、XPath() 関数を使用するか(8-8 ページの「XPath()」を参照)、または XML 要素のドット表記を使用します(8-11 ページの「XML 要素のドット表記」を参照)。また [式] ボタン  を使用して XPath Wizard を呼び出し、このウィザードを使ってサンプルの着信ドキュメントから XPath の式を自動的に生成できます。詳細については、8-31 ページの「XPath Wizard を使用する XPath 式の作成」を参照してください。

7. [OK] をクリックします。変数の初期化内容が、[例外ハンドラのプロパティ] ダイアログ ボックスの [変数] タブのリストに表示されます。
8. 初期化するすべての変数に対し、手順 4 から 7 を繰り返します。
9. [コミット時アクション] タブで、[追加]、[更新] または [削除] ボタンを使用して、現在アクティブなトランザクションのマークがロールバック専用として付けられていない場合に実行するアクションを指定します。アクションの実行に関する説明は、第 6 章「アクションの定義」を参照してください。
10. [ロールバック時アクション] タブで、現在アクティブなトランザクションがロールバック専用にプログラムでマークが付けられている場合に実行するアクションの追加、更新、または削除を指定します。ワークフローは、現在アクティブなトランザクションの先頭にロールバックされます。
11. [コミット時アクション] または [ロールバック時アクション] タブで、プログラムの制御をメイン ワークフローに戻すために、[例外ハンドラを編集] アクションを追加します。詳細については、9-8 ページの「例外ハンドラを終了する」を参照してください。
12. [OK] をクリックすると、カスタム例外ハンドラが追加されます。すると新しい例外ハンドラがテンプレート定義の [プロパティ] ダイアログ ボックスの [例外ハンドラ] タブに表示され、またフォルダ ツリーの [例外ハンドラ] フォルダの下に表示されます。

例外ハンドラを終了する

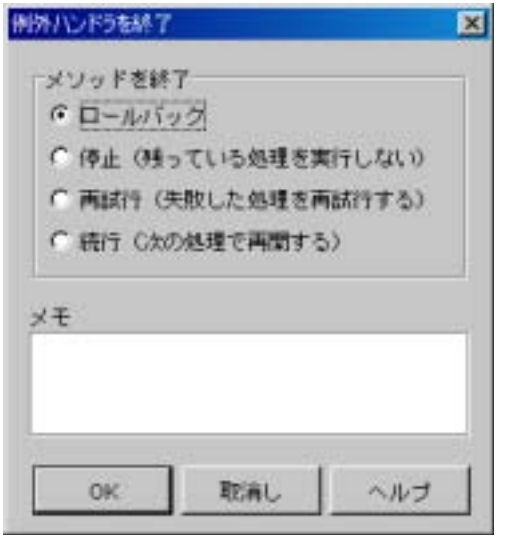
例外ハンドラを終了するには、[例外ハンドラのプロパティ] ダイアログ ボックスからのみ指定できる [例外ハンドラを編集] アクションを使用します（詳細は、9-3 ページの「例外ハンドラの定義」を参照）。このアクションを [例外ハンドラ] の [コミット時アクション] タブまたは [ロールバック時アクション] タブに必ず追加して、メイン フローに制御を返すことができるようにしてください。

[ロールバック] パスで使用可能な終了オプションは、[ロールバック] だけです。Commit パスでは、以下の 4 つのオプションから選択できます。

- ロールバック－ユーザ トランザクションにロールバックのみのマークを付け、トランザクションの開始前の状態にワークフローを戻します。例外は、例外ハンドラを設定としてクライアントに伝播される。
- 停止－例外ハンドラを終了し、後続のアクションの実行を取り止めます。
- 再試行－例外ハンドラを終了し、失敗したオペレーションの再試行を試みます（失敗した操作とは、アクションまたは変数設定を指す）。
- 続行－例外ハンドラの実行を停止し、次のオペレーションでワークフローの実行の継続を試みます（次の操作とは、次のアクションまたは次に設定する変数を指す）。

注意： 一部の Enterprise Java Beans (EJB) は、UserTransaction メソッドを呼び出してトランザクションにロールバックのみのマークを付けることができ、またコンテナ境界で非検査の例外を返すことができます。これら 2 つのケースのいずれの場合も、WebLogic Server はトランザクションをロールバックします。

図 9-3 [例外ハンドラを編集]



例外ハンドラを終了する手順は、次のとおりです。

1. [例外ハンドラのプロパティ] ダイアログ ボックスから、[アクションを追加] ダイアログ ボックスを呼び出し、[例外処理アクション] フォルダを展開し、[例外ハンドラを編集] を選択し、[OK] をクリックして [例外ハンドラを編集] ダイアログ ボックスを表示します。

   [ロールバック時アクション] タブにアクションを追加する場合、選択可能なオプションは [ロールバック] に限られます。[コミット時アクション] タブにアクションを追加する場合は、[メソッドを終了] オプションの 1 つを選択します。

2. [OK] をクリックするとアクションが追加されます。

カスタム例外ハンドラを更新する

1. 以下のいずれか 1 つを行い、[例外ハンドラのプロパティ] ダイアログ ボックスを表示します。
   - フォルダー ツリーで、[例外ハンドラ] フォルダを展開し、希望の例外ハンドラ フォルダを右クリックし、ポップアップ メニューから [プロパティ] を選択します。

- テンプレート定義のプロパティ ダイアログ ボックスの [例外ハンドラ] タブから、希望の例外ハンドラを選択し、[更新] をクリックします。

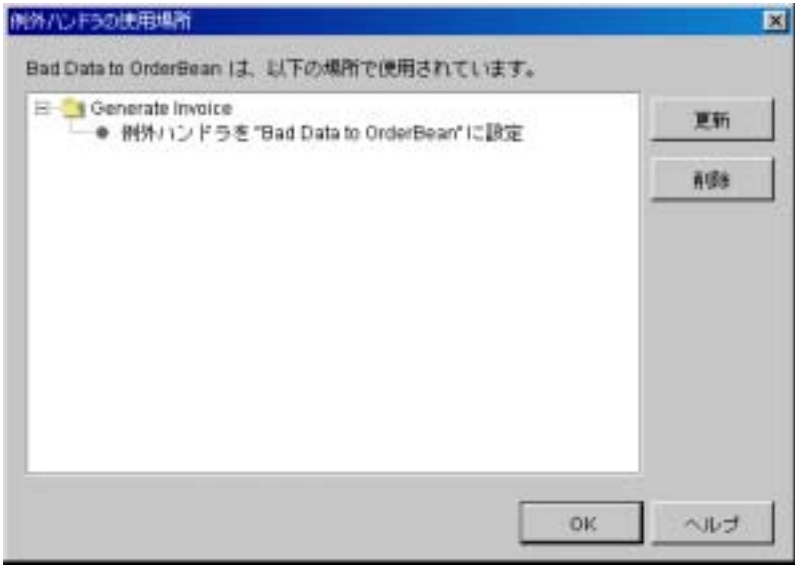
2. 例外ハンドラに対して必要な変更を行います。
3. [OK] をクリックして変更を保存します。

例外ハンドラの用途を表示する

例外ハンドラがワークフロー内で参照する場所を閲覧する手順は、次のとおりです。

1. フォルダー ツリーで、例外ハンドラのフォルダを右クリックし、ポップアップ メニューから [使用場所] を選択して [例外ハンドラの使用場所] ダイアログ ボックスを表示します。このダイアログ ボックスには、選択した例外ハンドラが参照するワークフロー内の場所が一覧で示されます。

図 9-4 [例外ハンドラの使用場所] ダイアログ ボックス



2. (省略可能) ダイアログ ボックスの以下のボタンを用いて以下の操作を行います。
   - [更新] ー選択したオブジェクトに対するダイアログ ボックスを開きます。

- [削除] ―選択したオブジェクトを削除します。

3. [OK] をクリックします。[例外ハンドラの使用場所] ダイアログ ボックスを終了します。

カスタム例外ハンドラを削除する

一度定義しておけば、例外ハンドラは例外ハンドラの呼び出しアクションまたはワークフロー例外ハンドラを設定アクションで参照しない場合にのみ削除されます（9-13 ページの「例外ハンドラを呼び出す」を参照）。例外ハンドラが参照する場所のリストを閲覧する場合は、9-10 ページの「例外ハンドラの用途を表示する」の手順に従います。

例外ハンドラを削除する手順は、次のとおりです。

1. 以下のいずれか 1 つを実行します。
   - フォルダー ツリーで、[例外ハンドラ] フォルダを展開し、該当する例外ハンドラ フォルダを右クリックし、ポップアップ メニューから [削除] を選択します。
   - テンプレート定義のプロパティ ダイアログ ボックスの [例外ハンドラ] タブから、該当する例外ハンドラを選択し、[削除] をクリックします。
2. メッセージが表示されたら削除の操作を確認します。

ワークフローからの例外ハンドラの呼び出し

ワークフローから例外ハンドラを呼び出すには、以下の例外処理アクションを用います。

- [ワークフロー例外ハンドラを設定] ―このアクションは、ワークフローの例外ハンドラをユーザが指定したものに設定します。このポイントからのすべての例外は、このアクションが再び指定されるまで、指定された例外ハンドラによってキャッチされます。詳細については、9-12 ページの「ワークフロー例外ハンドラを設定する」を参照してください。

- [例外ハンドラの呼び出し] ― このアクションはカスタム例外ハンドラ内で定義したアクションを呼び出し、また必要に応じて、ワークフロー内の所定のポイントで例外プロセッサに XML メッセージを送信します。これは一般的に、ワークフロー内で特に定義された例外に対して、代替アクションを行うような条件付きの状況で用いられます。詳細については、9-13 ページの「例外ハンドラを呼び出す」を参照してください。
- [例外ハンドラを編集] ― このアクションは、例外ハンドラ内からメイン ワークフローに制御を返すために使用されます。詳細については、9-8 ページの「例外ハンドラを終了する」を参照してください。

ワークフロー例外ハンドラを設定する

指定した例外ハンドラを、ワークフロー テンプレート定義でアクティブな例外ハンドラにするには、ワークフロー例外ハンドラを設定アクションを使用します。このアクションをワークフロー テンプレート定義内で使わず、かつ初期の例外ハンドラとして定義しマーク付けした例外ハンドラが他に存在しない場合、システムの例外ハンドラがデフォルトで使用され、例外が発生するたびに呼び出されます。

ワークフローに設定した例外ハンドラは、後続の [ワークフロー例外ハンドラを設定] アクションを使用してその例外ハンドラをシステム ハンドラまたは別のカスタム定義ハンドラにリセットするまで、ワークフローのインスタンス全体を通して持続します。

図 9-5 [ワークフロー例外ハンドラを設定]

ワークフロー例外ハンドラをカスタム ハンドラに設定する手順は、次のとおりです。

1. [アクションを追加] ダイアログ ボックスから [例外処理アクション] フォルダを展開し、[ワークフロー例外ハンドラを設定] を選択し、[OK] をクリックして [ワークフロー例外ハンドラを設定] ダイアログ ボックスを表示します。
2. [例外ハンドラ] ドロップダウン リストから、以下のいずれか 1 つを行います。
   - ワークフロー例外ハンドラを、ユーザが定義したカスタム例外ハンドラに設定するには、[例外ハンドラ] ドロップダウン リストからその例外ハンドラを選択します。
   - ワークフロー例外ハンドラをデフォルトのシステム例外ハンドラに戻すには、[例外ハンドラ] ドロップダウン リストから（システム例外ハンドラを）選択します。
3. [OK] をクリックするとアクションが追加されます。

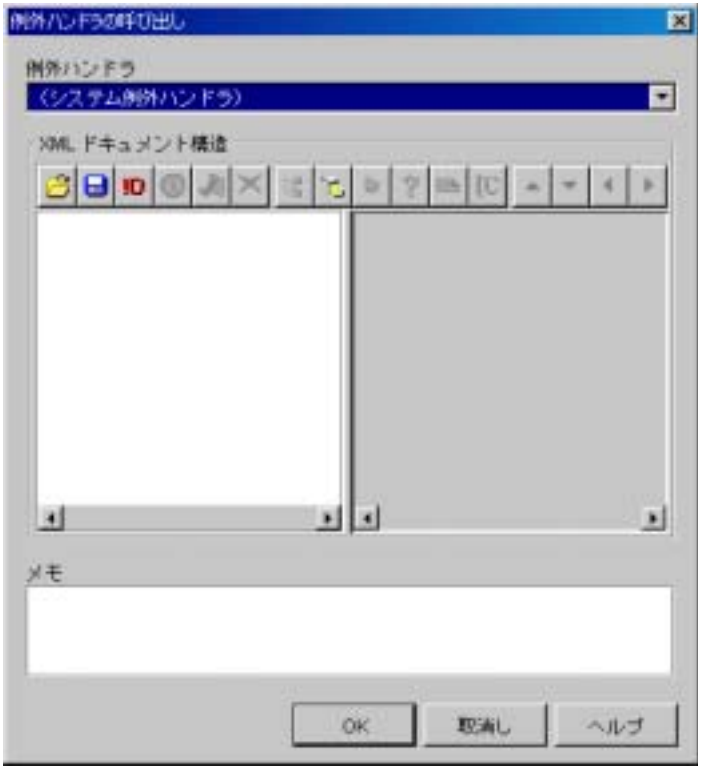
例外ハンドラを呼び出す

特定の例外ハンドラをワークフロー内で呼び出し、また必要に応じて、ユーザが定義した XML ドキュメントを例外ハンドラに送信するには、例外ハンドラの呼び出しアクションを使用します。XML ドキュメントを使うと、例外ハンドラを呼び出したときに変数入力に使用する値を送信できます。（詳細は、9-3 ページの「例外ハンドラの定義」を参照）。

しかし、このアクションによって、アクティブに設定されている例外ハンドラがオーバーライドされることはありません。例外が発生しても、引き続きその例外ハンドラによって処理されます。さらに、例外ハンドラの呼び出しアクションの実行時には、例外が発生したかどうかに関係なく、例外ハンドラで定義したアクションが実行されます。したがって、一般的にこのアクションは、ワークフローそのものの中から例外が送出されている結果、例外ハンドラで定義したアクションを呼び出したいという条件の下で使用します。このアクションは、ビジネス オペレーションによって送出された例外を処理するものではありません。

また、このアクションは XML ドキュメントを選択した任意の例外ハンドラに単に送信する場合にも使用できます。

図 9-6 [例外ハンドラの呼び出し] ダイアログ ボックス



カスタム例外ハンドラを呼び出す手順は、次のとおりです。

1. [アクションを追加] ダイアログ ボックスから [例外処理アクション] フォルダを展開し、[例外ハンドラの呼び出し] を選択し、[OK] をクリックして [例外ハンドラの呼び出し] ダイアログ ボックスを表示します。
2. [例外ハンドラ] ドロップダウン リストから、以下のいずれか 1 つを行います。
   - XML ドキュメントを、ワークフロー内のこのポイントでイベント プロセッサに単に送信する場合は、[例外ハンドラ] フィールドの設定をデフォルトのシステム例外ハンドラのままに残します。
   - 既に定義しているカスタム例外ハンドラを選択する。ここで指定するカスタム定義の例外ハンドラ内のアクションは、例外が実際に発生したか

どうかに関係なく、このアクションの実行時に呼び出されます。このポイントで発生する例外は、ワークフロー例外ハンドラとして設定された例外ハンドラによって引き続き処理されます。

3. （省略可能）例外ハンドラの呼び出し時にイベント プロセッサに送信する XML ドキュメントを定義します。XML ドキュメント構造を指定するには以下のいずれかを行います。
   - 新たな自由形式ドキュメントを作成するには、[子を追加] ボタンをクリックし、ノードを追加することにより、ドキュメントの作成を開始します。
   - 既存の XML ドキュメントを指定するには、[インポート] ボタンをクリックしてドキュメントをロードします。
   - 新たなタイプ指定 XML ドキュメントを作成するには、[コンテンツ タイプを設定] ボタンをクリックし、適切なスキーマ ドキュメントをロードします。

   上記のオプションの詳細な手順については、7-2 ページの「XML ドキュメントの作成と編集」を参照してください。

   ユーザ定義の XML ドキュメントは、ワークフロー テンプレート定義内に保存される。
4. [OK] をクリックするとアクションが追加されます。

システム エラー メッセージ

8-12 ページの「実行時のワークフロー データを収集する」で説明したように、例外ハンドラが以下の情報を問い合わせできるように、4 つの属性を指定して `WorkflowAttribute()` 関数を使用することができます。

- Java の例外クラス名
- エラー番号
- エラー メッセージ本文
- エラー重大度コード（ユーザには例外ハンドラがアクションまたは API 経由で呼び出されたか、キャッチした例外の結果として呼び出されたか通知される）。

- *0*: エラー タイプは不明。内部使用専用。
- 1: ユーザ要求の処理時に致命的な例外が発生した。
- 2: ワークフロー ステータスの不一致など、致命的で違法な条件が発生した。
- 3: ユーザが手作業で修正できる、致命的でないワークフロー条件が発生した。
- 4: `WorkflowProcessor.invokeWorkfowErrorHandler` を呼び出すアプリケーション、または例外ハンドラを呼び出しアクションを実行するワークフローでカスタム エラーが発生した。

次の表に、ワークフロー エラー メッセージを番号と本文別に一覧で示します。

表 9-1 ワークフロー エラー メッセージ

エラー番号	エラー メッセージ本文
0	不明なエラー
1	システム エラー
2	ワークフロー エラー
3	ワークフロー警告
4	ネスとされた例外
5	サーバではリクエストを完了できませんでした。
6	サーバではリクエストを完了できませんでした。
7	ワークフローは完了しているので、変更できません。
8	このタスクのプロパティでは、タスクに完了マークを付けられません。
9	このタスクのプロパティでは、タスクの k 何両マークを外せません。
10	このタスクのプロパティでは、タスクを再割り当てできません。
11	このタスクのプロパティでは、タスクを変更できません。
12	このタスクはすでに完了しているので、取得できません。
13	このタスクはすでに完了しているので、割り当てられません。

表 9-1 ワークフロー エラー メッセージ

エラー番号	エラー メッセージ本文
14	このタスクはすでに完了しているので、実行できません。
15	このタスクは非アクティブなので、実行できません。
16	ワークフローが完了しているので、このタスクを実行できません。
17	このタスクは割り当てられていないので、実行できません。
18	ルート変更の送り元と送り先は異なっている必要があります。
19	有効日は、期日以前とする必要があります。
20	ソース ユーザは、指定の期間中にすでにルート変更されています。
21	このルート変更で循環的な参照がされます。
22	タスク名はワークフロー内で一意である必要があります。
23	指定したオーガニゼーション "{0}" はすでに定義されています。
24	ロール名はオーガニゼーション内で一意である必要があります。
25	指定したユーザ "{0}" はすでに定義されています。
26	レポート名は一意である必要があります。
27	変数名は一意である必要があります。
28	ワークフロー テンプレート名は一意である必要があります。
29	作業負荷グラフ名は一意である必要があります。
30	ビジネス カレンダー "{0}" はすでに定義されています。
31	ユーザ "{0}" は現在ログオンされており、削除できません。
32	ユーザ "{0}" に割り当てられているタスクがあるため、このユーザを削除することはできません。先にすべてのタスクの再割り当てを行ってください。
33	ユーザを管理するパーミッションが付与されていません。
34	ロールを管理するパーミッションが付与されていません。

表 9-1 ワークフロー エラー メッセージ

エラー番号	エラー メッセージ本文
35	オーガニゼーションを管理するパーミッションが付与されていません。
36	ワークフローを定義するパーミッションが付与されていません。
37	ワークフローを監視するパーミッションが付与されていません。
38	タスクのルート変更を行うパーミッションが付与されていません。
39	サーバのコンフィグレーションを変更するパーミッションが付与されていません。
40	変数 "{0}" は ID 式またはトリガ定義によって参照されているので、削除できません。
41	変数 "{0}" は、1 つ以上のアクションによって参照されているので、削除できません。
42	変数 "{0}" は、1 つ以上の決定によって参照されているので、削除できません。
43	変数名は空白にできません。
44	ロール "{0}" に割り当てられているタスクがあるため、このロールを削除することはできません。先にすべてのタスクの再割り当てを行ってください。
45	ワークフロー "{0}" は、1 つ以上のアクションによって参照されているので、削除できません。
46	タスク "{0}" は、1 つ以上のアクションによって参照されているので、削除できません。
47	ロール "{0}" は、1 つ以上のアクションによって参照されているので、削除できません。
48	ユーザ "{0}" は、1 つ以上のアクションによって参照されているので、削除できません。
49	このワークフローはサスペンドされているので、変更できません。
50	ビジネス カレンダーのテンプレート ルールを削除できません。
51	式の中でビジネス カレンダー ID が指定されていません。
52	カレンダーの XML が定義されていません。
53	指定したビジネス カレンダー "{0}" が見つかりませんでした。

表 9-1 ワークフロー エラー メッセージ

エラー番号	エラー メッセージ本文
54	ビジネス カレンダー "{0}" で {1} 年のルールが定義されていません。
55	ビジネス カレンダー "{0}" のプロセッサが見つかりませんでした。
56	{0}, {1}: ビジネス カレンダー ルールに予期しないトークン "{2}" があります。
57	{0}, {1}: ビジネス カレンダー ルールに予期しない文字 "{2}" があります。
58	タイムゾーン識別子 "{0}" が無効です。
59	EJB ホーム環境が設定されていません。
61	初期コンテキストを取得できません。環境が無効である可能性があります : {0}
62	型 "{0}" のホーム インタフェースのクラス オブジェクトをロードできません。{1}
63	型 "{0}" のリモート インタフェースのクラス オブジェクトをロードできません。{1}
64	指定したコンテキストで "{0}" にバインドされたオブジェクトがありません。{1}
65	"{0}" にバインドされたオブジェクトが、ホーム インタフェース "{1}" を実装していません。
66	オブジェクト "{0}" にホーム メソッド "{1}" が含まれていません。{2}
67	オブジェクト "{0}" にリモート メソッド "{1}" が含まれていません。{2}
68	ビジネス オペレーションにより、不明なクラス "{0}" のオブジェクトが返されました。{1}
69	リモート オブジェクトが見つかりませんでした。
70	EJB "{0}" メソッドの呼び出しが失敗しました。{1}
71	"{0}" オブジェクトのメソッドにより、予期しない型 "{1}" の値が返されました。
72	メソッドに対するパラメータの数が違います。予想していたのは {1} 個でしたが、見つかったのは {2} 個でした。
73	型 "{1}" のパラメータ {0} に対するクラス（プリミティブの場合はラップされている）をロードできませんでした。{2}

表 9-1 ワークフロー エラー メッセージ

エラー番号	エラー メッセージ本文
74	型 "{1}" のパラメータ {0} に対するクラス（プリミティブの場合はラップされていない）をロードできませんでした。{2}
75	パラメータ {0} の値を要求された型 "{1}" にキャストできません。
76	ホーム メソッドがセッション Bean を返しませんでした。
77	指定したワークフロー テンプレート（ID={0}）が見つかりませんでした。
78	指定したワークフロー テンプレート定義（ID={0}）が見つかりませんでした。
79	開始すべきアクティブかつ有効なテンプレート定義が見つかりませんでした。
80	第 {0} 行、第 {1} 列に XML 構文エラーがあります。
81	"{0}" のクラス オブジェクトをロードできません。{1}
82	"{0}" で一致するコンストラクタが見つかりませんでした。{1}
83	新しい "{0}" を作成できませんでした。{1}
84	"{0}" で "{1}" に一致するメソッドが見つかりませんでした。{2}
85	"{0}" メソッドの呼び出しが失敗しました。{1}
86	ワークフロー テンプレートは現在、{0} によってロックされています。
87	間隔の「開始」日が無効です。
88	間隔の「終了」日が無効です。
89	日付 "{0}" は無効です。
90	月名 "{0}" は無効です。
91	曜日名 "{0}" は無効です。
92	指定したタスク インスタンス "{0}" が見つかりませんでした。
93	必須の入力変数 "{0}" が指定されていません。
94	指定した結合インスタンス "{0}" が見つかりませんでした。

表 9-1 ワークフロー エラー メッセージ

エラー番号	エラー メッセージ本文
95	指定した変数インスタンス "{0}" が見つかりませんでした。
96	指定したオーガニゼーション "{0}" が見つかりませんでした。
97	指定したロール "{0}" が見つかりませんでした。
98	指定したユーザ "{0}" が見つかりませんでした。
99	ユーザ "{0}" はオーガニゼーション "{1}" に所属していません。.
100	ユーザ "{0}" をオーガニゼーション "{1}" に追加できませんでした。
101	ユーザ "{0}" をロール "{1}" に追加できませんでした。
102	ユーザ "{0}" をオーガニゼーション "{1}" から削除できませんでした。
103	ユーザ "{0}" をロール "{1}" から削除できませんでした。
104	指定したユーザ "{0}" は、ロール "{1}" の定義が行われているオーガニゼーションには所属していません。
105	サーバにセキュリティ レルムがインストールされていません。
106	インストールされているセキュリティ レルム "{0}" はリストできません。
107	セキュリティ レルム "{0}" は管理できません。
108	データベースに接続できません。
109	ロール "{0}" にはメンバーが含まれていません。
110	準備エラー : 参照 "{0}" が無効です。
111	感覚の単位 "{0}" が無効です。
112	変数値の割り当てが不正です。型 "{1}" の変数 "{0}" に値の型 "{2}" を割り当ててみてください。
113	テンプレート定義が非アクティブであるため、ワークフローをインスタンス化できません。
114	対象タスク "{0}" が見つかりませんでした。

表 9-1 ワークフロー エラー メッセージ

エラー番号	エラー メッセージ本文
115	対象イベント "{0}" が見つかりませんでした。
116	対象アクション "{0}" が見つかりませんでした。
117	指定したビジネス オペレーション "{0}" が見つかりませんでした。
118	引数 "{1}" を有するプログラム "{0}" の呼び出し中にエラーが発生しました。
119	ユーザ "{0}" には電子メール アドレスがありません。
120	ロール "{1}" のユーザ "{0}" には電子メール アドレスがありません。
121	ルーティング テーブルで、適切な割り当て対象を識別できませんでした。
122	ユーザ "{0}" に対してワークフローオーガニゼーションが定義されていません。
123	開始日の式 "{0}" が違います。\n"(1)"
124	例外ハンドラ "{0}" が見つかりませんでした。
125	例外ハンドラの処理中にエラーが発生しました。
126	例外ハンドラが、許容されている最大再試行回数を超えました。
127	アプリケーションまたはワークフローのインスタンスが、例外ハンドラを呼び出しました。
128	指定したワークフロー インスタンス {0} が見つかりませんでした。
129	XML ドキュメントの解析時にエラーが発生しました。
130	この定義は、新しいバージョンで作成されています。\n\ WebLogic Integrator Studio を {0} 以降のバージョンにアップグレードしてください。

10 ワークフローのモニタリング

この章では、ワークフローのモニタリングについて説明します。

- ワークフロー モニタリング タスクの概要
- ワークフロー インスタンスの操作
- ユーザ ワークリストとロールのワークリストの表示
- タスクのパーミッションと優先度の変更
- タスクのステータスと割り当ての変更
- 作業負荷レポートの使用
- 統計レポートの使用

ワークフロー モニタリング タスクの概要

Studio でワークフローのモニタ機能を使うと、ワークフローの設計者は実行時にモニタしながら、設計環境内でワークフロー設計のデバッグおよびトラブルシューティングを行うことができます。また、システム管理者は、本稼動環境でリアルタイムに実行されるワークフローをモニタして介入できます。

さらに本稼動環境では、管理者は手動で割り当てられたタスクのユーザおよびロールの作業負荷をモニタできます。

最後にビジネス アナリストは、手動で割り当てたタスクについて実行後のデータ収集を実行し、作業負荷とパフォーマンスに関する生の履歴データと統計データを収集し蓄積することにより、ビジネス処理におけるボトルネックと非効率性を分析できます。この節では、以下のタスクについて説明します。

- 実行中または完了したワークフロー インスタンスのステータスを表示します。手順の詳細は、10-5 ページの「ワークフロー インスタンスの状態を表示する」を参照してください。

- 実行中のインスタンスにおける変数の現在の値を表示して修正します。手順の詳細は、10-8 ページの「ワークフロー インスタンスの変数を表示し、更新する」を参照してください。
- ワークフロー インスタンスに関連するタスク（たとえば、タスクの再割り当てや、ワークフローの強制的な再実行など）を修正します。手順の詳細は、10-16 ページの「タスクのステータスと割り当ての変更」を参照してください。
- タスクごとに定義されたパーミッションを変更して、Worklist のユーザや他の管理者がタスクを変更できるようにします。手順の詳細は、10-14 ページの「タスクのパーミッションと優先度の変更」を参照してください。
- ユーザまたはロールのタスク リストを表示し、タスクの割り当て、ステータス、またはパーミッションを修正します。手順の詳細は、10-12 ページの「ユーザ ワークリストとロールのワークリストの表示」を参照してください。
- タスク数をユーザまたはロール別にステータス（保留中、非アクティブ、完了、期限切れ）ごとに表示して、システムの作業負荷ステータスを表示します。手順の詳細は、10-18 ページの「作業負荷レポートの使用」を参照してください。
- 未使用データを蓄積し、作業負荷および実行回数の統計レポートをユーザおよびロール別に作成します。手順の詳細は、10-22 ページの「統計レポートの使用」を参照してください。

注意： ユーザおよびロールのワークリストの表示を除き、この節で説明するモニタ タスクを実行するには、インスタンスのモニタ パーミッションが必要です。パーミッションの詳細は、3-26 ページの「ユーザおよびロールへのパーミッションの割り当て」を参照してください。

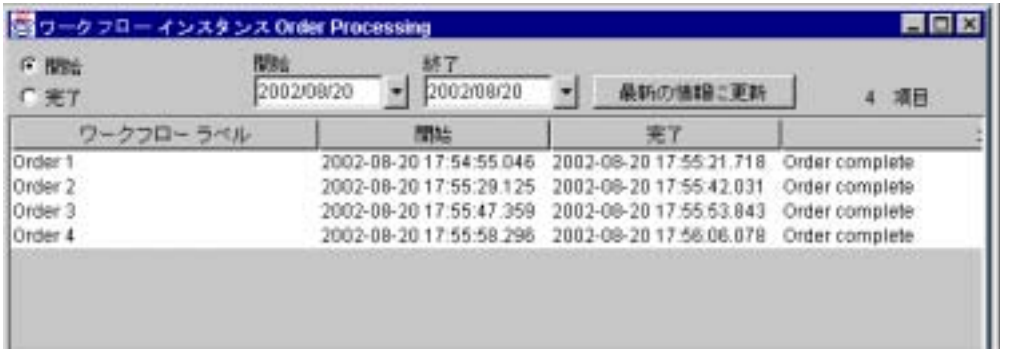
ワークフロー インスタンスの操作

ワークフロー インスタンスは、実行時に配置されたワークフロー テンプレート定義のセッションです。ワークフロー インスタンスのステータス、および変数の現在の値を表示できます。設計環境では、インスタンスを表示して、ワークフロー設計のデバッグおよびトラブルシューティングを行うことができます。プロダクション環境では、実行中のワークフローのステータスを表示したり、介入して変数の更新またはタスクの修正を行うことができます。

ワークフロー インスタンスのリストを表示する手順は、次のとおりです。

1. フォルダ ツリー内で、インスタンスを表示するワークフロー テンプレートまたはテンプレート定義を右クリックします。ポップアップ メニューから [インスタンス] を選択し、[ワークフロー インスタンス] ダイアログ ボックスを表示します。
2. 以下のオプションからいずれか 1 つを選択します。
   - 開始－指定した時間の範囲内に開始した、処理中または完了後のすべてのワークフローを表示します。
   - 完了－指定した時間の範囲内に完了した、すべてのワークフローを表示します。
3. [開始] および [終了] ドロップダウン ボックスから開始日および終了日を選択し、選択したワークフロー テンプレート定義のインスタンスを表示する期間を指定します。

図 10-1 [ワークフロー インスタンス] ダイアログ ボックス



各ワークフロー インスタンスについて、以下の情報が表示されます。

ワークフロー ラベル	テンプレート定義の [プロパティ] ダイアログ ボックスの [ワークフロー ラベル] フィールドで指定した式からラベルが生成される。詳細については、5-13 ページの「テンプレート定義を更新、ラベリングおよびアクティブ化する」 を参照。
開始	ワークフローがインスタンス化された日付。

完了	ワークフローが完了した日付。まだ完了していない場合、このカラムは空白になる。
コメント	ワークフロー内の [ワークフロー コメントを設定] アクションで指定した式から生成されたコメント。詳細については、6-47 ページの「ワークフロー コメントを設定する」を参照してください。このアクションが定義されていない場合、このカラムは空白になる。

性能上の理由から、プロセス エンジンは一度に 100 件の項目のみを返します。さらに追加項目がある場合は、ボタン（+）が表示されます。このボタンをクリックすると、次の 100 個の項目が取り出されます。追加項目がなくなると、ボタンは表示されません。

インスタンス リストを更新するには、[最新の情報に更新] をクリックします。

[ワークフロー インスタンス] ダイアログ ボックスで、以下の操作を行うことができます。

- ワークフロー インスタンスのステータスを、グラフまたはリストに表示します。詳細については、10-5 ページの「ワークフロー インスタンスの状態を表示する」を参照してください。
- ワークフロー インスタンス変数の現在の値を表示および更新します。詳細については、10-8 ページの「ワークフロー インスタンスの変数を表示し、更新する」を参照してください。
- タスクの割り当てとプロパティを修正します。詳細については、10-14 ページの「タスクのパーミッションと優先度の変更」を参照してください。
- ワークフロー インスタンスを削除します。詳細については、10-11 ページの「ワークフロー インスタンスを削除する」を参照してください。

ワークフロー インスタンスの状態を表示する

特定のワークフロー インスタンスのステータスを表示するには、[ワークフロー インスタンス] ダイアログ ボックスのリスト内でインスタンスをダブルクリックする、またはインスタンスを右クリックして、ポップアップ メニューから [ワークフロー ステータス] を選択します。

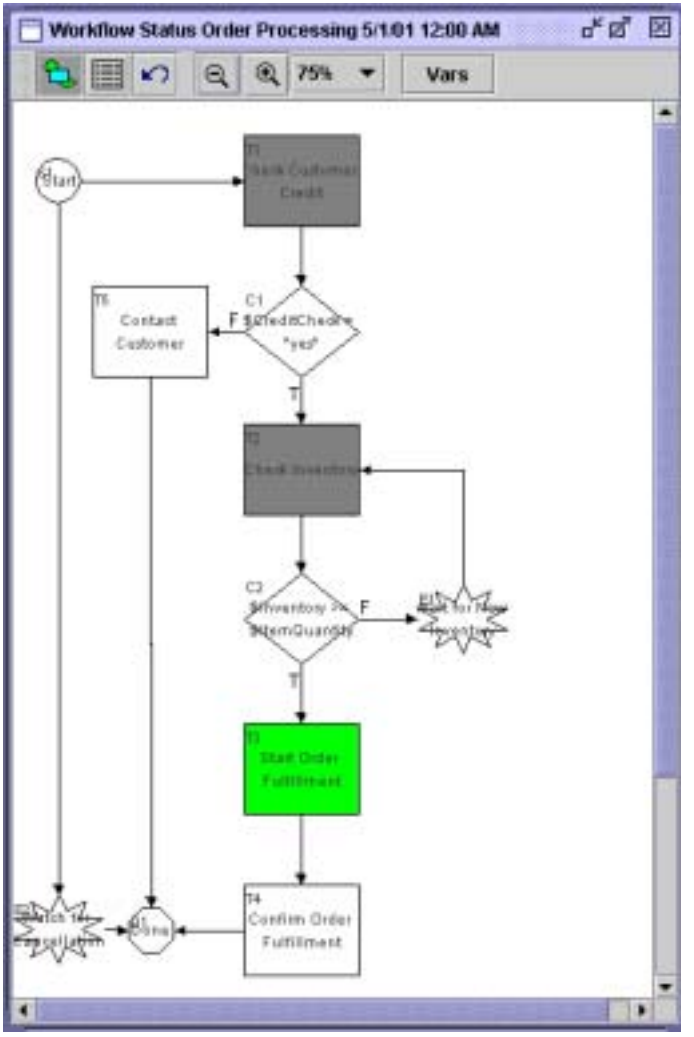
ウィンドウが開き、実行中のワークフローの現在のステータスを表すフローチャートまたはタスク リストが表示されます。ウィンドウの上部にあるボタンを使って、以下の操作を行います。

表 10-1 [ワークフロー ステータス] ウィンドウのボタン

ボタン	目的
	ワークフロー インスタンスのフローチャートをグラフで表示する。
	ワークフロー インスタンス内のすべてのタスクを表示する。
	ワークフロー ビューを更新する。

ワークフロー インスタンスの [フローチャート] ビューを表示するには、  ボタンをクリックします。

図 10-2 [ワークフロー ステータス] ウィンドウ : [フローチャート] ビュー



アクティブなタスクは緑、実行済みのタスクはグレー、非アクティブなタスクやその他のノードは白で表示されます（タスクのステータスについては、5-57 ページの 「タスクの状態を理解する」 を参照）。

デバッグの際にタスクのステータスを表す色の情報を使うと、ワークフロー内のどこで停止したかを識別できます。完了したはずのタスクが緑で表示されている（タスクはアクティブ化されたが実行されなかったことを示す）場合、ノードの定義にエラーがある可能性があります。

ワークフロー内の全タスクとその状態のリストを表示するには、  ボタンをクリックします。

図 10-3 [ワークフロー ステータス] ウィンドウ : [リスト] ビュー

リスト ビューには、各タスクに対して以下の情報が表示されます。

タスク	タスク名。
割り当て対象	タスクが割り当てられたユーザまたはロール。タスクの割り当てに関する詳細については、6-48 ページの「手動タスクの設定」を参照。
開始	タスクが開始された日付と時刻。
期限	タスクの期限。これは、ワークフロー内の [タスク期日を設定] アクションによって指定する。詳細については、6-56 ページの「タスク期日を設定する」を参照。
完了	タスクが完了した日付と時刻。
優先度	タスクに割り当てられた優先度。詳細については、5-59 ページの「タスクの優先順位について」を参照。

コメント	ワークフロー内の [タスク コメントを設定] アクションで指定した式から生成されたコメント。詳細については、6-58 ページの「タスクのコメントを設定する」を参照。このアクションが定義されていない場合、このカラムは空白になる。

タスクのステータスによって、以下のボックスが表示されます。

- ボックス無しで表示されたタスクは、非マニュアル タスクか、またはまだアクティブ化されていないタスクです（開始日および完了日は表示されない）。
- アクティブ化されたが保留中、つまり実行待ちのタスク（開始日付は表示されているが、完了日付が表示されていない）には、空のボックスが表示されます。
- 完了後のタスク（完了日付が表示される）には、チェックされたボックスが表示されます。
- 期限切れのタスク（期限が現在の日付の前）には、赤いボックスが表示されます。

[ワークフロー インスタンス] ダイアログ ボックスで、以下の操作を行うことができます。

- タスクのステータスと割り当てを変更します。詳細については、10-14 ページの「タスクのパーミッションと優先度の変更」を参照してください。
- タスクのパーミッションと優先度を変更します。詳細については、10-14 ページの「タスクのパーミッションと優先度の変更」を参照してください。

ワークフロー インスタンスの変数を表示し、更新する

ワークフローについて定義したすべての変数の現在値を、実行中いつでも表示および更新できます。

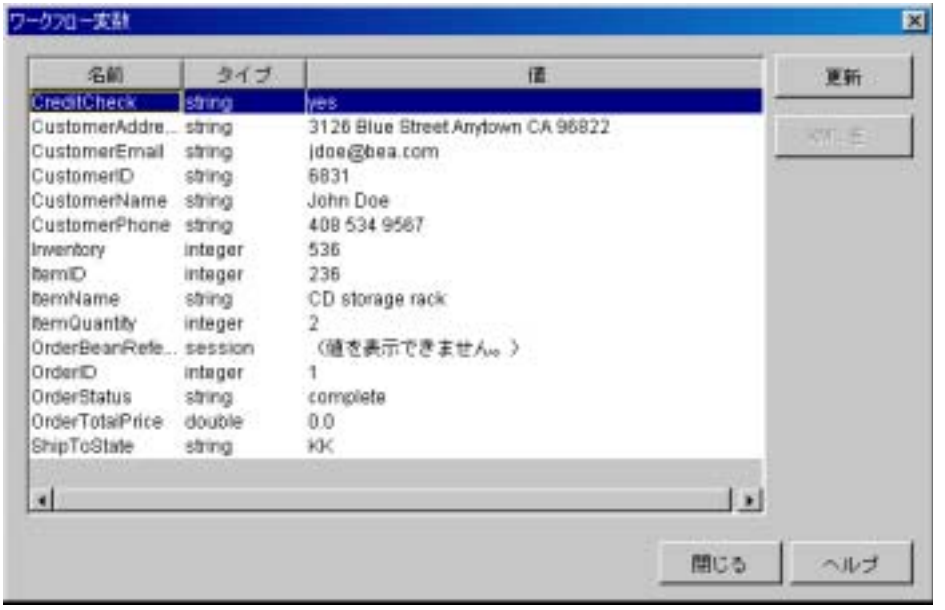
設計時にワークフローをデバッグするとき、変数値を表示すれば、設計エラーのトラブルシューティングがやりやすくなります。ワークフローが正しく実行されたように見える場合でも、設計のバグを示す間違った変数の値の設定が見つかる可能性があります。

ワークフロー インスタンス変数を表示する手順は、次のとおりです。

1. 以下のいずれか 1 つを実行します。
   - [ワークフロー インスタンス] ダイアログ ボックス内で、必要なワークフロー インスタンスを右クリックし、ポップアップ メニューから [変数] を選択します。
   - インスタンスの [ワークフロー ステータス] ウィンドウが表示されたら、[変数] ボタンをクリックします。

   [ワークフロー変数] ダイアログ ボックスに各変数がリストで表示され、その名前、型、およびカレント値が示されます（変数の型については、5-28 ページの「変数に関する作業」を参照）。

図 10-4 [ワークフロー変数] ダイアログ ボックス



2. XML 型の変数のみの内容を表示するには、リストで XML 変数を選択し、[XML を表示] をクリックします。変数の XML の内容を示す [XML を表示] ウィンドウが表示されます。

図 10-5 [XML を表示] ウィンドウ

3. [XML を表示] ウィンドウを終了するには、[閉じる] をクリックします。

変数の値を更新する手順は、次のとおりです。

1. [ワークフロー変数] ダイアログ ボックスから必要な変数を選択し、[更新] をクリックして [変数を設定] ダイアログ ボックスを表示します。

図 10-6 [変数を設定] ダイアログ ボックス

2. [値] フィールドに、変数の新しい値になる定数を入力します。

注意 : ワークフロー ロジックは変数の値に依存する場合が多いので、手動による変数値の変更は、注意して行ってください。また、変数の日付型に対して有効な値を入力する必要があります。

3. [OK] をクリックして変更を保存し、変数をリセットします。

ワークフロー インスタンスを削除する

日付に従って、1 つまたは複数のワークフロー インスタンスを削除できます。

1 つのワークフロー インスタンスを削除する手順は、次のとおりです。

1. [ワークフロー インスタンス] ダイアログ ボックス内のリストからワークフロー インスタンスを右クリックし、表示されたメニューから [削除] を選択します。
2. 「ワークフロー インスタンスを削除」という警告メッセージが表示されたら、[はい] をクリックしてインスタンスを削除する、または [いいえ] をクリックして操作を取り消します。

1 つのテンプレートについて複数のワークフロー インスタンスを削除する手順は、次のとおりです。

1. フォルダ ツリー内で、削除するワークフロー インスタンスのテンプレートまたはテンプレート定義フォルダを右クリックし、ポップアップ メニューから [インスタンスを削除] を選択します。[ワークフロー インスタンスを削除] ダイアログ ボックスが表示されます。

図 10-7 [ワークフロー インスタンスを削除] ダイアログ ボックス

2. 以下のオプションからいずれか 1 つを選択します。
   - [開始] ― このオプションを選択すると、指定した日付の範囲内に開始された、完了後のすべてのワークフロー インスタンスが削除されます。
   - [完了] ― このオプションを選択すると、指定した日付の範囲内に完了したワークフロー インスタンスが削除されます。
3. [開始] および [終了] ボックスで日付を選択し、選択したワークフロー内のすべてのインスタンスを削除する日付の範囲を指定します。
4. [OK] をクリックして選択したワークフロー インスタンスを削除する、または [取消し] をクリックして操作を取り消します。

ユーザ ワークリストとロールのワークリストの表示

ユーザのワークリストを表示するには、必要なオーガニゼーションのフォルダツリー内のリストからユーザを右クリックし、ポップアップ メニューから [ユーザ ワークリストを開く] を選択します。[ユーザ ワークリスト] ダイアログ ボックスに、ユーザに割り当てられたタスクのリストが表示されます。

図 10-8 [ユーザ ワークリスト] ダイアログ ボックス

ロールのワークリストを表示するには、必要なオーガニゼーションのフォルダツリー内のリストからロールを右クリックし、ポップアップ メニューから [ロール ワークリストを開く] を選択します。[ロール ワークリスト] ダイアログ ボックスに、ロールに割り当てられたタスクのリストが表示されます。

図 10-9 [ロール ワークリスト] ダイアログ ボックス

リストには、各タスクについて以下の情報が表示されます。

タスク	タスク名。
ワークフロー	タスクが定義されたワークフローの名前。
開始	タスクが開始された日付と時刻。
期限	タスクの期限。これは、ワークフロー内の [タスク期日を設定] アクションによって指定する。詳細については、6-56 ページの「タスク期日を設定する」を参照。
完了	タスクが完了した日付と時刻。
優先度	タスクに割り当てられた優先度。詳細については、5-59 ページの「タスクの優先順位について」を参照。
コメント	ワークフロー内の [タスク コメントを設定] アクションで指定した式から生成されたコメント。詳細については、6-58 ページの「タスクのコメントを設定する」を参照。このアクションが定義されていない場合、このカラムは空白になる。

タスクのステータスによって、以下のボックスが表示されます。

- まだアクティブ化されていないタスク（開始日付または完了日付が表示されない）には、ボックスが表示されません。
- アクティブ化されたが保留中、つまりユーザによって実行されていないタスク（開始日付は表示されているが、完了日付が表示されていない）には、空のボックスが表示されます。

- 完了後のタスク（完了日付が表示される）には、チェックされたボックスが表示されます。
- 期限切れのタスク（期限が現在の日付の前）には、赤いボックスが表示されます。

ユーザまたはロールに対するタスク リストをリフレッシュするには、  ボタンをクリックします。

[ロール ワークリスト] ダイアログボックスおよび [ユーザ ワークリスト] ダイアログ ボックスで、以下の操作を行うこともできます。

- タスクのステータスと割り当てを変更します。詳細については、10-14 ページの「タスクのパーミッションと優先度の変更」 を参照してください。
- タスクのパーミッションと優先度を変更します。詳細については、10-14 ページの「タスクのパーミッションと優先度の変更」 を参照してください。

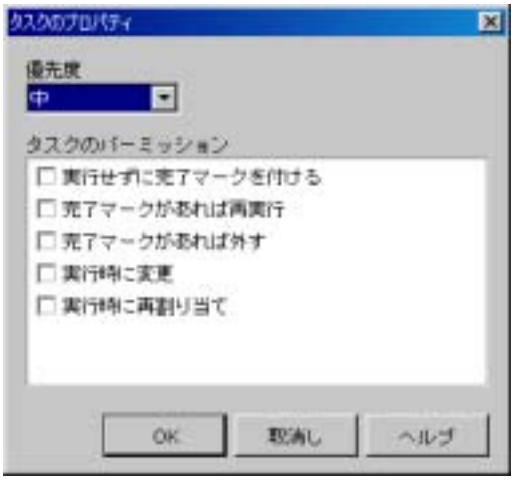
タスクのパーミッションと優先度の変更

タスクの定義に 実行時に変更パーミッションが割り当てられている場合は、タスクの優先度とパーミッションを変更できます（詳細は、5-55 ページの「タスクのプロパティを定義する」 を参照）。また、実行時に変更パーミッションがあると、Worklist のユーザはタスクのステータスと割り当てを変更できます。詳細については、10-16 ページの 「タスクのステータスと割り当ての変更」 を参照してください。

タスクのパーミッションと優先度を変更する手順は、次のとおりです。

1. [ワークフロー ステータス]、[ユーザ ワークリスト] または [ロール ワークリスト] の各ダイアログ ボックスで、再割り当てを行うタスクを右クリックします。ポップアップ メニューから [プロパティ] を選択して、[タスクのプロパティ] ダイアログ ボックスを表示します。

図 10-10 [タスクのプロパティ] ダイアログ ボックス



2. （省略可能）[優先度] フィールドで、タスクの優先度レベル（[低]、[中]、[高]）を選択します。
3. 以下の [タスクのパーミッション] チェック ボックスを必要に応じてチェックします。

パーミッション	説明
実行せずに完了マークを付ける	Worklist ユーザまたは Studio 管理者に、実行されていないタスクに完了マークを付けることを許可し、タスクの完了日付を手動で設定できる。
完了マークがあれば再実行	Worklist ユーザは、完了済みのタスクを再実行できる。
完了マークがあれば外す	Worklist ユーザまたは Studio 管理者に、完了マークが付いているタスクの状態を未完了に戻すことを許可する（ユーザが完了済みのタスクに完了済みではないというマークを付けると、タスクのステータスは Active に変更されるが、実行済みのアクションによる効果が取り消されることはない）。
実行時に変更	Worklist ユーザまたは Studio 管理者に、タスク実行前にタスクに対するパーミッションを変更することを許可する。

パーミッション	説明
実行時に再割り当て	Worklist ユーザに、タスクを受けるか再割り当てすることを許可する。あるいは、Studio 管理者に、タスク実行前にタスクを別のユーザまたはロールに再り割当てすることを許可する。

4. [OK] をクリックして、タスクへの変更を保存します。

タスクのステータスと割り当ての変更

タスクに設定されたパーミッションの範囲内で、現在実行中のワークフローに介入してタスクを別のユーザまたはロールに再割り当てしたり、タスクの完了済みのマークを付けたり削除したりできます。

注意： タスクの再割り当て、完了済みマークの添付と削除、プロパティの変更などを行うことができますが、Studio から起動したワークリストについてタスクを実行することはできません。タスクを実行するには、Worklist またはカスタム クライアント アプリケーションから行います。詳細については、『*WebLogic Integration Worklist ユーザーズ ガイド*』を参照してください。

タスクを再割り当てする

タスクの定義に 実行時に再割り当てパーミッションが割り当てられていて、タスクがまだ完了していない場合、別の Worklist またはカスタム クライアント アプリケーションのユーザ、ロール、またはロール内のユーザに、タスクを再割り当てできます（これらの違いやタスクの割当てについては、6-48 ページの「手動タスクの設定」を参照）。

タスクを再割り当てする手順は、次のとおりです。

1. [ワークフロー ステータス]、[ユーザ ワークリスト] または [ロール ワークリスト] の各ダイアログ ボックスで、再割り当てを行うタスクを右クリックします。ポップアップ メニューから [タスクの再割り当て] を選択して、[タスクを再割り当て] ダイアログ ボックスを表示します。

図 10-11 [タスクを再割り当て] ダイアログ ボックス

2. 以下のオプションからいずれか 1 つを選択します。
   - [ユーザ] ー特定のユーザにタスクを再割り当てします。
   - [ロール] ー指定したロールに属する任意のユーザにタスクを再割り当てします。
   - [ロール内のユーザ] ー指定したロールに属するユーザのうち、割り当てられたタスクの数が最小の人にタスクを再割り当てします。
3. [割り当て先] ドロップダウン リストから、タスクを再割り当てするユーザまたはロールの名前を選択します。
4. [OK] をクリックして、タスクの再割り当てを行います。

タスクのステータスに完了マークを付ける

タスクの定義に 実行せずに完了マークを付けるパーミッションが割り当てられている場合、ユーザによって実際にタスクが実行される前に、タスクのステータスを完了に変更できます。

タスクのステータスを完了に変更するには、[ワークフロー ステータス]、[ユーザ ワークリスト] または [ロール ワークリスト] の各ダイアログ ボックスで、完了ステータスに変更するタスクを右クリックします。ポップアップ メニューから、[タスクに完了マークを付ける] を選択します。タスクの完了日は現在の日付が設定され、ワークフローは先に進みます。

タスクのステータスの完了マークを外す

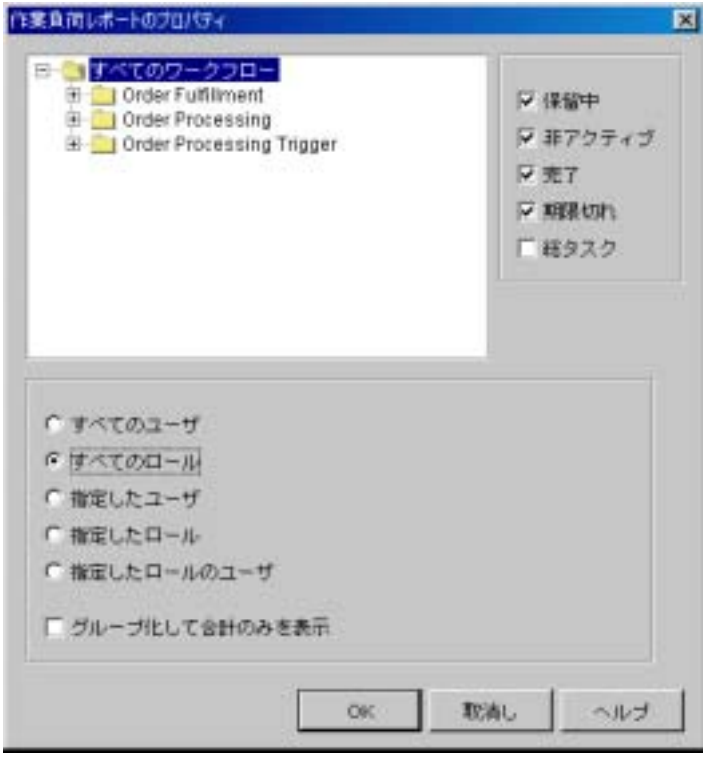
タスクの定義に 完了マークがあれば外すパーミッションが割り当てられている場合、完了済みのタスクのステータスを変更できます。

タスクのステータスを完了以外に変更するには、[ワークフロー ステータス]、[ユーザ ワークリスト] または [ロール ワークリスト] の各ダイアログ ボックスで、変更するタスクを右クリックします。ポップアップ メニューから、[タスクの完了マークを外す] を選択します。タスクのステータスが保留中にリセットされ、完了日付が消去されます。

作業負荷レポートの使用

Studio では、ワークフロー、タスク、ユーザまたはロール、およびタスクのステータスごとにタスク数を示す作業負荷レポートを表示できます。Studio のフォルダー ツリーで、[作業負荷レポート] を右クリックし、ポップアップ メニューから [開く] を選択して [作業負荷レポートのプロパティ] ダイアログ ボックスを表示します。

図 10-12 [作業負荷レポートのプロパティ] ダイアログ ボックス



作業負荷レポートの情報をコンパイルする

作業負荷レポートに、[作業負荷レポートのプロパティ] ダイアログ ボックスで選択した以下の内容に応じて、情報が表示されます。

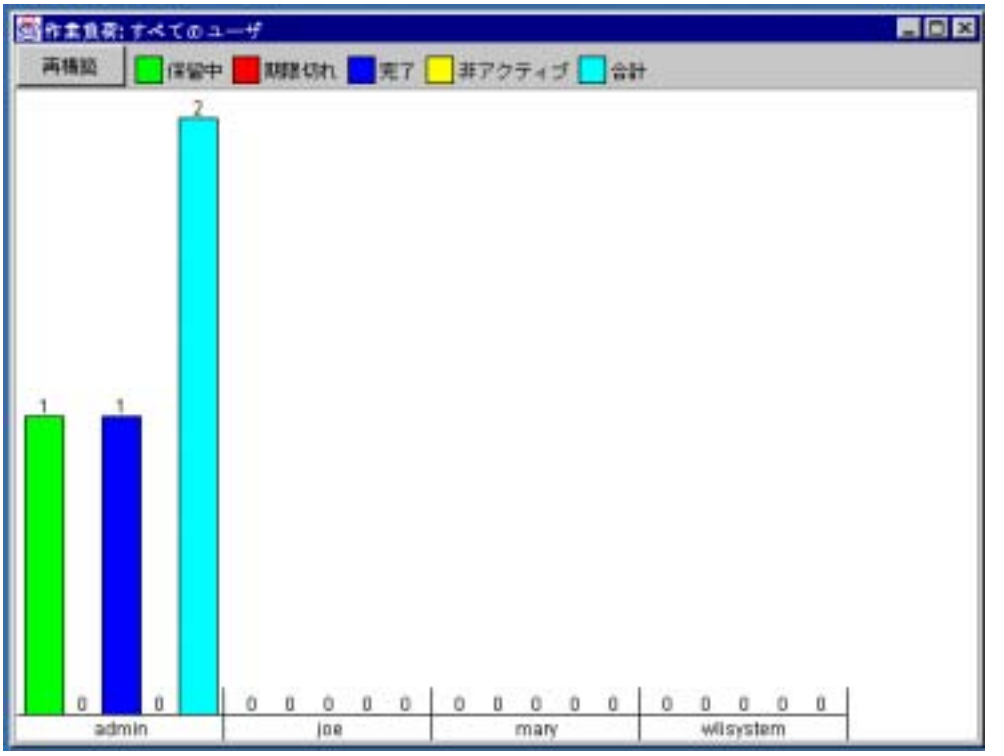
- [ワークフロー テンプレート] フォルダーワークフロー テンプレート内のすべてのワークフロー テンプレート定義に関する情報を含める場合、またはワークフロー テンプレートを展開して 1 つのワークフロー テンプレート定義のみを選択する場合に、ワークフロー テンプレートを選択します。選択したワークフロー テンプレートまたはワークフロー テンプレート定義内の情報のみが、レポートに表示されます。

- [保留中] －保留中のタスクを表示します。保留中のタスクとは、開始してまだ完了していないタスクです。
- [非アクティブ] －非アクティブ状態のタスクを表示します。非アクティブなタスクとは、開始前のタスクです。
- [完了] －完了したタスクを表示します。
- [期限切れ] －期日超過のタスクを表示します。期限切れのタスクとは、現在の日付またはそれ以前に期限が設定されている保留中のタスクです。
- [総タスク] －タスクの状態に関係なく、すべてのタスクを表示します（つまり、非アクティブ状態、保留中、および完了したタスクがすべて含まれる）。
- [すべてのユーザ] －各ユーザの作業負荷の情報を表示します。
- [すべてのロール] －各ロールの作業負荷の情報を表示します。
- [指定したユーザ] －表示されたユーザ リスト内で選択されたユーザの作業負荷の情報を表示します。
- [指定したロール] －表示されたロールリスト内で選択されたロールの作業負荷の情報を表示します。
- [指定したロールのユーザ] －表示されたロールリスト内で選択されたロールのメンバーであるユーザの作業負荷の情報を表示します。
- [グループ化して合計のみを表示] －上記の選択したオプションで示されるグループに関して、タスクの総件数のみを表示します。選択しなかった場合は、個々のユーザまたはロールが作業負荷レポートに示されます。

作業負荷レポートを表示する

作業負荷レポートの選択を行った後、[OK] をクリックして現在の作業負荷をグラフィック表示します。

図 10-13 [作業負荷レポート] ダイアログ ボックス



レポートの内容は、選択したオプションに基づいて、ユーザ、ロール、または総件数別に細分化されます。情報量が多い場合、この表示画面は左右にスクロールできます。棒グラフは、該当する各種の状態の合計タスク件数を示します。ウィンドウの最上部の凡例は、設計時に選択したオプションに応じて、それぞれの表示色の意味を示します。さらに正確な表示のために、実際のタスク件数が各棒グラフの上に表示されます。

統計レポートの使用

タスク、ユーザ、ロールなどを基本にした統計レポートを表示することができます。統計レポートには、完了したワークフローの履歴データに基づいた統計的情報が表示されます。このレポートには、完了したタスクの件数、各タスクに要した合計時間、平均時間、最小および最大時間が示されます。また、平均時間からの標準偏差も含まれます。

統計レポートの情報をコンパイルする

Studio のフォルダー ツリーで、[統計レポート] を右クリックし、ポップアップ メニューから [開く] を選択して [統計レポートのプロパティ] ダイアログ ボックスを表示します。このレポートには、[統計レポートのプロパティ] ダイアログ ボックスで選択した内容に応じた情報が含まれています。

図 10-14 [作業負荷レポートのプロパティ] ダイアログ ボックス

- [ワークフロー テンプレートのリスト] ―利用可能なワークフロー テンプレートの一覧。テンプレートを展開して、統計レポートに含めるワークフロー テンプレート定義を選択します。
- [すべての日付] ―統計情報の計算値を、日付と関係なく収集します。
- [開始] ―当該日付以降に完了したワークフロー内の全タスクを、統計計算に含めます。
- [終了] ―当該日付以前に完了したワークフロー内の全タスクを、統計計算に含めます。

- [すべてのユーザ] ―各ユーザの統計情報を表示します。
- [すべてのロール] ―各ロールの統計情報を表示します。
- [指定したユーザ] ―選択された各ユーザの統計情報を表示します。
- [指定したロール] ―選択された各ロールの統計情報を表示します。
- [指定したロールのユーザ] ―表示されたロールリスト内で選択されたロールのメンバーである各ユーザの統計情報を表示します。
- [グループ化] ―上記の選択したオプションで示されるグループに関して、タスクの総件数のみをレポートで表示します。選択しなかった場合は、個々のユーザまたはロールが作業負荷レポートに示されます。

統計レポートを表示する

統計レポートの選択を行った後、[OK] をクリックして統計レポートを表示します。

図 10-15 統計レポート

ダイアログ ボックスの最上部に、以下のオプションが表示されます。

- [構築 / 再構築] ―時間の単位または制度レベルの単位を変更した場合に、レポートを再作成します。
- [秒]、[分]、[時間]、[日] ―レポートで使用する時間の単位。

- [精度] ー統計情報で設定する小数点以下の桁数を選択できるドロップダウンリスト。

レポートの設計時に選択したオプションに応じて、レポートには以下のカラムが表示されます。

- [タスク] ー測定対象のタスクの名前。
- [ユーザ] ータスクに割り当てるユーザまたはロール。レポートの設計時に [グループ化] オプションを選択した場合、結果は全ユーザまたは全ロールに対して合計されるため、このカラムは表示されません。
- [数] ー完了したタスクの総件数。
- [合計] ータスクの実行に要した時間の合計。つまり、タスクの開始日付と完了日付の間の時間。
- [平均] ーこのタスクの実行に要した時間の平均値。
- [StdDev] ーこのタスクの実行に要した時間の平均値の標準偏差。
- [最小] ーこのタスクの実行に要した時間の最小値。
- [最大] ーこのタスクの実行に要した時間の最大値。

レポートの定義は、結果とは別の場所に格納されるので、統計レポートはいつでも実行できます。

11 ワークフロー パッケージのインポートとエクスポート

この章では、Java アーカイブ ファイルに対するワークフロー パッケージのインポートとエクスポートの方法について説明し、また XML ファイルに対するテンプレート定義のインポートとエクスポートの方法について説明します。

- インポートおよびエクスポート
- ワークフロー パッケージのエクスポート
- ワークフロー パッケージのインポート
- XML ファイルに対するワークフロー テンプレート定義のインポートとエクスポート

インポートおよびエクスポート

WebLogic Integration には、Java アーカイブ（JAR）ファイルにワークフロー オブジェクトをエクスポートする機能と、同ファイルからワークフロー オブジェクトをインポートする機能があります。この機能を使用すると、テンプレート、テンプレート定義、ビジネス オペレーション、ビジネス カレンダー、イベント キー テーブル、および XML リポジトリ項目をエクスポートおよびインポートできます。オーガニゼーション、ユーザ、ロールは、エクスポートもインポートもできません。

エクスポート済みのパッケージには*発行済み*パッケージのマークを付けることができますが、これはインポート時にその内容が読み取り専用であることを示します。発行済みのテンプレート内には、新しいテンプレート定義を作成できません。ただし、発行済みのテンプレートおよびテンプレート定義は削除できます。

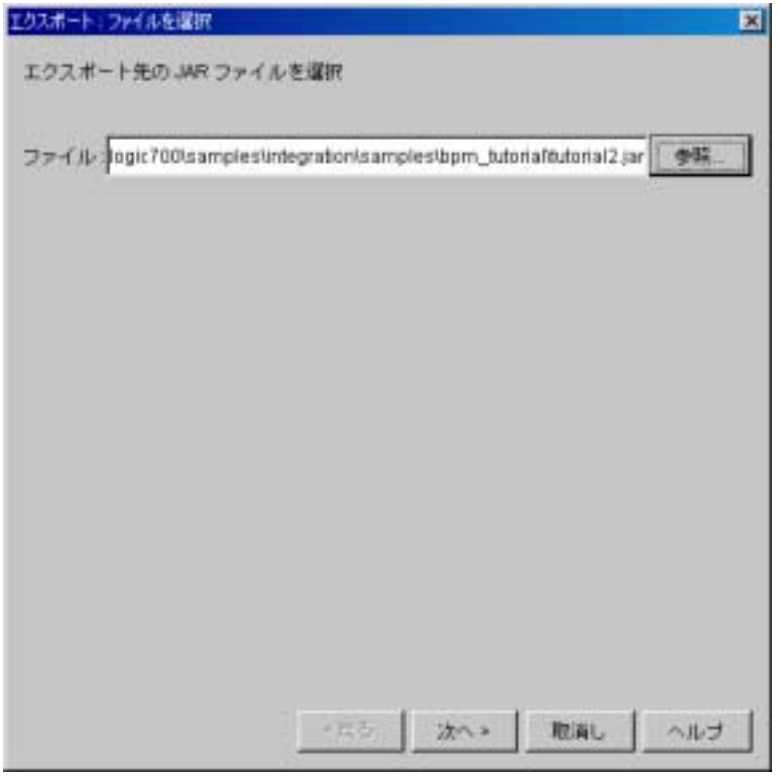
さらに、許可のないインポートから、エクスポートされたパッケージを保護するためのパスワードを割り当てることもできます。

ワークフロー パッケージのエクスポート

ワークフロー パッケージをエクスポートする手順は、以下のとおりです。

1. [ツール | パッケージをエクスポート] の順に選択して、[エクスポート : ファイルを選択] ダイアログ ボックスを表示します。

図 11-1 [エクスポート : ファイルを選択] ダイアログ ボックス



2. 以下のいずれかを実行して、パッケージをエクスポートする JAR ファイルのフル パス名を指定します。
   - パスとファイル名を、[ファイル] フィールドに直接入力します。

- [参照] をクリックして [保存] ダイアログ ボックスを表示し、[参照] フィールドからフォルダを選択します。次に、ファイルの名前を [ファイル名] フィールドに入力し、[保存] をクリックしてパスとファイル名を保存します。

3. [次] をクリックします。

   パスワードで保護されていない既存の JAR ファイルを指定した場合は、[エクスポート : エクスポートするコンポーネントを選択] ダイアログ ボックスが表示されます。

   パスワードで保護されている既存の JAR ファイルを指定した場合は、[エクスポート : パスワードを入力] ダイアログ ボックスが表示されます。必要なパスワードを入力し、[次] をクリックして [エクスポート : エクスポートするコンポーネントを選択] ダイアログ ボックスを表示する、または [戻る] をクリックして別のファイルを指定します。

図 11-2 [エクスポート : エクスポートするコンポーネントを選択] ダイアログ ボックス

注意 : [エクスポート : ファイルを選択] ダイアログ ボックスで、パッケージをエクスポートするユーザ システム上の既存の JAR ファイルを選択した場合は、選択した JAR ファイルの内容に基づいて、ツリー内の各項目が事前にチェックされます。

4. 該当するチェック ボックスをチェックする、またはチェックを外して、エクスポートするコンポーネントを指定する。エクスポート機能ではテンプレート定義の整合性が強制されないので、任意のオブジェクトの選択を解除して、エクスポートするパッケージから除外できる。

   注意 : エクスポート対象のテンプレート定義を選択した場合、テンプレート定義で参照するサブワークフローまたはビジネス オペレーションが自動的に選択されます。ただし、ビジネス カレンダーと XML エンティティは自動的には選択されない。したがって、エクスポート

パッケージにこれらを含める場合は、これらの項目を手動で選択する必要がある。

タスクの期日を指定するテンプレート定義をエクスポートし、かつその期日をビジネス カレンダーを使用して設定している場合は、テンプレート定義と共にカレンダーをエクスポートする必要がある。

同様に、XSL 変換 アクションを用いるテンプレート定義をエクスポートする場合は、その XSL 変換 アクションで必要なリポジトリの XML エンティティをエクスポートしてください。XML エンティティは、XSLT テンプレート ドキュメントまたは XML 入力ドキュメントのこと。

5. （省略可能）[パブリッシュされたパッケージ形式] のチェックボックスを選択し、対象システムにインポートした後で、発行済みパッケージのオブジェクトに読み取り専用のマークを付けます。
6. （省略可能）[パスワード保護] チェックボックスを選択してパッケージにパスワードを割り当て、[パスワード] および [確認] フィールドにパスワードを入力します。対象システム上のユーザは、パッケージをインポートするために、このパスワードを指定しなければなりません。
7. [エクスポート] をクリックして、エクスポートを開始します。エクスポート操作が完了すると、エクスポート操作が成功したかどうかを示す [エクスポートの概略を確認] レポートが表示されます。
8. [閉じる] をクリックして、[エクスポートの概略を確認] ダイアログ ボックスを閉じます。

ワークフロー パッケージのインポート

発行済みのパッケージ（以前にエクスポートして、発行済みのマークを付けたパッケージ）からテンプレート定義をインポートする場合に、テンプレートが対象システム上に存在しない場合は、発行済みのテンプレートが自動的に作成されます。ただし、既存のテンプレートにパブリッシュされたテンプレート定義をインポートしている場合、既存のテンプレートもパブリッシュされたものである必要があります。発行済みのテンプレート定義は、未発行のテンプレートにインポートできません。

同様に、パブリッシュされていないパッケージからテンプレート定義をインポートしていて、インポート先のシステム上にテンプレートが存在しない場合、パブリッシュされていないテンプレートが自動的に作成されます。パブリッシュされていないテンプレート定義を既存のテンプレートにインポートするには、既存のテンプレートもパブリッシュされていないものでなければなりません。未発行のテンプレート定義は、発行済みのテンプレートにインポートできません。

システム上の既存のオブジェクトと同じ名前を持つオブジェクトをインポートすると、インポート処理中に警告が表示されます。テンプレート、ビジネス オペレーション、およびビジネス カレンダーは上書きできます。テンプレート定義の上書きは許可されていないため、代わりに複数のコピーが作成されます。

インポートの後、[テンプレートのプロパティ] ダイアログ ボックスを使用し、また追加のオーガニゼーションをチェックすることにより、それぞれのテンプレートに追加のオーガニゼーションを対応付けることができます（手順については、5-6 ページの「テンプレート プロパティを更新する」を参照）。同じテンプレートを追加のオーガニゼーションに再インポートすると、複数のテンプレート定義が作成されてしまうため、決して再インポートしないでください。

ワークフロー パッケージをインポートする手順は、以下のとおりです。

1. [ツール | パッケージをインポート] の順に選択して、[インポート：ファイルを選択] ダイアログ ボックスを表示します。

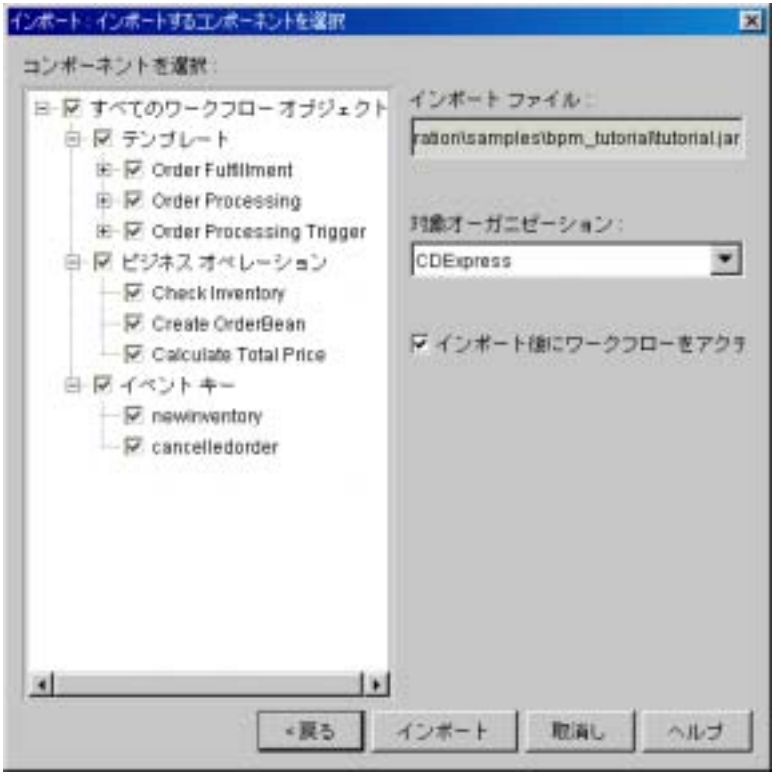
図 11-3 [インポート：ファイルを選択] ダイアログボックス

2. 以下のいずれかを実行して、パッケージをインポートする JAR ファイルのフル パス名を指定します。
   - パスとファイル名を、[ファイル] フィールドに直接入力します。
   - [参照] をクリックして [開く] ダイアログ ボックスを表示し、[参照] フィールドからフォルダを選択します。次に、ファイルの名前を [ファイル名] フィールドに入力し、[開く] をクリックしてパスとファイル名を保存します。
3. [次] をクリックします。

   インポートするパッケージがパスワードで保護されていない場合は、[インポート：インポートするコンポーネントを選択] ダイアログ ボックスが表示されます。

インポートするパッケージがパスワードで保護されていれば、[インポート：パスワードを入力] ダイアログ ボックスが表示されます。必要なパスワードを入力して [次] をクリックすると、[インポート：インポートするコンポーネントを選択] ダイアログ ボックスが表示されます。

図 11-4 [インポート：インポートするコンポーネントを選択] ダイアログ ボックス



4. インポートしないオブジェクトのチェック ボックスのチェックはすべて外す。

   注意： タスクの期日を指定するテンプレート定義をインポートし、かつその期日をビジネス カレンダーを使用して設定している場合は、テンプレート定義と共にカレンダーをインポートする必要がある。同様に、XSL 変換 アクションを用いるテンプレート定義をインポートする場合は、その XSL 変換 アクションで必要なリポジトリに XML エン

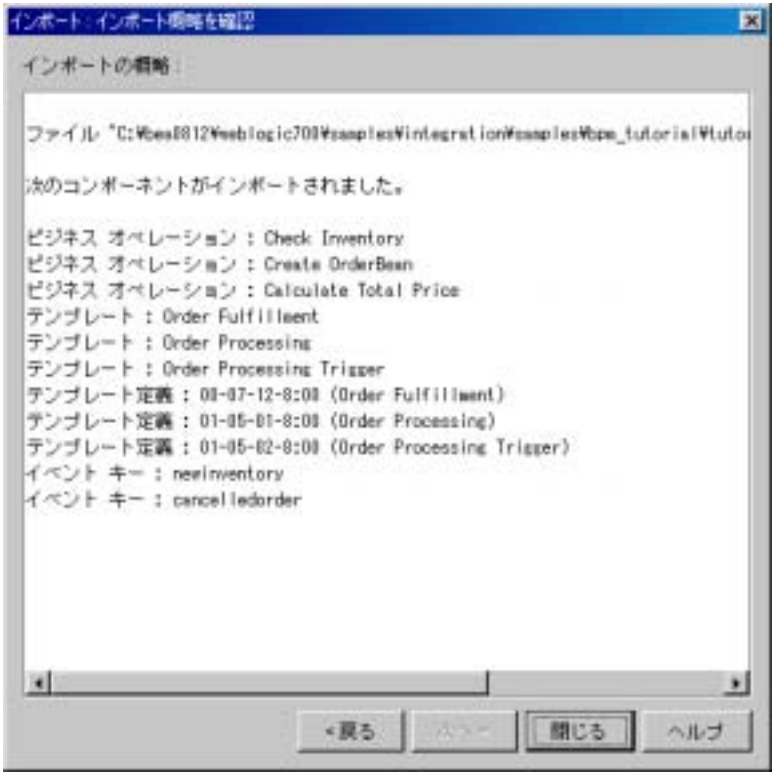
ティティを確実にインポートしてください。XML エンティティは、XSLT テンプレート ドキュメントまたは XML 入力ドキュメントのこと。

5. [対象オーガニゼーション] ドロップダウン メニューから、インポートしたテンプレート定義を最初に割り当てるオーガニゼーションを選択します。
6. （省略可能）インポートしたテンプレート定義をインポートの後に自動的にアクティブ化する場合は、インポート チェック ボックスの後の [アクティブ時] を選択します。テンプレート定義のアクティブ化については、5-13 ページの「テンプレート定義を更新、ラベリングおよびアクティブ化する」を参照してください。

   注意： テンプレート定義に、オブジェクトに対して未解決の参照がある場合は、アクティブ化できません。
7. [インポート] をクリックして、インポートを開始します。[取消し] をクリックすると操作はキャンセルされます。

   インポート オペレーションが完了すると、[インポート概略を確認] レポートが表示されます。このレポートには、インポートされたすべてのオブジェクトと、インポート操作中に検出された問題がリストアップされます。

図 11-5 [インポート概略を確認] ウィンドウ



8. [インポート概略を確認] ダイアログ ボックスを閉じるには [閉じる] をクリックし、前の画面に戻るには [戻る] をクリックします。

XML ファイルに対するワークフロー テンプレート定義のインポートとエクスポート

旧バージョンの WebLogic Process Integrator との互換性については、ユーザ コンピュータ上でマッピングされた任意ドライブの XML ファイルに対してワークフロー テンプレート定義をエクスポートでき、また XML フォーマットの（前にエクスポートした）ワークフロー テンプレート定義を Studio にインポートして、新しいワークフロー テンプレート定義を作成することができます。

注意： 旧バージョンの WebLogic Process Integrator との互換性を維持する必要がない場合は、この章で前述したパッケージのインポートおよびエクスポート機能の活用をお勧めします。

XML にワークフロー テンプレート定義をエクスポートする

ワークフロー テンプレート定義をエクスポートする手順は、次のとおりです。

1. フォルダ ツリーでエクスポートするテンプレート定義を右クリックします。
2. ポップアップ メニューから [エクスポート] を選択すると、[保存] ダイアログ ボックスが表示されます。
3. エクスポートしたワークフロー テンプレート定義を保存するドライブとディレクトリを選択します。
4. （省略可能）[ファイル名] フィールドで、ファイルの名前を選択するか、キー入力します。デフォルト名は、テンプレート名です。
5. [保存] をクリックしてテンプレート定義をエクスポートします。[取消し] をクリックすると操作はキャンセルされます。

XML からワークフロー テンプレート定義をインポートする

XML ファイルからインポートしたワークフロー テンプレート定義には、常に*非アクティブ状態*のマークが付けられます。インポートしたワークフロー テンプレート定義をインスタンス化するには、[テンプレート定義] ダイアログ ボックスで定義を*アクティブ*に変更する必要があります。詳細については、5-13 ページの「テンプレート定義を更新、ラベリングおよびアクティブ化する」 を参照してください。

テンプレートが存在しない場合は、テンプレート定義をインポートする前にテンプレートを作成する必要があります。テンプレートの作成の詳細は、5-4 ページの「ワークフロー テンプレートを作成する」 を参照してください。

注意 : ワークフローのインポートで警告メッセージが表示されることもあります。インポートしたワークフローが参照するビジネス オペレーションやイベント キーがある場合はそれらを定義し、インポートされたワークフローに含まれる [ビジネス オペレーションを実行] アクションや [ワークフローを開始] アクションを再定義する必要があります。

ワークフロー テンプレート定義をインポートする手順は、以下のとおりです。

1. Studio のフォルダ ツリーで、ワークフロー テンプレート定義のインポート先とするワークフロー テンプレートを右クリックします。

   注意 : ワークフローをインポートするテンプレートの名前が、インポートしている XML ファイルの名前と異なる場合は、警告メッセージが表示されます。

2. ポップアップ メニューから [テンプレート定義をインポート] を選択して、[開く] ダイアログ ボックスを表示します。

3. [開く] ダイアログ ボックスで、ハード ドライブ上の XML ファイルのカレント位置を選択し、[開く] をクリックします。

   注意 : ワークフロー テンプレート定義がエクスポートされたときに割り当てられた名前と異なる名前を使用してワークフロー テンプレート定義をインポートすることはできません。その場合は、警告メッセージが表示されます。

4. ファイルのインポート後に、インポート確認のダイアログ ボックスが表示され、インポートの確認を求めてきます。[はい] をクリックして、ワークフロー テンプレート定義をインポートします。最初に指定した日付と時間で、テンプレート定義が作成されます。

索引

い

え

お

か

き

け

こ

さ

し

せ

そ

た

つ

て

と

は

れ

ろ

わ