BEA WebLogic Integration™

BPM - Workshop サンプル ユーザーズ ガイド

リリース 7.0 SP2
マニュアルの日付 : 2003 年 2 月

著作権

限定的権利条項

商標または登録商標

BEA WebLogic Integration 7.0 サンプル 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	2003 年 2 月	7.0 SP2

目次

このマニュアルの内容

このマニュアルでは、WebLogic Integration BPM - Worklist 相互運用性サンプルの使用方法について説明します。

このマニュアルの内容は以下のとおりです。

- 第 1 章「概要」では、サンプルに示されている SOAP over HTTP および XML over HTTP という 2 種類のメッセージ交換について説明します。
- 第 2 章「サンプルの実行」では、サンプルを設定および実行する方法について説明します。
- 第 3 章「BPM-Workshop 相互運用性プロセス」では、サンプルの実行内容について説明します。

対象読者

このマニュアルは、Web サービスを使用して XML ドキュメントを交換するユーザを対象としています。

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と、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、WebLogic Integration の Web サイトで入手できます。PDF 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。

関連情報

以下の WebLogic Integration ドキュメントには、この製品の使用に関連した情報が含まれています。

- *WebLogic Integration BPM ユーザーズ ガイド*
- *WebLogic Integration Studio ユーザーズ ガイド*

サポート情報

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@beasys.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 ドキュメントのリリース番号をお書き添えください。

本バージョンの WebLogic Integration Worklist について不明な点がある場合、または Worklist 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。

- お名前、電子メール アドレス、電話番号、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。

表記法	適用

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`***
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。

表記法	適用

...	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 概要

WebLogic Integration BPM ñ Workshop サンプルは、Web サービスと Web ワークフローの間で XML ドキュメント交換による相互呼び出しがどのように行われるかを示します。このサンプルには次の 2 つのシナリオが示されています。

- Web サービスを呼び出して XML を渡し、後で Web サービスから XML を受け取るワークフロー
- ワークフローを呼び出して XML を渡し、後でワークフローから XML を受け取る Web サービス

また、このサンプルには次の 2 種類のメッセージ フォーマットが示されています。

- HTTP (Hypertext Transfer Protocol protocol) を使用した SOAP (Simple Object Access Protocol)。メッセージ交換は対話的に行われるか、非同期応答を伴います。
- HTTP を使用した未加工の XML。メッセージ交換は非対話的に行われます。

注意： BPM - Workshop サンプルは Windows システムでのみ動作します。

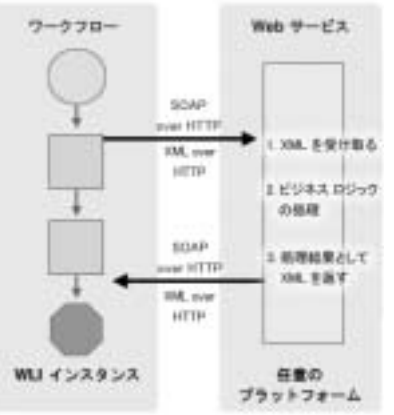
この章の内容は以下のとおりです。

- 相互運用のシナリオ
- サンプルでのシナリオの実行方法

相互運用のシナリオ

図 1-1 は、ワークフローから Web サービスを呼び出して XML メッセージを渡し、後で Web サービスからワークフローに XML メッセージを受け取る方法を示しています。

図 1-1 Web サービスを呼び出して XML を渡し、後で Web サービスから XML を受け取るワークフロー



このワークフローでは、Web サービスを呼び出して XML ドキュメントを渡すために、SOAP-HTTP または HTTP-XML を使用して XML メッセージを渡します。XML メッセージを受け取った Web サービスは、ビジネス ロジックの処理を開始します。その処理途中または処理後に、Web サービスは XML ドキュメントを含むメッセージをワークフローに送信します。

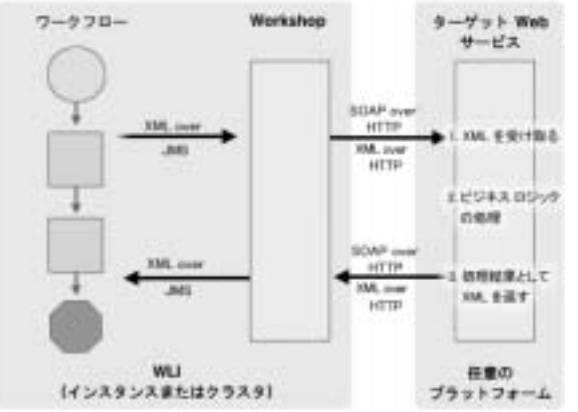
このシナリオは非同期の Web サービスの特徴を表しています。ワークフローは Web サービスをインスタンス化した後、ワークフロー自身の処理を自由に実行し、後から Web サービスに応答を要求できます。Web サービスの各インスタンスは、サービスを呼び出したワークフロー インスタンスに XML の結果を返さなければなりません。

注意： 厳密に言うと、XML over HTTP のサンプルの Web サービス メソッドは、同期メソッドです。これは、HTTP による XML のみのバインドをサポートする JWS メソッドが空の戻り値をサポートしないという WebLogic Workshop の特徴によるものです。そのため、Workshop JWS 内で非同期 XML/HTTP 専用メソッドを使用することはできません。サンプルでは、JWS でダミーの整数を戻り値として返すことにより、この制限に対処しています。

シナリオの実装方法

図 1-2 に示すように、仲介 Web サービスを使用することで、ターゲット Web サービスのプロトコルやメッセージ フォーマットからワークフローを切り離しています。

図 1-2 Web サービスを呼び出して XML を渡し、後で Web サービスから XML を受け取るワークフロー



仲介 Web サービスは、ターゲット Web サービスから受け取った応答 XML が、元の XML メッセージを送信して呼び出した Web サービス インスタンスと同一のインスタンスから送信されたことを保証します。また、ターゲット Web サービスから、そのサービスに元の XML ドキュメントを渡した正しいワークフロー インスタンスへ、XML の結果を伝達することも保証します。

ワークフローには、未加工の XML を仲介 Web サービスに転送し、後で仲介 Web サービスから XML 応答を受け取るメカニズムが必要です。これらの XML 転送は両方とも、同じ仲介 Web サービス インスタンスとワークフロー インスタンスの間で実行される必要があります。仲介 Web サービスを使用すると、ワークフローは、SOAP/XML over HTTP インタフェースをサポートする必要から解放されます。ワークフローは JMS プロシージャおよびコンシューマとして機能し、Web サービスが SOAP/XML メッセージ フォーマット変換を処理します。

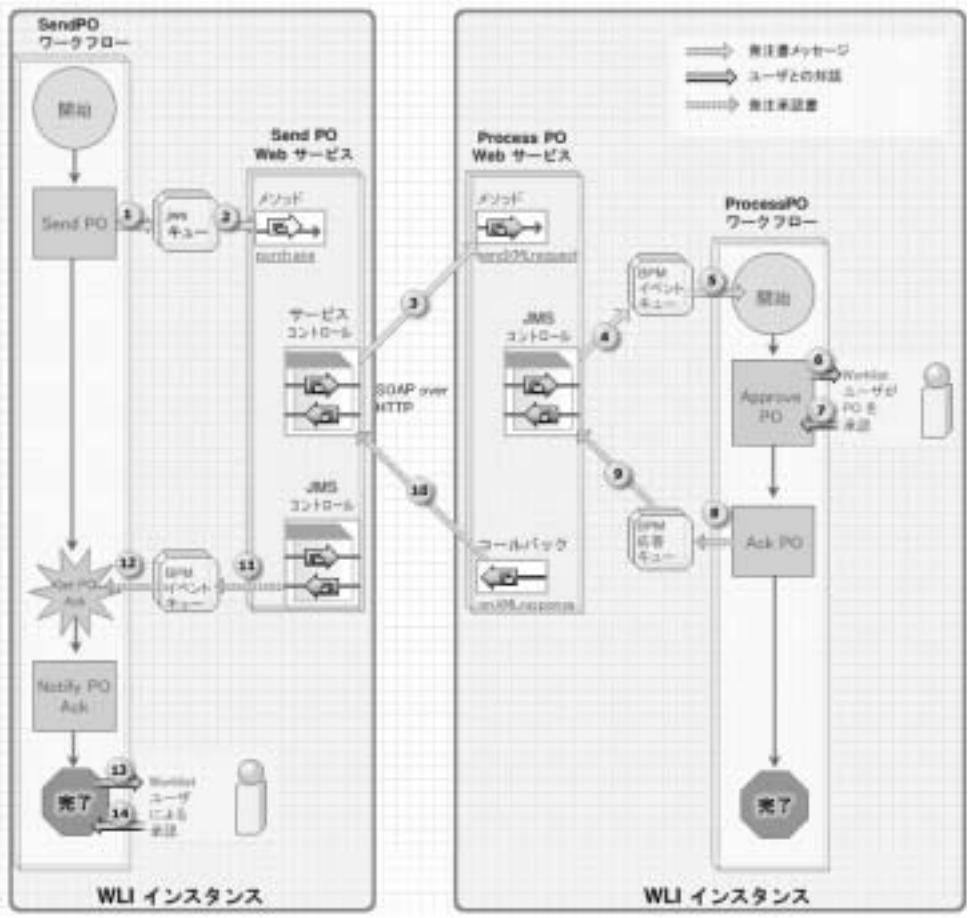
サンプルでのシナリオの実行方法

前述のとおり、このサンプルは、発注書情報を含む XML ドキュメントをワークフローで生成する方法を示しています。このサンプルでは、SendPO ワークフローは XML ドキュメントを使用して ProcessPO ワークフローをトリガします。ProcessPO ワークフローは最終的に、発注書情報の処理を完了した後、結果の XML ドキュメントを SendPO ワークフローに返します。

図 1-3 は、SOAP over HTTP サンプルの相互運用性プロセスを示しています。表 1-1 は、プロセスの詳細です。

図 1-4 は、XML over HTTP サンプルの相互運用性プロセスを示しています。表 1-2 は、プロセスの詳細です。

図 1-3　ワークフローと Workshop の対話 － SOAP



注意： 各ステップについては、表 1-1 を参照してください。

図 1-3 に示すように、ProcessPO ワークフローは SendPO ワークフローから呼び出されます。ProcessPO ワークフローを呼び出してその応答 XML を非同期に取得する手段として、Process PO Web サービスがデプロイされています。これにより、Process PO Web サービスが ProcessPO ワークフローを呼び出してその結果を受け取る際、SendPO ワークフローは実質的に Process PO Web サービスのク

ライアントとして動作することができます。これを可能にするには、SendPO ワークフローを実行する WebLogic Integration インスタンスから、HTTP 経由で Process PO Web サービスにアクセスできることが必要です。

Send PO Web サービスは、Process PO Web サービスの呼び出しと結果の受信に関する詳細を処理します。Send PO Web サービスは、結果の XML を受け取ると、それを SendPO ワークフローに伝達します。SendPO ワークフローと ProcessPO ワークフローは、HTTP 経由で非同期に XML ドキュメントを相互に送信する手段として、Web サービス Send PO および Process PO を使用します。

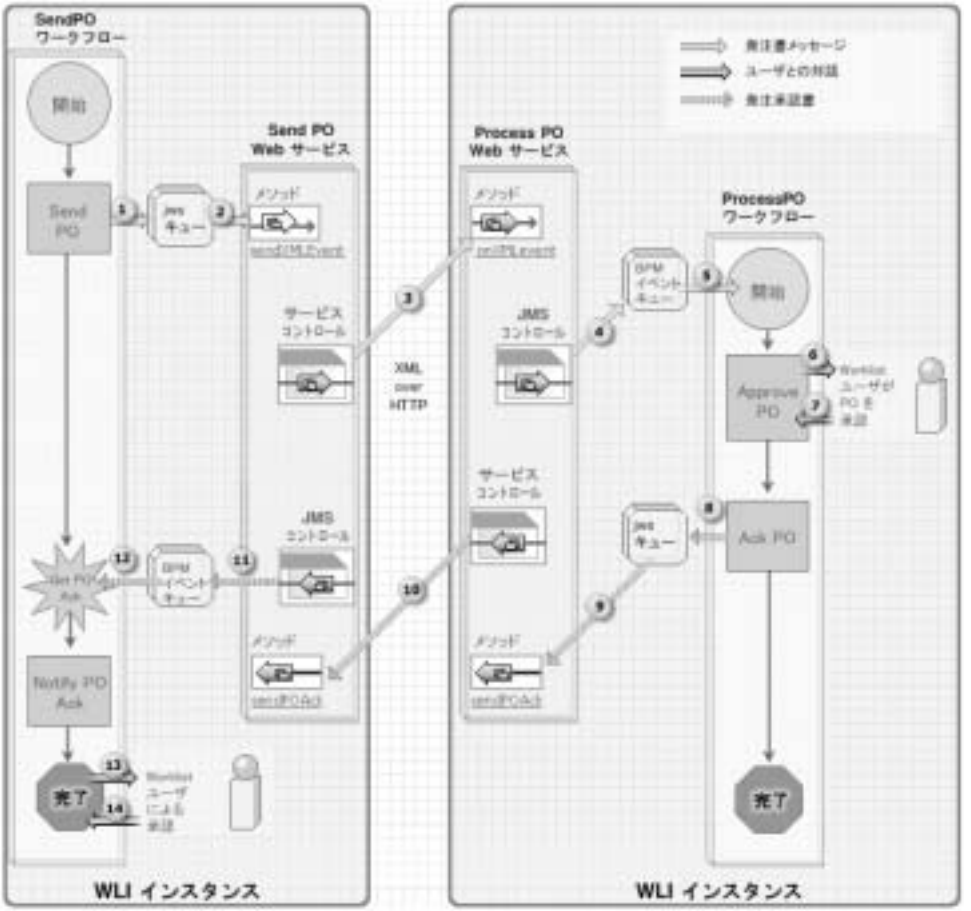
表 1-1 SOAP over HTTP サンプルで実行されるシナリオのステップ

ステップ	説明
1	SendPO ワークフローが Web サービス専用の JMS キュー (`jws.queue`) に、発注書を含む XML ドキュメントを送信する。XML ドキュメントには SendPO ワークフローのインスタンス ID も含まれている。SendPO Web サービスは、応答を返すときに (ステップ 11)、そのインスタンス ID を WLPIInstanceIDs という JMS プロパティに割り当てる。これにより次のことが保証される。 ■ 発注書を送信した SendPO ワークフロー インスタンスが、応答を受け取るワークフロー インスタンスと同一であること。 ■ 応答が到着した時点で SendPO ワークフロー イベント ノードである Get PO Ack がまだアクティブになっていない場合でも、応答が受け取られること。
2	Send PO Web サービスの purchase メソッドには、発注書 XML ドキュメントにマップされるパラメータがある。さらに、このメソッドは `jws.queue` に到着した XML メッセージを処理する。 **注意：** JMS コントロールの受信キューは、非請求メッセージを受信するためには使用できない。詳細については、WebLogic Workshop のオンライン ヘルプで「JMS コントロール : Web サービスから Java Message Service のキューとトピックを使用する」にある「JMS コントロールでサポートされないメッセージングのシナリオ」を参照。

ステップ	説明
3	Send PO Web サービスが発注書ドキュメントを Process PO Web サービスに送信する。Send PO Web サービスは、Process PO の jws ファイルを使用して作成されたサービス コントロールを使用する。Send PO Web サービスと Process PO Web サービスは、SOAP フォーマットの XML メッセージを HTTP 経由で交換する。SOAP メッセージ交換により、Send PO Web サービスと Process PO Web サービスが異なるマシン上で動作でき、SendPO ワークフローと ProcessPO ワークフロー間のリモート通信が可能になる。Send PO Web サービスと Process PO Web サービス間の SOAP メッセージ交換は対話的に行われ、2 つの Web サービス間の要求と応答の相関を保証するために、SOAP メッセージ会話 ID が SOAP ヘッダに格納される。このサンプルは 1 台のマシンで実行することもできる。
4	Process PO Web サービスが JMS コントロールを使用して、発注書 XML ドキュメントを BPM イベント キューに入れる。このキューは ProcessPO ワークフローによってモニタされている。JMS メッセージの JMSCorrelatioID ヘッダ フィールドには、送信メッセージを送った Process PO Web サービス インスタンスに JMS 応答メッセージが返されることを保証するために、SOAP 会話 ID が格納されている。
5	ProcessPO ワークフローが BPM イベント キューから XML メッセージを取り出し、処理を開始する。
6	ProcessPO が WLI Worklist ユーザにタスクを割り当て、発注書の受け入れを要求する。
7	ユーザがタスクを実行し、発注書を受け入れる。
8	ProcessPO ワークフローが発注承認書 XML メッセージを BPM 応答キューに入れる。承認書メッセージのヘッダ フィールド JMSCorrelationID には、意図された Process PO Web サービス インスタンスだけがこのメッセージを受け取ることを保証するために、SOAP メッセージ会話 ID が格納されている。
9	Process PO Web サービスには JMS コントロールがあり、そのコントロールによって、BPM 応答キューから発注承認書 XML メッセージが取り出される。

ステップ	説明
10	Process PO Web サービスがコールバック メソッドを呼び出し、発注承認書メッセージを SendPO Web サービスに転送する。前述のとおり、いずれかの SOAP over HTTP を使用して Web サービス間の通信が行われる。
11	サービス コントロールのコールバックが JMS コントロールを使用して、発注承認書 XML メッセージを BPM イベント キューに入れる。
12	SendPO ワークフローが BPM イベント キューから発注承認書 XML メッセージを取り出す。Send PO ワークフローのワークフロー インスタンス ID がメッセージ ヘッダに格納されたアドレス指定メッセージの受信によって、Get PO Ack イベント ノードがトリガされる。これにより、発注書ドキュメントを送信したワークフロー インスタンスのイベント ノードが PO 承認メッセージの受信でトリガされることを保証している。
13	SendPO ワークフローが WebLogic Integration Worklist ユーザにタスクを割り当て、発注承認書の受け入れを要求する。
14	ユーザがタスクを実行し、発注承認書を受け入れる。

図 1-4　ワークフローと Workshop の対話 － XML



注意： 各ステップについては、表 1-2 を参照してください。

図 1-4 に示すように、ProcessPO ワークフローは SendPO ワークフローから呼び出されます。ProcessPO ワークフローを呼び出し、その応答 XML を非同期に取得するための手段として、Process PO Web サービスがデプロイされています。これにより、Process PO Web サービスが ProcessPO ワークフローを呼び出してその結果を受け取る際、SendPO ワークフローは実質的に Process PO Web サービ

スのクライアントとして動作することができます。これを可能にするには、SendPO ワークフローを実行する WebLogic Integration インスタンスから、HTTP 経由で Process PO Web サービスにアクセスできることが必要です。

Send PO Web サービスは、Process PO Web サービスの呼び出しと結果の受信に関する詳細を処理します。Send PO Web サービスは、結果の XML を受け取ると、それを SendPO ワークフローに伝達します。SendPO ワークフローと ProcessPO ワークフローは、HTTP 経由で非同期に XML ドキュメントを相互に送信する手段として、Web サービス Send PO および Process PO を使用します。

表 1-2 XML over HTTP サンプルで実行されるシナリオのステップ

ステップ	説明
1	SendPO ワークフローが Web サービス専用の JMS キュー (`jws.queue`) に、発注書を含む XML ドキュメントを送信する。XML ドキュメントには SendPO ワークフローのインスタンス ID も含まれている。Send PO Web サービスが PO Ack 応答を返した時点で (ステップ 11)、メッセージ ペイロードには、SendPO ワークフローのインスタンス ID が含まれている。SendPO ワークフロー インスタンスの Get PO Ack イベント ノードは、メッセージ ペイロードにそのインスタンス ID を含むイベント キーを検索する。 **注意：** この形式の要求と応答の相関は、SOAP のサンプルとは異なる。SOAP のサンプルでは、SendPO ワークフローの Get PO Ack イベント ノードは、アドレス指定メッセージの受信によってトリガされる。
2	Send PO Web サービスの `sendXMLEvent` メソッドには、発注書 XML ドキュメントにマップされるパラメータがある。さらに、このメソッドは `jws.queue` に到着した XML メッセージを処理する。 **注意：** JMS コントロールの受信キューは、非請求メッセージを受信するためには使用できない。詳細については、WebLogic Workshop のオンライン ヘルプで「JMS コントロール : Web サービスから Java Message Service のキューとトピックを使用する」にある「JMS コントロールでサポートされないメッセージングのシナリオ」を参照。

ステップ	説明
3	Send PO Web サービスが、Process PO Web サービスの `onXMLEvent` メソッドを呼び出すことによって、Process PO Web サービスに発注書 XML ドキュメントを送信する。Send PO Web サービスは、Process PO Web サービスの jws ファイル (`WLW2BPM.jws`) を使用して作成されたサービス コントロールを使用する。Send PO Web サービスと Process PO Web サービスは、未加工の XML メッセージを HTTP 経由で交換する。 **注意：** これは SOAP サンプルとは異なる。SOAP サンプルでは、2 つの Web サービスが SOAP メッセージを使用して相互に通信する。
4	Process PO Web サービスが JMS コントロールを使用して、発注書 XML ドキュメントを BPM イベント キューに入れる。このキューは ProcessPO ワークフローによってモニタされている。メッセージ ペイロードには SendPO ワークフローのインスタンス ID が格納されている。
5	ProcessPO ワークフローが BPM イベント キューから XML メッセージを取り出し、処理を開始する。
6	ProcessPO が WebLogic Integration Worklist ユーザに発注書の受け入れを要求するタスクを割り当てる。
7	ユーザがタスクを実行し、発注書を受け入れる。
8	ProcessPO ワークフローが発注承認書 XML メッセージを Workshop JMS キュー (`jws.queue`) に入れる。メッセージ ヘッダには、メッセージが JMS キューで受信されたときに呼び出される Process PO Web サービス メソッド (`WLW2BPM.jws` の sendPOAck) を識別する情報が格納されている。
9 **10**	Process PO Web サービスの `sendPOAck` メソッドが、Send PO Web サービスに対して定義されているサービス コントロール (`BPM2WLWControl.ctrl`) を通して、Send PO Web サービスの `sendPOAck` メソッドを呼び出す。前述のとおり、Process PO Web サービスと Send PO Web サービス間のメッセージ交換は、HTTP を使用して未加工の XML フォーマットで行われる。

ステップ	説明
11	Send PO Web サービスが JMS コントロールを使用して、発注承認書 XML メッセージを BPM イベント キューに入れる。承認書メッセージのメッセージ ペイロードにはSendPO ワークフローのインスタンス ID が格納されている。
12	SendPO ワークフローが BPM イベント キューから発注承認書 XML メッセージを取り出す。Get PO Ack イベント ノードは、イベント キー値式を使用して、それ自身と同じワークフロー インスタンス ID を含むメッセージだけを受け入れる。
13	SendPO ワークフローが WebLogic Integration Worklist ユーザに発注承認書の受け入れを要求するタスクを割り当てる。
14	ユーザがタスクを実行し、発注承認書を受け入れる。

WebLogic Integration のインスタンスが 1 つである理由

サンプルは WebLogic Integration の 1 つのインスタンスで実行されます。実際には、2 つのインスタンスが HTTP 経由で相互に接続できることを条件に、SendPO ワークフローと Send PO Web サービスを 1 つのインスタンスにデプロイし、Process PO Web サービスと ProcessPO ワークフローを別のインスタンスにデプロイできます。

サンプルのステップを一度に 1 つずつ実行できるように、サンプル実行中は WebLogic Integration Swing Worklist クライアントを使用して、設定された間隔の 2 回に 1 回ユーザからの入力が取得されます。Worklist との対話が果たす機能は、ユーザが一度に 1 つのセクションの実行を見ることができるように、サンプルの実行をセクションに分割することだけです。

2　サンプルの実行

ワークフローの設定

この章では、BEA WebLogic Integration BPM と WebLogic Workshop の相互運用性サンプルの実行方法を説明します。

要件

注意： このサンプルを実行するには、標準インストール オプションを使用して WebLogic Platform SP2 をインストールしておく必要があります。カスタム インストールでは、サンプルで使用する Platform Domain はインストールされません。詳細については、『WebLogic Platform のインストール』(http://edocs.beasys.co.jp/e-docs/platform/docs70/install/index.html) を参照してください。

サンプルを実行するには、WebLogic Integration および WebLogic Workshop に関する基本知識が必要です。以下の概念に精通している必要があります。

- HTTP または JMS をサポートする Web サービスの作成
- Web サービスでのサービス コントロールまたは JMS コントロールの使用
- Web サービスへの対話サポートの追加

詳細については、WebLogic Workshop のマニュアル (http://edocs.beasys.co.jp/e-docs/workshop/docs70/index.html) および『*BPM ユーザーズ ガイド*』(http://edocs.beasys.co.jp/e-docs/wli/docs70/bpmtutor/index.htm) を参照してください。

サンプルの設定方法

サンプルを設定するには、次の手順を実行します。

手順 1: Domain Configuration Wizard を実行する

注意： WLW_BPM ドメインを作成済みの場合は、2-7 ページの「手順 2 : WebLogic Server をコンフィグレーションする」に進んでください。

1. [スタート | プログラム | BEA WebLogic Platform 7.0 | Domain Configuration Wizard] を選択し、Domain Configuration Wizard を起動します。
2. Configuration Wizard が実行中の状態になった後、図 2-1 に示すように、Platform Domain テンプレートを選択します。

図 2-1　BEA Configuration Wizard

3. [名前] フィールドに「WLW_BPM」と入力し、[Next] をクリックします。

4. [サーバ タイプを選択] ウィンドウで [Single Server] を選択し、[Next] をクリックします。
5. [ドメインの場所を選択] ウィンドウで、WLW_BPM ドメインの場所として以下のディレクトリを指定し、[Next] をクリックします。

   BEA_HOME\user_projects\

   この指定によって *BEA_HOME*\user_projects\WLW_BPM ディレクトリが作成されます。
6. 図 2-2 に示すように、[スタンドアロン / 管理サーバのコンフィグレーション] ウィンドウでデフォルト値を選択し、[Next] をクリックします。

図 2-2 スタンドアロン / 管理サーバのコンフィグレーション

7. [管理ユーザを作成] ウィンドウ (図 2-3) で、ユーザ名とパスワードを入力し、[Next] をクリックします。

図 2-3 管理ユーザを作成

8. [データベース メール セッションのコンフィグレーション] ウィンドウで、電子メールのアドレスとホストを入力し、[Next] をクリックします。

   注意 : このサンプルでは、ワークフローから電子メールは送信されません。しかし、先に進むにはこの情報を入力する必要があります。

9. *Windows のみ* : [サーバの [スタート] メニュー エントリを作成] ウィンドウで、目的のラジオ ボタンを選択し、[Next] をクリックします。不確かな場合は [No] を選択します。

図 2-4 サーバの [スタート] メニュー エントリを作成

10. [コンフィグレーションの概要] ウィンドウ (図 2-5) で、[Create] をクリックします。ドメインが作成されます。

図 2-5 コンフィグレーションの概要

11. [コンフィグレーション ウィザードが完了しました] ウィンドウ (図 2-6) で、[コンフィグレーションを終了します] ラジオ ボタンが選択されていることを確認し、[Done] をクリックします。

図 2-6 コンフィグレーション ウィザード完了

手順 2 : WebLogic Server をコンフィグレーションする

1. 以下を実行して WebLogic Server を起動します。

   ```
   BEA_HOME\user_projects\WLW_BPM\startWebLogic.cmd
   ```

   サーバが起動開始し、Pointbase データベースが作成されます (データベースの作成は、新しいドメインのサーバを最初に起動した場合にのみ実行されます)。

2. データベースが作成された後、コマンド ラインでユーザ名とパスワードの入力を求められます。Domain Configuration Wizard で作成したユーザ名とパスワードを入力します。

3. 起動プロセスの終わりに、コマンド ラインでクレジット カードの暗号化 / 暗号化解除キーを作成するかどうか確認を求められます。「y」と入力します。

   「< サーバが **RUNNING** モードで起動しました >」という行が表示されるまで待ってから、次の手順に進みます。

注意： JMS 応答キューを以前に作成済みの場合は、2-10 ページの「手順 3 : WebLogic Integration Studio を設定する」に進みます。

4. サーバが実行中であることがコマンド ラインに表示された後、次のようにして JMS 応答キューを作成します。

   a. ブラウザを開き、次の URL を入力します。

      ```
      http://localhost:7501/console
      ```

      WebLogic Server Administration Console が開きます。

   b. Domain Configuration Wizard で作成したユーザ名とパスワードを入力し、[サインイン] をクリックします。図 2-7 に示す WebLogic Server ホーム ページが表示されます。

図 2-7 WebLogic Server Administration Console

   c. 左ペインで、[WLW_BPM ｜サービス｜ JMS ｜サーバ] の順にノードを開きます。

   d. [WLIJMSServer] をクリックします。図 2-8 に示す [WLIJMSServer] ページが表示されます。

図 2-8　WLIJMSServer

e. [送り先のコンフィグレーション] をクリックします。[JMS 送り先] ページが表示されます。

f. [新しい JMSQueue のコンフィグレーション] をクリックします。図 2-9 に示す [新しい JMSQueue の作成] ページが表示されます。

図 2-9　新しい JMSQueue の作成

g. [名前] フィールドに「WLW_BPM_EVENT_RESPONSE」と入力します。

h. [JNDI 名] フィールドに「com.bea.wli.bpm.WLWResponseQueue」と入力します。

i. [作成] をクリックします。新しい送り先がツリーの [サーバ \WLIJMSSERVER\ 送り先] ディレクトリの下に表示されます。

手順 3 : WebLogic Integration Studio を設定する

1. [スタート | プログラム | BEA WebLogic Platform 7.0 | WebLogic Integration 7.0 | Studio] を選択して、Studio を起動します。
2. 次のように指定して Studio にログインします (図 2-10 を参照)。
   - [ユーザ名]: wlisystem
   - [パスワード]: wlisystem
   - [サーバ URL]: t3://localhost:7501

   注意 : 使用するポート番号が、Domain Wizard で選択したポート番号と同じであることを確認してください。

図 2-10 図 4-1: WebLogic Integration Studio へのログオン

3. 次のように BPMWLW という名前のオーガニゼーションを作成します。

   注意 : SOAP-HTTP または XML-HTTP サンプルを実行して BPMWLW オーガニゼーションをすでに作成済みの場合は、すべてのテンプレート定義を削除してから、次の手順に進んでください。テンプレート定義を

削除するには、[オーガニゼーション] ツリーで個々の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 [削除] を選択します。その後、手順 4 に進みます。

a. Studio のメニューから [コンフィグレーション | オーガニゼーション] を選択します。[オーガニゼーションを定義] ウィンドウが表示されます。

b. [追加] をクリックし、「BPMWLW」というオーガニゼーションを作成します。

c. [オーガニゼーションを定義] ウィンドウが表示されたら、BPMWLW オーガニゼーションを選択し、[閉じる] を選択します。

4. Studio の [オーガニゼーション] ドロップダウン リストから BPMWLW を選択します。

5. 次のように BPM_WLW ワークフローをインポートします。

a. Studio のメニューから [ツール | パッケージをインポート] を選択します。

b. SOAP over HTTP サンプルを設定する場合は、*BEA_HOME*/weblogic700/samples/integration/samples/BPM-WLW/workflows ディレクトリから bpmwlwsoaphttp.jar をインポートします。

XML over HTTP サンプルを設定する場合は、*BEA_HOME*/weblogic700/samples/integration/samples/BPM-WLW/workflows ディレクトリから bpmwlwxmlhttp.jar ファイルをインポートします。

この *BEA_HOME* は、WebLogic Platform のホーム ディレクトリです。

c. [インポート：インポートするコンポーネントを選択] ウィンドウで、すべてのコンポーネントが選択されていること、および [インポート後にワークフローをアクティブ化] チェック ボックスがチェックされていることを確認し、[インポート] を選択します。

注意： 既存のテンプレート定義とイベント キーを上書きするかどうか確認するメッセージが表示されたら、[すべてはい] をクリックします。これにより、以前に実行したサンプルのテンプレート定義が新しい定義で上書きされます。

図 2-11 インポートするコンポーネントを選択

d. [インポート概略を確認] ウィンドウで、[閉じる] をクリックします。テンプレートが [オーガニゼーション] ツリーに表示されます。

手順 4 : WebLogic Workshop を設定する

1. サンプルの jar ファイルを次のように圧縮解除します。

   a. SOAP over HTTP サンプル ー *BEA_HOME*\weblogic700\samples\integration\samples\BPM-WLW\lib\wlw_bpm_soap.jar ファイルを BEA_HOME\user_projects\WLW_BPM\applications ディレクトリに圧縮解除します。

      wlw_bpm_soap ディレクトリが作成されます。このディレクトリには BPM2WLW.jws ファイルと WLW2BPM.jws ファイルが入っています。

b. XML over HTTP サンプル ー *BEA_HOME*\weblogic700\samples\integration\samples\BPM-WLW\lib\wlw_bpm_xml.jar ファイルを BEA_HOME\user_projects\WLW_BPM\applications ディレクトリに圧縮解除します。

wlw_bpm_xml ディレクトリが作成されます。このディレクトリには BPM2WLW.jws ファイルと WLW2BPM.jws ファイルが入っています。

2. [スタート | プログラム | BEA WebLogic Platform 7.0 | WebLogic Workshop | WebLogic Workshop] を選択して、WebLogic Workshop を起動します。Workshop のウィンドウが表示されます。

注意： サンプルをすでに実行したことがある場合は、Workshop で自動的にプロジェクトが開始され、サーバが起動します。[ファイル] メニューから [プロジェクトを閉じる] を選択します。手順 5 に進みます。

3. Workshop のメニューで、[ツール | プリファレンス | パス] を選択し、次のように設定します。

a. [コンフィグレーション ディレクトリ] を「BEA_HOME\user_projects」に設定します。

b. その他のパラメータを次のように設定します。

- [名前]: localhost
- [ポート]: 7501
- [ドメイン]: WLW_BPM

c. [OK] をクリックします。

図 2-12　パス情報の設定

4. [OK] をクリックします。[プリファレンス] ウィンドウが閉じます。
5. Workshop のメニューから [ファイル | プロジェクトを開く] を選択します。
6. [プロジェクトを開く] ウィンドウで、`wlw_bpm_soap` または `wlw_bpm_xml` プロジェクトを選択し、[開く] をクリックします。

手順 5 : Web サービスを起動する

1. 次のように Send PO Web サービス (`BPM2WLW.jws`) を起動します。
   a. Workshop のプロジェクト ツリーで、`bpm2wlw` フォルダを開きます。

b. BPM2WLW.jws ノードをダブルクリックします。Send PO Web サービスが右ペインに表示されます。図 2-13 (SOAP over HTTP) または図 2-14 (XML over HTTP) を参照してください。

図 2-13　Send PO Web サービス － SOAP over HTTP

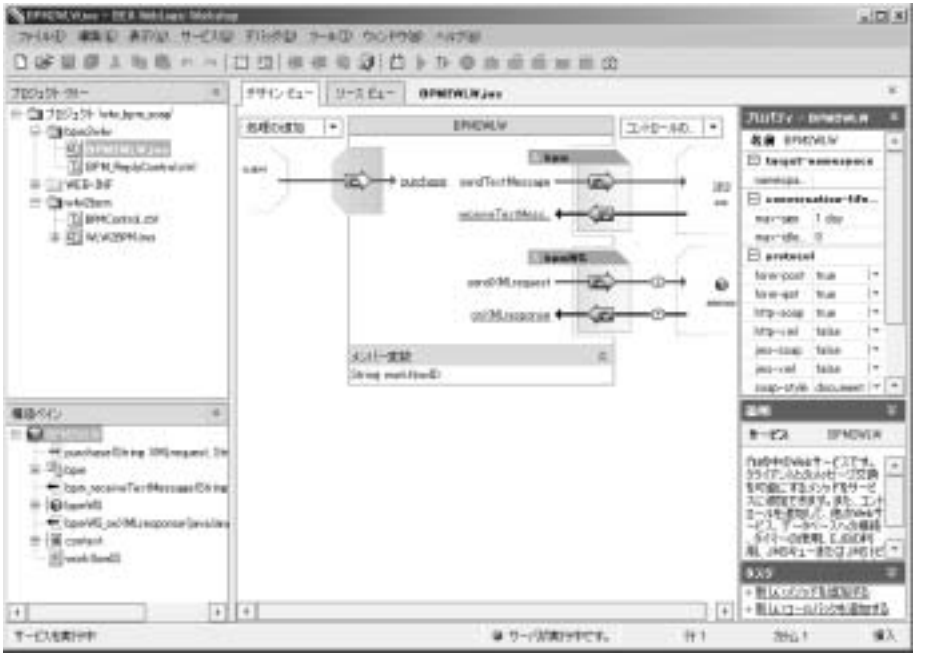
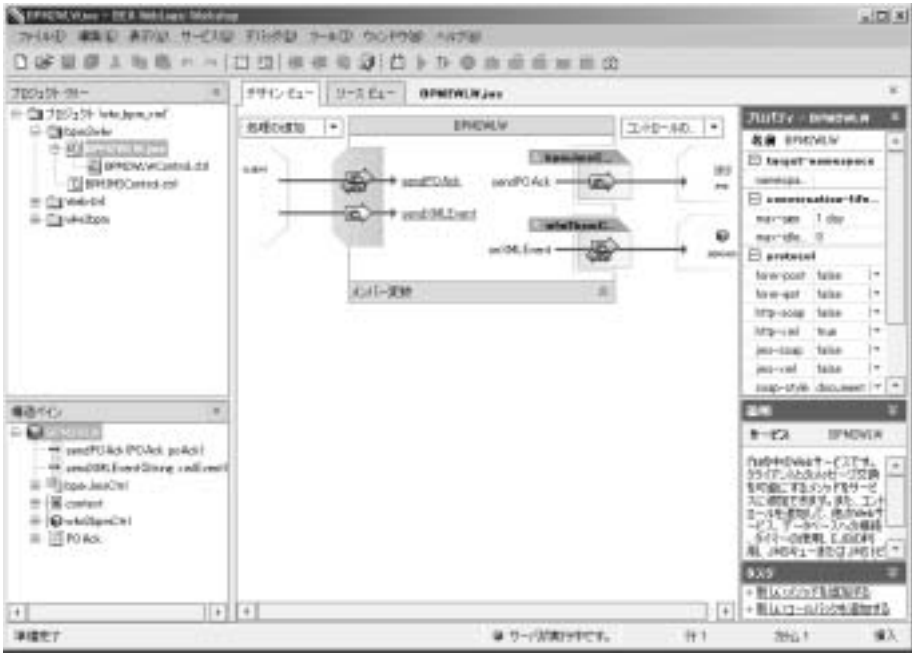


図 2-14 Send PO Web サービス ー XML over HTTP



c. [デバッグ] メニューから [開始] を選択します。Workshop の画面左下に「ビルドを開始しました」というメッセージが表示されます。ビルド完了後、メッセージは「サービスを実行中」に変わります。次に、ブラウザウィンドウが開き、Send PO Web サービス (`BPM2WLW.jws`) のテストフォームが表示されます。図 2-15 (SOAP over HTTP) または図 2-16 (XML over HTTP) を参照してください。

図 2-15　Send PO Web サービス － SOAP over HTTP

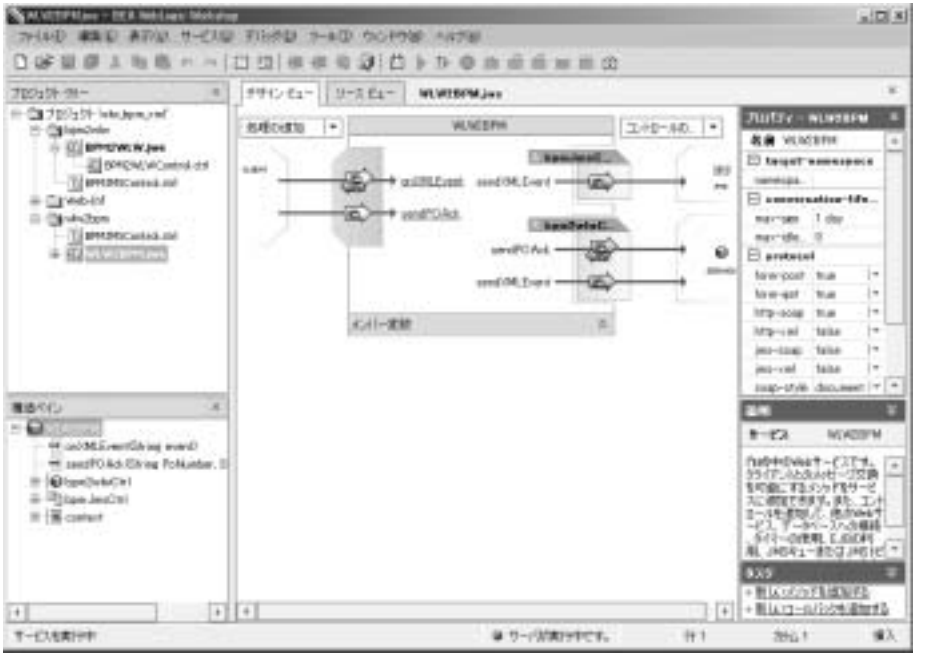
図 2-16　Send PO Web サービス － XML over HTTP

d. Workshop のプロジェクト ツリーで、`wlw2bpm` フォルダを開き、`WLW2BPM.jws` ノードをダブルクリックします。Process PO Web サービスが右ペインに表示されます。図 2-17 (SOAP over HTTP) または図 2-18 (XML over HTTP) を参照してください。

注意： Process PO Web サービス (`WLW2BPM.jws`) がまだ実行中でない場合は、[デバッグ] メニューで [開始] を選択します。Workshop の画面左下に「ビルドを開始しました」というメッセージが表示されます。ビルド完了後、メッセージは「サービスを実行中」に変わります。次に、ブラウザ ウィンドウが開き、Process PO Web サービスのテスト フォームが表示されます。図 2-19 (SOAP over HTTP) または図 2-20 (XML over HTTP) を参照してください。

図 2-17　Process PO Web サービス ー SOAP over HTTP

図 2-18　Process PO Web サービス － XML over HTTP



e. Process PO Web サービス (`WLW2BPM.jws`) のテスト フォームを表示するには、ブラウザに次の URL を入力します。

- SOAP over HTTP:
  `http://localhost:7501/wlw_bpm_soap/wlw2bpm/WLW2BPM.jws?.EXPLORE=.TEST`
- XML over HTTP:
  `http://localhost:7501/wlw_bpm_xml/wlw2bpm/WLW2BPM.jws?.EXPLORE=.TEST`

図 2-19 は、SOAP over HTTP のテスト フォーム、図 2-20 は XML over HTTP のテスト フォームを示しています。

図 2-19　Process PO Web サービス ― SOAP over HTTP

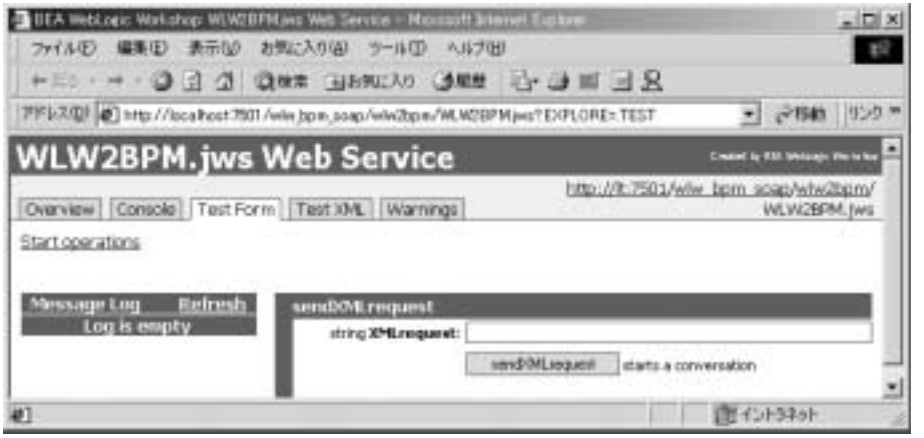


図 2-20　Process PO Web サービス ― XML over HTTP

注意： WebLogic Workshop の詳細については、WebLogic Workshop オンラインヘルプで以下のトピックを参照してください。

- 「*Web サービス構築ガイド*」の「コントロール : Web サービスからリソースを使用する」

- 「*Web サービス構築ガイド*」の「コントロール : Web サービスからリソースを使用する」にある「サービス コントロール : 他の Web サービスを使用する」
- 「*Web サービス構築ガイド*」の「会話を使用してステートを保持する」
- 「*Web サービス構築ガイド*」の「非同期性を利用して長時間の処理を実現する」にある「コールバックを使用してイベントのクライアントに通知する」
- 「*WebLogic Workshop リファレンス*」の「クラス リファレンス」にある「JwsContext Interface」

手順 6 : WebLogic Integration Swing Worklist を設定する

1. 以下を実行して Swing Worklist クライアントを起動します。

   ```
   BEA_HOME\weblogic700\integration\bin\worklist_swing.cmd
   ```

2. 次のように指定して Worklist にログオンします

   - [ユーザ名]: wlisystem
   - [パスワード]: wlisystem
   - [サーバ URL]: t3://localhost:7501

   Worklist が開き、「ワークリストには保留中のタスクはありません。」というメッセージが表示されます。

3. [了解] をクリックしてメッセージを閉じます。

   以上でサンプルの設定は完了です。

4. 第 3 章「BPM-Workshop 相互運用性プロセス」に進み、サンプルの実行を完了します。

3 BPM-Workshop 相互運用性プロセス

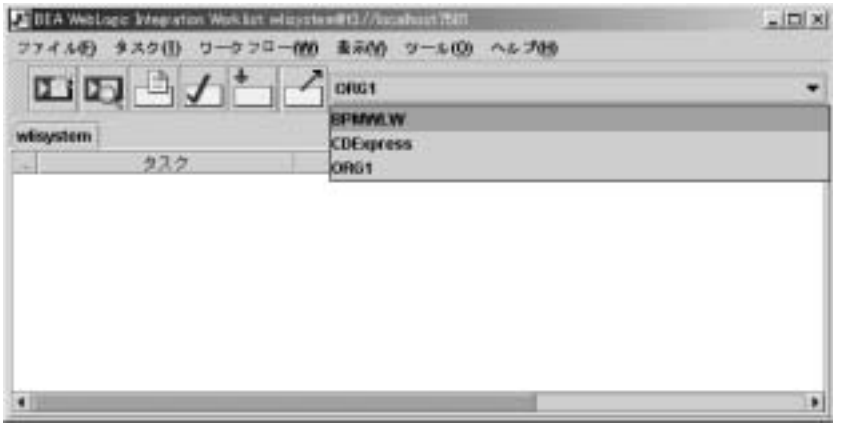
この章では、相互運用性プロセスの完了に必要な手順、および WebLogic Integration BPM - Workshop 相互運用性サンプルで交換されるメッセージについて説明します。

サンプルの実行

サンプルを実行するには、以下の手順を実行します。

1. 図 3-1 に示すように、Worklist の [オーガニゼーション] ドロップダウン リストから BPMWLW を選択します。

図 3-1 Worklist でのオーガニゼーションの選択



2. [ワークフロー] メニューから [ワークフローを開始] を選択します。図 3-2 に示す [ワークフローを開始] ウィンドウが表示されます。

図 3-2 ワークフローの開始

3. 選択したオーガニゼーションが BPMWLW であることを確認します。
4. SendPO ワークフローを選択し、[OK] をクリックします。ワークフローが開始したことを示すメッセージ ボックスが表示されます。
5. メッセージ ボックスで [了解] を選択します。

   Worklist で SendPO ワークフローを開始すると、ワークフローは次のいずれかの URI を使用して PO XML メッセージを `jws.queue` JMS キューに送信します。

   - SOAP over HTTP: `wlw_bpm_soap/bpm2wlw/BPM2WLW.jws`.
   - XML over HTTP: `wlw_bpm_xml/bpm2wlw/BPM2WLW.jws`

   SOAP over HTTP サンプルでは、SendPO Web サービスが PO を受信し、次に purchase メソッドを起動します。purchase メソッドは `WLW2BPMControl.sendXMLrequest` メソッドを呼び出し、次に、そのメソッドが `BPMControl.sendXMLrequest` メソッドを呼び出します。`BPMControl.sendXMLEvent` メソッド呼び出しにより、PO が `com.bea.wli.bpm.EventQueue` JMS キューに入れられます。

   XML over HTTP サンプルでは、Send PO Web サービスが PO を受信し、次に `sendXMLEvent` メソッドを起動します。`BPM2WLW.sendXMLEvent` メソッドは `WLW2BPMControl.onXMLEvent` メソッドを呼び出し、次に、そのメソッドが `BPMControl.sendXMLEvent` メソッドを呼び出します。`BPMControl.sendXMLEvent` メソッド呼び出しにより、PO が `com.bea.wli.bpm.EventQueue` JMS キューに入れられます。

次に、ProcessPO ワークフローの開始ノードは、キューから PO を取り出し、そのコンテンツを変数に抽出します。ProcessPO ワークフローの Approve PO ノードは、タスクを wlisystem ユーザに割り当て、ユーザが Worklist でタスクを実行するまで待機します。図 3-3 を参照してください。

図 3-3 Approve PO ワークフロー ノードでのユーザへのタスクの割り当て

6. タスクを実行するには、Approve PO タスクをダブルクリックします。Approve PO ノードが開き、図 3-4 に示すように、PO の承認を求めるウィンドウが表示されます。

図 3-4 [Approve PO] ウィンドウ

SOAP over HTTP　　XML over HTTP

7. [はい] をクリックするとタスクが承認されます。この質問に対するコールバック アクションにより、Approve PO ノードに完了マークが付きます。

   SOAP over HTTP サンプルでは、以下の処理が実行されます。

   ワークフローは Ack PO ノードに進みます。このノードでは、POAck XML メッセージを `com.bea.wli.bpm.WLWResponseQueue` に送信しま

す。WLW2BPM Web サービス内の BPMControl コントロールがキューから POAck メッセージを取り出し、WLW2BPM Web サービス内の BPM_onXMLresponse コールバック ハンドラを起動します。このコールバック ハンドラは onXMLresponse ハンドラを呼び出します。呼び出された onXMLresponse ハンドラは、POAck メッセージを com.bea.wli.bpm.EventQueue JMS キューに送信します。

SendPO ワークフローの Get PO Ack ノードでは、POAck メッセージを取得し、実行を Notify PO Ack ノードに進めます。Notify PO Ack ノードでは、Notify PO Ack タスクを Worklist に割り当てます。図 3-5 を参照してください。

XML over HTTP サンプルでは、以下の処理が実行されます。

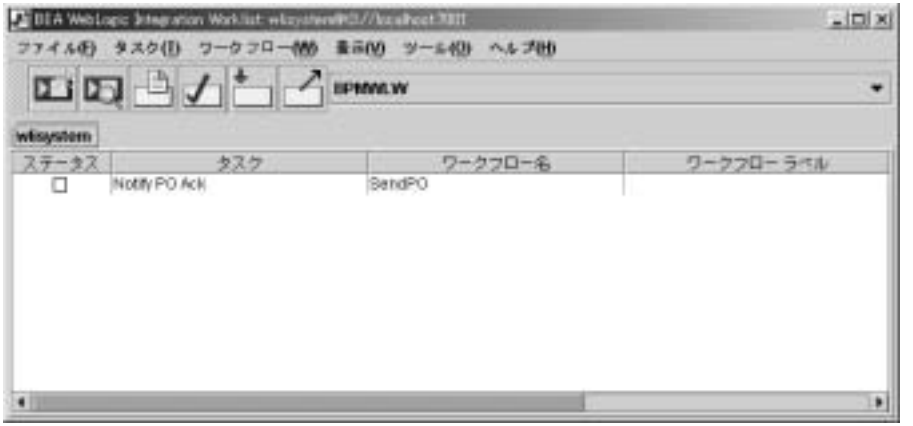
ワークフローは Ack PO ノードに進みます。このノードでは、POAck XML メッセージを jws.queue に送信します。POAck XML メッセージは、SendPO ワークフローのインスタンス ID をメッセージのペイロードに追加します。メッセージには「URI」ヘッダもあり、このヘッダには wlw_bpm_xml/wlw2bpm/WLW2BPM.jws が指定されています。そのため、POAck XML メッセージによって WLW2BPM Web サービスが呼び出されます。

WLW2BPM Web サービスは POAck メッセージをキューから取り出し、WLW2BPM.sendPOAck メソッドをトリガします。WLW2BPM.sendPOAck メソッドは、BPM2WLW コントロールの sendPOAck メソッドを呼び出し、PO 番号を渡します。

sendPOAck メソッドは、未加工の XML メッセージを HTTP 経由で BPM2WLW Web サービスに送信します。BPM2WLW Web サービスは、承認を受信すると、POAck メッセージを com.bea.wli.bpm.EventQueue JMS キューに送信します。

SendPO ワークフローには、POAck メッセージ ペイロード内のインスタンス ID を検索するイベント キーがあります。該当するインスタンス ID がキー値式に含まれるイベントがイベント キューに到着すると、そのインスタンスが Get PO Ack ノードをトリガします。その結果、POAck メッセージが取得され、実行は Notify PO Ack ノードに進みます。このノードでは、Notify PO Ack タスクを Worklist に割り当てます。図 3-5 を参照してください。

図 3-5 Notify PO Ack ワークフロー ノードでのユーザへのタスクの割り当て



8. Notify PO Ack タスクをダブルクリックしてタスクを実行します。タスクが Worklist から消え、実行は停止ノードに進みます。これでサンプルは終了です。

9. Web サービス間で渡されるメッセージを表示するには、次の URL を使用します。

   - SOAP over HTTP:

     ```
     http://localhost:7501/wlw_bpm_soap/wlw2bpm/WLW2BPM.jws?.EXPLORE=.TEST
     ```

     ```
     http://localhost:7501/wlw_bpm_soap/bpm2wlw/BPM2WLW.jws?.EXPLORE=.TEST
     ```

   - XML over HTTP:

     ```
     http://localhost:7501/wlw_bpm_xml/wlw2bpm/WLW2BPM.jws?.EXPLORE=.TEST
     ```

     ```
     http://localhost:7501/wlw_bpm_xml/bpm2wlw/BPM2WLW.jws?.EXPLORE=.TEST
     ```

   注意： メッセージ ログが空の場合は、[Web Service] ページの [Message Log] セクションで [Refresh] をクリックします。

10. メッセージを表示するには、[Message Log] のメッセージをクリックします。

    次の図にメッセージを示します。

図 3-6 Send PO Web サービスのメッセージ ログ ― SOAP over HTTP

図 3-7　Process PO Web サービスのメッセージ ログ ― SOAP over HTTP

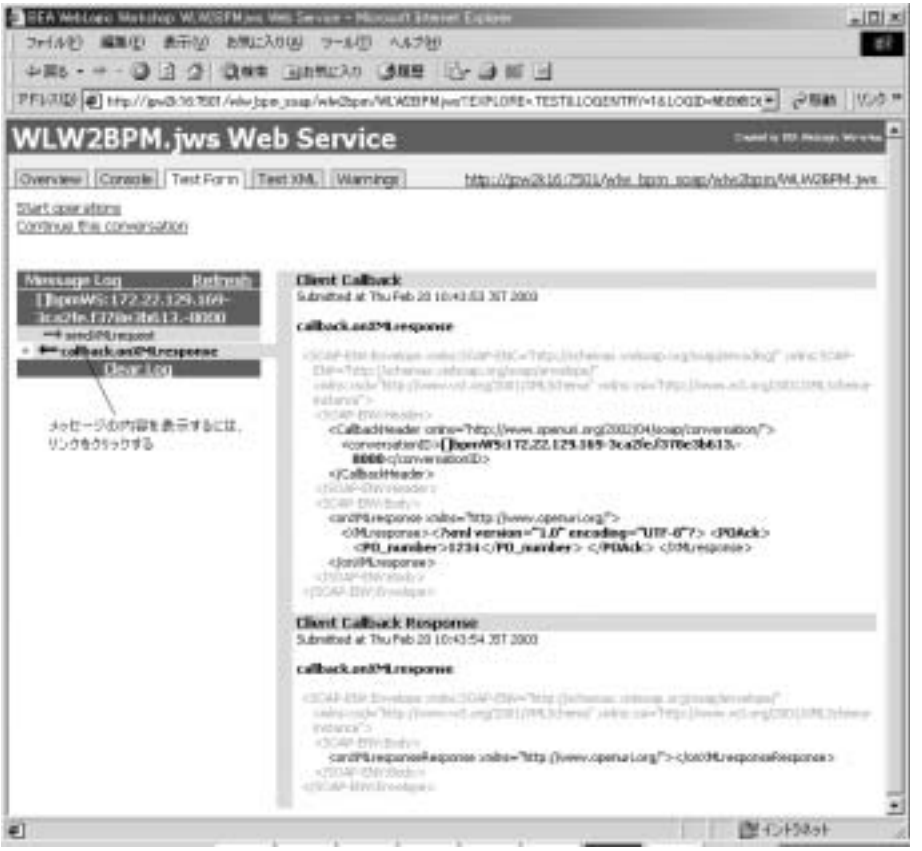


図 3-8 Send PO Web サービスのメッセージ ログ ― XML over HTTP

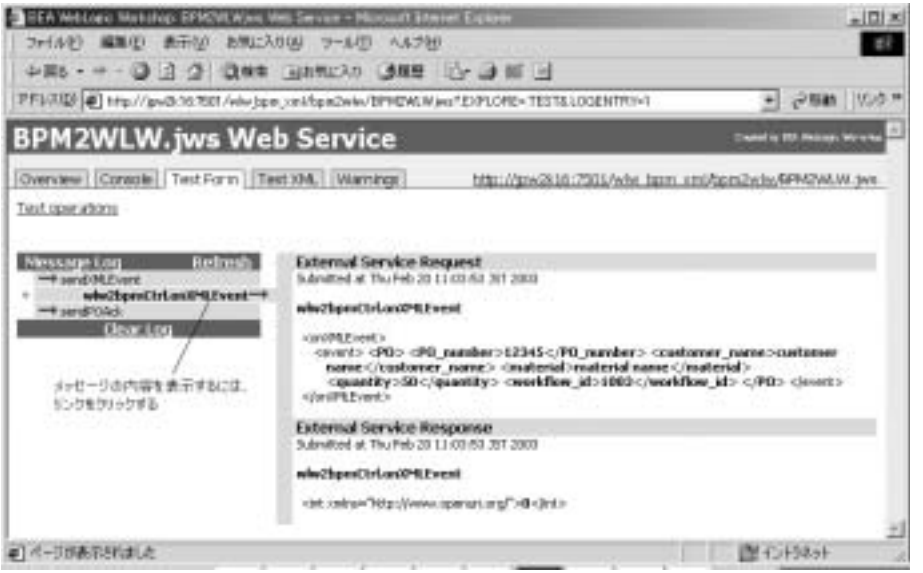
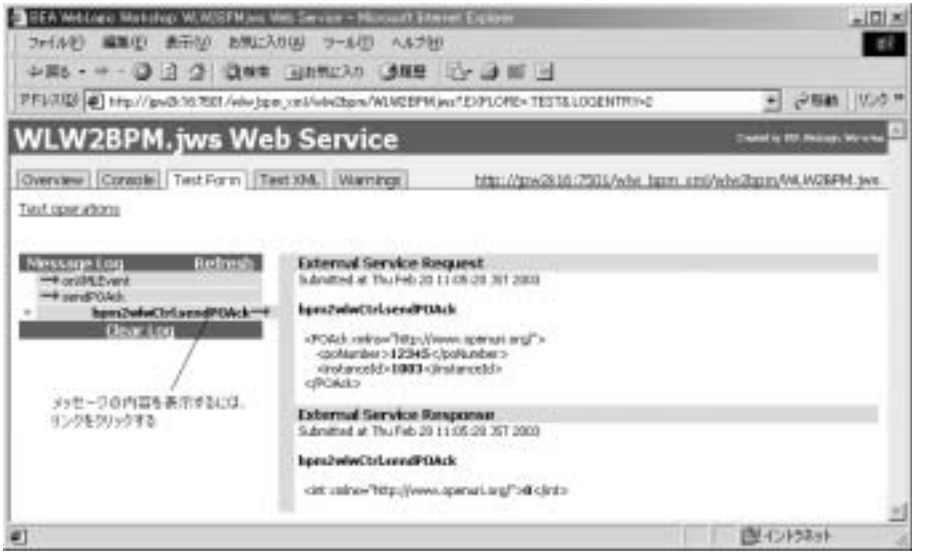


図 3-9 Send PO Web サービスのメッセージ ログ ― XML over HTTP



索引

わ