



**ORACLE®**

# Oracle BPM 11g JMS Integration アプリケーション作成ハンズオン

日本オラクル株式会社

# Agenda

- ハンズオンの目的
- 環境準備
- JMS Integrationアプリケーションの作成  
(以下のプロジェクトを作成する)

BpmJmsSend	JMSアダプタで、ターゲット・プロセスの開始イベント・メッセージと、ターゲット・サービスのビジネス・イベント・メッセージの2種類のメッセージを、JMSキューへ送信
BpmJmsReceive	JMSアダプタでJMSキューから2種類のメッセージを受信し、受信のメッセージのタイプによって、別々の方法でターゲット・サービスを呼び出す ・開始イベント・メッセージの場合、ターゲット・サービスは直接呼び出す ・ビジネス・イベント・メッセージの場合、メディエータを経由して呼び出す

- 補足情報
- リファレンス

# ハンズオンの目的

- Oracle BPMアプリケーション開発ライフサイクルの理解
- JMSサーバー、JMSモジュール、キューなどの概念の理解
- JMSメッセージを介してリモート・プロセスを呼び出す方法の理解
- JMSメッセージとメディアータを介してリモート・プロセスを呼び出す方法の理解

# 環境準備 (1/6)

- WebLogic Server 10.3.5 のインストール
- SOA Suite 11gPS4 Feature Pack のインストール
- BPM 11gPS4 Feature Pack のインストール
- Oracle JDeveloper 11gR1 のインストール

# 環境準備 (2/6)

## JMSキュー「jms/sampleQueue」の作成

- WebLogic Server管理コンソールへログイン
- 左側の「ドメイン構造」から「サービス > メッセージング > JMSモジュール」を選択し、リストから「BPMJMSModule」を選択

The screenshot displays the Oracle WebLogic Server Administration Console. On the left, the 'ドメイン構造' (Domain Structure) tree is visible, with 'サービス' (Services) > 'メッセージング' (Messaging) > 'JMSモジュール' (JMS Modules) selected. A red box and a green circle with the number '1' highlight this selection. The main content area shows the 'JMSモジュール' (JMS Modules) configuration page. Below the 'この表のカスタマイズ' (Customize this table) section, there is a table of JMS modules. A red box and a green circle with the number '2' highlight the 'BPMJMSModule' entry in the table.

名前	タイプ
BPMJMSModule	システム
...	...

# 環境準備 (3/6)

## JMSキュー「jms/sampleQueue」の作成

- 「新規」をクリック

BPMJMSModuleの設定

構成 サブデプロイメント ターゲット セキュリティ ノート

このページは、JMSシステム・モジュールおよびそのリソースに関する全般的な情報が表示されます。また、新規リソースの構成や既存リソースへのアクセスを行うこともできます。

名前: BPMJMSModule このJMSシステム・モジュールの名前。 [詳細...](#)

ディスクリプタ・ファイル名: jms/bpmjmsmodule-jms.xml JMSモジュール・ディスクリプタ・ファイルの名前。 [詳細...](#)

このページでは、このJMSシステム・モジュール用に作成されたキューおよびトピック宛先、接続ファクトリ、JMSテンプレート、宛先ソート・キー、宛先割当、分散宛先、外部サーバー、およびストア・アンド・フォワード・パラメータなどのJMSリソースの概要を示します。

この表のカスタマイズ

リソースのサマリー

1

表示項目 1 - 8/8 前へ | 次へ

<input type="checkbox"/> 名前	タイプ	JNDI名	サブデプロイメント	ターゲット
<input type="checkbox"/> BAMCommandXAConnectionFactory	接続ファクトリ	jms/bpm/BAMCommandXAConnectionFactory	デフォルトのターゲット指定	AdminServer
<input type="checkbox"/> CubeCommandXAConnectionFactory	接続ファクトリ	jms/bpm/CubeCommandXAConnectionFactory	デフォルトのターゲット指定	AdminServer
<input type="checkbox"/> MeasurementQueue	割当	...	...	...

# 環境準備 (4/6)

## JMSキュー「jms/sampleQueue」の作成

- 「キュー」を選択して「次へ」
- 「名前」と「JNDI名」を入力して「次へ」

新しいJMSシステム・モジュール・リソースの作成

戻る 次へ 2 取消し

作成するリソースのタイプを選択してください

これらのページでは、キュー、トピック、テンプレート

選択したリソースのタイプに応じて、リソースの作り、分散キュー、分散トピック、外部サーバー、JMSサーバーをターゲットとして選択することもできます。キュー・リソースとそのメンバーをサーバー・リソース

接続ファクトリ

キュー 1

新しいJMSシステム・モジュール・リソースの作成

戻る 次へ 終了 取消し

JMS宛先のプロパティ

次のプロパティは、新しいキューを識別するために使用されます。現在のモジュールはBPMIntegrationJMSModuleです。

\*は必須フィールドです

\*名前: 3 JmsSampleQueue

JNDI名: 4 jms/sampleQueue

名前: JmsSampleQueue

JNDI名: jms/sampleQueue

テンプレート: None

戻る 次へ 5 取消し

# 環境準備 (5/6)

## JMSキュー「jms/sampleQueue」の作成

- 「サブデプロイメント」に「BPMSubDeployment」を選択し、「終了」をクリックして、JMSキュー「jms/sampleQueue」の作成が完了

新しいJMSシステム・モジュール・リソースの作成

戻る 次へ **終了** 2

次のプロパティは、新しいJMSシステム・モジュール・リソースをターゲット指定するために使用されます。

このページでは、このシステム・モジュール・リソースを割り当てるサブデプロイメントを選択します。サブデプロイメントスタ、またはSAFエージェントに割り当てるメカニズムです。必要に応じて、「新しいサブデプロイメントの作成」ボタンでサブデプロイメントのターゲットは、親モジュールのサブデプロイメント管理ページで後から再構成することもできます。

使用するサブデプロイメントを選択してください。(なし)を選択した場合、ターゲット指定は行われません。

サブデプロイメント: **BPMSubDeployment** 1 サブデプロイメントの作成

このサブデプロイメントに割り当てるターゲットを指定してください。

ターゲット:

JMSサーバー
<input type="radio"/> BAMJMSServer
<input checked="" type="radio"/> BPMJMSServer



# 環境準備 (6/6)

## JMSキュー「jms/sampleQueue」の作成

- ドメイン構造パネルから「環境 > サーバー > AdminServer > 構成 > 全般」の「JNDIツリーの表示」をクリックして、JMSサーバー「BPMJMSServer」のターゲット・サーバーのJNDIツリーに「jms/sampleQueue」が存在することを確認



The screenshot displays the Oracle WebLogic Server Administration Console. On the left, the JNDI tree structure is visible, with the path `jms/sampleQueue` highlighted. On the right, the configuration page for `jms.sampleQueue` is shown, with the `bindName` field set to `jms.sampleQueue` and circled in red. The `className` is `weblogic.jms.common.DestinationImpl` and the `hashCode` is `1917741501`.

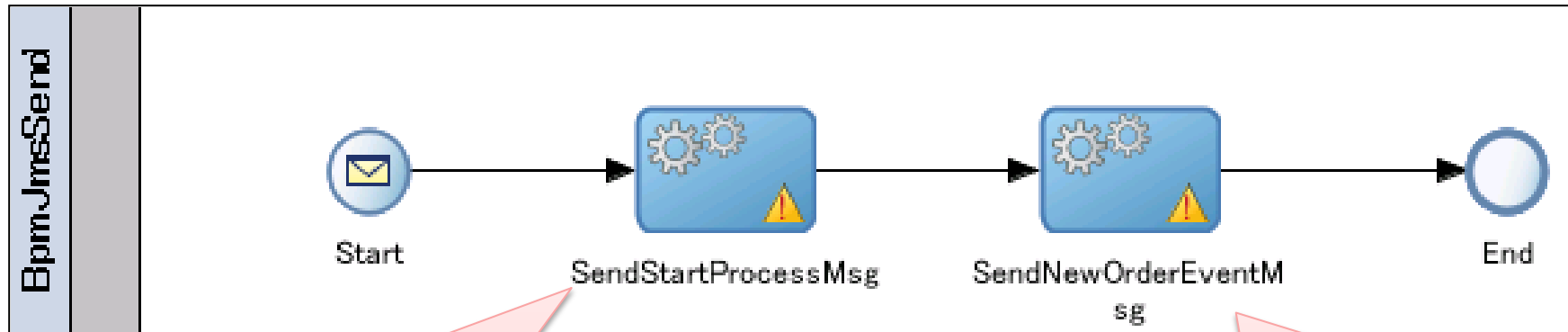
フィールド名	値
バインド名:	jms.sampleQueue
クラス:	weblogic.jms.common.DestinationImpl
ハッシュ・コード:	1917741501
toStringの結果:	BPMJMSModule!JmsSampleQueue

# JMSIntegrationアプリケーションの作成



# プロセスのモデリング

## 送信プロセスの全体像



### SendStartProcessMsg サービス・タスク

シナリオ1: JMSメッセージで直接プロセスを呼び出す

ターゲット・プロセスの開始イベントで定義された操作と同じXMLスキーマ型を持つJMSメッセージを送信し、ターゲット・プロセスを呼び出します。ターゲット・サービスのWSDLを持つ必要があります。

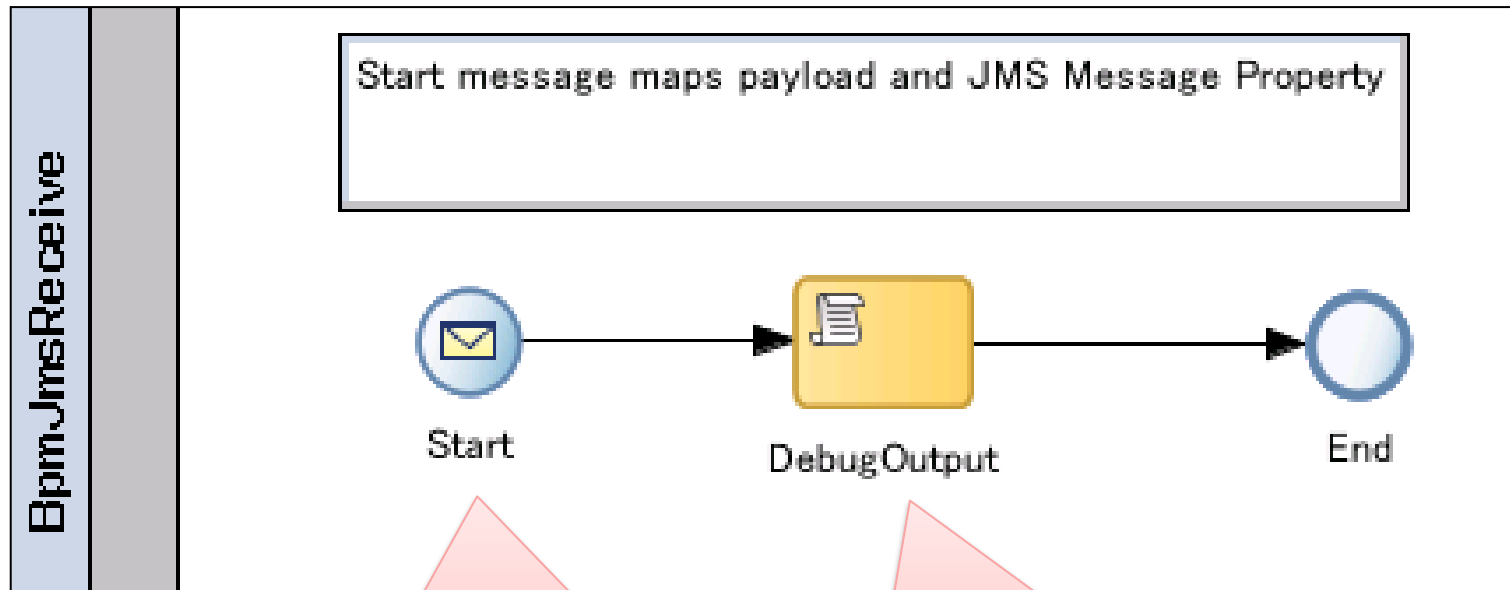
### SendNewOrderEventMsg サービス・タスク

シナリオ2: JMSメッセージからメディエータ経由でプロセスを呼び出す

ターゲット・サービスのWSDLを持たず、XMLスキーマ型を持つJMSメッセージを送信します。受信側のコンポジットでは、メディエータが受信したJMSメッセージを変換して、ターゲット・プロセスの開始メッセージを作成します

# プロセスのモデリング

## ターゲット・プロセスの全体像



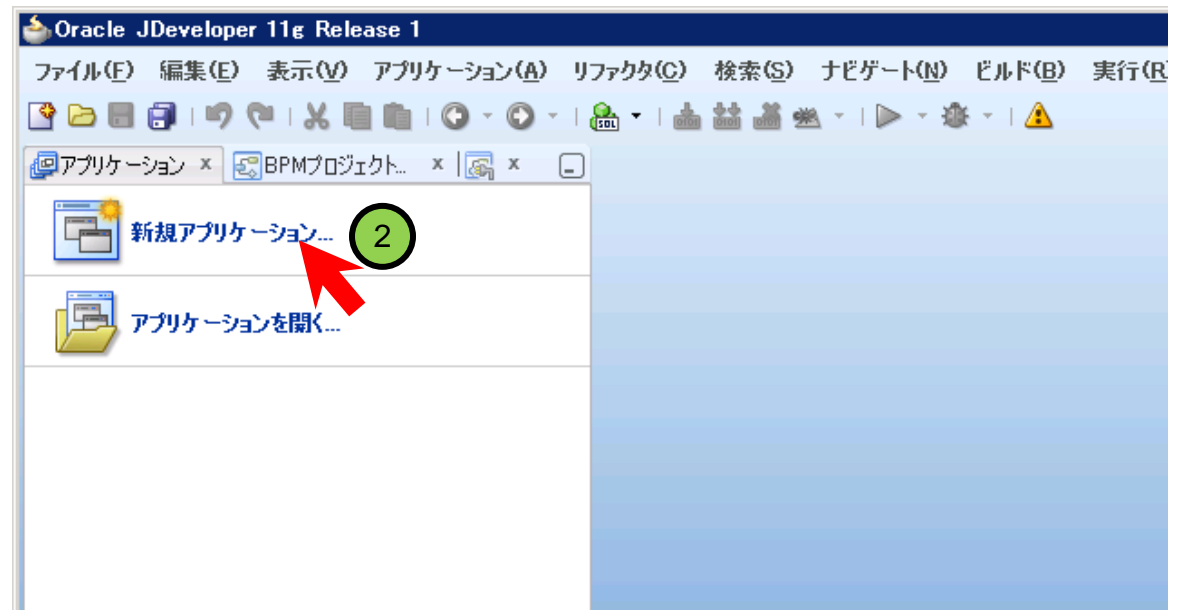
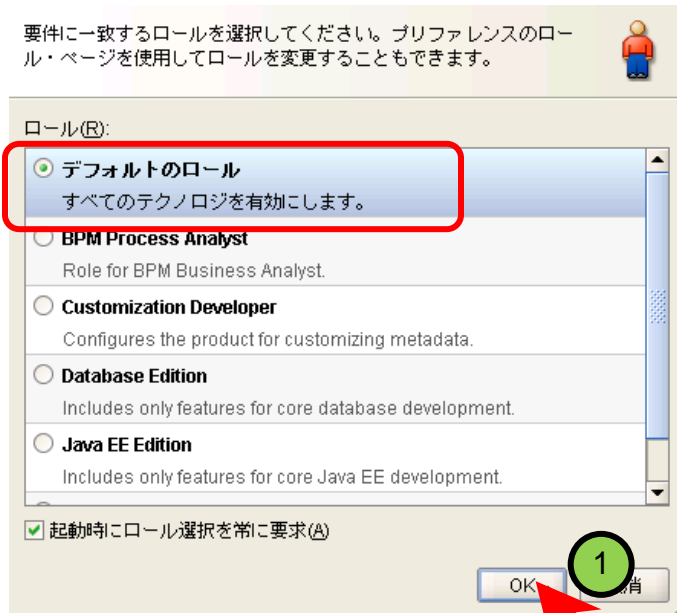
**Startイベント:**  
ペイロードとJMSメッセージ・プロパティ  
のデータ・マッピングをおこないます。

**DebugOutputスクリプト・タスク:**  
ペイロードとJMSメッセージ・プロパティ  
の確認用です。必須ではありません。

# プロセスのモデリング

## 送信プロセスBpmJmsSend (1/9)

- Oracle JDeveloper を開き、「新規アプリケーション」をクリック



# プロセスのモデリング

## 送信プロセスBpmJmsSend (2/9)

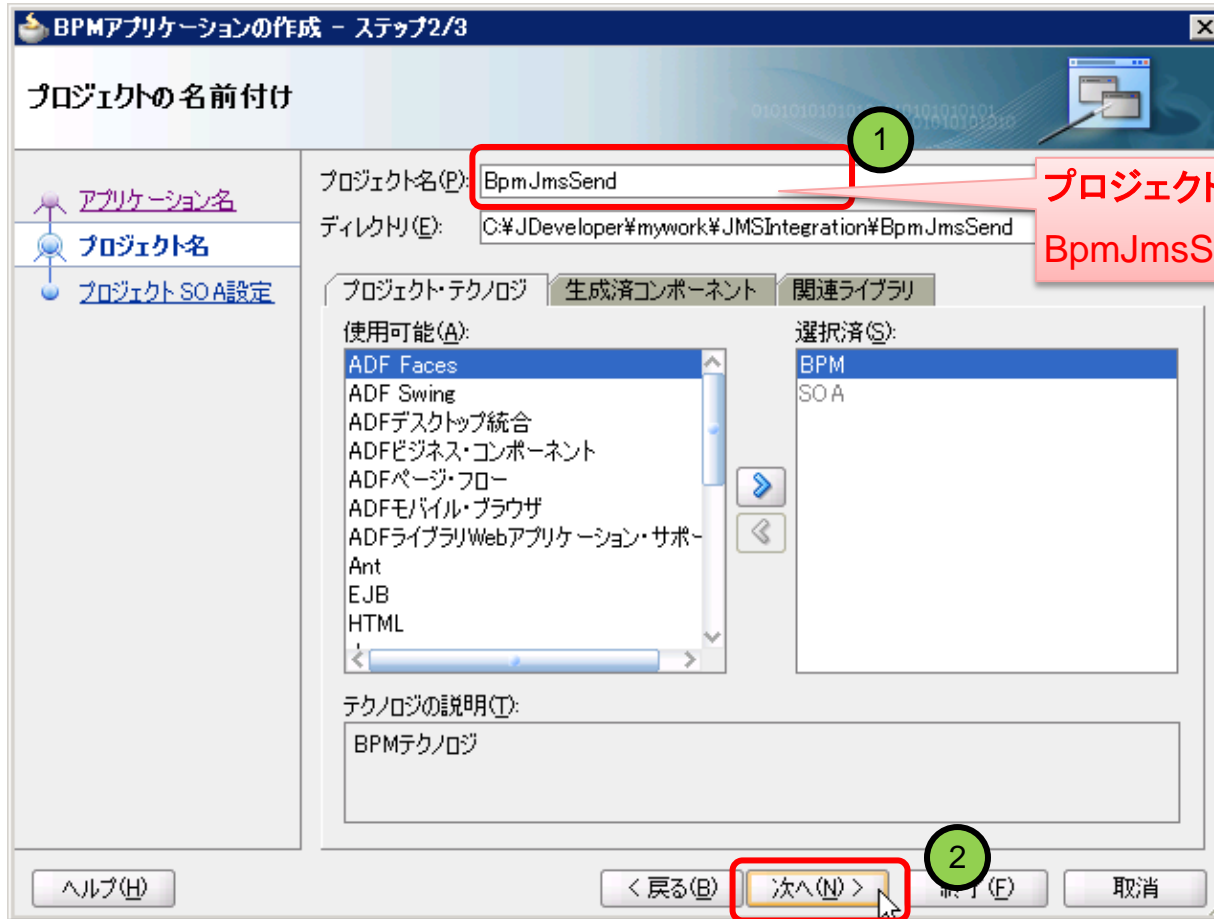
- アプリケーション名を入力し、「BPMアプリケーション」を選択して「次へ」をクリック



# プロセスのモデリング

## 送信プロセスBpmJmsSend (3/9)

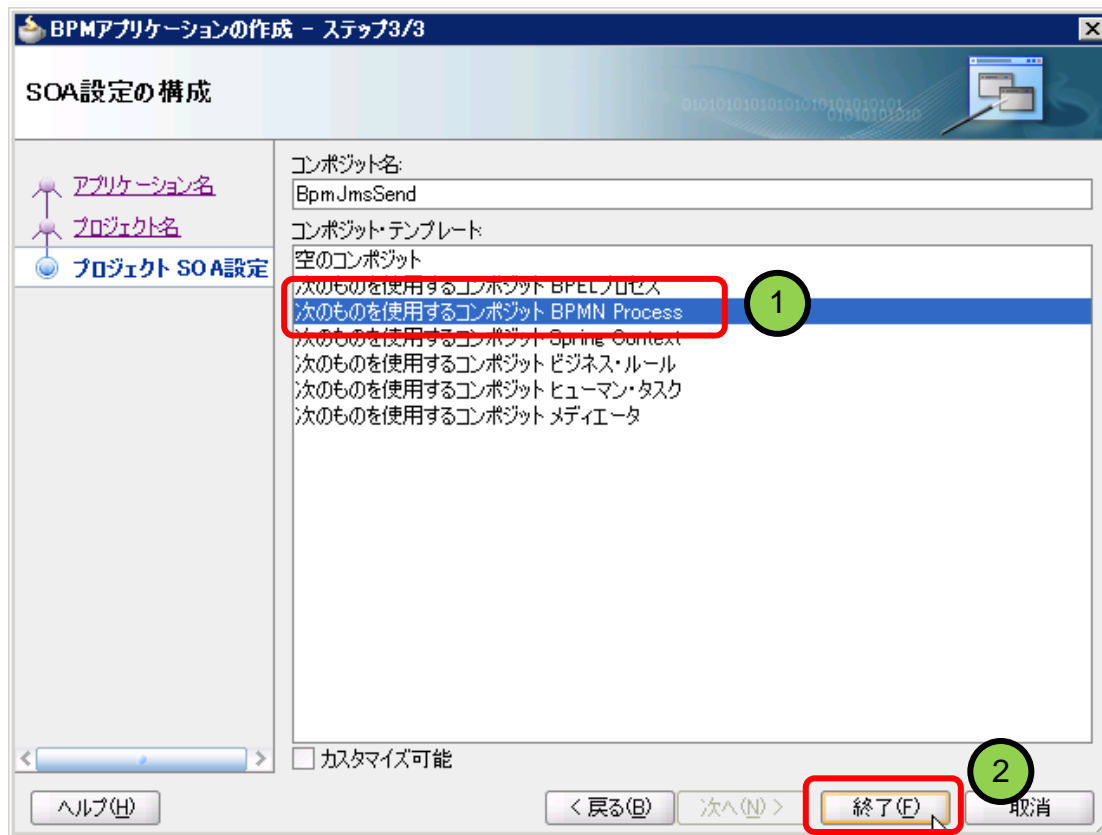
- プロジェクト名を入力して「次へ」をクリック



# プロセスのモデリング

## 送信プロセスBpmJmsSend (4/9)

- 「次のものを使用するコンポジット BPMN Process」を選択し、「終了」をクリック

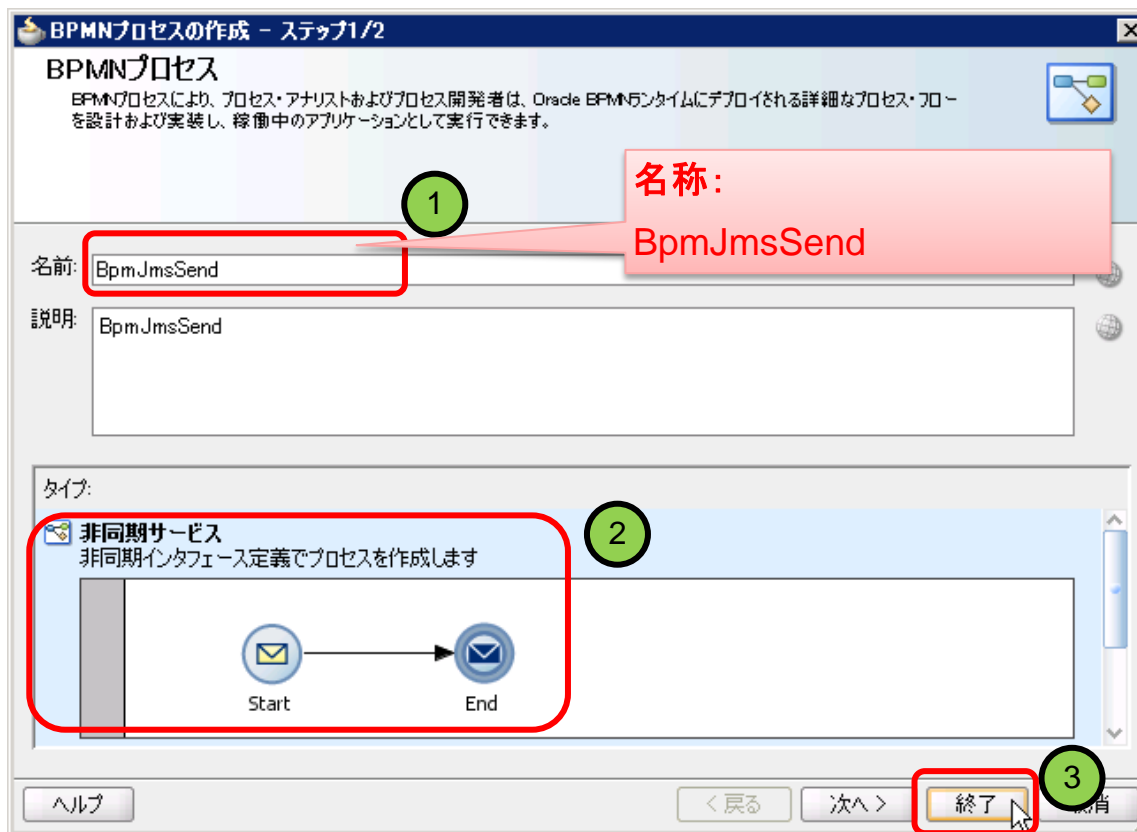




# プロセスのモデリング

## 送信プロセスBpmJmsSend (5/9)

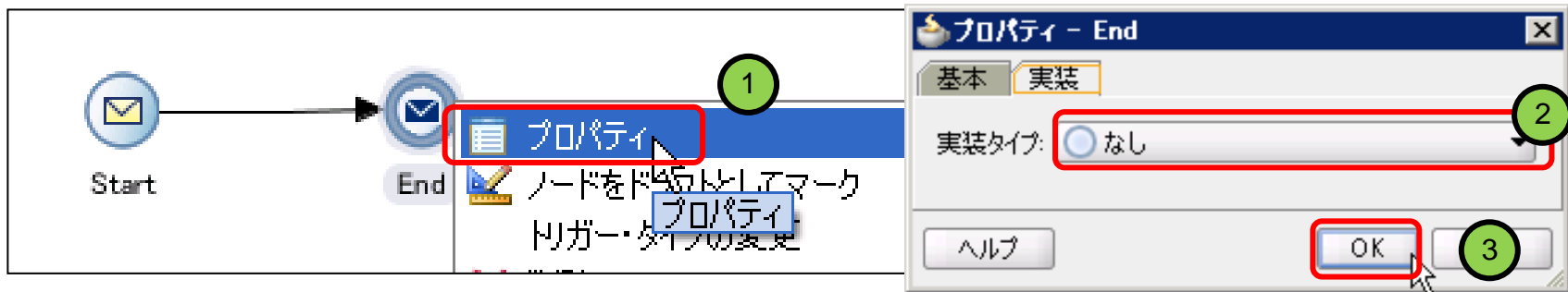
- BPMNプロセスの名前を入力し、タイプとして「非同期サービス」を選択して「終了」をクリック



# プロセスのモデリング

## 送信プロセスBpmJmsSend (6/9)

- 「End」イベントを右クリックし、プロパティを開く  
「実装タイプ」を「なし」に変更して「OK」をクリック



- 「End」イベントをクリックし、右側にドラッグしてデザイン・パネルの右側にドロップ



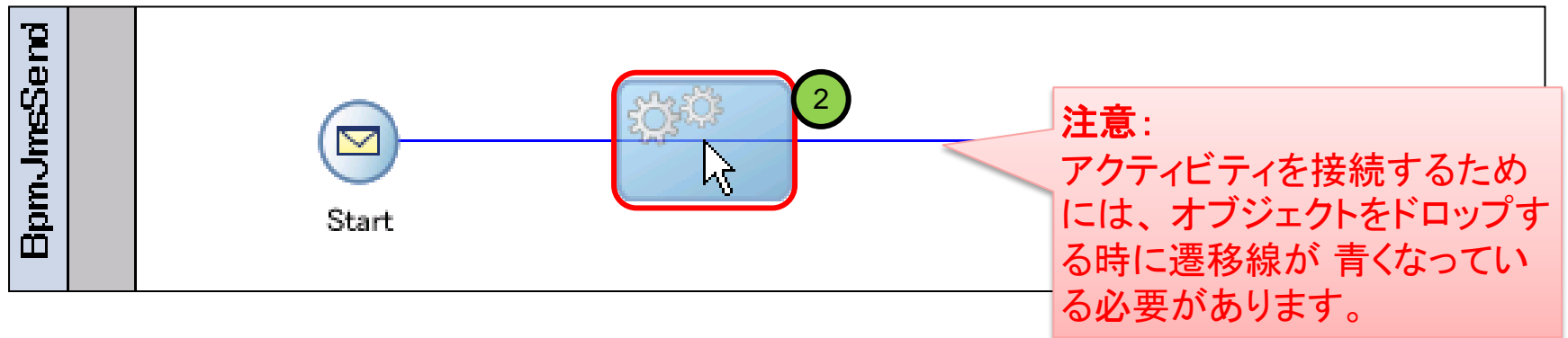
# プロセスのモデリング

## 送信プロセスBpmJmsSend (7/9)

- 「アクティビティ > サービス」を選択



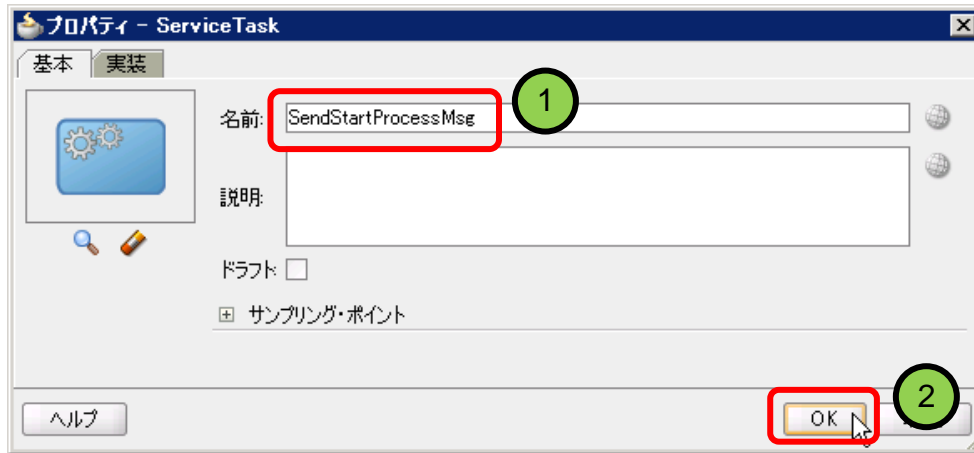
- 「Start」と「End」イベントの間のシーケンス・フロー上にドロップ



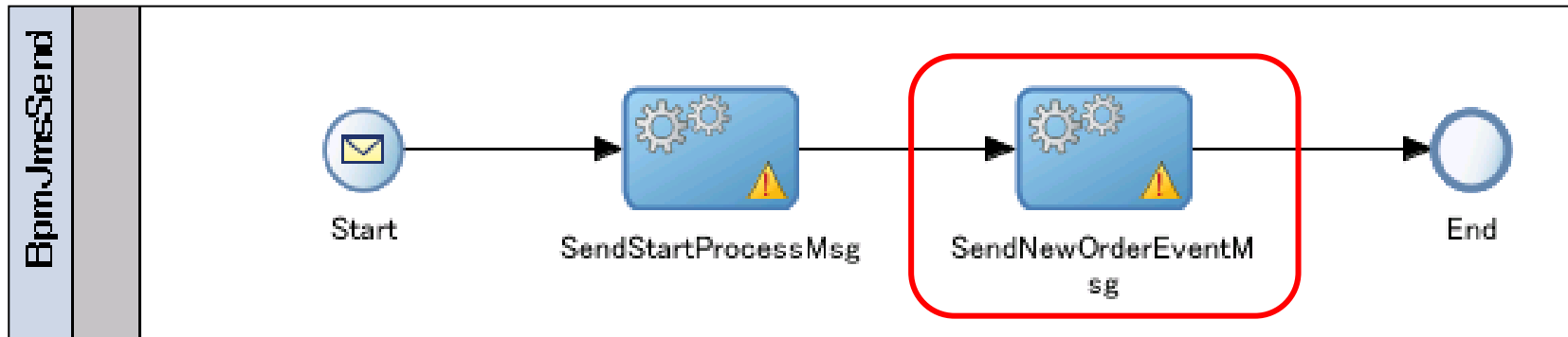
# プロセスのモデリング

## 送信プロセスBpmJmsSend (8/9)

- 名前に「**SendStartProcessMsg**」を設定して、「**OK**」をクリック



- 同じようにサービス・タスク「**SendNewOrderEventMsg**」を追加



# プロセスのモデリング

## 送信プロセスBpmJmsSend (9/9)

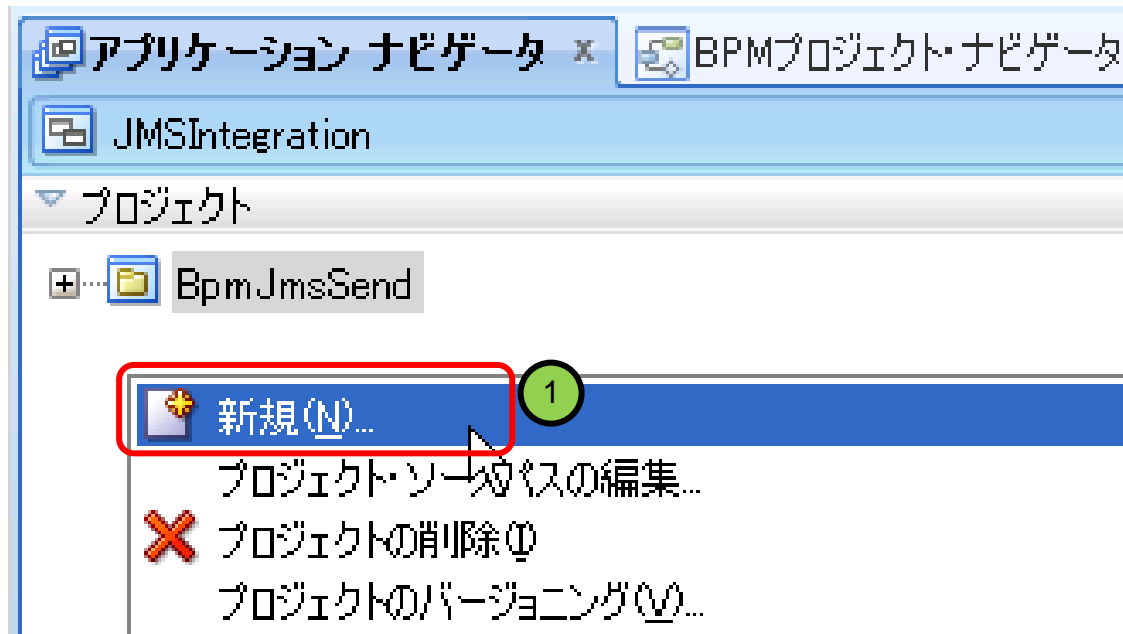
- 「すべて保存」をし、「BpmJmsSend」タブ以外のタブを閉じる

The screenshot displays the Oracle JDeveloper 11g interface for modeling the 'BpmJmsSend' process. The top menu bar includes options like 'ファイル(E)', '編集(E)', '表示(V)', 'アプリケーション(A)', 'リファクタ(C)', '検索(S)', 'ナビゲート(N)', 'ビルド(B)', '実行(R)', 'バージョンing(O)', 'ツール(T)', and 'ウィンドウ(W)'. The toolbar contains icons for saving (1), running (2), and closing (3). The project tree on the left shows the structure of the 'BpmJmsSend' project, including 'BPM Content', 'processes', 'アクティビティ・ガイド', '組織', 'SOAコンテンツ', 'classes', 'testsuites', 'xsd', 'xsl', 'ビジネス・ルール', and 'BpmJmsSend.componentType'. The process diagram on the right shows a flow from 'Start' to 'SendStartProcessMsg' to 'SendNewOrderEventM sg' to 'End'.

# プロセスのモデリング

## ターゲット・プロセスBpmJmsReceive (1/9)

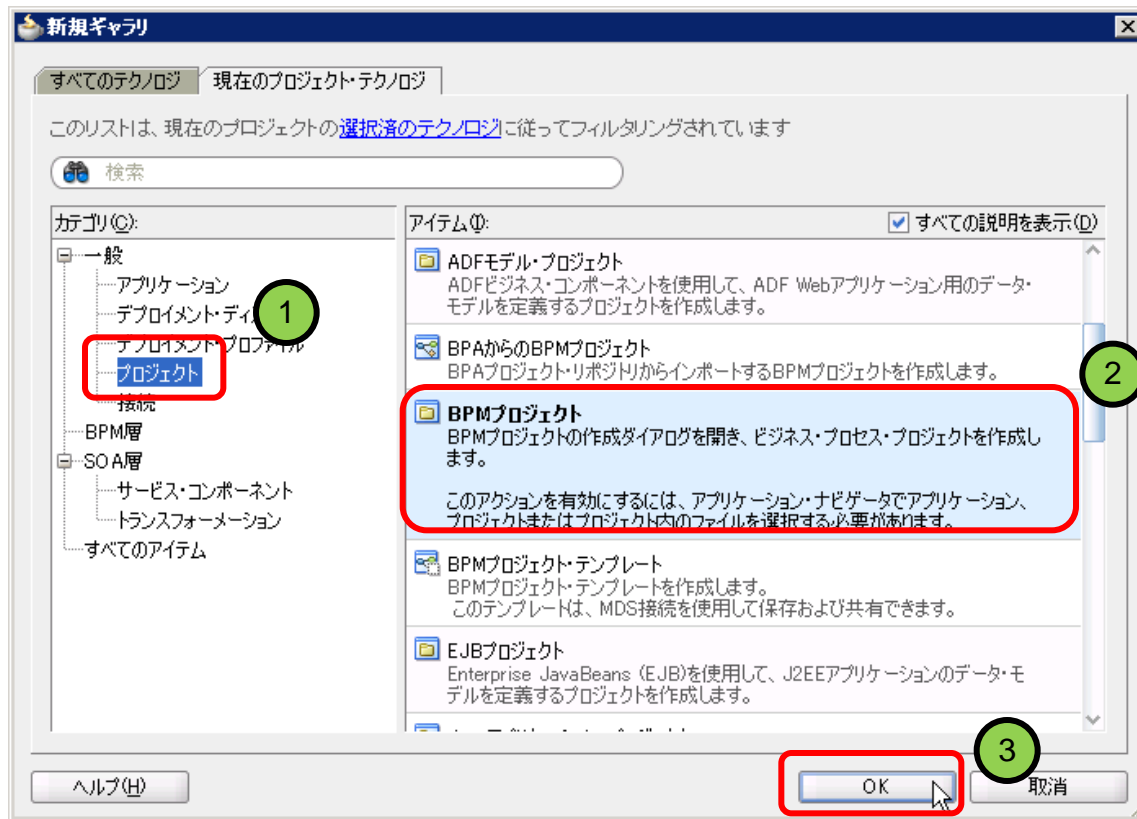
- 「アプリケーション・ナビゲータ」の空白部分で右クリックをし、「新規」をクリック



# プロセスのモデリング

## ターゲット・プロセスBpmJmsReceive (2/9)

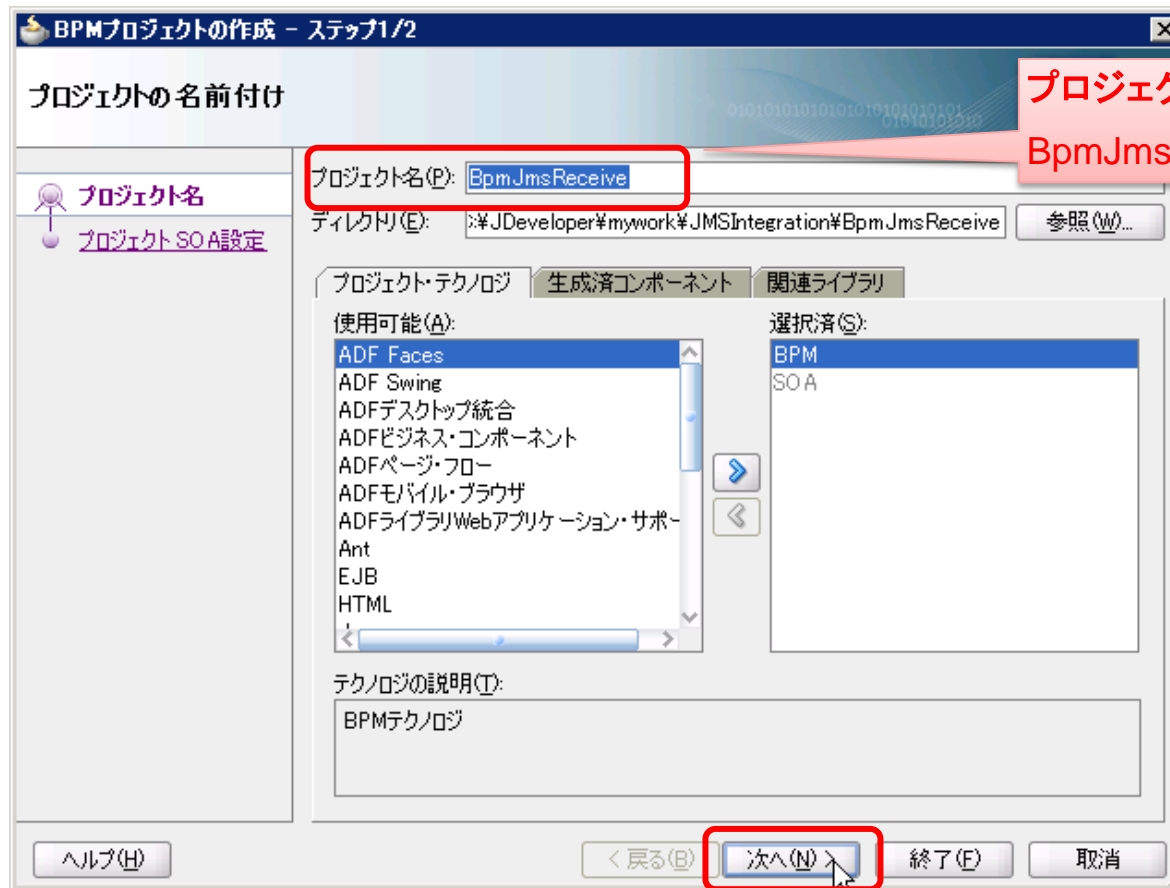
- カテゴリにて「一般 > プロジェクト」を選択し、「BPMプロジェクト」を選択して「次へ」をクリック



# プロセスのモデリング

## ターゲット・プロセスBpmJmsReceive (3/9)

- プロジェクト名を入力して「次へ」をクリック



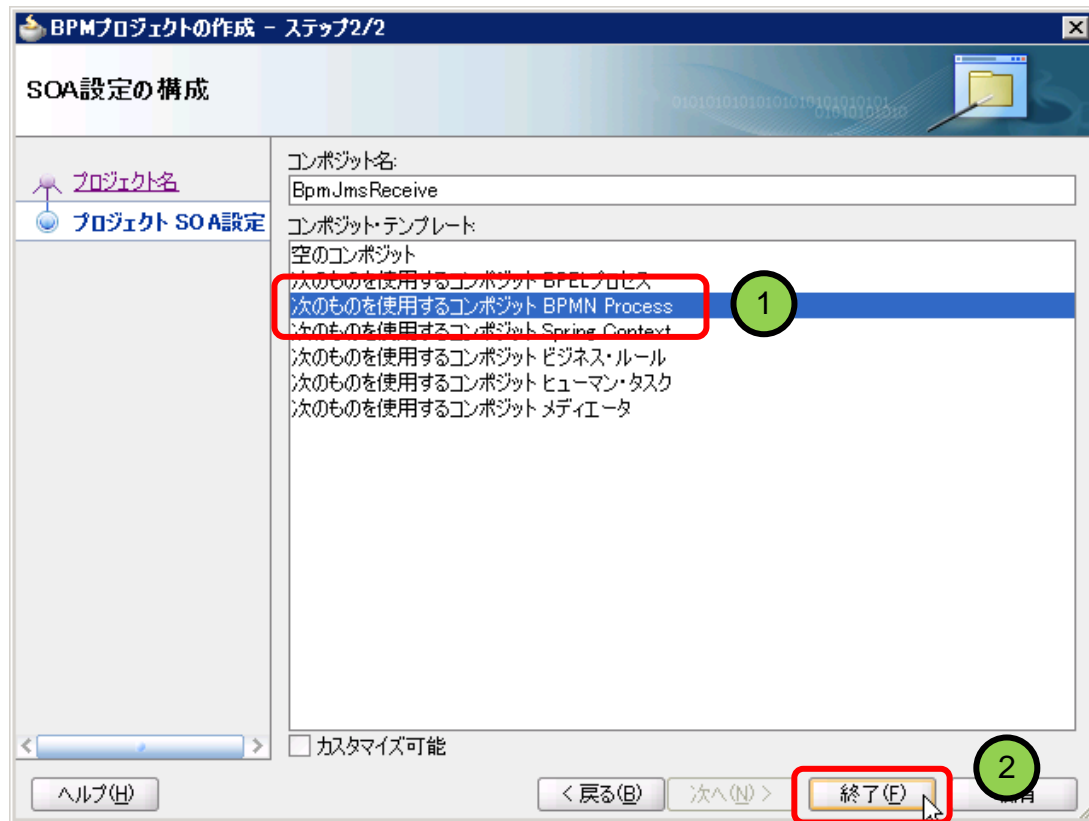
プロジェクト名:  
BpmJmsReceive



# プロセスのモデリング

## ターゲット・プロセスBpmJmsReceive (4/9)

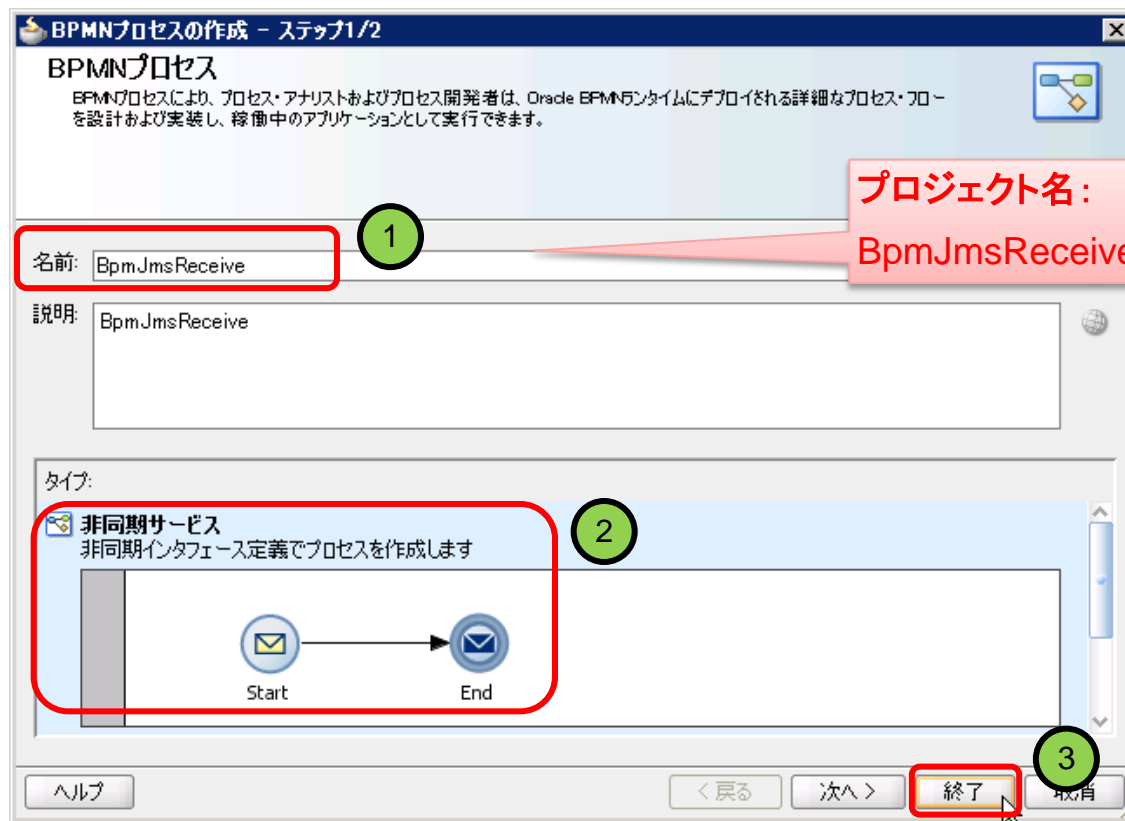
- 「次のものを使用するコンポジット BPMN Process」を選択し、「終了」をクリック



# プロセスのモデリング

## ターゲット・プロセスBpmJmsReceive (5/9)

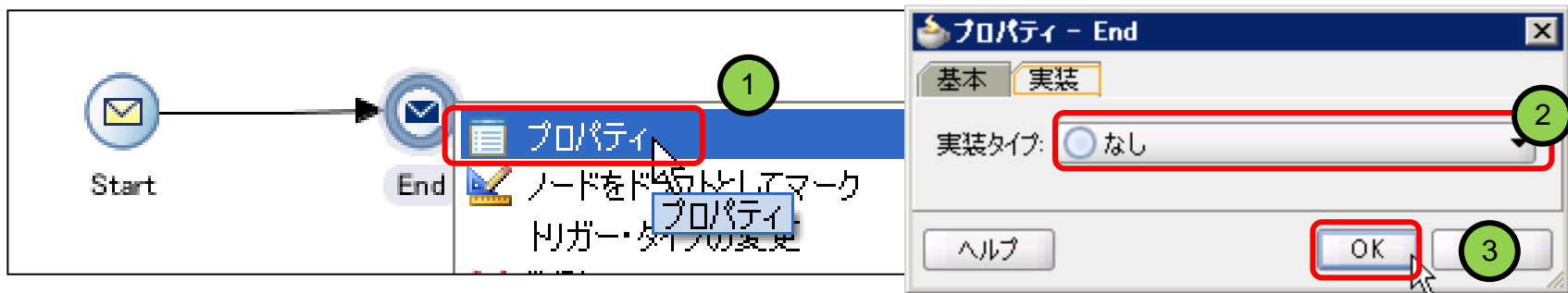
- BPMNプロセスの名前を入力し、タイプとして「非同期サービス」を選択して「終了」をクリック



# プロセスのモデリング

## ターゲット・プロセスBpmJmsReceive (6/9)

- 「End」 イベントを右クリックし、プロパティを開く  
「実装タイプ」を「なし」に変更して「OK」をクリック



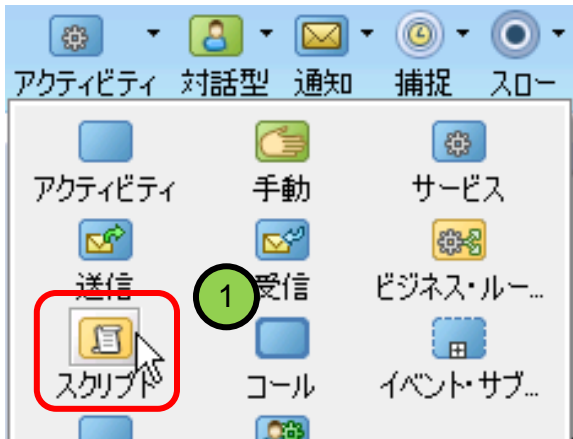
- 「End」イベントをクリックし、右側にドラッグしてデザイン・パネルの右側にドロップ



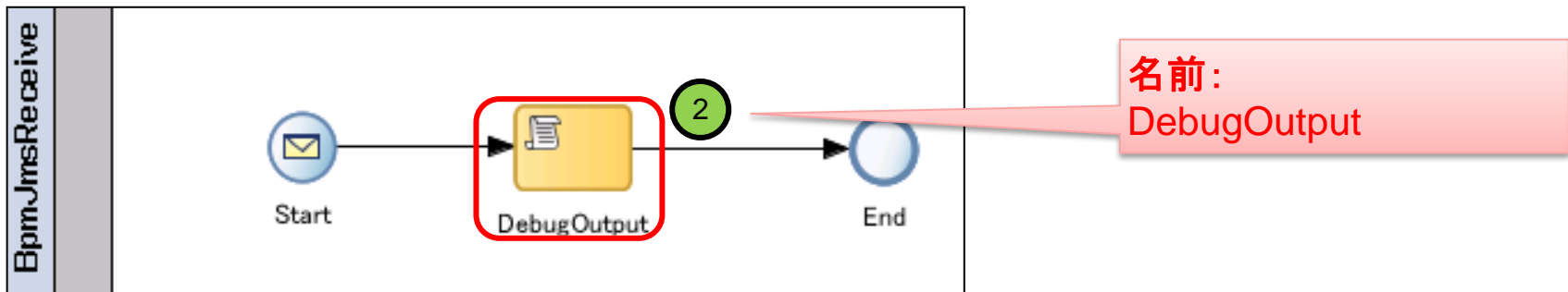
# プロセスのモデリング

## ターゲット・プロセスBpmJmsReceive (7/9)

- 「アクティビティ > スクリプト」を選択



- 「Start」と「End」イベントの間のシーケンス・フロー上にドロップ



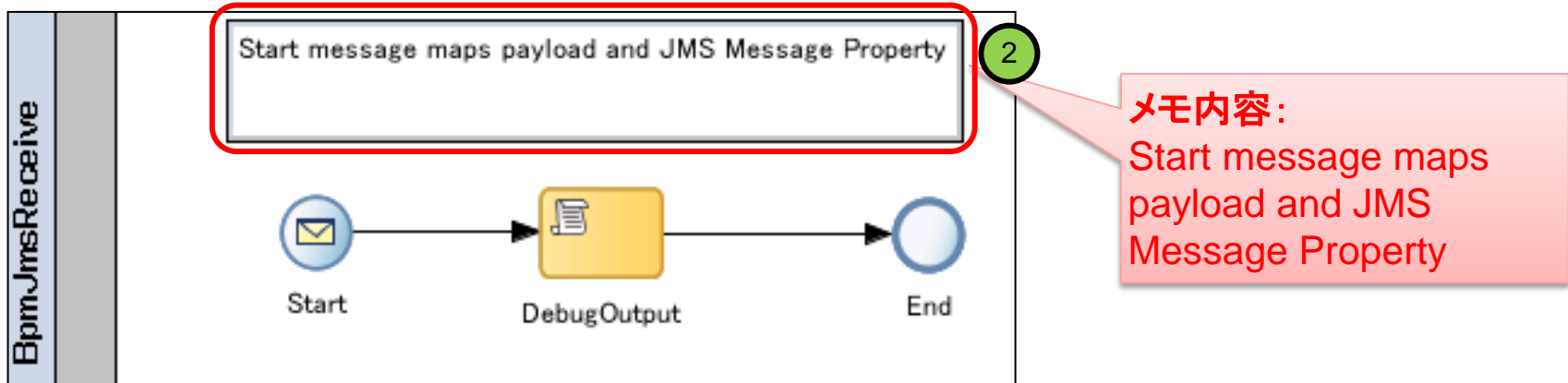
# プロセスのモデリング

## ターゲット・プロセスBpmJmsReceive (8/9)

- 「アーティファクト > テキスト注釈」を選択



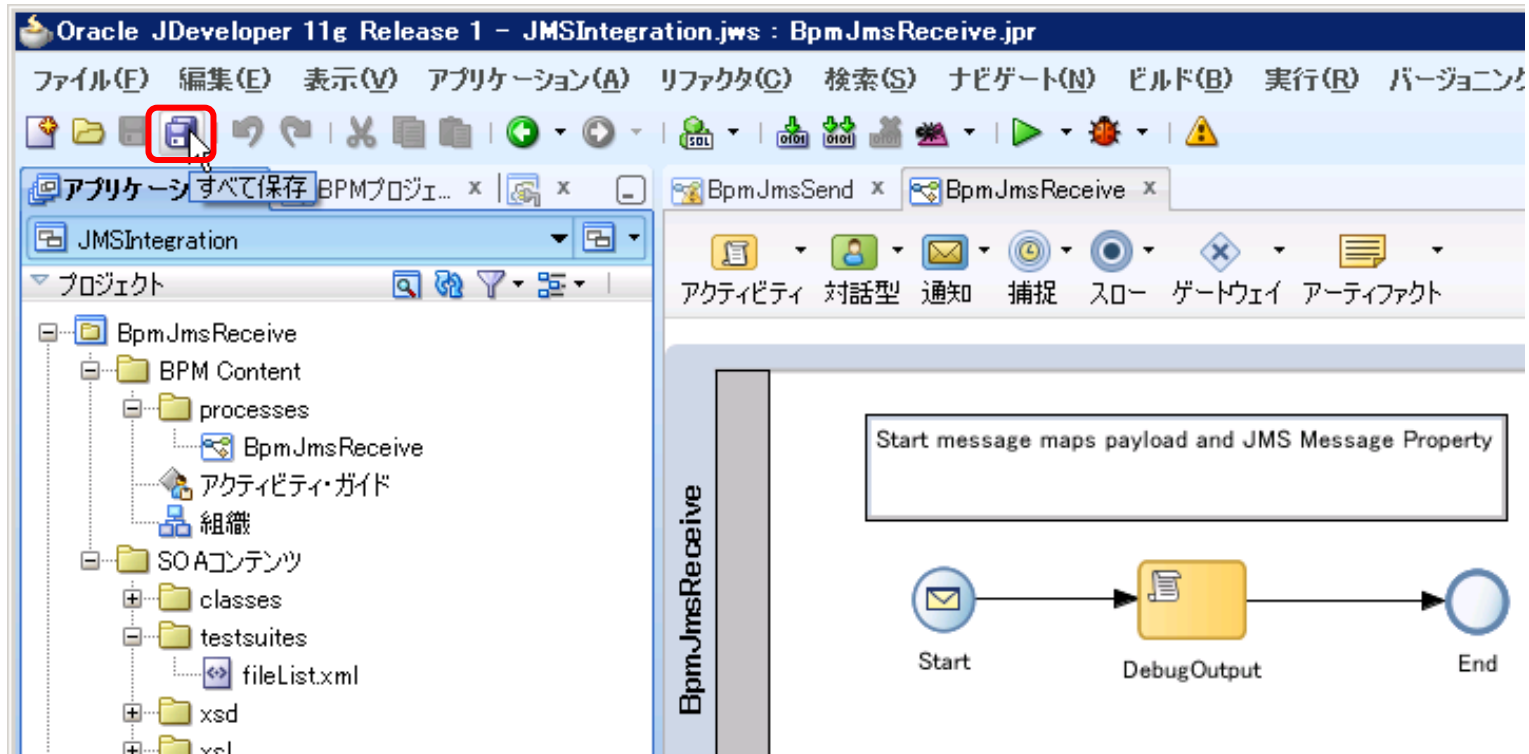
- 「Start」と「End」イベントの間のシーケンス・フロー上にドロップ



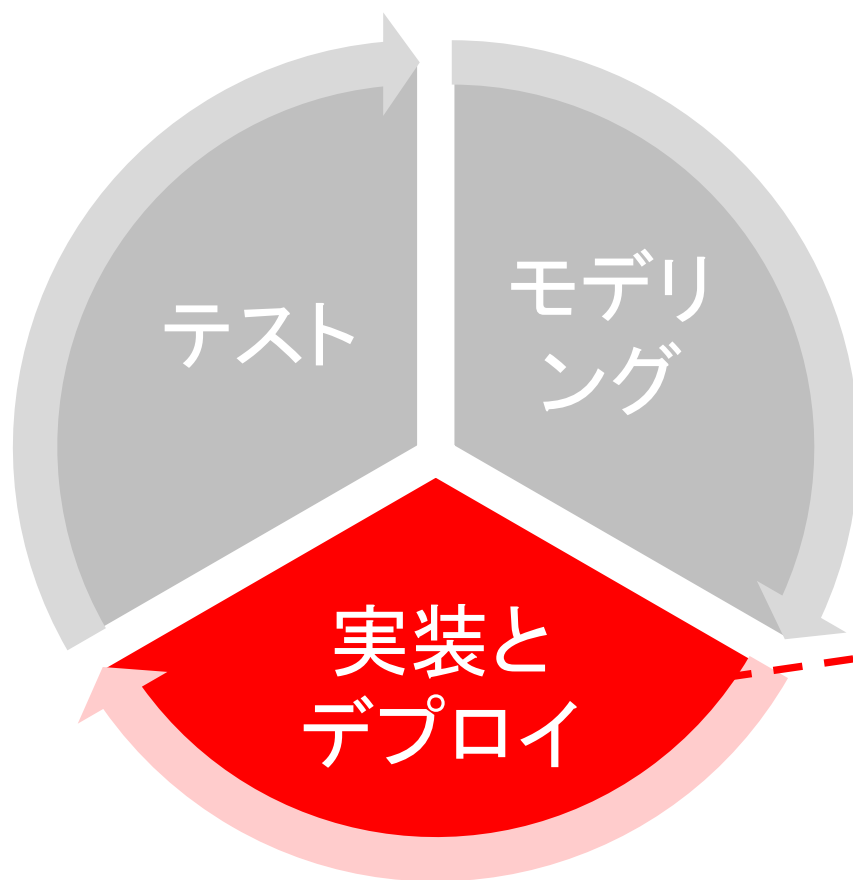
# プロセスのモデリング

## ターゲット・プロセスBpmJmsReceive (9/9)

- 「すべて保存」をし、「BpmJmsSend」タブと「BpmJmsReceive」タブ以外のタブを閉じる



# JMSIntegrationアプリケーションの作成



データ・オブジェクトの作成

各要素の実装  
～ シナリオ1

各要素の実装  
～ シナリオ2

プロセスのデプロイ

# データ・オブジェクトの作成 (1/8)

- ビジネス・オブジェクトの定義(ターゲット・プロセス)
  - Data.NewOrderEvent  
(**NewOrderEvent.xsd**のNewOrderEvent要素に基づく)

**注意:**

NewOrderEvent.xsd ファイルは、左の添付ファイルより、ローカル上に保存しておく



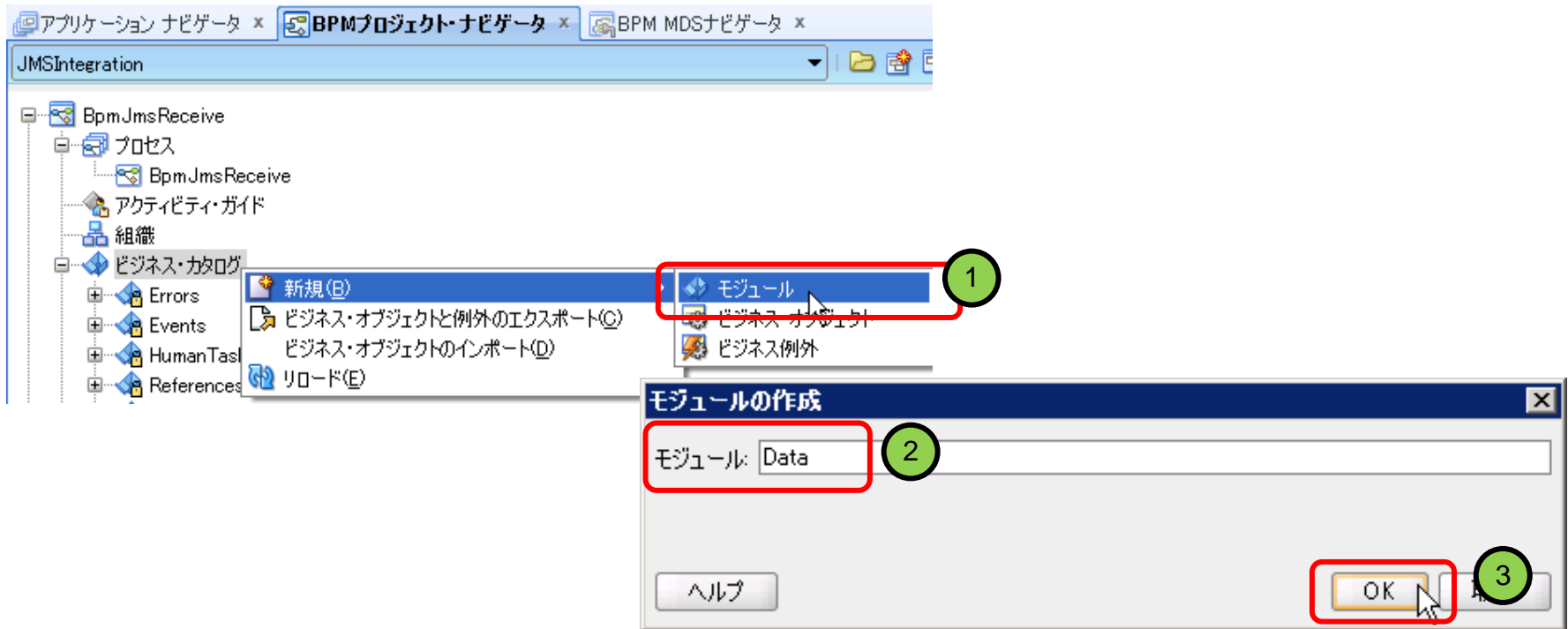
NewOrderEvent.xsd

- データ・オブジェクトの定義(ターゲット・プロセス)
  - **orderAmount** (Int型、入力データ)
  - **orderId** (String型、入力データ)
  - **msgType** (String型、入力データ)
  - **debugString** (String型、入力データ)



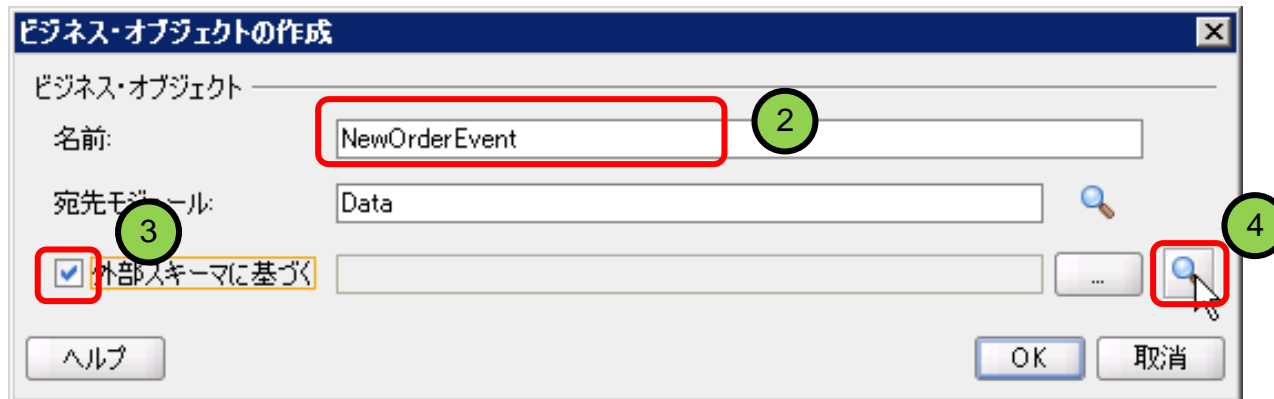
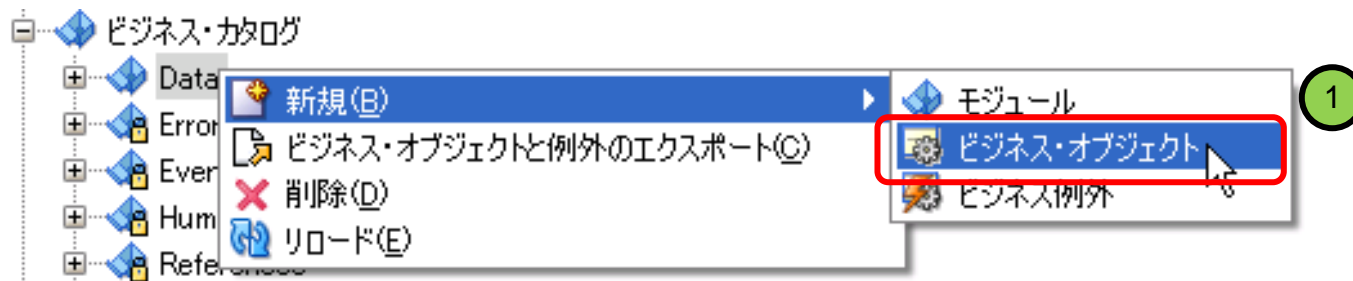
# データ・オブジェクトの作成 (2/8)

- BPMプロジェクト・ナビゲータで、「BpmJmsReceive」プロジェクト配下の「ビジネス・カタログ」を右クリックし、「新規 > モジュール」を選択し、「Data」モジュールを作成



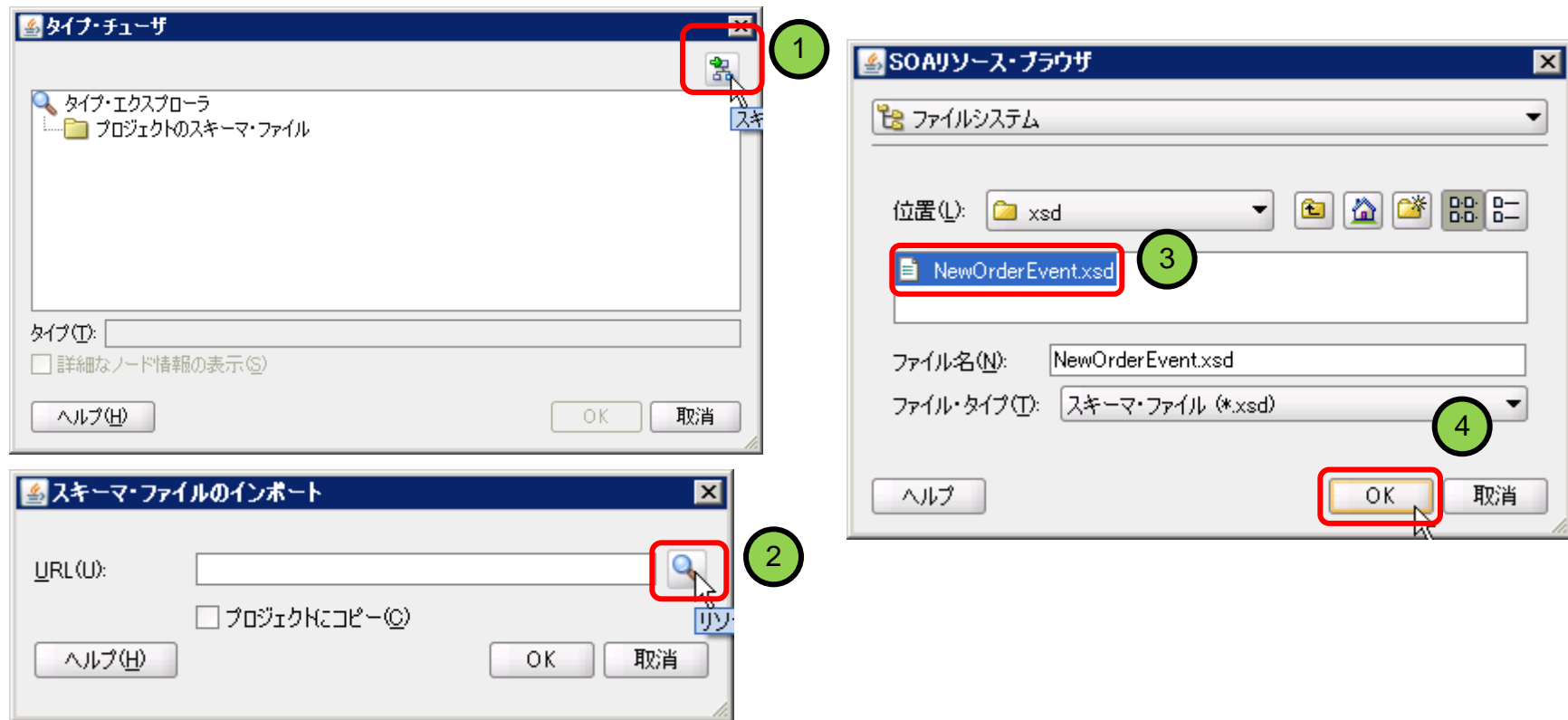
# データ・オブジェクトの作成 (3/8)

- 「Data」の配下に「NewOrderEvent」のビジネス・オブジェクトを作成



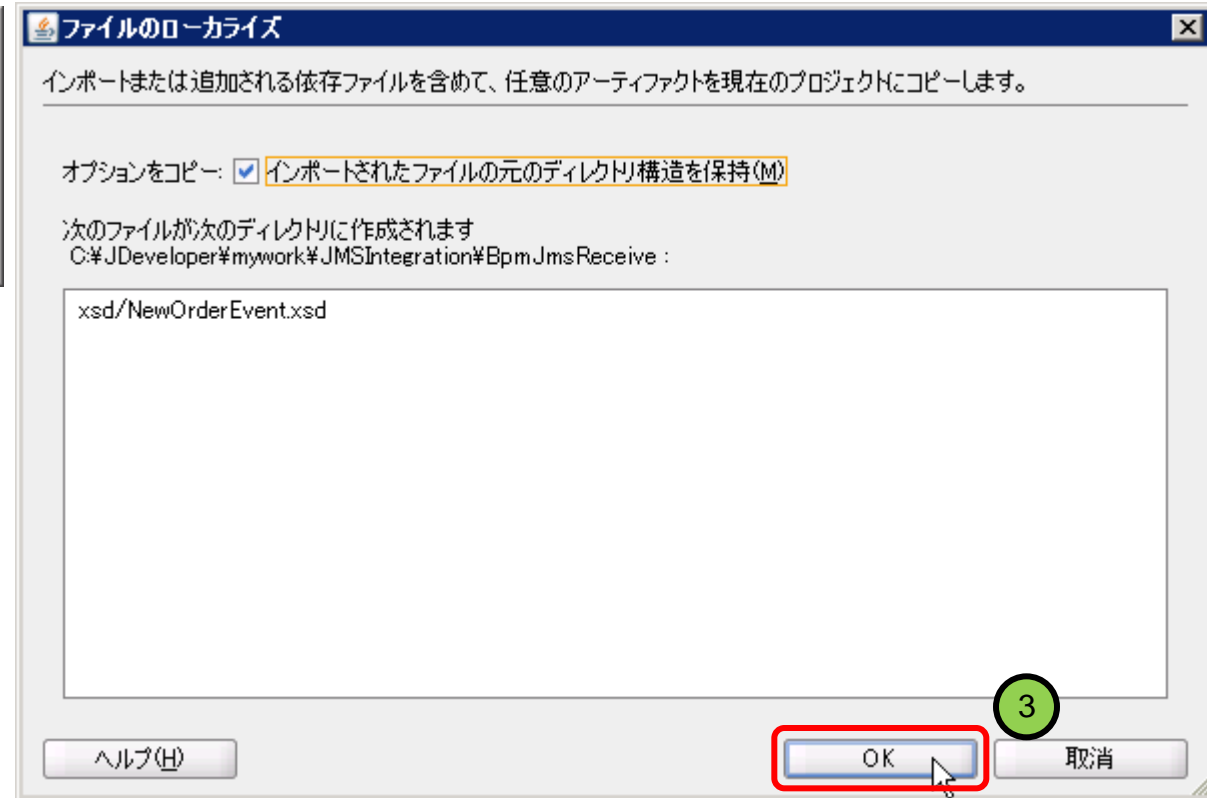
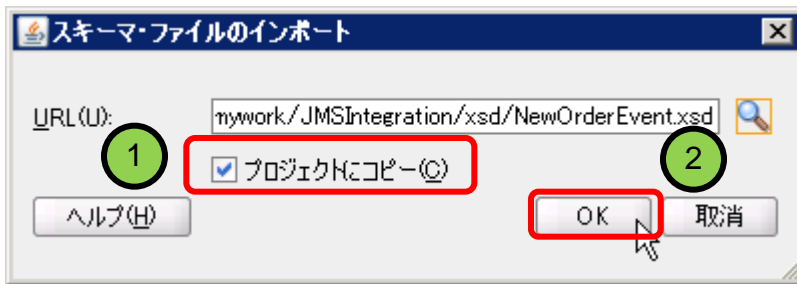
# データ・オブジェクトの作成 (4/8)

- ローカルに保存した添付のスキーマ・ファイル「NewOrderEvent.xsd」を選択



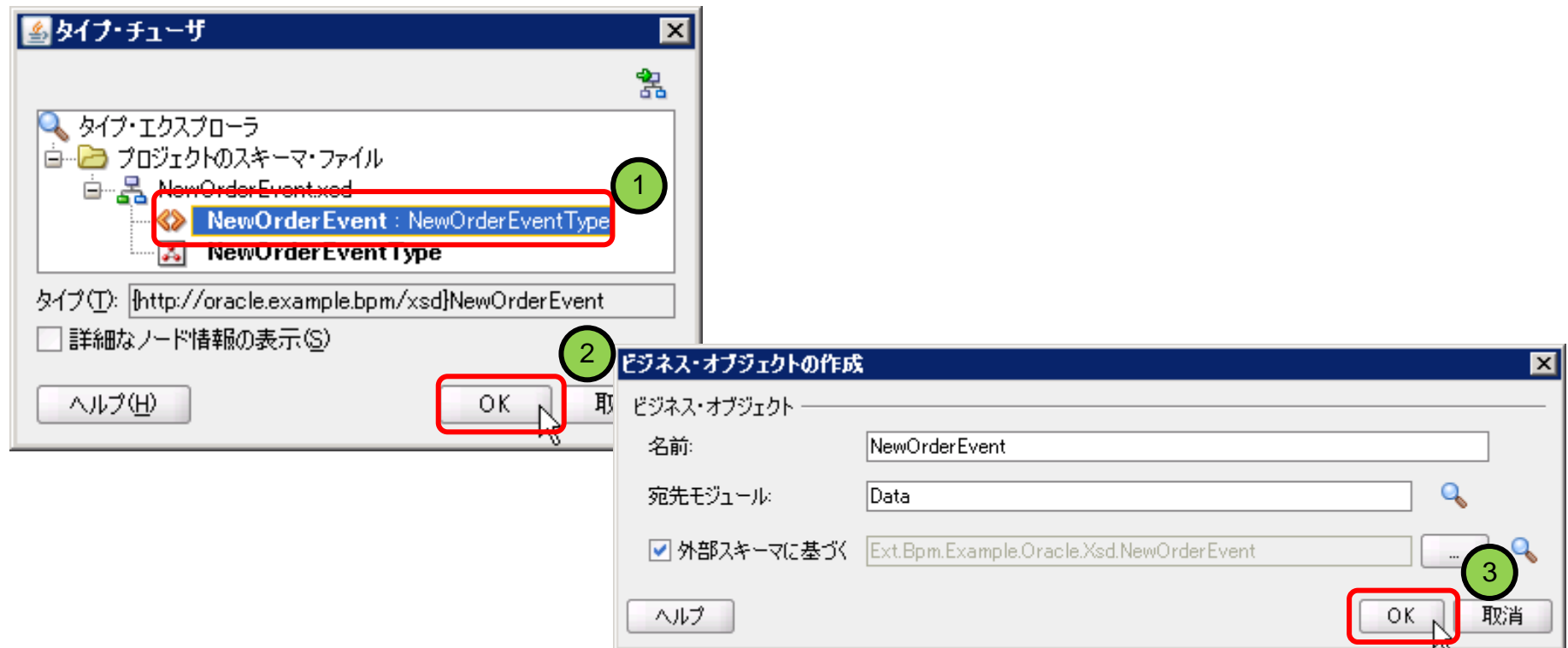
# データ・オブジェクトの作成 (5/8)

- 外部スキーマを「プロジェクトにコピー」するように選択



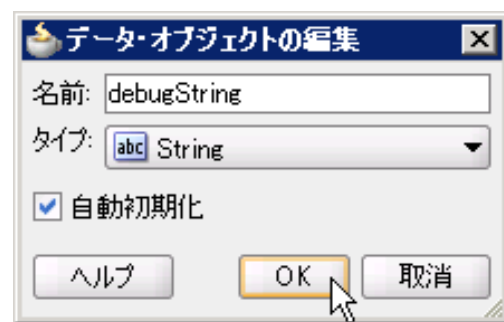
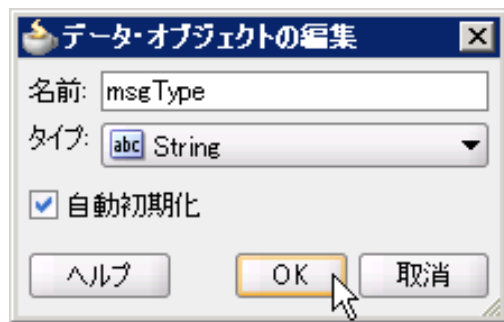
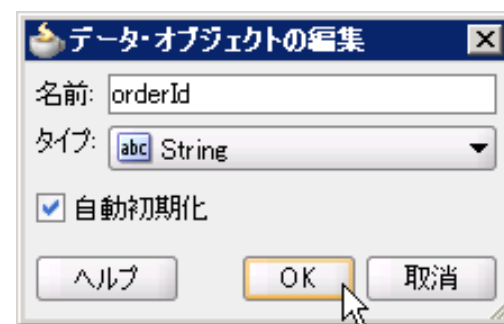
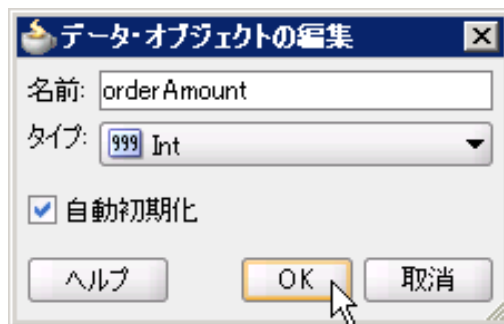
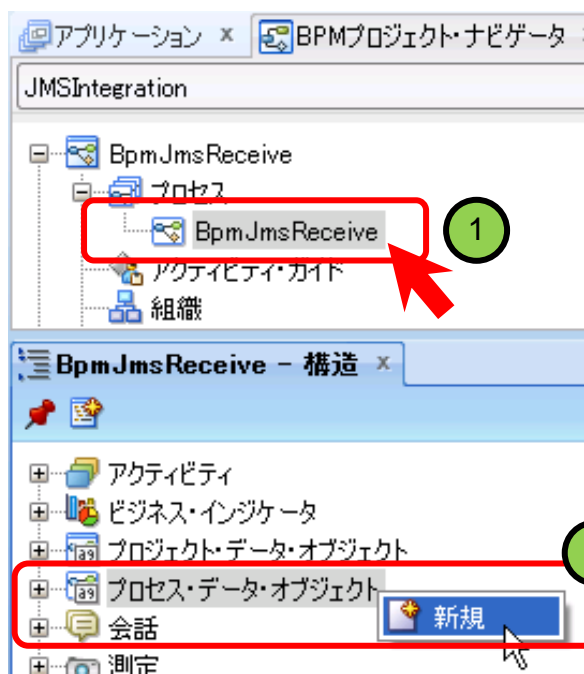
# データ・オブジェクトの作成 (6/8)

- 「NewOrderEvent」要素を選択し、「NewOrderEvent」ビジネス・オブジェクトの作成を完了する



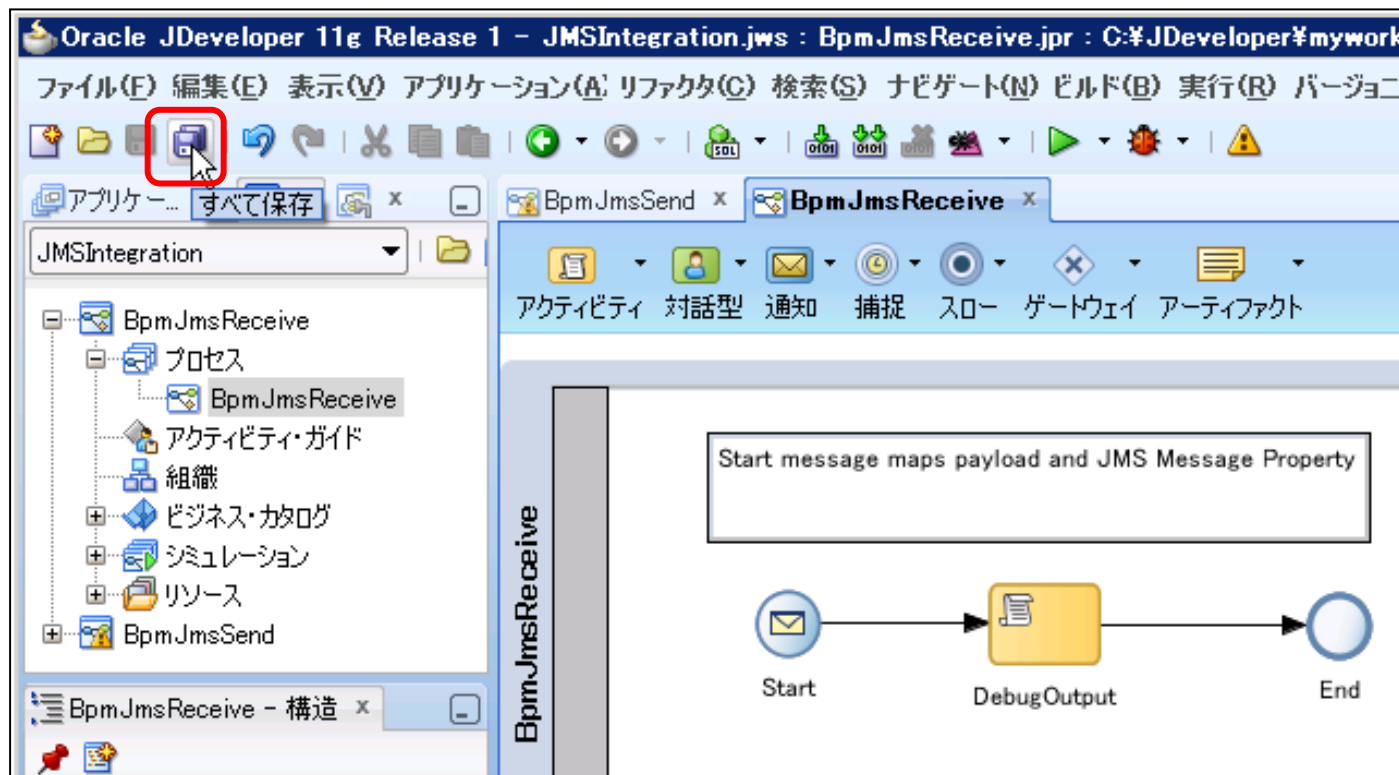
# データ・オブジェクトの作成 (7/8)

- 「BpmJmsReceive」プロセスを選択し、左下に表示される構造情報の「プロセス・データ・オブジェクト」を右クリックし、「新規」より4つのデータ・オブジェクトを新規作成

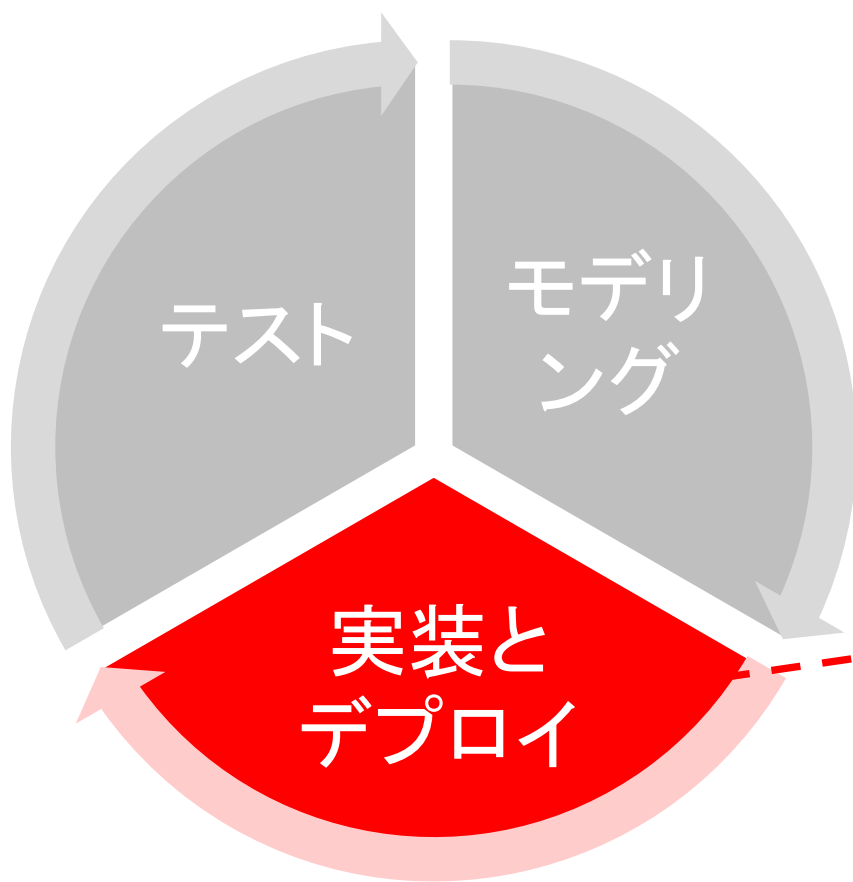


# データ・オブジェクトの作成 (8/8)

- 「すべて保存」をして、「BpmJmsSend」タブと「BpmJmsReceive」タブ以外のタブを閉じる



# JMSIntegrationアプリケーションの作成



データ・オブジェクトの作成

各要素の実装  
～ シナリオ1

各要素の実装  
～ シナリオ2

プロセスのデプロイ



# 各要素の実装 ～ シナリオ1

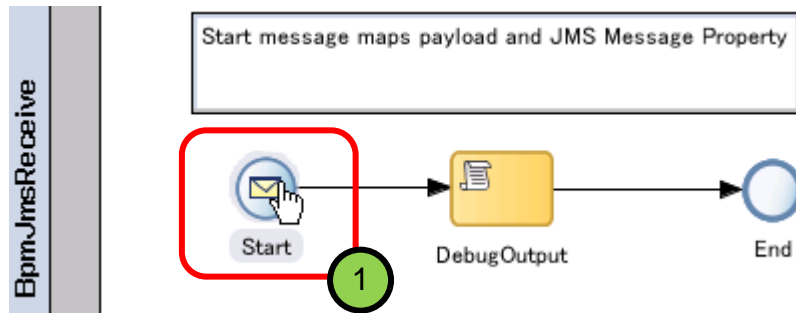
## 実装の手順

手順	概要	詳細	キーワード
1	[ターゲット・プロセス] 開始イベントの実装	ターゲット・プロセス「BpmJmsReceive」のWSDLが必要のため、事前に開始イベントとスクリプト・タスクを実装	WSDL 開始イベント 操作
2	[送信プロセス] JMSキューに送信	JMS送信プロセス「BpmJmsSend」の「SendStartProcessMsg」サービス・タスクを実装し、ターゲット・プロセスのWSDLを使用して、JMSキューにターゲット・プロセスの開始イベント・メッセージを送信	JMSアダプタ メッセージ発行
3	[ターゲット・プロセス] JMSキューから受信	ターゲット・プロセスで、上記のJMSキューから開始イベント・メッセージを受信して、プロセスを呼び出すように実装	JMSアダプタ メッセージ消費

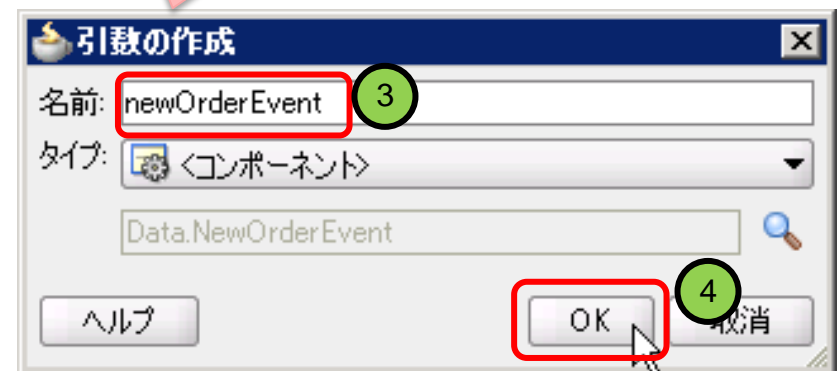
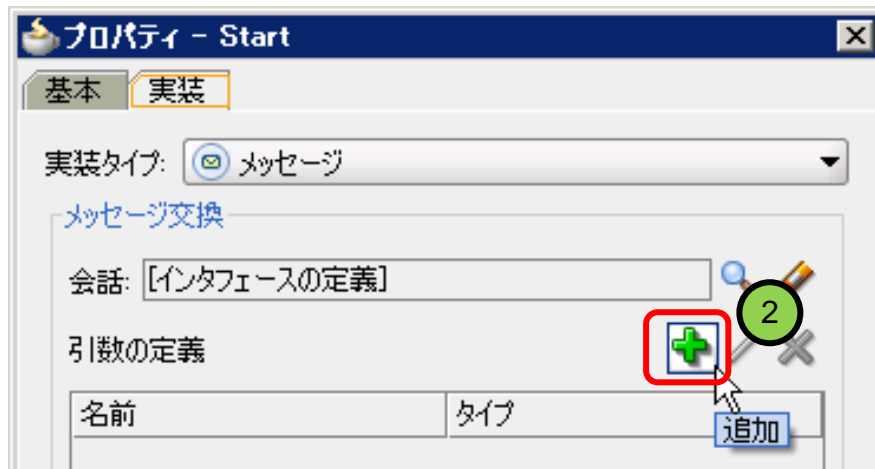
# 各要素の実装 ~ シナリオ1

## [ターゲット・プロセス] 開始イベントの実装 (1/8)

- プロセス・エディターで、「Start」イベントをダブル・クリックし、プロパティを開き、「実装」タブで引数「newOrderEvent」を追加



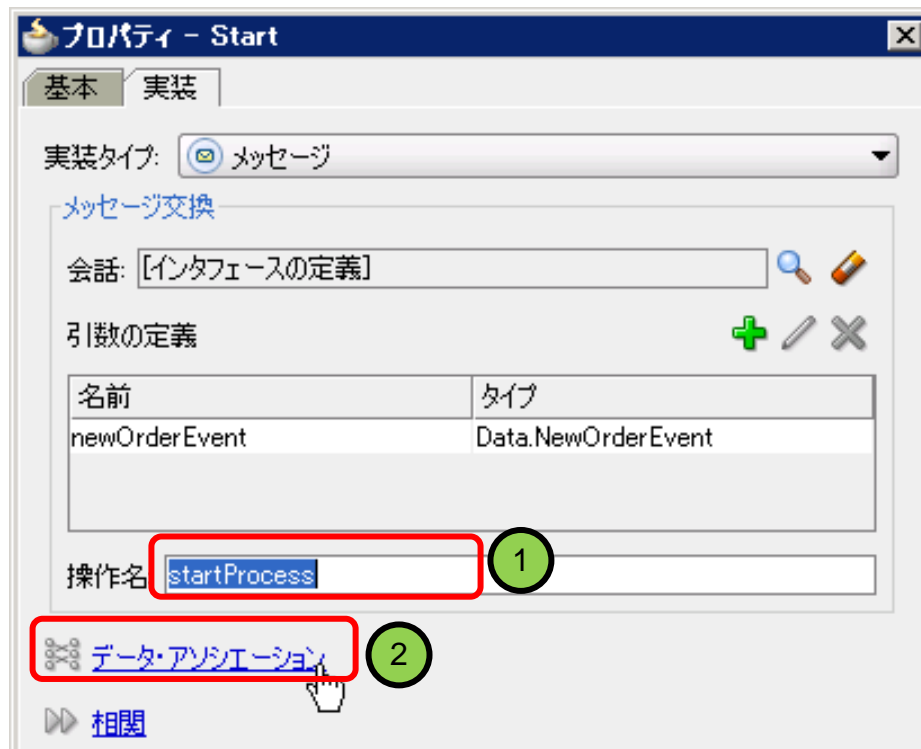
**newOrderEvent:**  
タイプは、「Data.NewOrderEvent」  
ビジネス・オブジェクトとなります。



# 各要素の実装 ～ シナリオ1

## [ターゲット・プロセス] 開始イベントの実装 (2/8)

- 「操作名」に「startProcess」を設定し、「データ・アソシエーション」リンクをクリック



# 各要素の実装 ～ シナリオ1

## [ターゲット・プロセス] 開始イベントの実装 (3/8)

- 「出力」タブで、左の引数をドラッグして右側のプロセスのデータ・オブジェクトにドロップすることにより、以下のデータ・アソシエーションを追加して、「OK」をクリック

データ・アソシエーション

出力

Start

引数

newOrderEvent

999 orderAmount

abc orderId

BpmJmsReceive

データ・オブジェクト

orderAmount 999

orderId abc

msgType abc

コピー 自: newOrderEvent 至: orderId

From	To
999 newOrderEvent.orderAmount	999 orderAmount
abc newOrderEvent.orderId	abc orderId

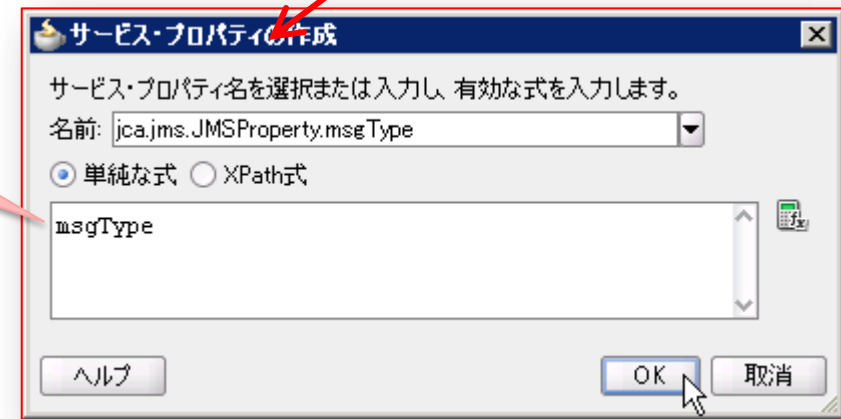
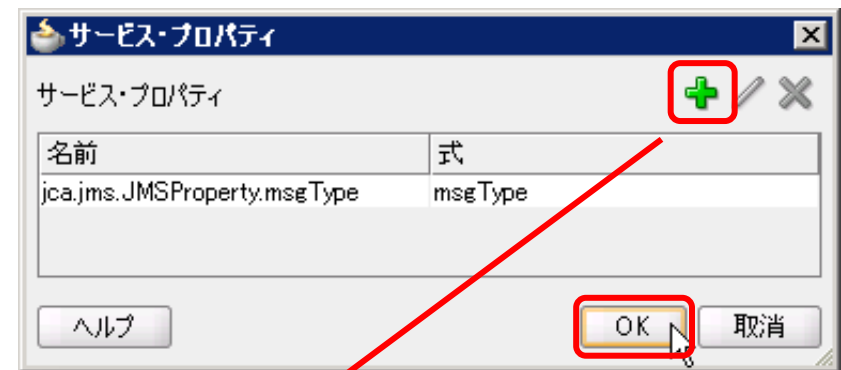
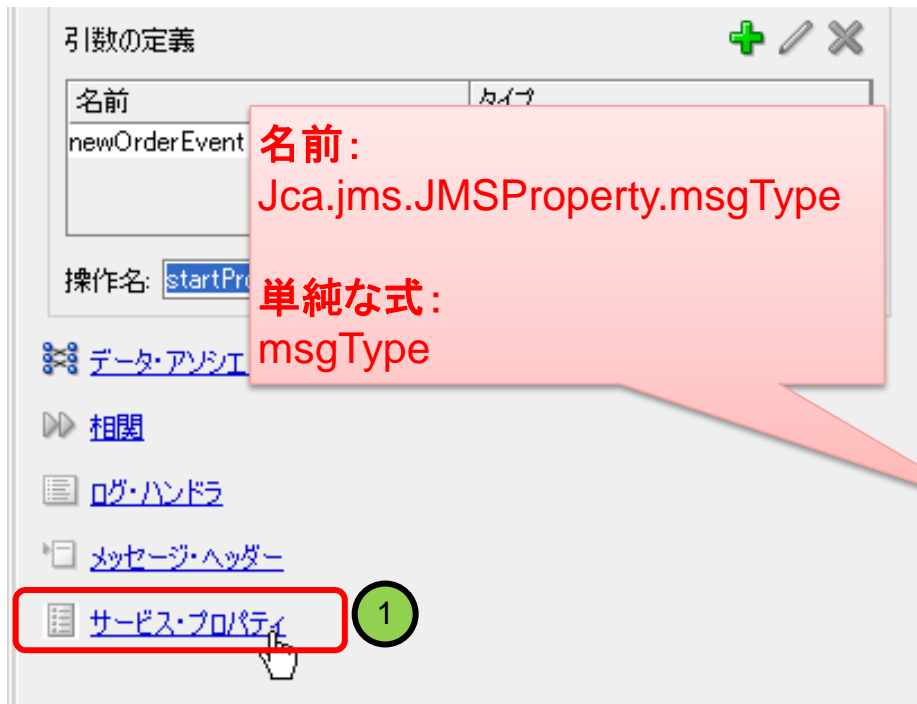
出力データ・アソシエーションの割当て後にターゲットを検証

ヘルプ OK 取消

# 各要素の実装 ～ シナリオ1

## [ターゲット・プロセス] 開始イベントの実装 (4/8)

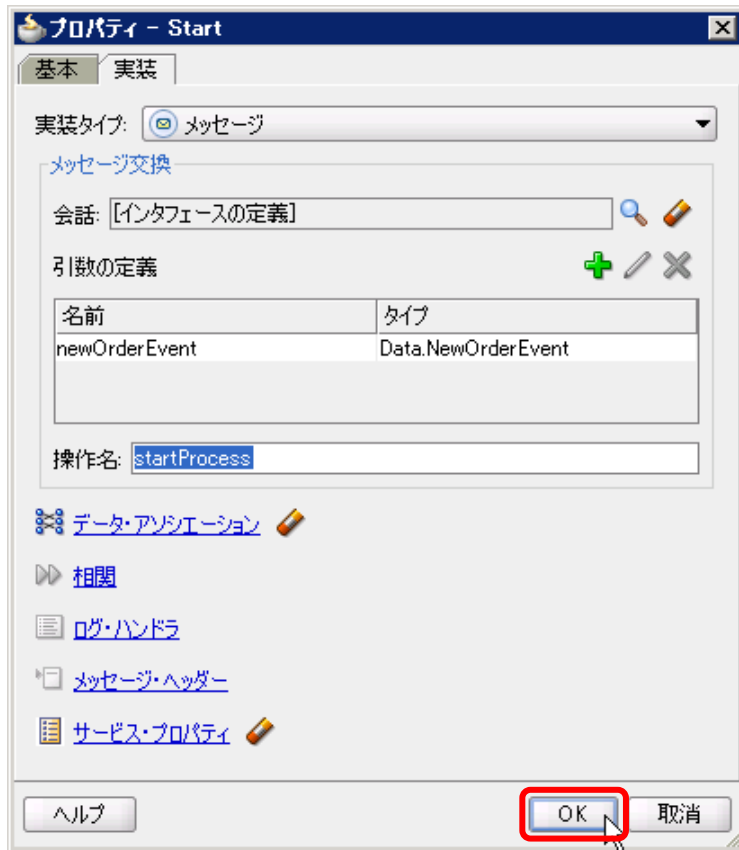
- 「サービス・プロパティ」リンクをクリックして、サービス・プロパティを追加



# 各要素の実装 ~ シナリオ1

## [ターゲット・プロセス] 開始イベントの実装 (5/8)

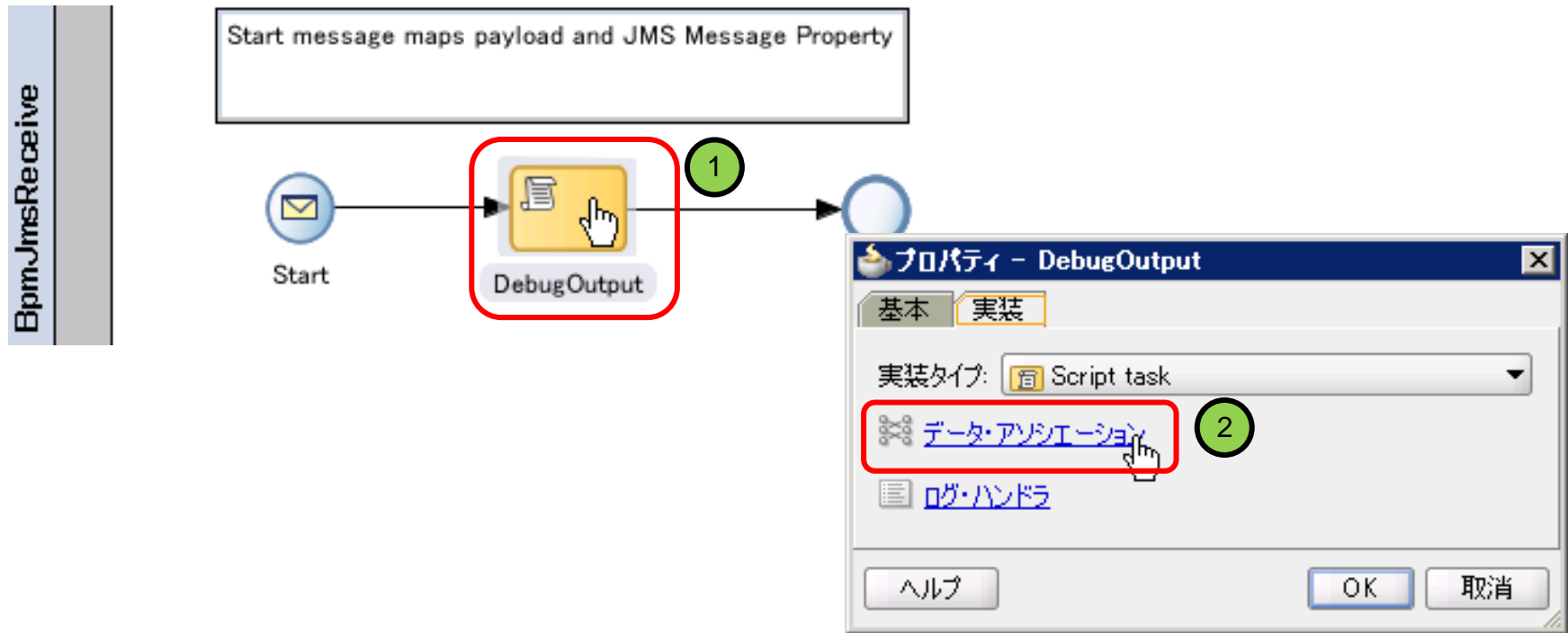
- 「OK」をクリック



# 各要素の実装 ～ シナリオ1

## [ターゲット・プロセス] 開始イベントの実装 (6/8)

- プロセス・エディターで、「DebugOutput」スクリプト・タスクをダブル・クリックし、プロパティを開き、「実装」タブの「データ・アソシエーション」をクリック



# 各要素の実装 ~ シナリオ1

## [ターゲット・プロセス] 開始イベントの実装 (7/8)

- 「式」アイコンを右側のデータ・オブジェクトの「debugString」にドラッグ・アンド・ドロップし、「式ビルダー」に式を設定して、「OK」をクリック

The screenshot displays two windows from Oracle BPM Studio. On the left, the 'データ・アソシエーション' (Data Association) window shows a tree view with 'BpmJmsReceive' selected. The 'debugString' property is highlighted with a red box and a green circle labeled '1'. On the right, the '式ビルダー' (Expression Builder) window shows the formula: "[msgType:" + msgType + "][id:" + orderId + "][amt:" + orderAmount + "]" entered in the '式(E):' field, which is also highlighted with a red box and a green circle labeled '2'. A red callout box at the bottom contains the formula: 式: "[msgType:" + msgType + "][id:" + orderId + "][amt:" + orderAmount + "]". The 'OK' button in the Expression Builder window is highlighted with a red box and a green circle labeled '3'.



# 各要素の実装 ～ シナリオ1

## [ターゲット・プロセス] 開始イベントの実装 (8/8)

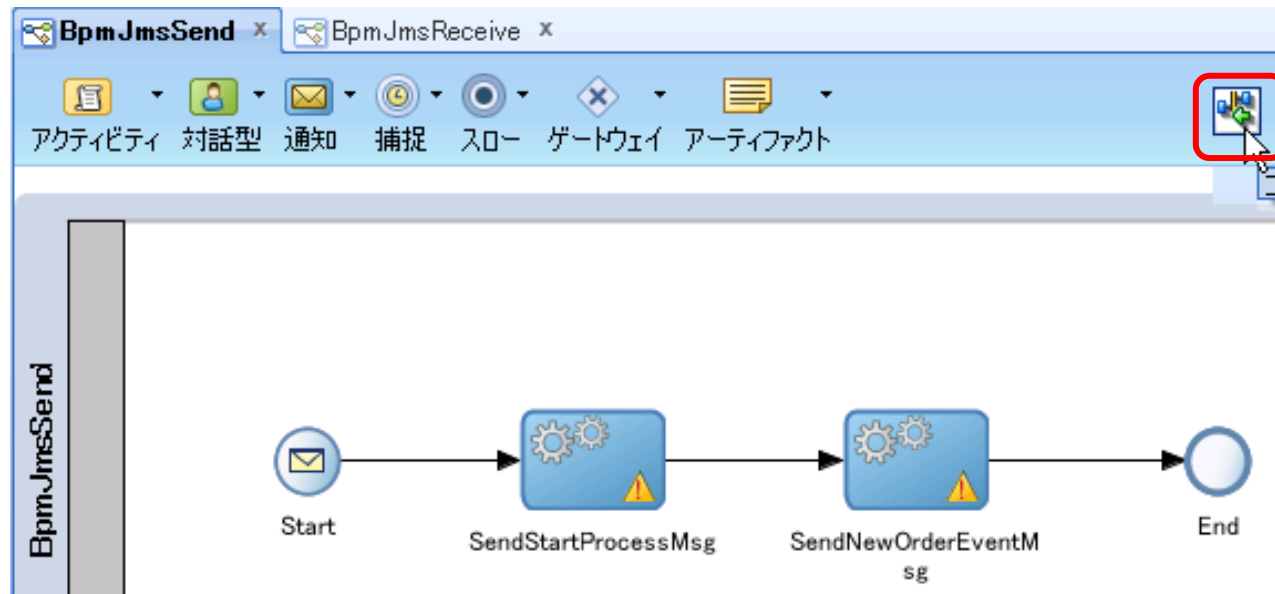
- 2回「OK」をクリックし、「DebugOutput」スクリプト・タスクの実装が完了し、「すべて保存」をする

The screenshot displays the Oracle BPM Studio interface. The main window is titled "データ・アソシエーション" (Data Association) and shows a process diagram with a "DebugOutput" task. A "プロパティ - DebugOutput" (Properties - DebugOutput) dialog box is open, showing the "実装" (Implementation) tab. The "実装タイプ" (Implementation Type) is set to "Script task". The "データ・アソシエーション" (Data Association) is selected, and the "ログ・ハンドラ" (Log Handler) is also visible. The "OK" button is highlighted with a red box and a green circle containing the number "1". A second "OK" button is highlighted with a red box and a green circle containing the number "2".

# 各要素の実装 ～ シナリオ1

## [送信プロセス] JMSキューに送信 (1/18)

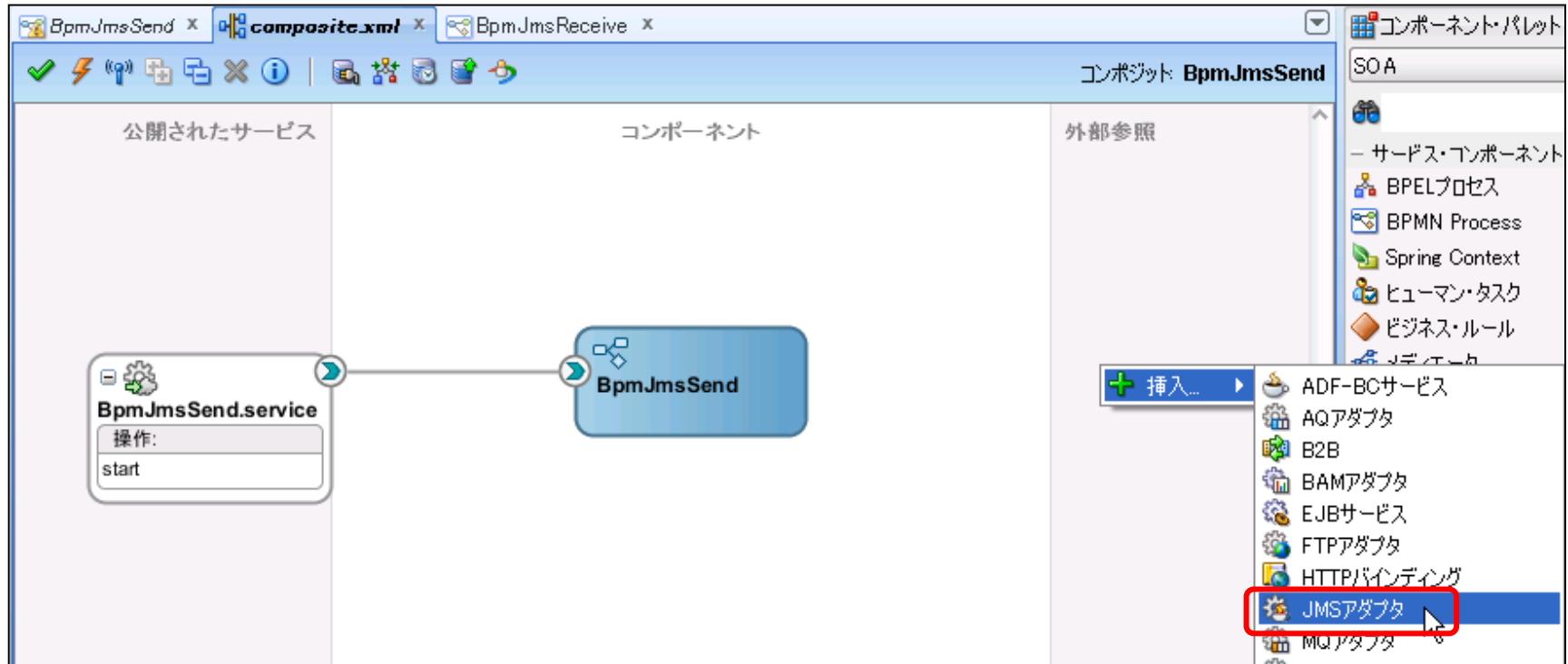
- 「BpmJmsSend」プロセス・エディタで、「コンポジット・エディタ」アイコンをクリック、コンポジット・エディターへ移動



# 各要素の実装 ~ シナリオ1

## [送信プロセス] JMSキューに送信 (2/18)

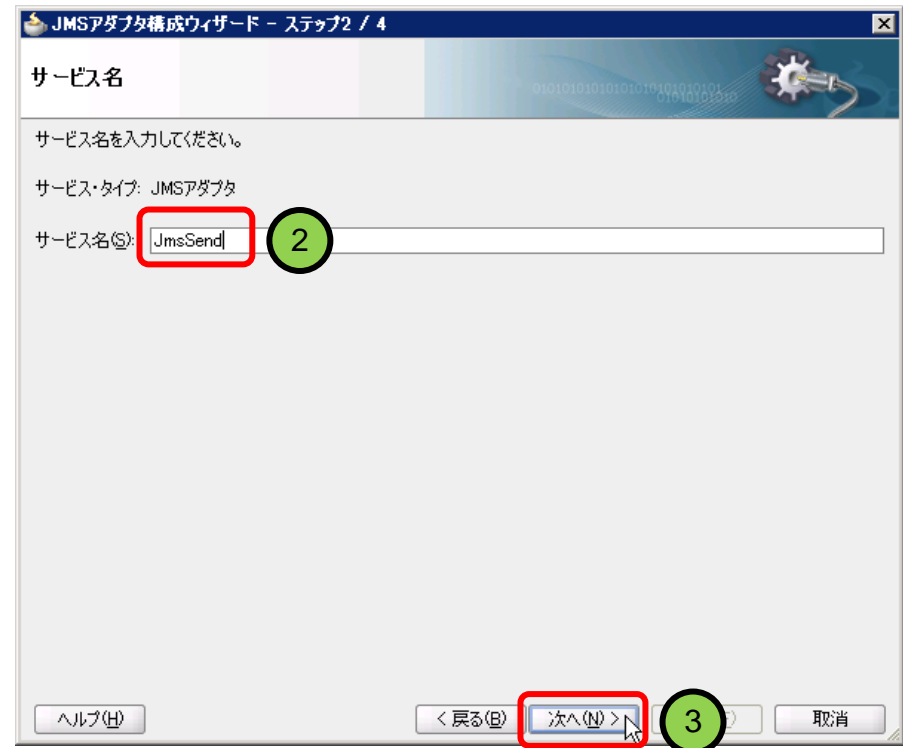
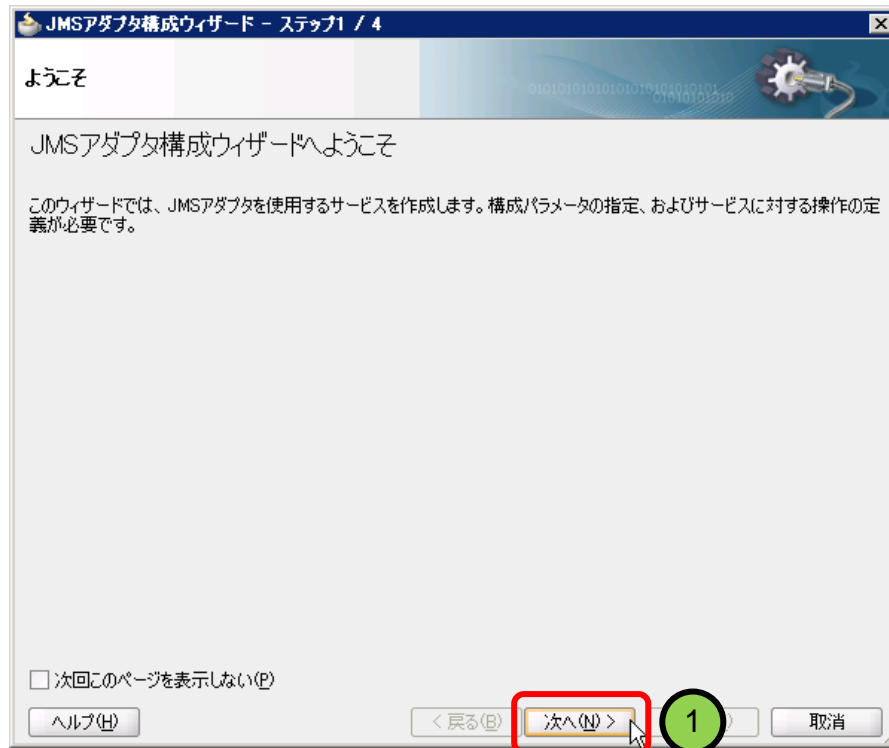
- コンポジット・エディタの「外部参照」欄で、右クリックして「挿入 > JMSアダプタ」を選択



# 各要素の実装 ~ シナリオ1

## [送信プロセス] JMSキューに送信 (3/18)

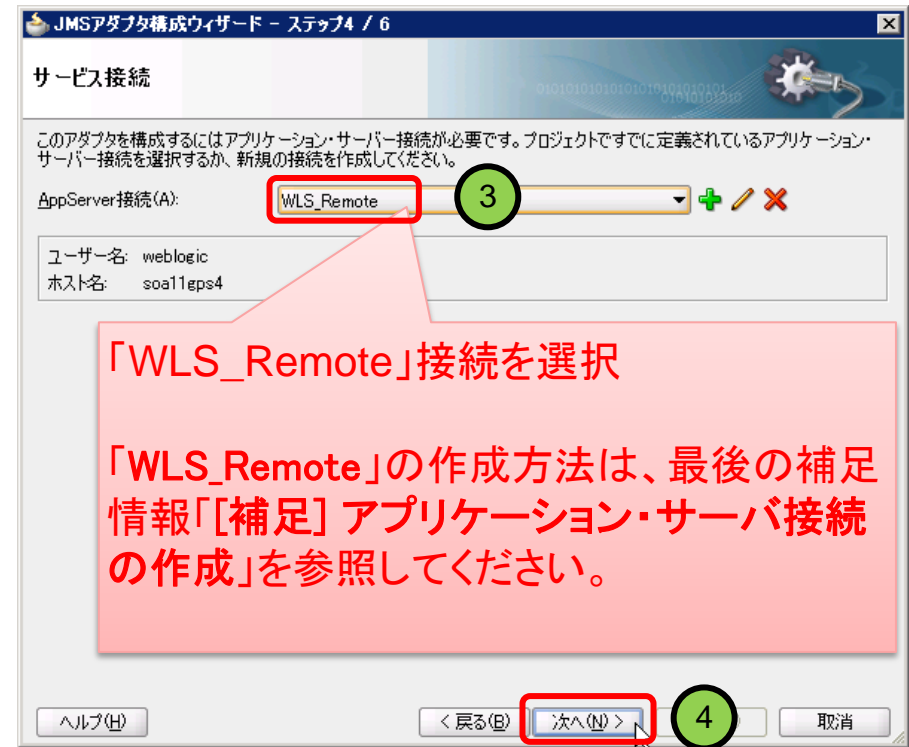
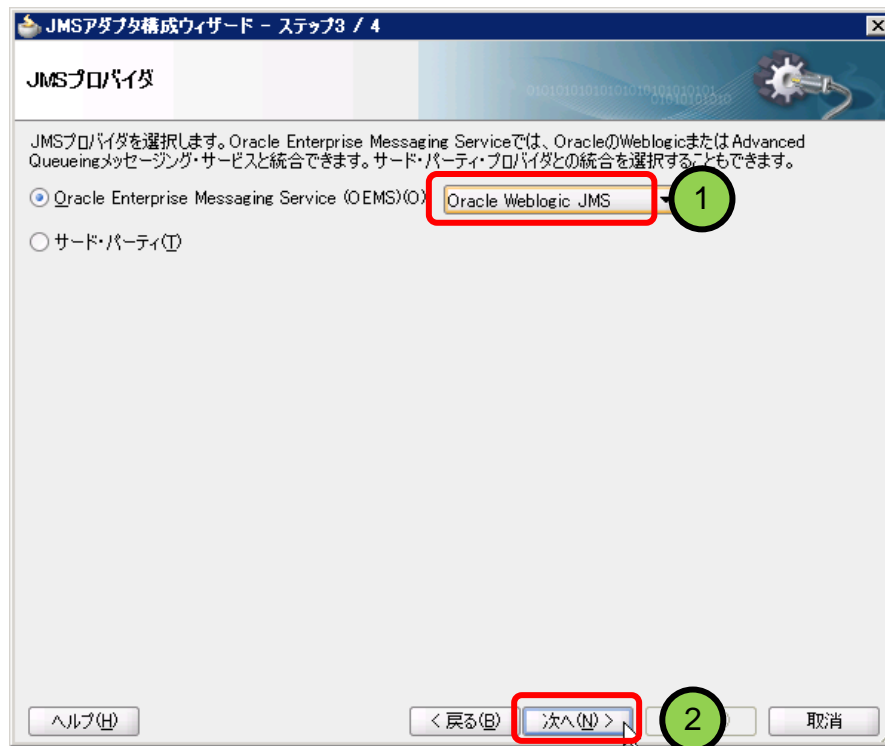
- 「ようこそ」画面で「次へ」
- 「サービス名」に「JmsSend」を入力して「次へ」



# 各要素の実装 ～ シナリオ1

## [送信プロセス] JMSキューに送信 (4/18)

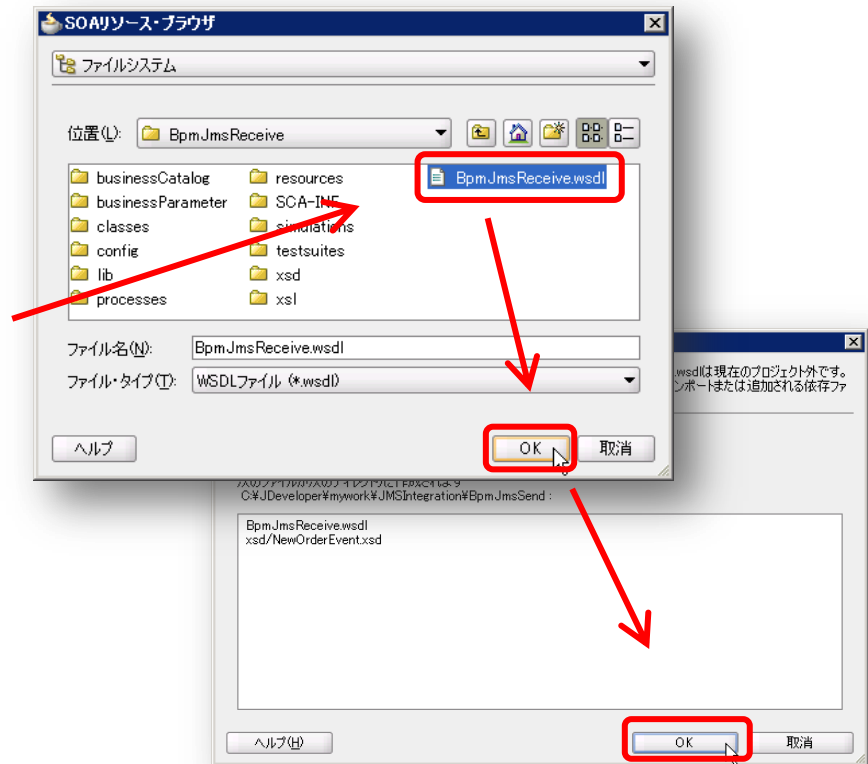
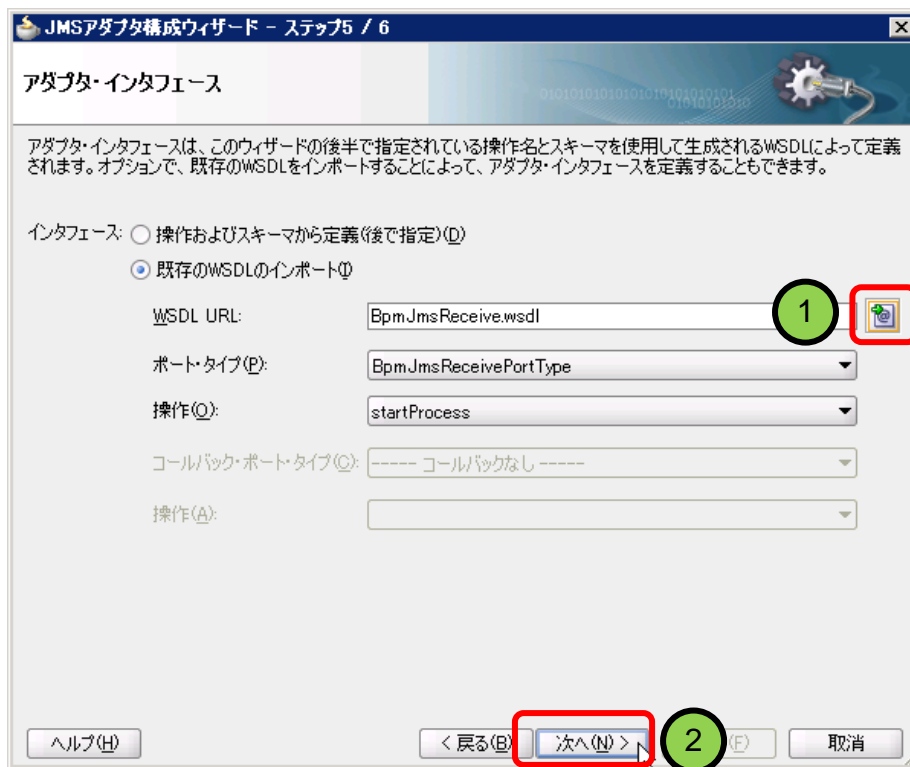
- 「Oracle Weblogic JMS」を選択して「次へ」
- 事前作成した「WLS\_Remote」接続を選択して「次へ」



# 各要素の実装 ~ シナリオ1

## [送信プロセス] JMSキューに送信 (5/18)

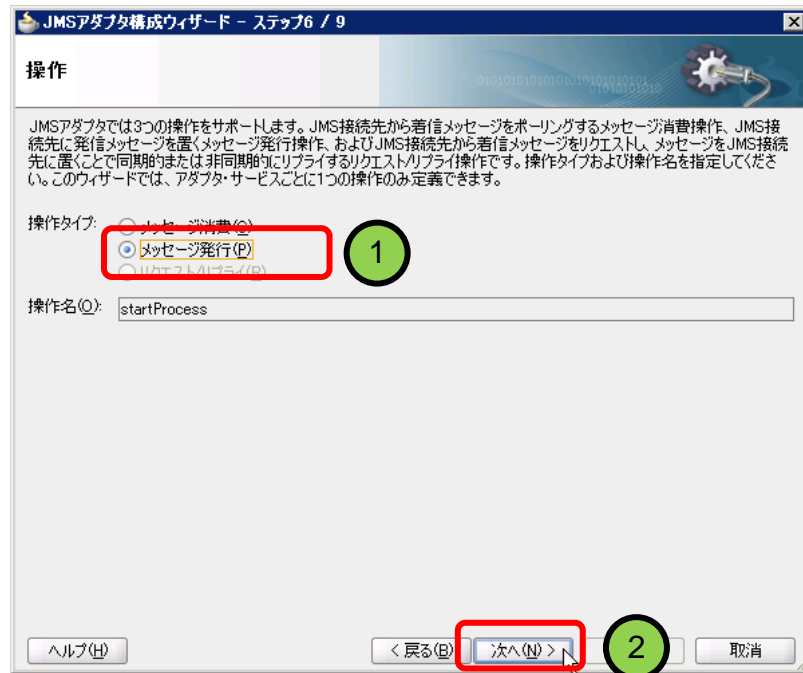
- ターゲット・プロセス「BpmJmsReceive」のプロジェクトのディレクトリから、「BpmJmsReceive.wsdl」を選択して「次へ」



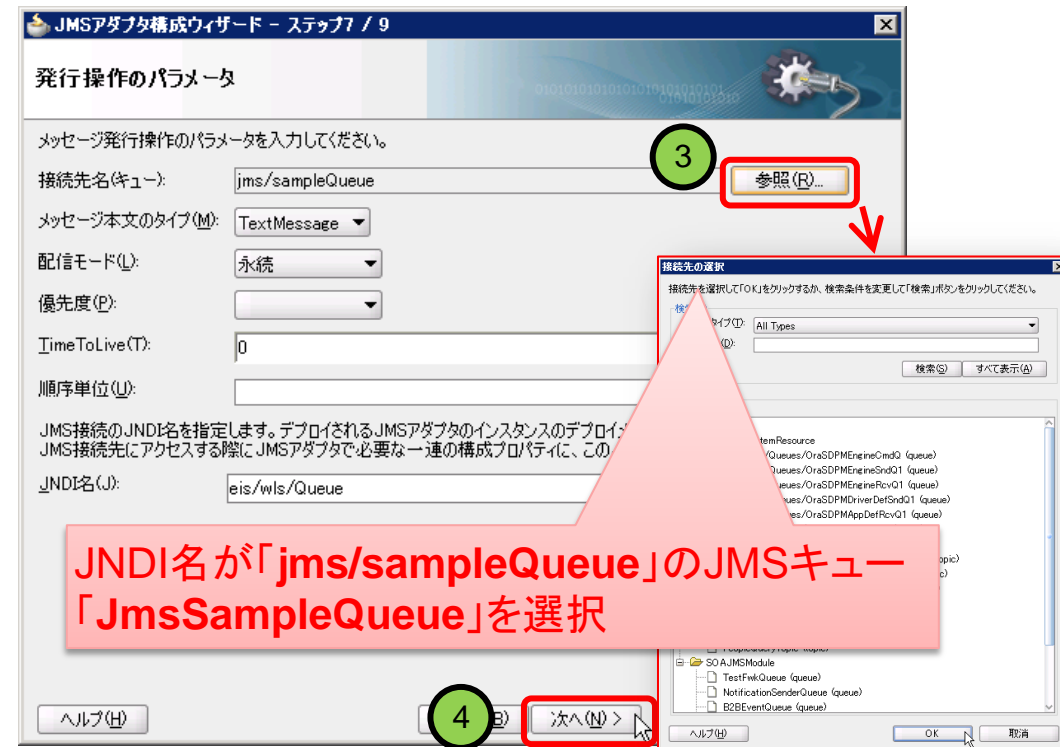
# 各要素の実装 ~ シナリオ1

## [送信プロセス] JMSキューに送信 (6/18)

- 「メッセージ発行」を選択して「次へ」



- 環境準備で作成したJMSキューを選択して「次へ」

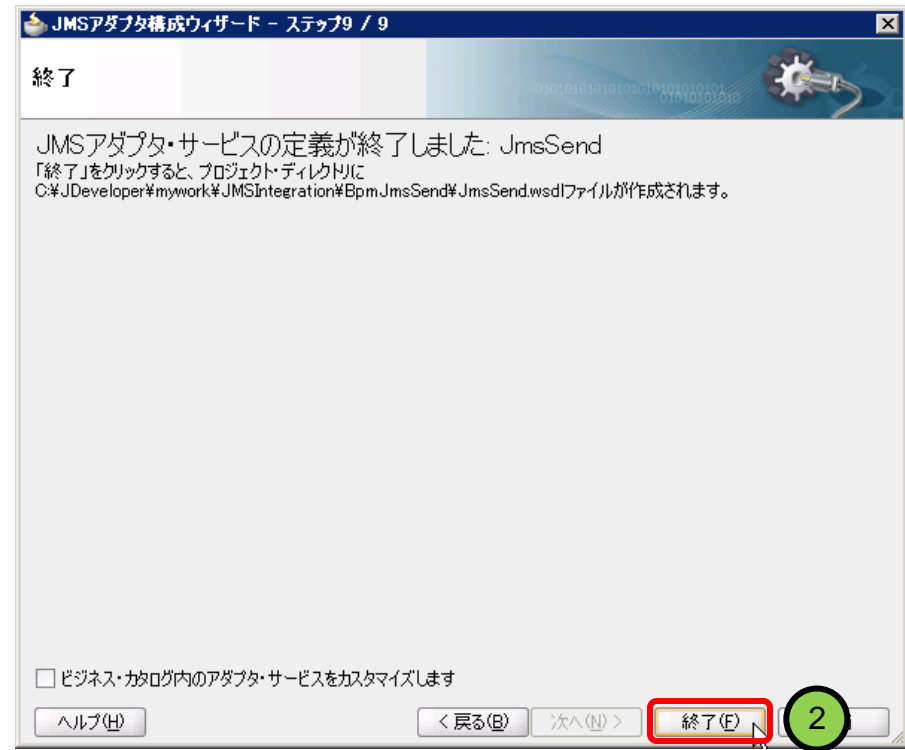
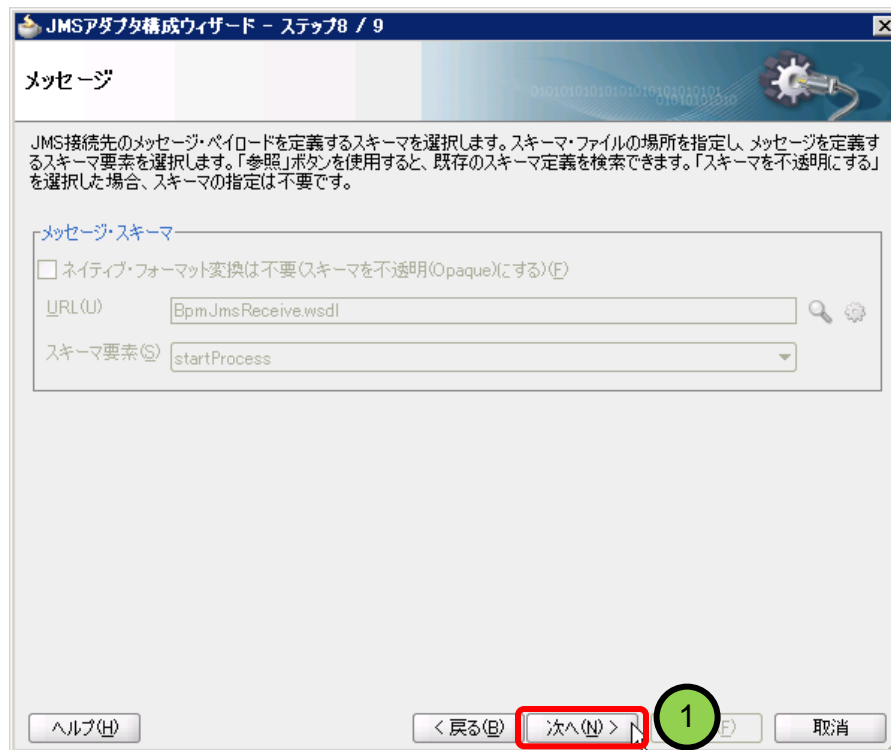


# 各要素の実装 ~ シナリオ1

## [送信プロセス] JMSキューに送信 (7/18)

- 「次へ」をクリック

- 「終了」をクリックし、「すべて保存」をする

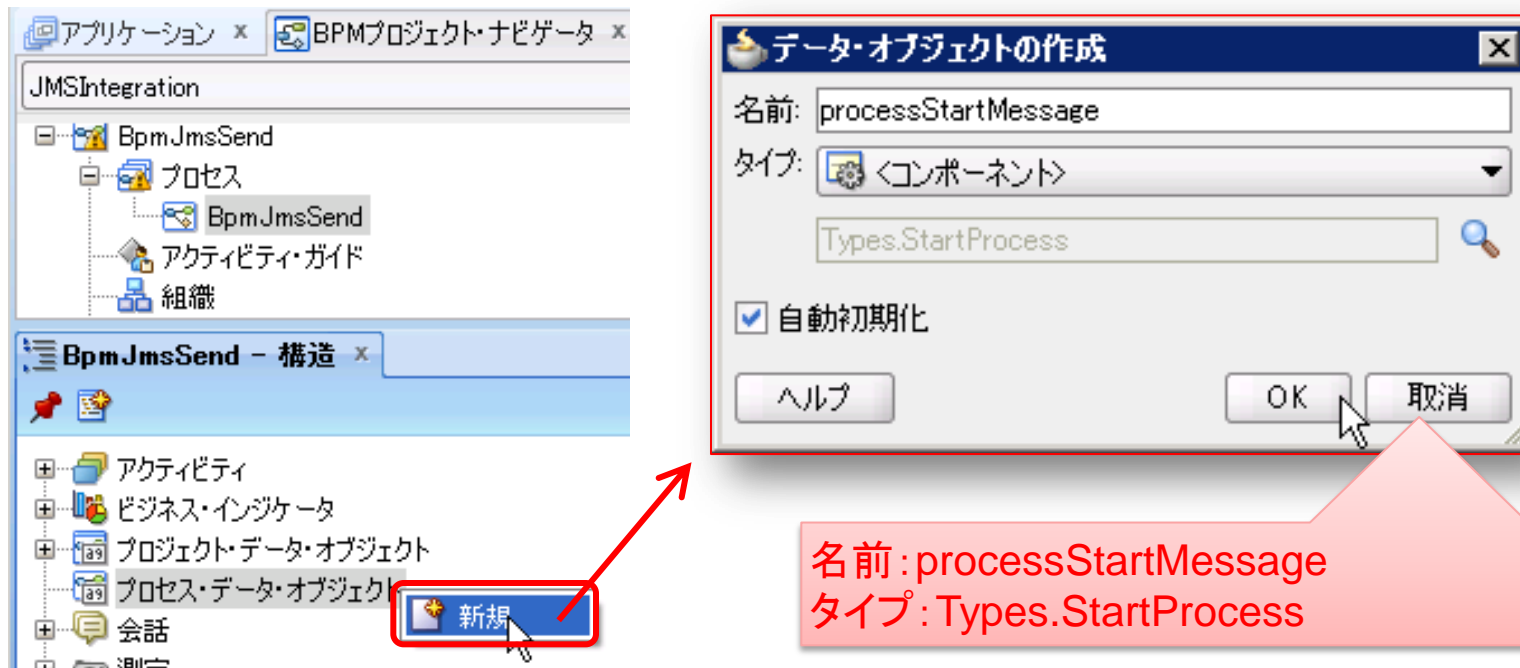




# 各要素の実装 ~ シナリオ1

## [送信プロセス] JMSキューに送信 (8/18)

- 「BPMプロジェクト・ナビゲータ」で「BpmJmsSend」プロセスを選択し、左下に表示される構造情報の「プロセス・データ・オブジェクト」を右クリックし、「新規」より以下のデータ・オブジェクトを新規作成



名前: processStartMessage  
タイプ: Types.StartProcess

# 各要素の実装 ～ シナリオ1

## [送信プロセス] JMSキューに送信 (9/18)

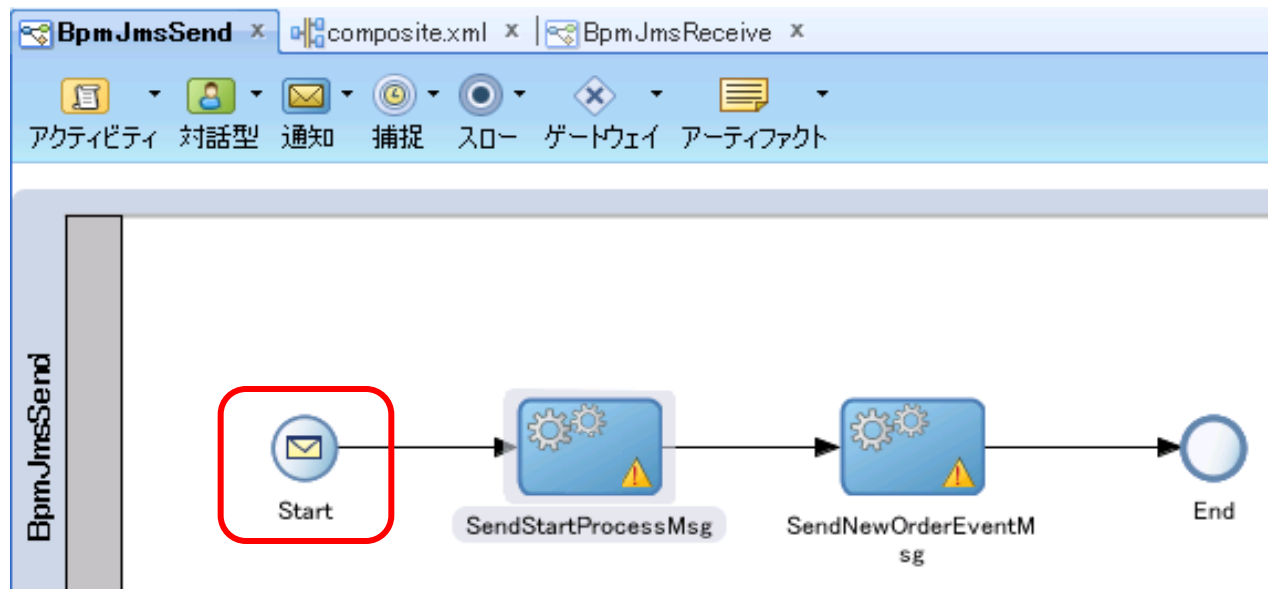
- 構造情報の「会話」を右クリックし、「新規」をクリックしてJMSアダプタ・サービス「JmsSend」のサービス・コールの会話を作成



# 各要素の実装 ～ シナリオ1

## [送信プロセス] JMSキューに送信 (10/18)

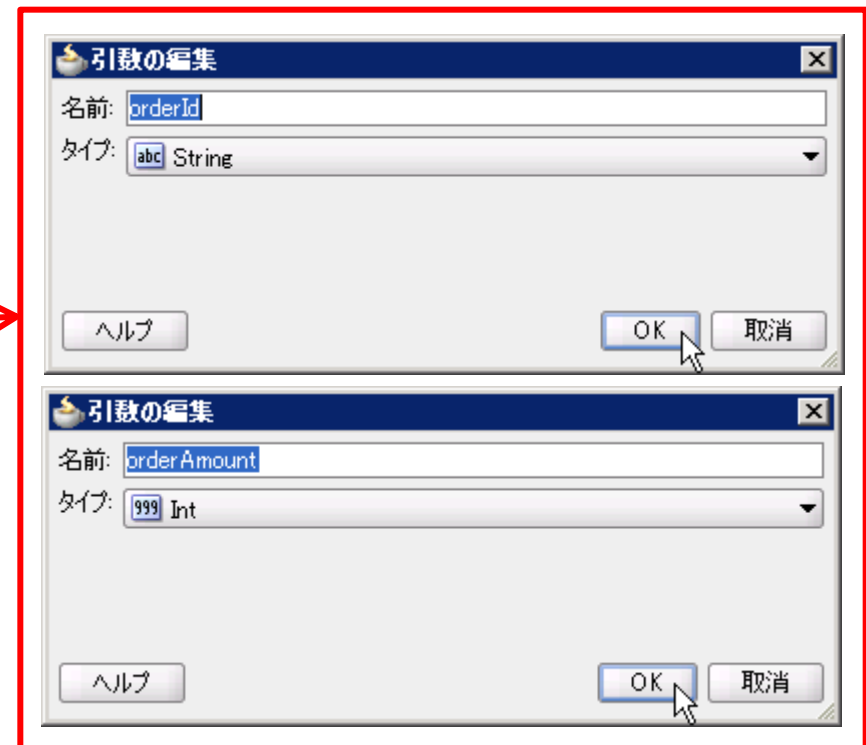
- 「Start」イベントをダブル・クリックし、「プロパティ」を開く



# 各要素の実装 ～ シナリオ1

## [送信プロセス] JMSキューに送信 (11/18)

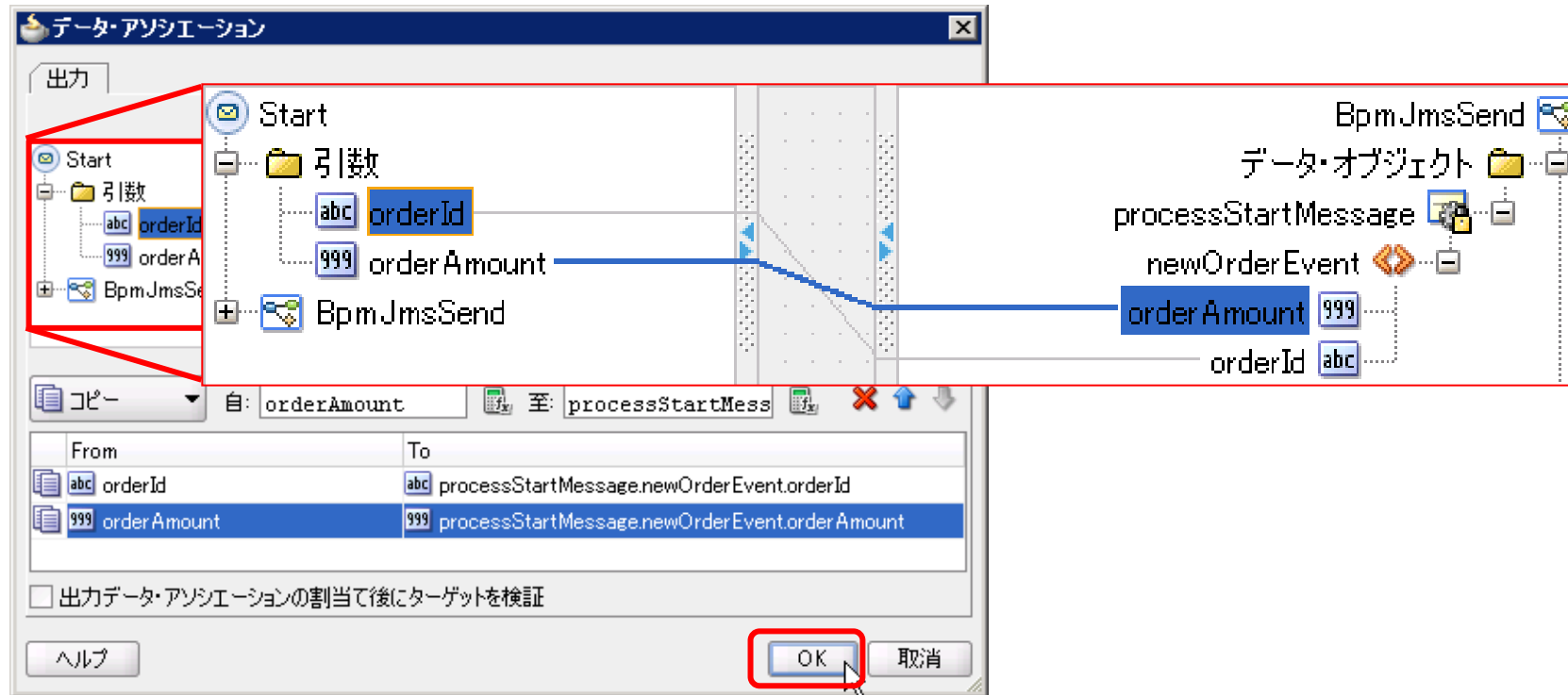
- 引数「orderId」と「orderAmount」を追加し、「操作名」に「operation」を設定して、「データ・アソシエーション」をクリック



# 各要素の実装 ～ シナリオ1

## [送信プロセス] JMSキューに送信 (12/18)

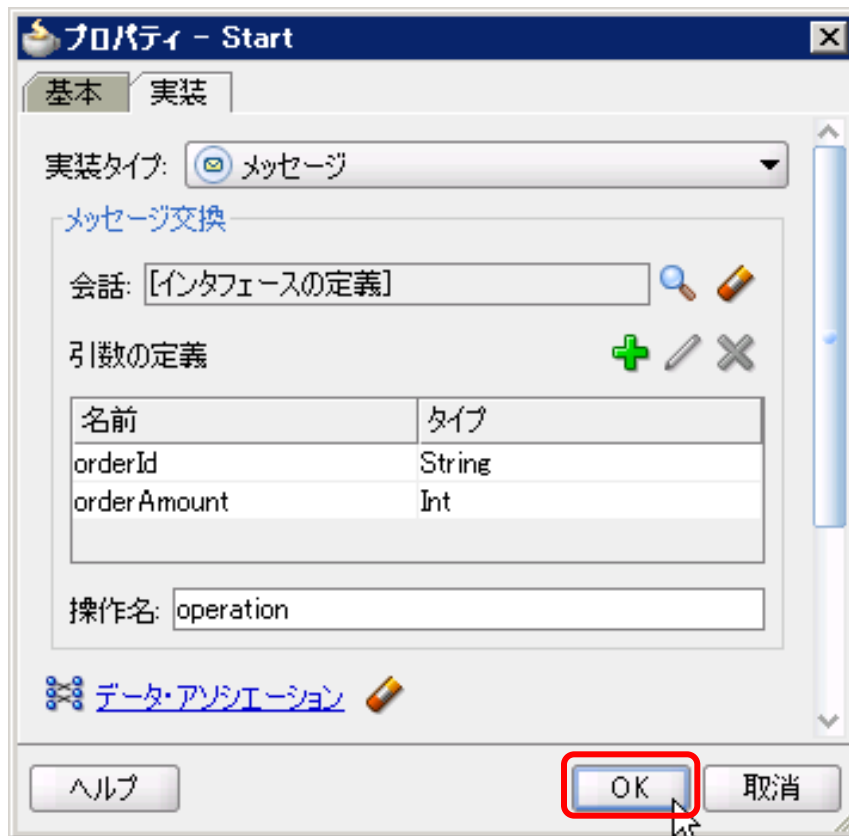
- 「出力」タブで、左の引数をドラッグして右側のプロセスのデータ・オブジェクトにドロップすることにより、以下のデータ・アソシエーションを追加して、「OK」をクリック



# 各要素の実装 ～ シナリオ1

## [送信プロセス] JMSキューに送信 (13/18)

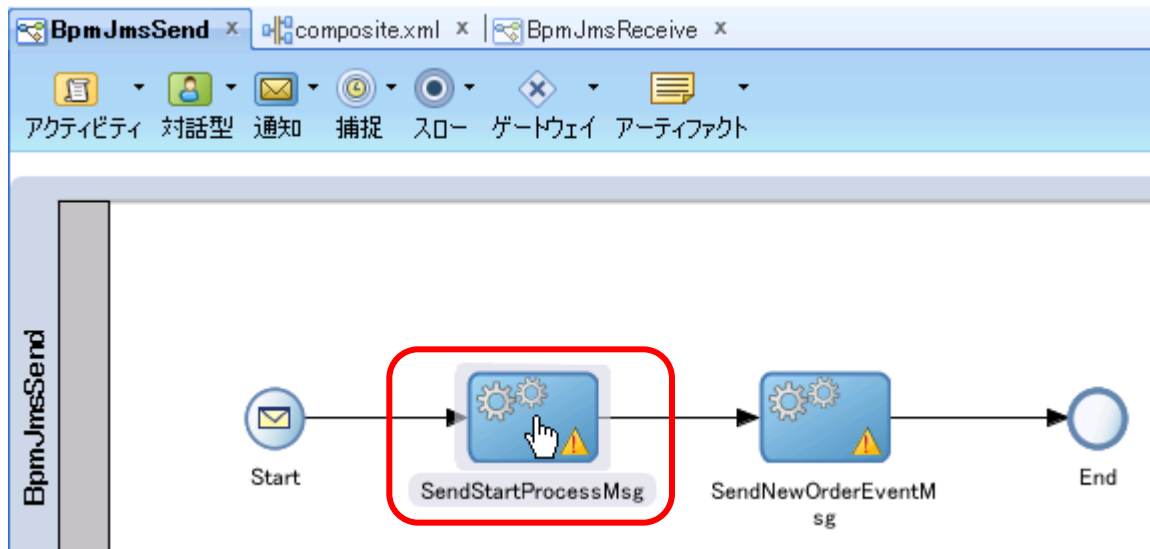
- 「OK」をクリックし、「Start」イベントの実装が完了



# 各要素の実装 ～ シナリオ1

## [送信プロセス] JMSキューに送信 (14/18)

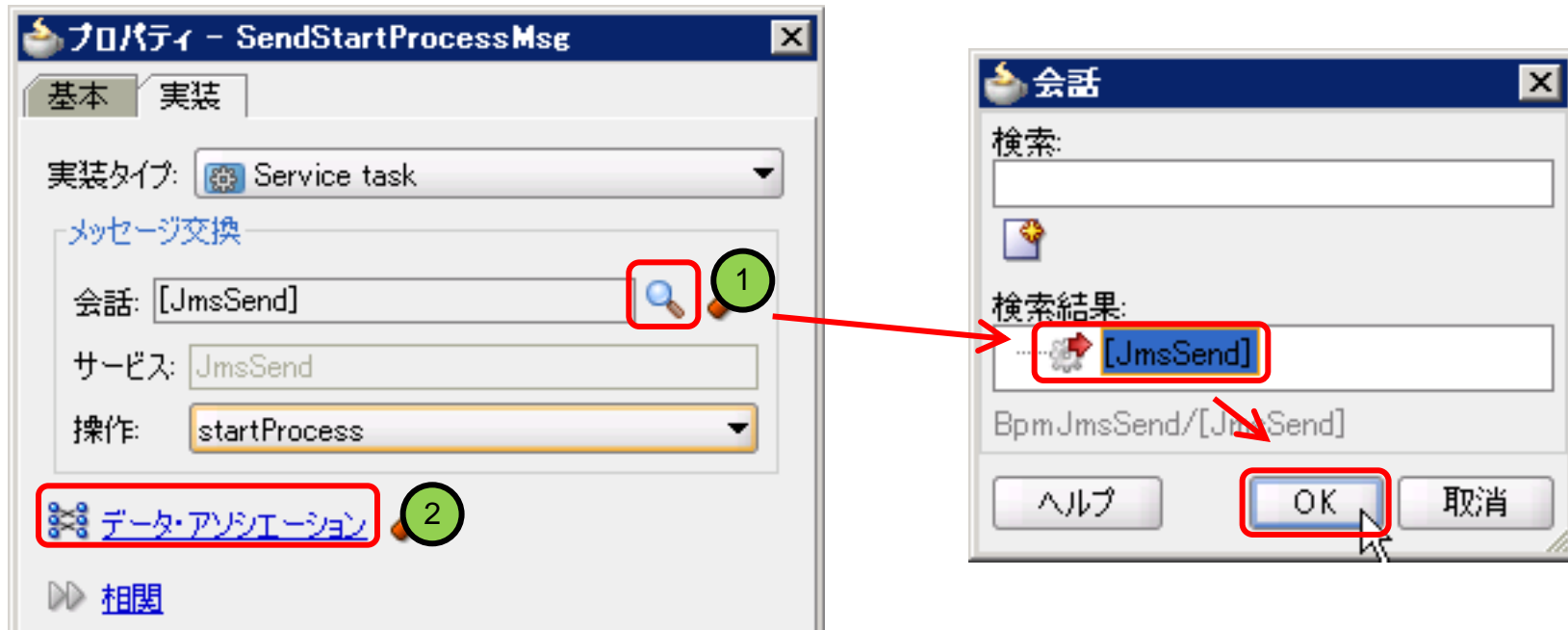
- 「SendStartProcessMsg」サービス・タスクをダブル・クリックし、「プロパティ」を開く



# 各要素の実装 ~ シナリオ1

## [送信プロセス] JMSキューに送信 (15/18)

- 作成した会話「JmsSend」を選択して、「データ・アソシエーション」をクリック

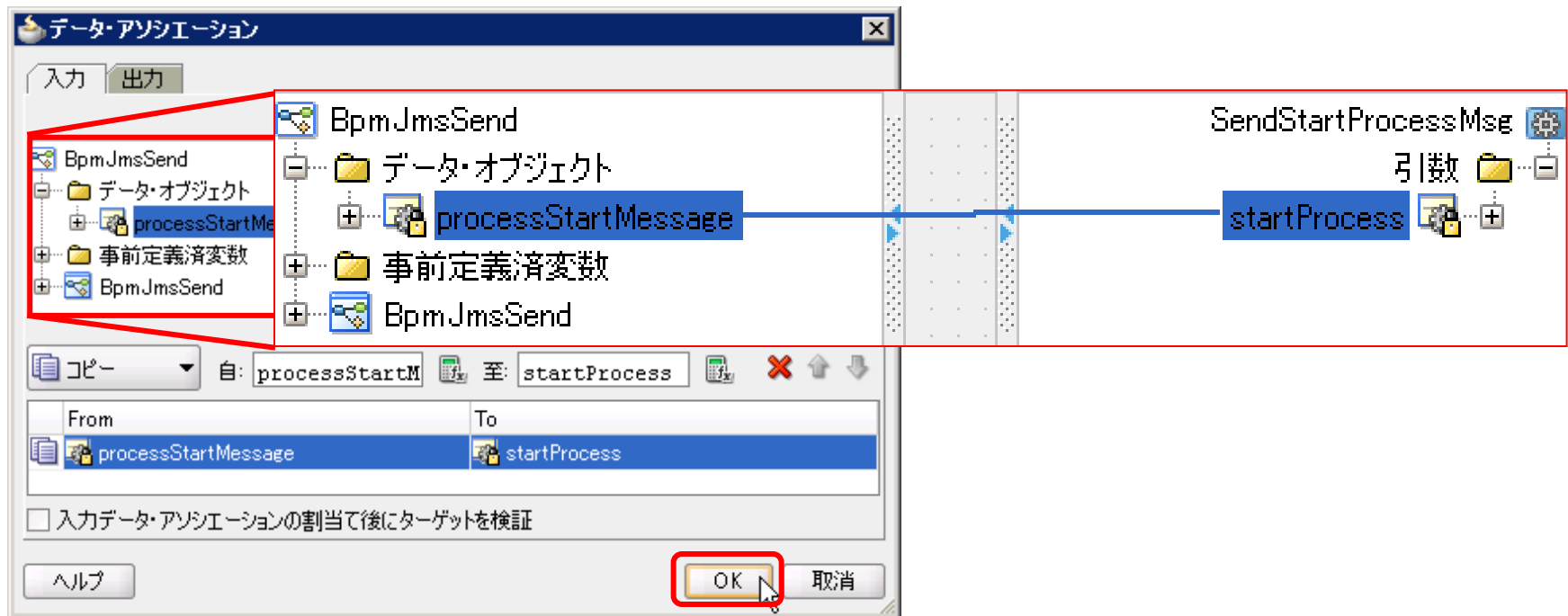




# 各要素の実装 ～ シナリオ1

## [送信プロセス] JMSキューに送信 (16/18)

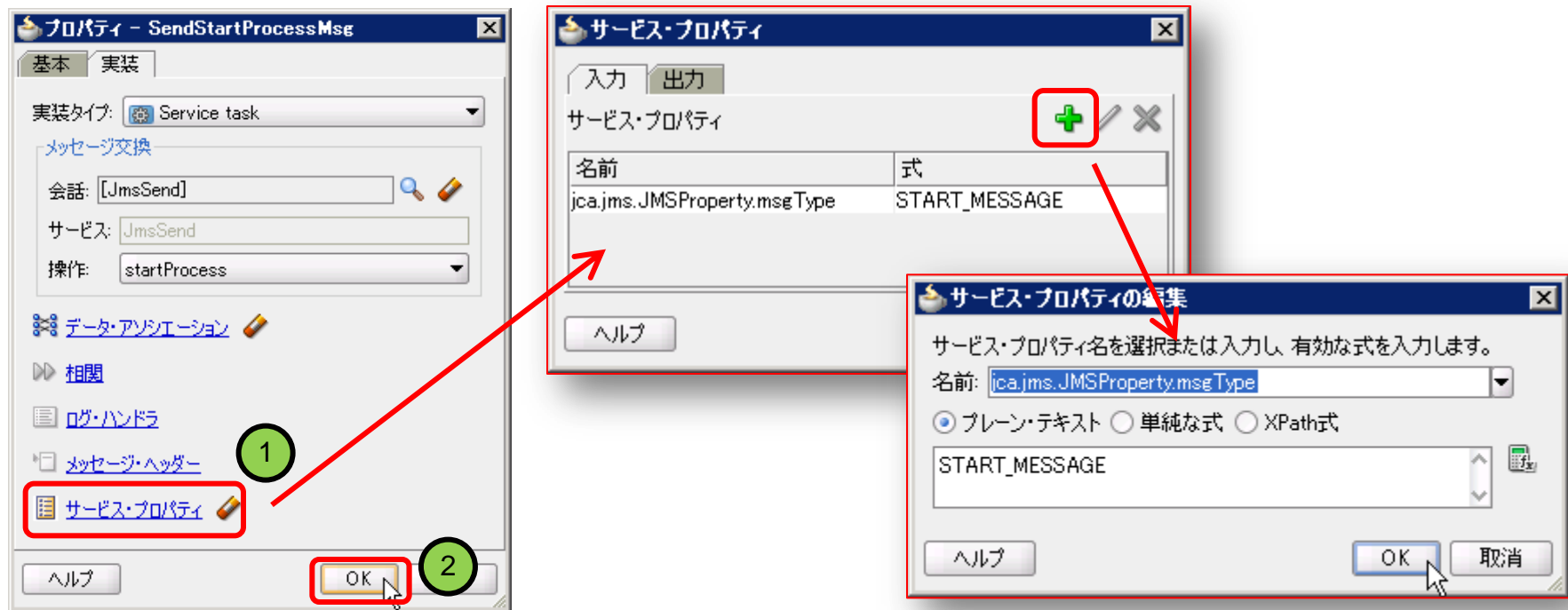
- 「入力」タブで、左のデータ・オブジェクトをドラッグして右側の引数にドロップすることにより、以下のデータ・アソシエーションを追加して、「OK」をクリック



# 各要素の実装 ～ シナリオ1

## [送信プロセス] JMSキューに送信 (17/18)

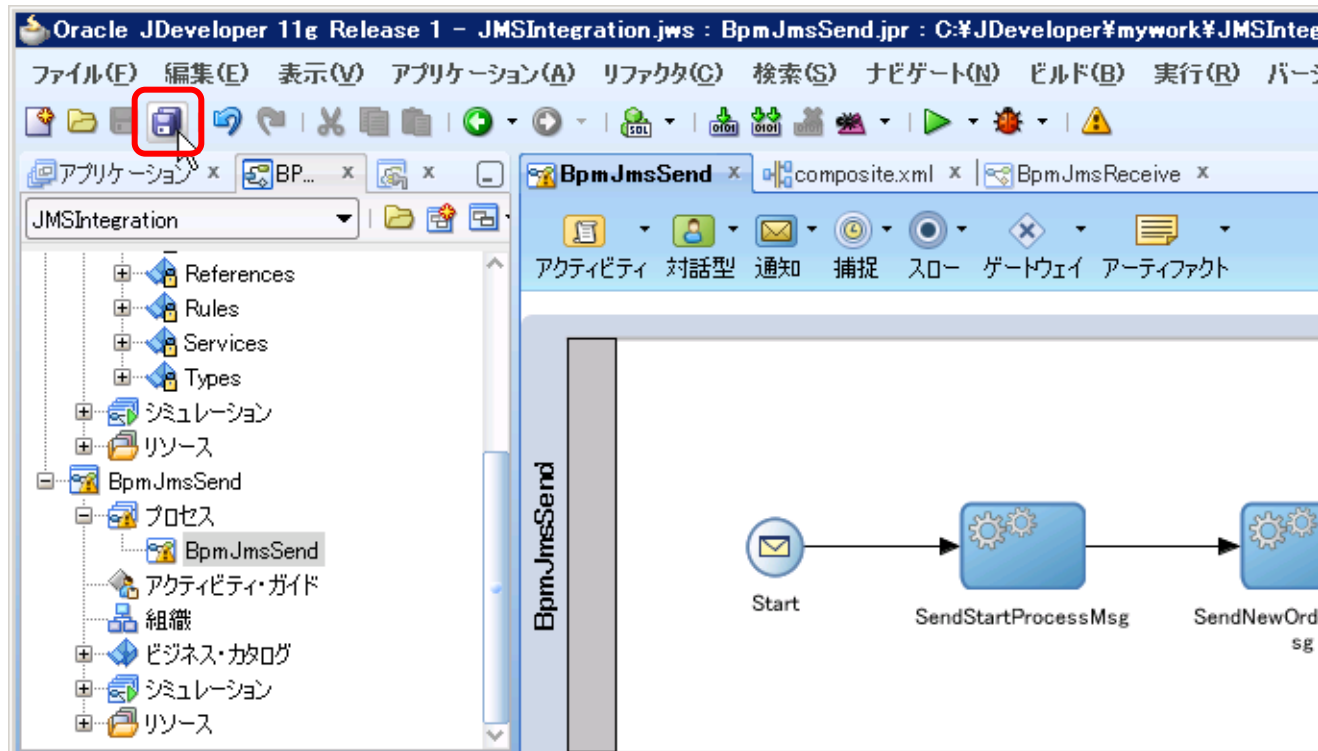
- 「サービス・プロパティ」をクリックして、サービス・プロパティ「jca.jms.JMSProperty.msgType」に文字列「START\_MESSAGE」を設定して、「OK」をクリック



# 各要素の実装 ～ シナリオ1

## [送信プロセス] JMSキューに送信 (18/18)

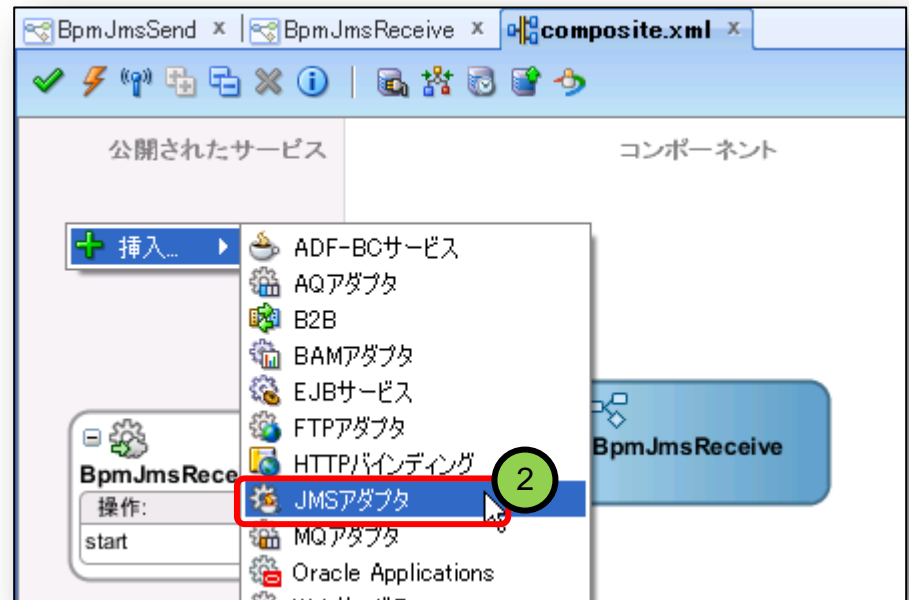
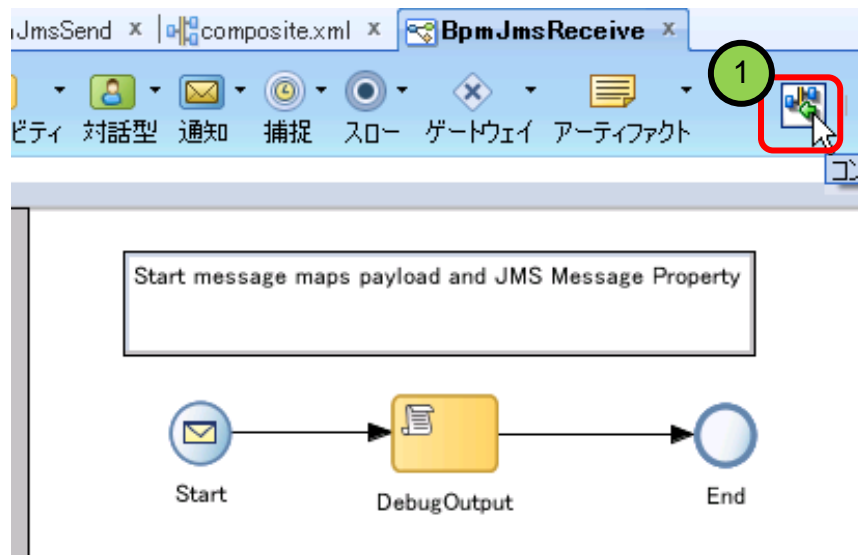
- 「すべて保存」をして、「SendStartProcessMsg」サービス・タスクの実装が完了



# 各要素の実装 ～ シナリオ1

## [受信プロセス] JMSキューから受信 (1/7)

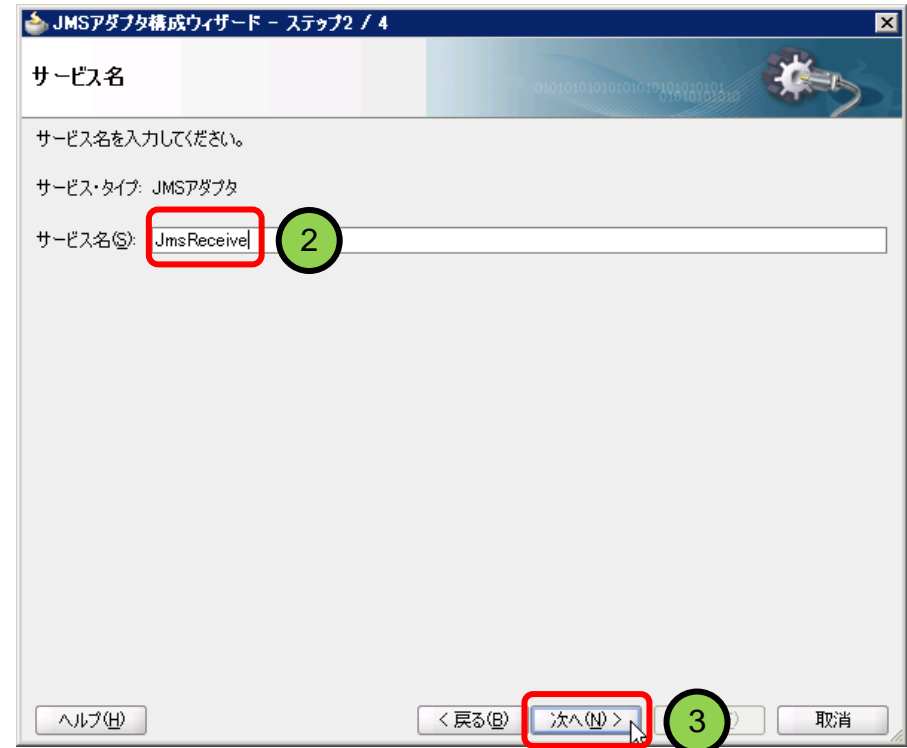
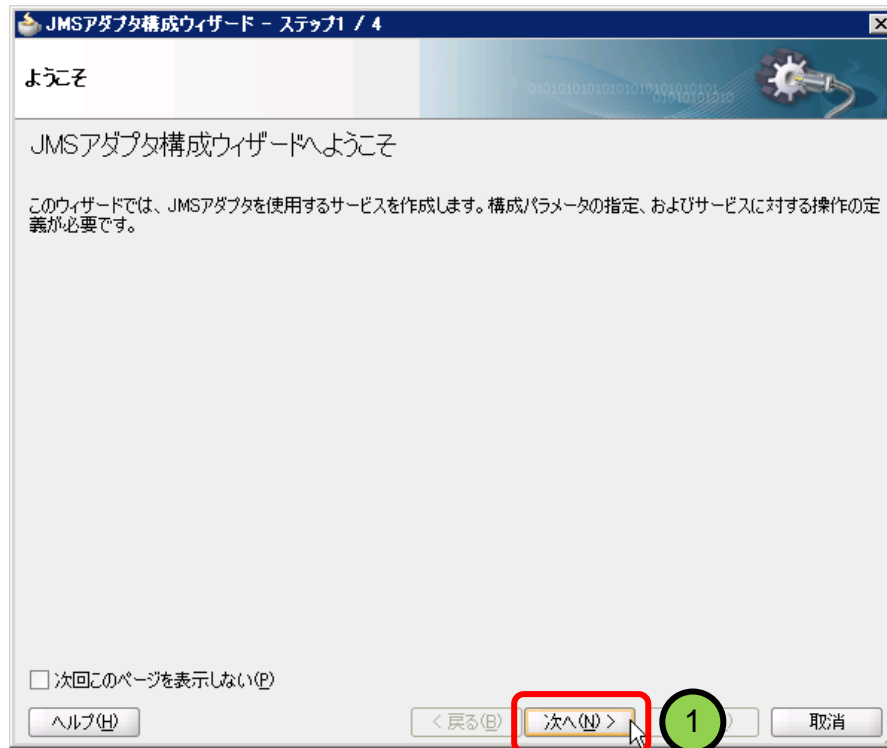
- 「BpmJmsReceive」プロセス・エディタで、「コンポジット・エディタ」アイコンをクリック、コンポジット・エディターへ移動
- コンポジット・エディタの「公開されたサービス」欄で、右クリックして「挿入 > JMSアダプタ」を選択



# 各要素の実装 ~ シナリオ1

## [受信プロセス] JMSキューから受信 (2/7)

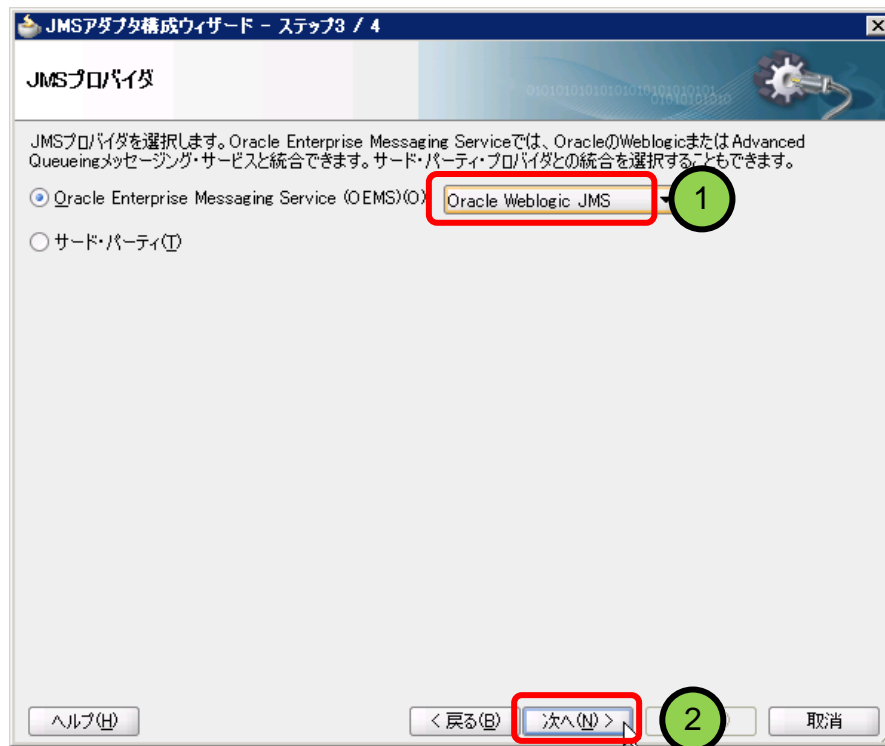
- 「ようこそ」画面で「次へ」
- 「サービス名」に「JmsReceive」を入力して「次へ」



# 各要素の実装 ～ シナリオ1

## [受信プロセス] JMSキューから受信 (3/7)

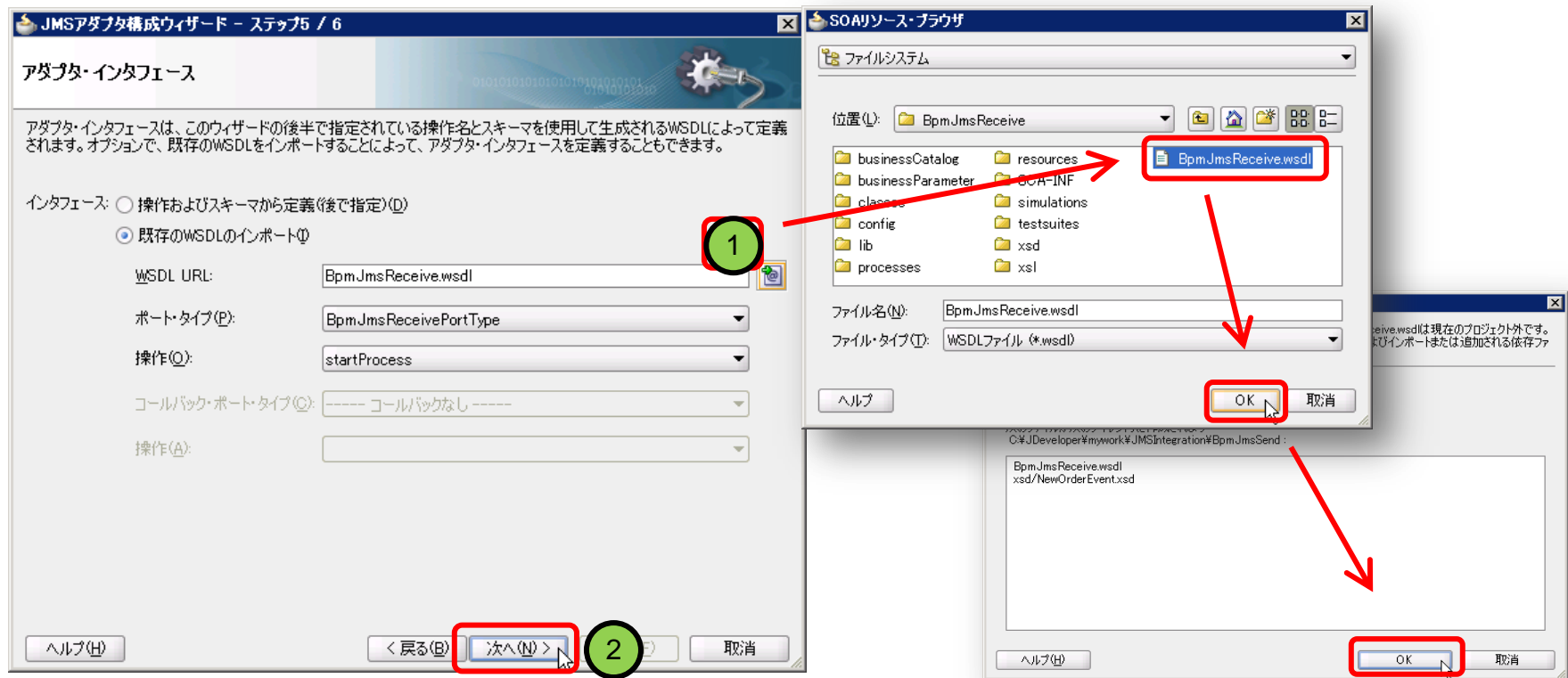
- 「Oracle Weblogic JMS」を選択して「次へ」
- 事前作成した「WLS\_Remote」接続を選択して「次へ」



# 各要素の実装 ～ シナリオ1

## [受信プロセス] JMSキューから受信 (4/7)

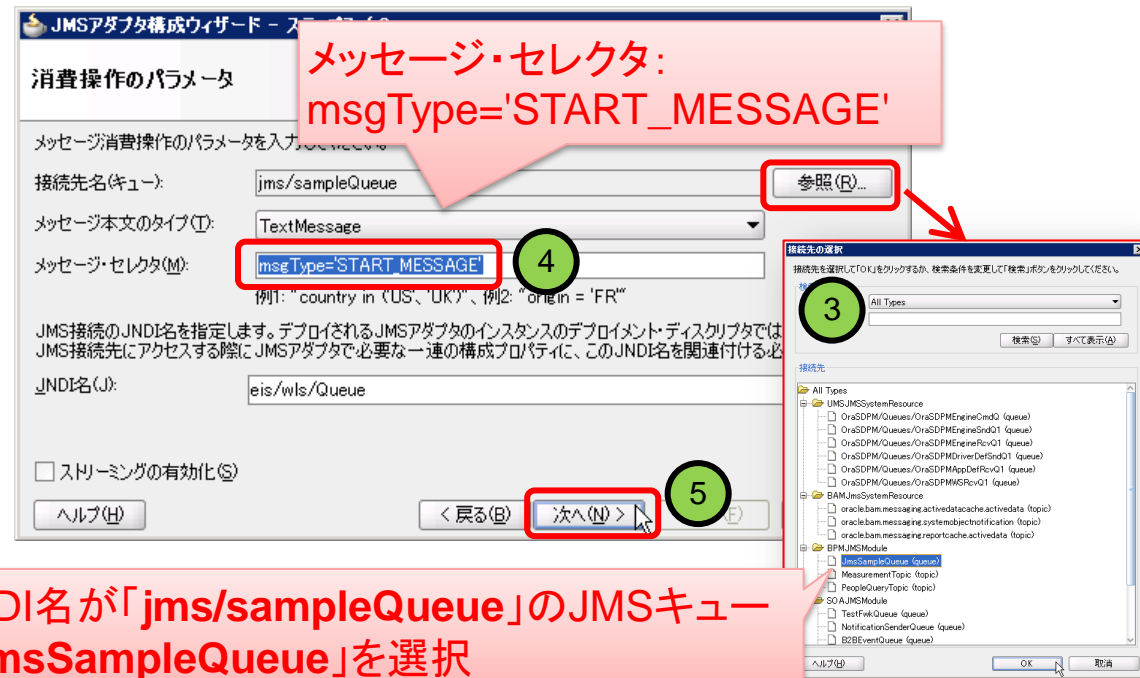
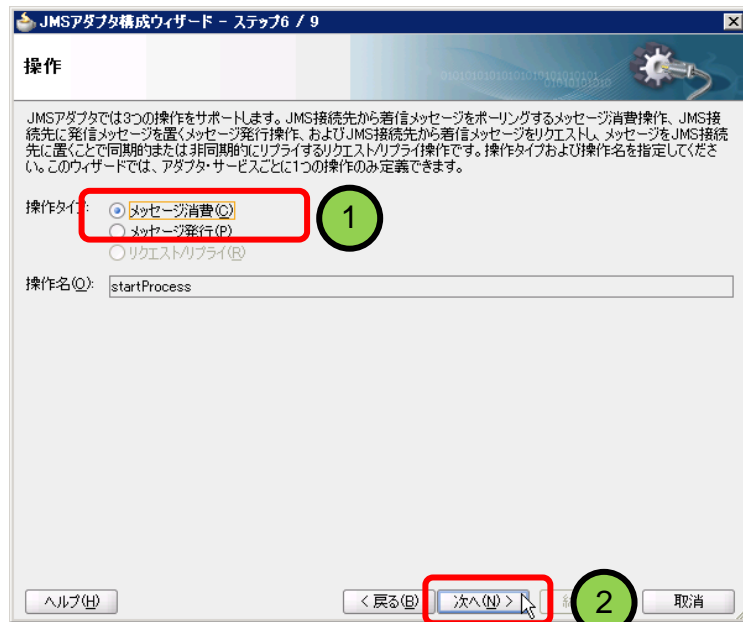
- ターゲット・プロセス「BpmJmsReceive」プロジェクトのディレクトリから、「BpmJmsReceive.wsdl」を選択して「次へ」



# 各要素の実装 ~ シナリオ1

## [受信プロセス] JMSキューから受信 (5/7)

- 「メッセージ消費」を選択して「次へ」
- 環境準備で作成したJMSキューを選択し、メッセージ・セレクタを入力して「次へ」



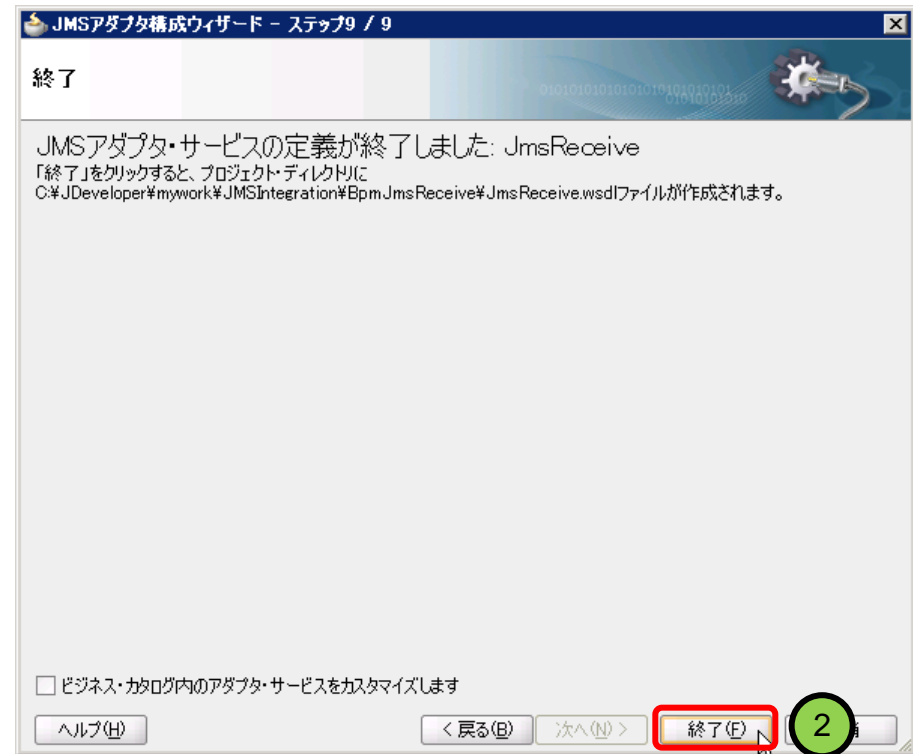
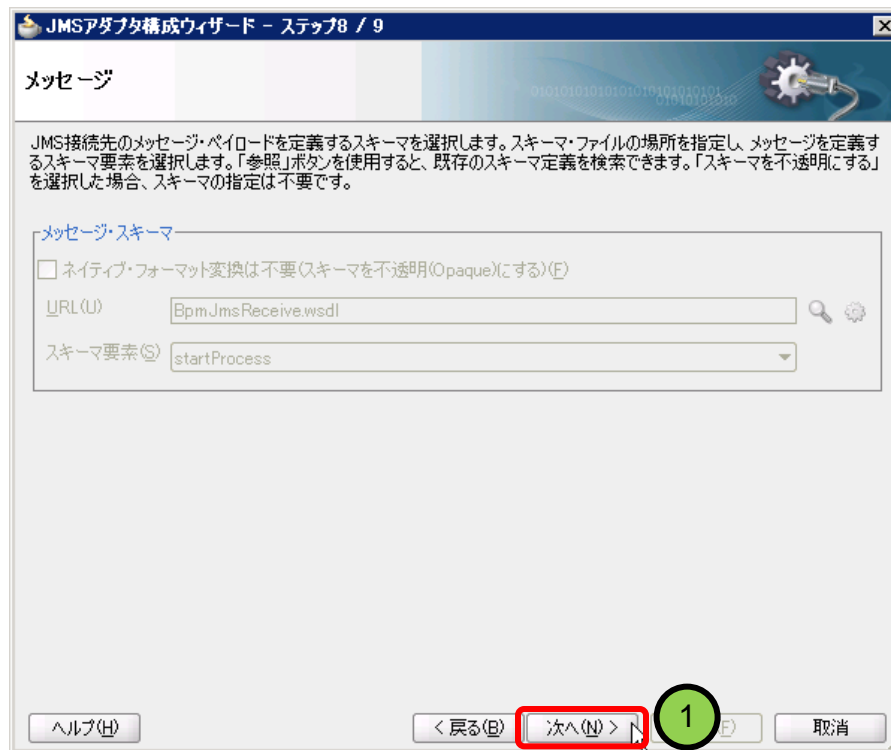


# 各要素の実装 ～ シナリオ1

## [受信プロセス] JMSキューから受信 (6/7)

- 「次へ」をクリック

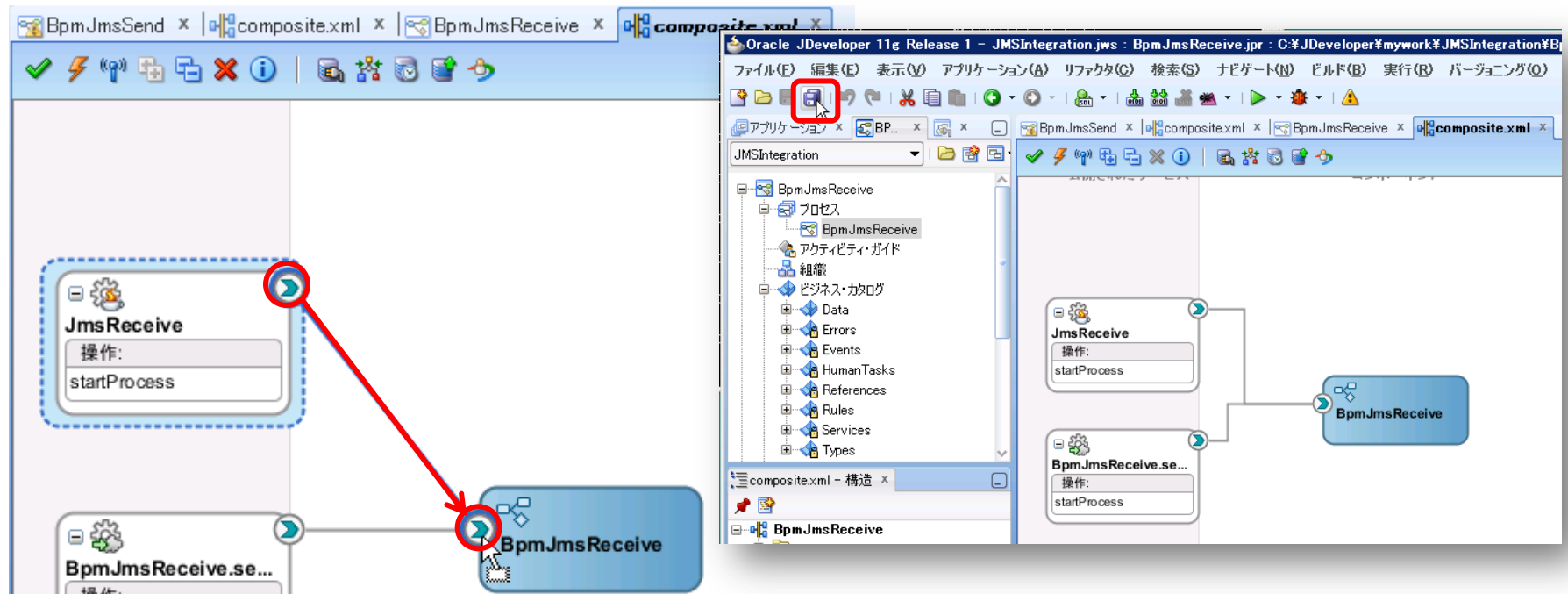
- 「終了」をクリックし、「すべて保存」をする



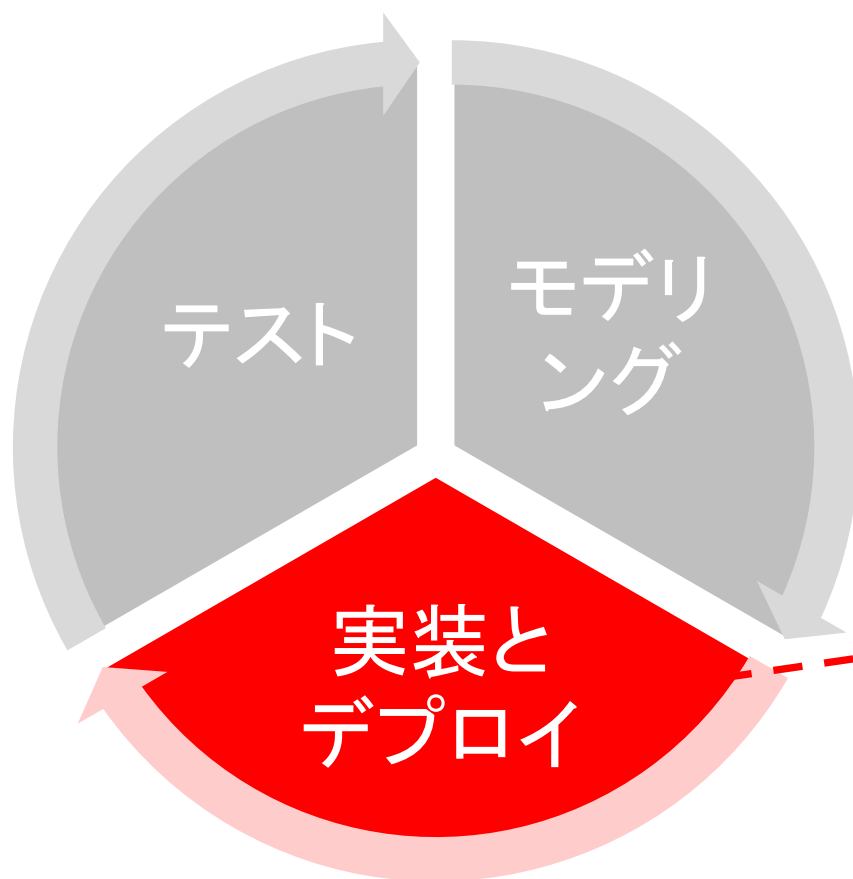
# 各要素の実装 ～ シナリオ1

## [受信プロセス] JMSキューから受信 (7/7)

- ドラッグ・アンド・ドロップで「JmsReceive」サービスから「BpmJmsReceive」プロセス・サービスまでのワイヤーを追加
- 「すべて保存」をして、「シナリオ1」の実装が完了



# JMSIntegrationアプリケーションの作成



データ・オブジェクトの作成

各要素の実装  
～ シナリオ1

各要素の実装  
～ シナリオ2

プロセスのデプロイ

# 各要素の実装 ～ シナリオ2

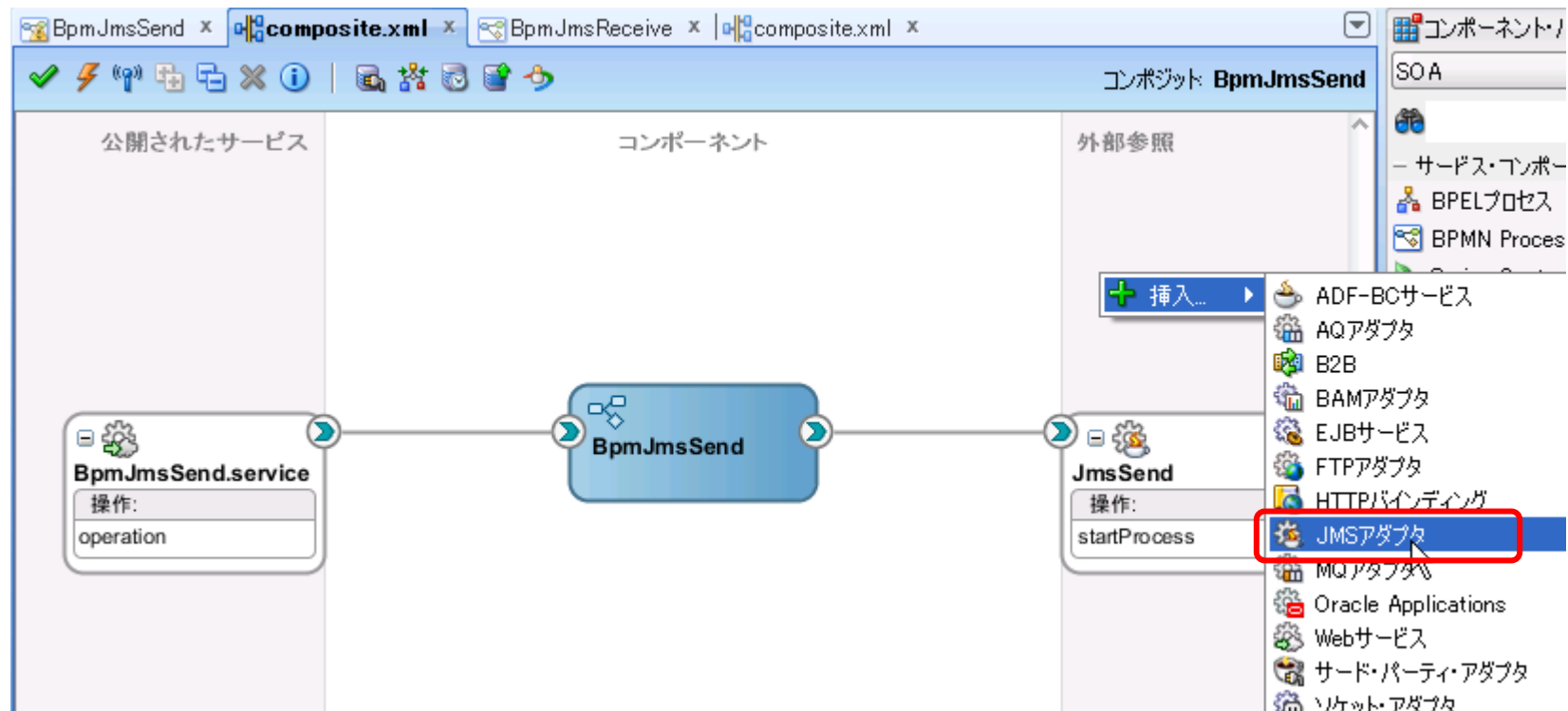
## 実装の手順

手順	概要	詳細	キーワード
1	[送信プロセス] JMSキューに送信	JMS送信プロセス「BpmJmsSend」の「SendNewOrderEventMsg」サービス・タスクを実装し、ターゲット・サービスのビジネス・イベント・スキーマを使用して、JMSキューにターゲット・サービスの入力データを含めるメッセージを送信するJMSアダプタを作成	JMSアダプタ メッセージ発行
2	[ターゲット・プロセス] JMSキューから受信	ターゲット・プロセスで、上記のJMSキューからメッセージを受信するJMSアダプタを作成し、受信したJMSメッセージからターゲット・プロセスの開始メッセージに変換するメディエータを作成して、プロセスを呼び出すように実装	JMSアダプタ メッセージ消費 メディエータ

# 各要素の実装 ～ シナリオ2

## [送信プロセス] JMSキューに送信 (1/13)

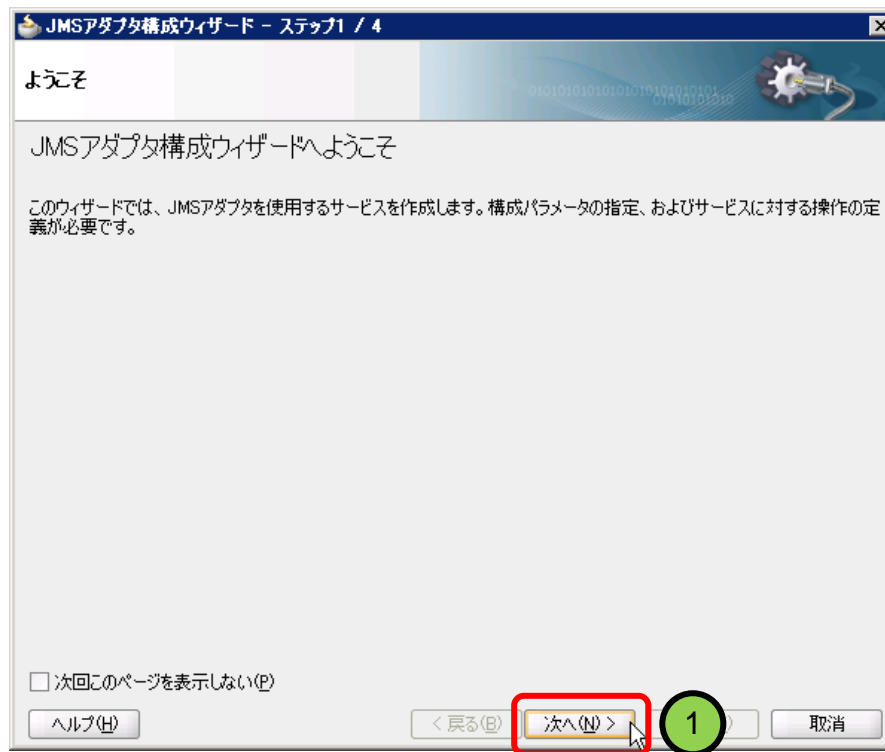
- 「BpmJmsSend」プロセスのコンポジット・エディタの「外部参照」欄で、右クリックして「挿入 > JMSアダプタ」を選択



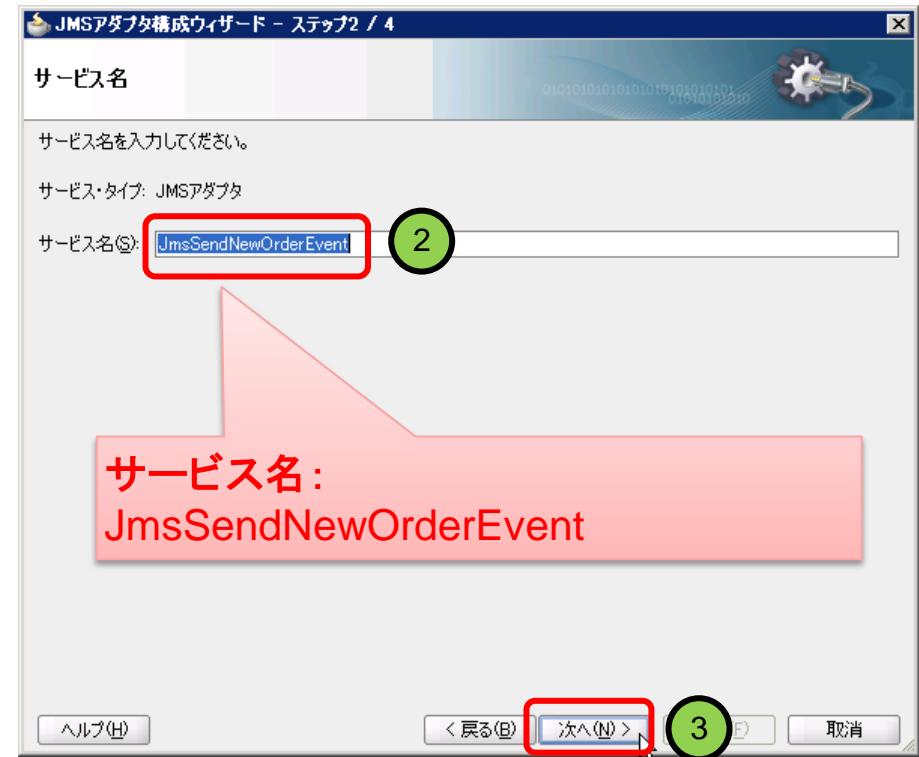
# 各要素の実装 ~ シナリオ2

## [送信プロセス] JMSキューに送信 (2/13)

- 「ようこそ」画面で「次へ」



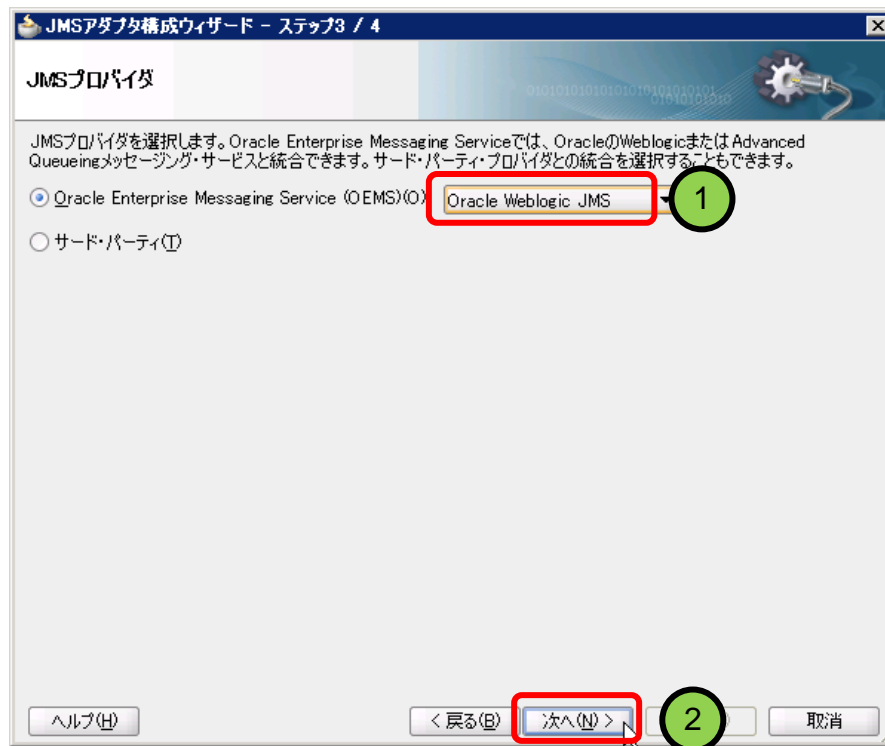
- 「サービス名」を入力して「次へ」



# 各要素の実装 ～ シナリオ2

## [送信プロセス] JMSキューに送信 (3/13)

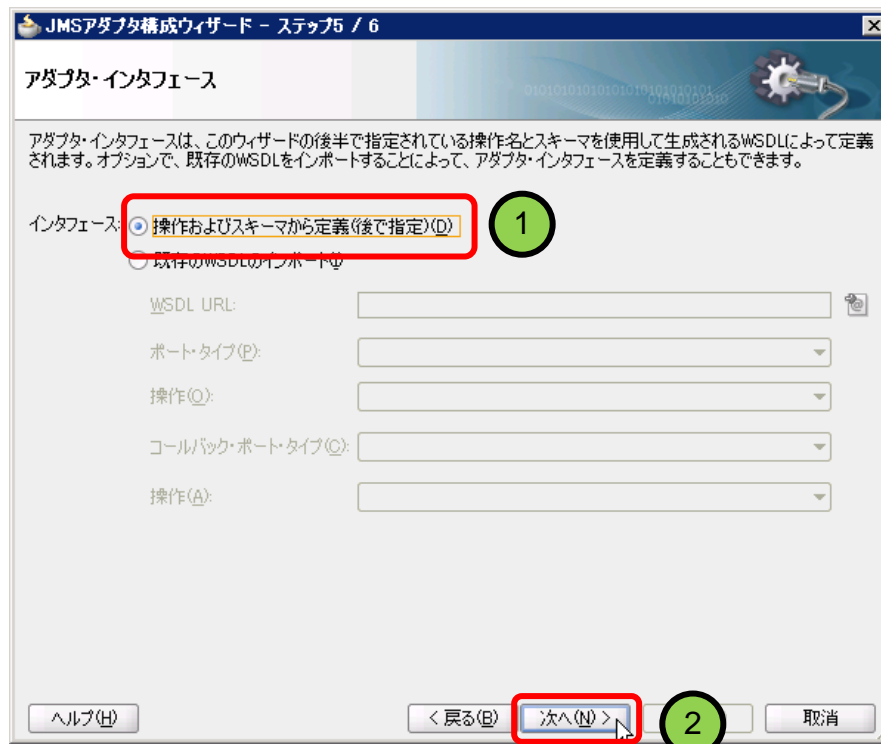
- 「Oracle Weblogic JMS」を選択して「次へ」
- 事前作成した「WLS\_Remote」接続を選択して「次へ」



# 各要素の実装 ～ シナリオ2

## [送信プロセス] JMSキューに送信 (4/13)

- 「操作およびスキーマから定義」を選択して「次へ」
- 「メッセージ発行」を選択して、「操作名」を設定して「次へ」





# 各要素の実装 ~ シナリオ2

## [送信プロセス] JMSキューに送信 (5/13)

- 環境準備で作成したJMSキューを選択して「次へ」

JMSアダプタ構成ウィザード - ステップ7 / 9

発行操作のパラメータ

メッセージ発行操作のパラメータを入力してください。

接続先名(キュー):  参照(R)...

メッセージ本文のタイプ(M):

配信モード(L):

優先度(P):

TimeToLive(T):  秒

順序単位(U):

JMS接続のJNDI名を指定します。デプロイされるJMSアダプタのインスタンスのデプロイメント・ディスクリプタでは、実行時にJMS接続先にアクセスする際にJMSアダプタに必要な一連の構成プロパティに、このJNDI名を関連付ける必要があります。

JNDI名(J):

ヘルプ(H) < 戻る(B) 次へ(N) > 取消

JNDI名が「jms/sampleQueue」のJMSキュー  
「JmsSampleQueue」を選択

接続先の選択

接続先を選択して「OK」をクリックするか、検索条件を変更して「検索」ボタンをクリックしてください。

検索(S)

接続先タイプ(T):

接続先名(D):

検索(S) すべて表示(A)

接続先

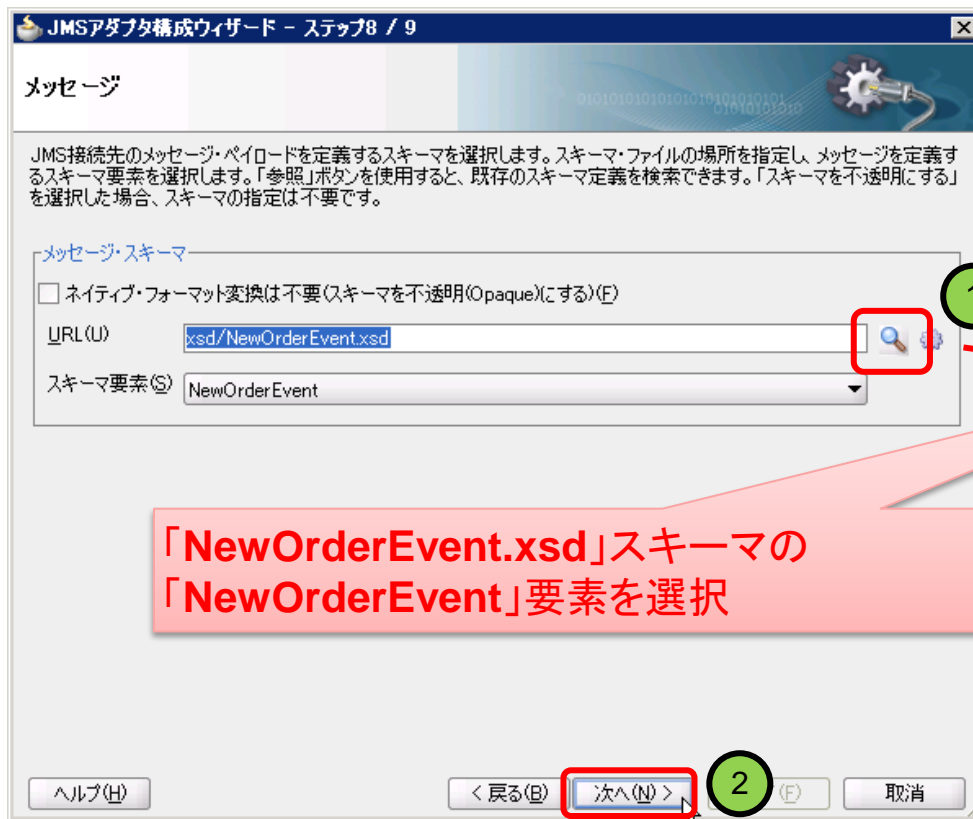
- All Types
  - UMS.JMSSystemResource
    - OraSDPM/Queues/OraSDPMEngineCmdQ (queue)
    - OraSDPM/Queues/OraSDPMEngineSndQ1 (queue)
    - OraSDPM/Queues/OraSDPMEngineRcvQ1 (queue)
    - OraSDPM/Queues/OraSDPMDriverDefSndQ1 (queue)
    - OraSDPM/Queues/OraSDPMDriverDefRcvQ1 (queue)
    - OraSDPM/Queues/OraSDPMAppDefRcvQ1 (queue)
    - OraSDPM/Queues/OraSDPMWSRcvQ1 (queue)
  - BAM.JMSSystemResource
    - oracle.bam.messaging.activedatacache.activedata (topic)
    - oracle.bam.messaging.systemobjectnotification (topic)
    - oracle.bam.messaging.reportcache.activedata (topic)
  - BPM.JMSModule
    - JmsSampleQueue (queue)**
    - MeasurementTopic (topic)
    - PeopleQueryTopic (topic)
  - SOA.JMSModule
    - TestFwkQueue (queue)
    - NotificationSenderQueue (queue)
    - B2BEventQueue (queue)

ヘルプ(H) OK 取消

# 各要素の実装 ~ シナリオ2

## [送信プロセス] JMSキューに送信 (6/13)

- メッセージ・スキーマを選択して「次へ」

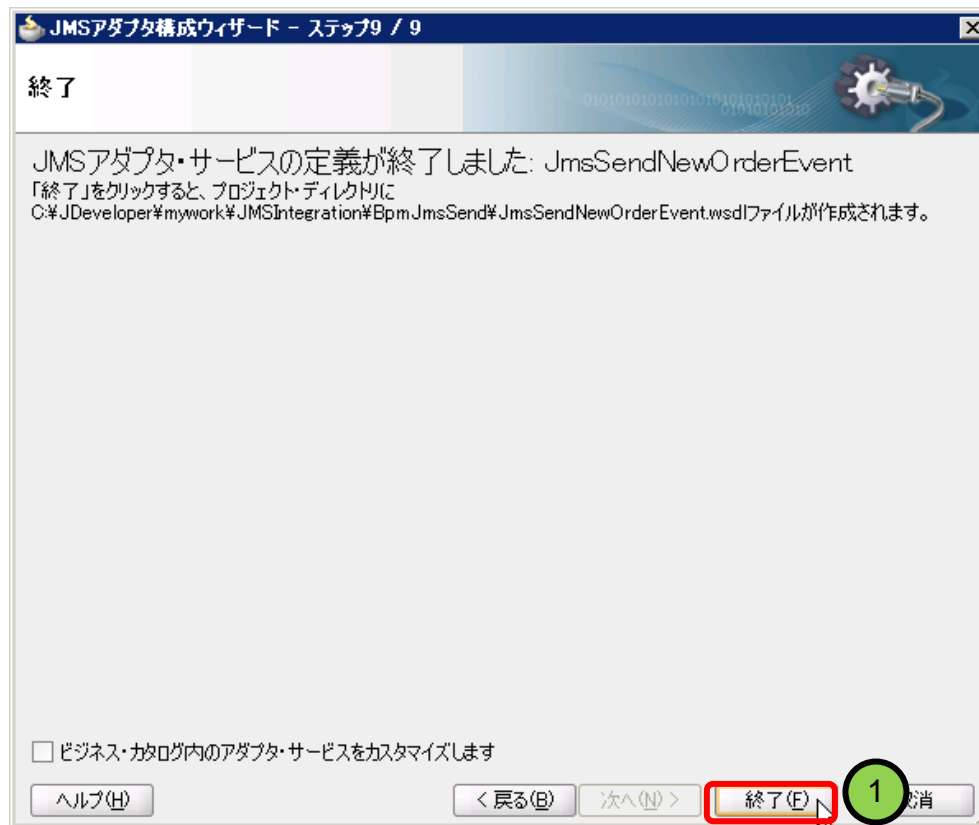


「NewOrderEvent.xsd」スキーマの  
「NewOrderEvent」要素を選択

# 各要素の実装 ～ シナリオ2

## [送信プロセス] JMSキューに送信 (7/13)

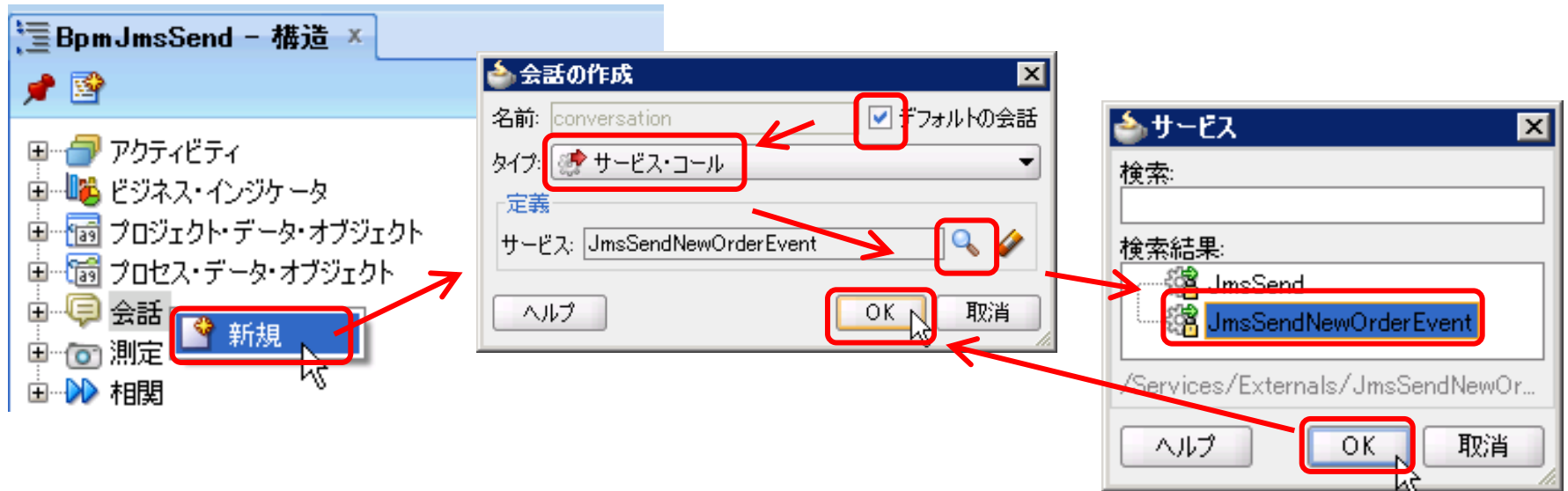
- 「終了」をクリックし、「すべて保存」をする



# 各要素の実装 ～ シナリオ2

## [送信プロセス] JMSキューに送信 (8/13)

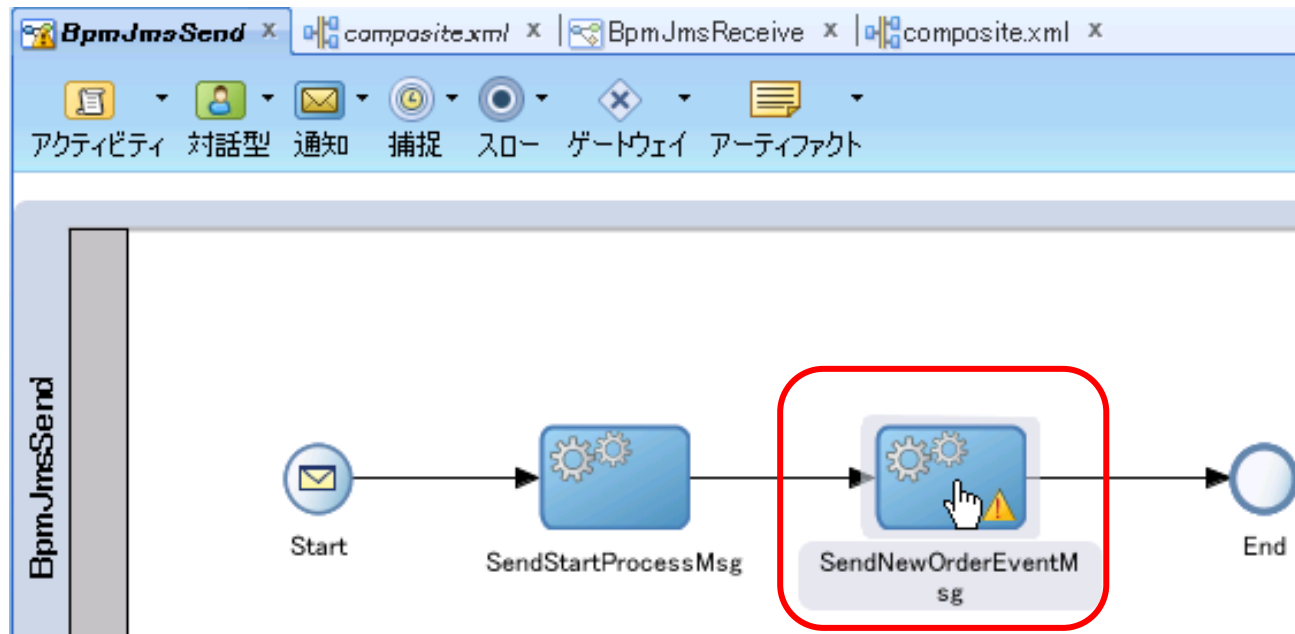
- 「BPMプロジェクト・ナビゲータ」で「BpmJmsSend」プロセスを選択し、左下に表示される構造情報の「会話」を右クリックし、「新規」をクリックしてJMSアダプタ・サービス「JmsSendNewOrderEvent」サービス・コールの会話を作成



# 各要素の実装 ～ シナリオ2

## [送信プロセス] JMSキューに送信 (9/13)

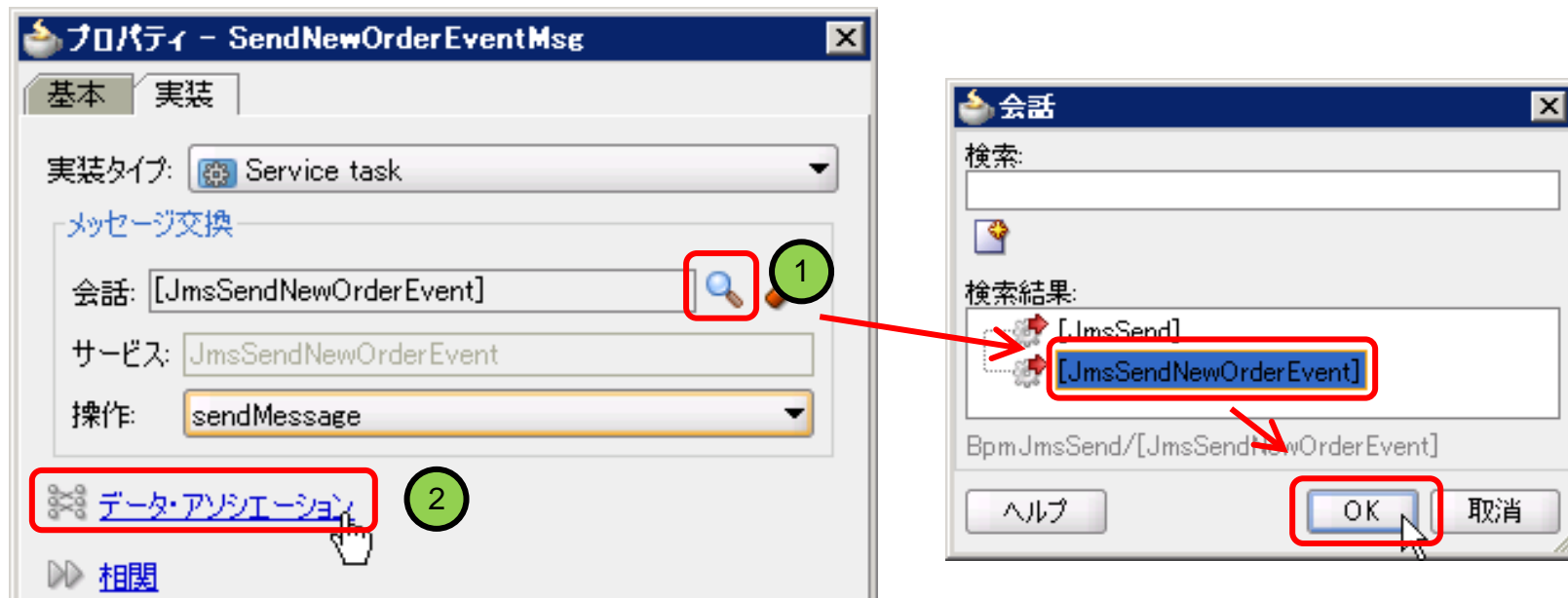
- 「SendNewOrderEventMsg」サービス・タスクをダブル・クリックし、「プロパティ」を開く



# 各要素の実装 ～ シナリオ2

## [送信プロセス] JMSキューに送信 (10/13)

- 作成した会話「JmsSendNewOrderEvent」を選択して、「データ・アソシエーション」をクリック



# 各要素の実装 ～ シナリオ2

## [送信プロセス] JMSキューに送信 (11/13)

- 「入力」タブで、左のデータ・オブジェクトをドラッグして右側の引数にドロップすることにより、以下のデータ・アソシエーションを追加して、「OK」をクリック



# 各要素の実装 ～ シナリオ2

## [送信プロセス] JMSキューに送信 (12/13)

- 「サービス・プロパティ」をクリックして、サービス・プロパティ「jca.jms.JMSProperty.msgType」に文字列「ORDER\_EVENT」を設定して、「OK」をクリック

The screenshot shows the configuration of a Service Task in Oracle BPM Studio. The 'サービス・プロパティ' (Service Properties) dialog is open, showing the property 'jca.jms.JMSProperty.msgType' with the value 'ORDER\_EVENT'. A 'サービス・プロパティの作成' (Create Service Property) dialog is also open, showing the same property name and value. Red boxes and arrows highlight the 'サービス・プロパティ' menu item, the 'サービス・プロパティ' dialog, and the 'サービス・プロパティの作成' dialog.

1. Click on 'サービス・プロパティ' in the left sidebar.

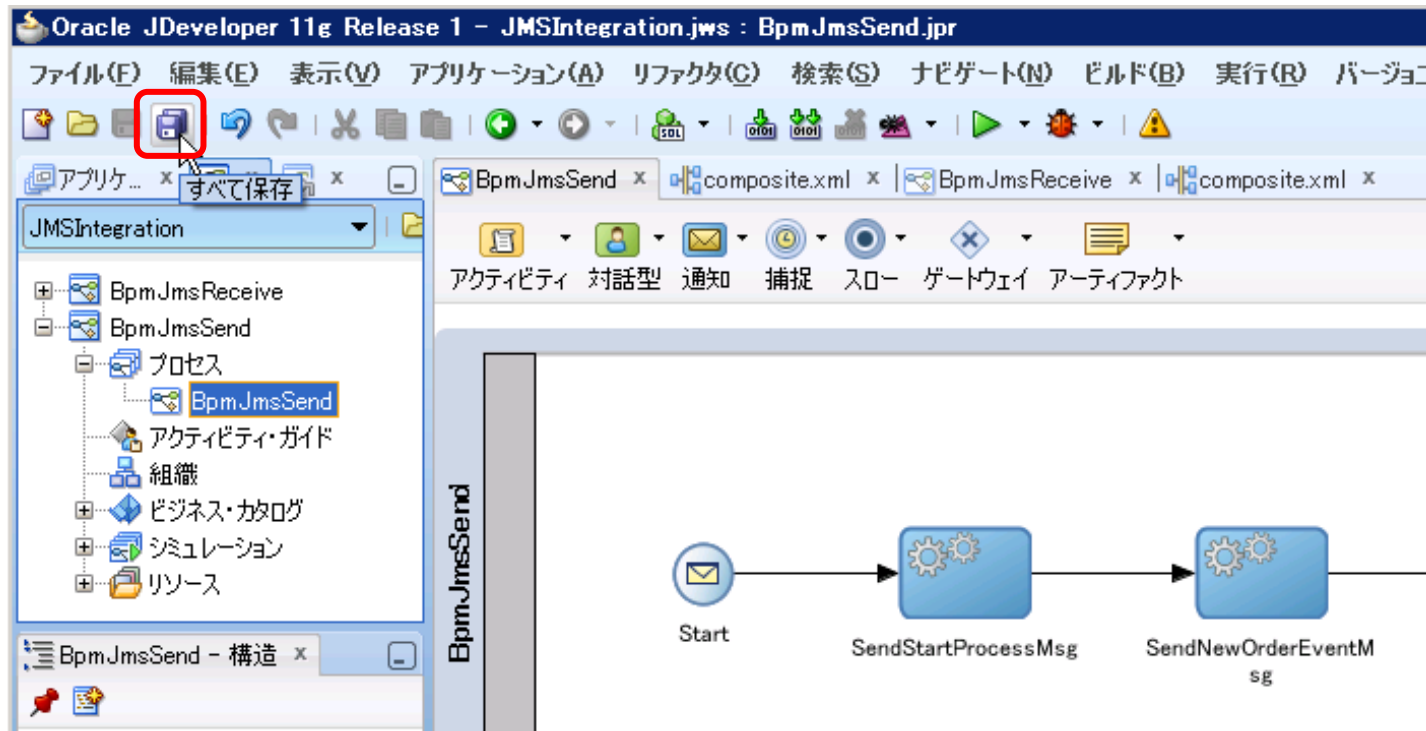
2. Click on 'OK' in the 'サービス・プロパティ' dialog.



# 各要素の実装 ～ シナリオ2

## [送信プロセス] JMSキューに送信 (13/13)

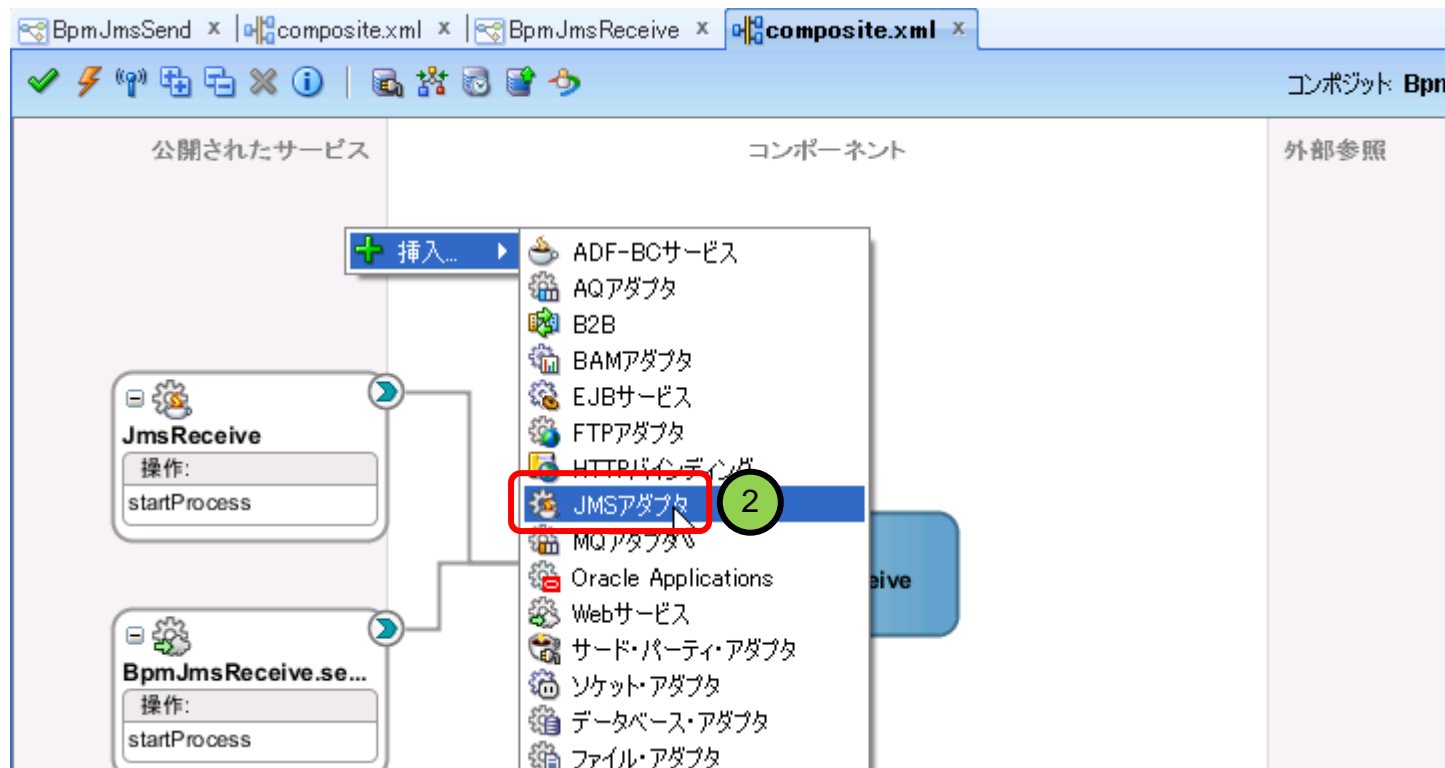
- 「すべて保存」をして、「SendNewOrderEventMsg」サービスタスクの実装が完了



# 各要素の実装 ～ シナリオ2

## [受信プロセス] JMSキューから受信 (1/17)

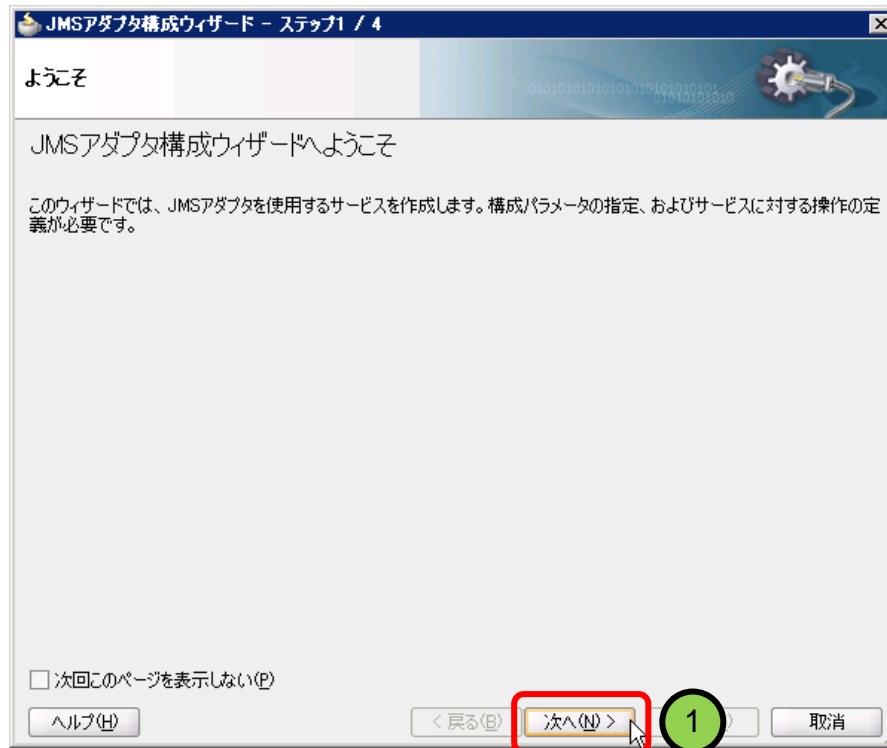
- 「BpmJmsReceive」プロセスのコンポジット・エディタの「公開されたサービス」欄で、右クリックして「挿入 > JMSアダプタ」を選択



# 各要素の実装 ~ シナリオ2

## [受信プロセス] JMSキューから受信 (2/17)

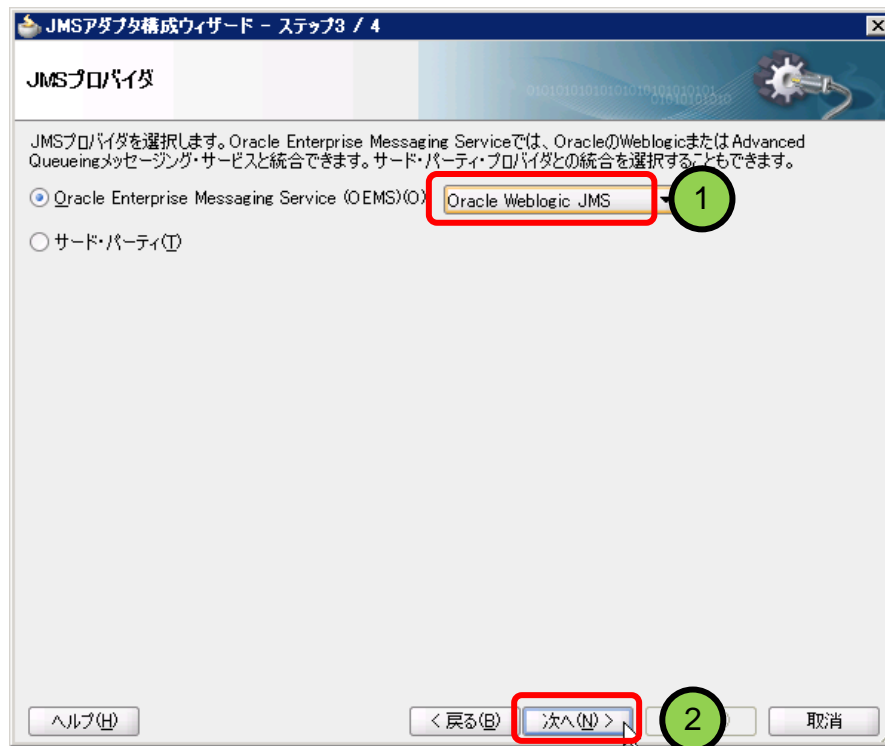
- 「ようこそ」画面で「次へ」
- 以下の「サービス名」を入力して「次へ」



# 各要素の実装 ～ シナリオ2

## [受信プロセス] JMSキューから受信 (3/17)

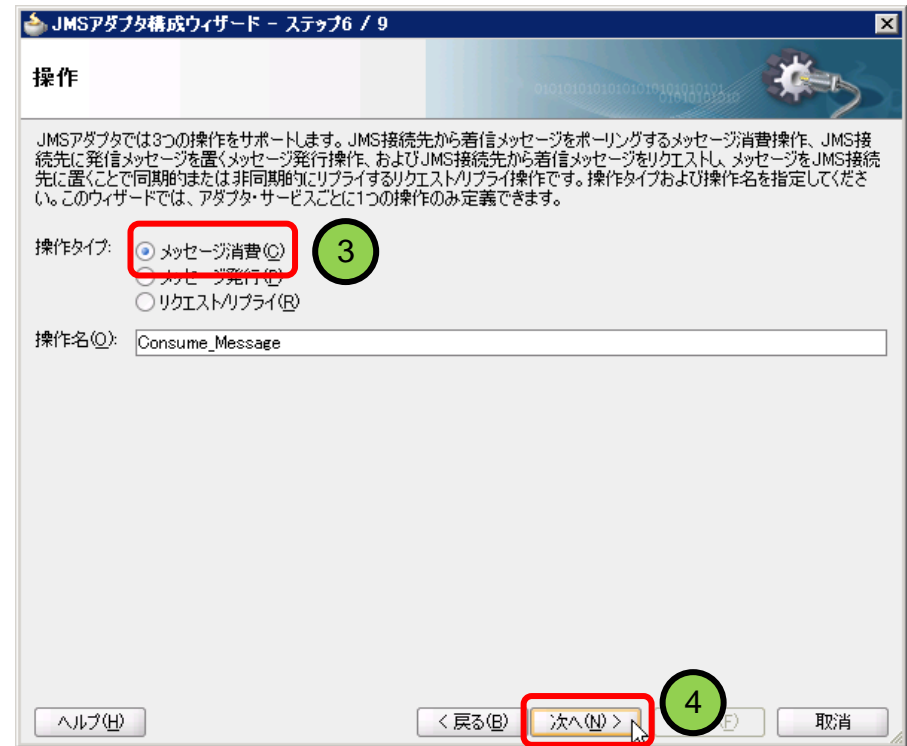
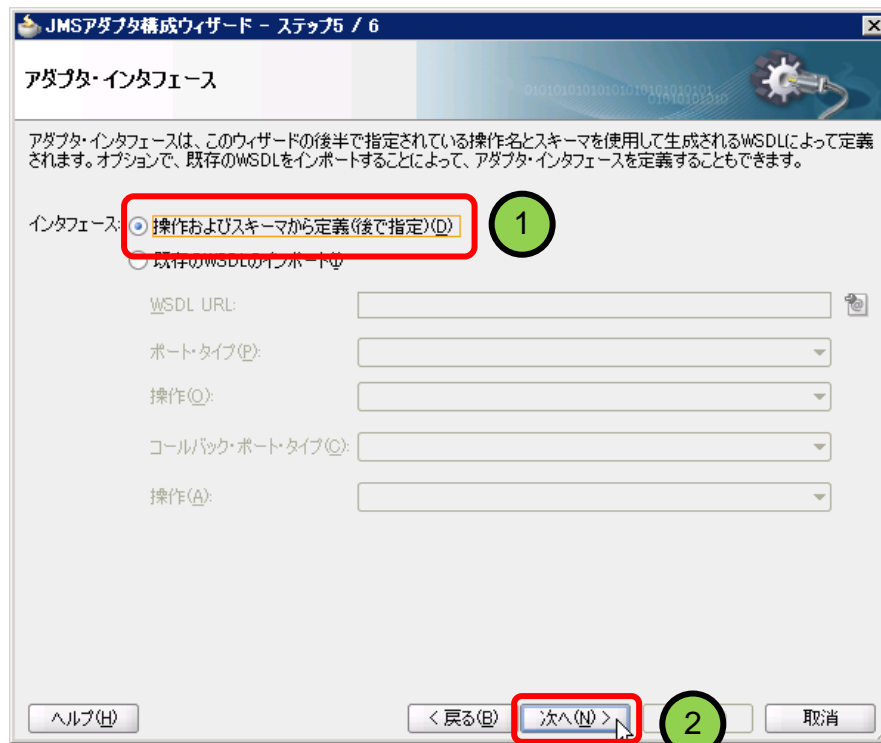
- 「Oracle Weblogic JMS」を選択して「次へ」
- 事前作成した「WLS\_Remote」接続を選択して「次へ」



# 各要素の実装 ～ シナリオ2

## [受信プロセス] JMSキューから受信 (4/17)

- 「操作およびスキーマから定義」を選択して「次へ」
- 「メッセージ消費」を選択して「次へ」



# 各要素の実装 ~ シナリオ2

## [受信プロセス] JMSキューから受信 (5/17)

- 環境準備で作成したJMSキューを選択し、「メッセージ・セレクタ」を入力して「次へ」

消費操作のパラメータ

メッセージ消費操作のパラメータを入力してください。

接続先名(キュー):  参照(R)... ①

メッセージ本文のタイプ(T):

メッセージ・セレクタ(M):  ②

例1: "country in ('US', 'UK')", 例2: "origin = 'FR'"

JMS接続のJNDI名を指定します。デプロイされるJMSアダプタのインスタンスのデプロイメント・ディスクリプタでは、実行JMS接続先にアクセスする際にJMSアダプタで必須の構成プロパティに、このJNDI名を関連付ける必要があります。

JNDI名(J):

ストリーミングの有効化(S)

ヘルプ(H) < 戻る(B) 次へ(N) > ③ 取消

接続先の選択

接続先を選択して「OK」をクリックするか、検索条件を変更して「検索」ボタンをクリックしてください。

検索(S)

接続先タイプ(T):

接続先名(Q):

検索(S) すべて表示(A)

接続先

All Types

OraSDPM/Queues/OraSD...  
OraSDPM/Queues/OraSD...  
BAMJmsSystemResource  
oracle.bam.messaging.act...  
oracle.bam.messaging.sy...  
oracle.bam.messaging.re...  
BPM.JMSModule  
JmsSampleQueue (queue)  
MeasurementTopic (topic)  
PeopleQueryTopic (topic)  
SOA.JMSModule  
TestFwkQueue (queue)  
NotificationSenderQueue (queue)  
B2BEventQueue (queue)

ヘルプ(H) OK 取消

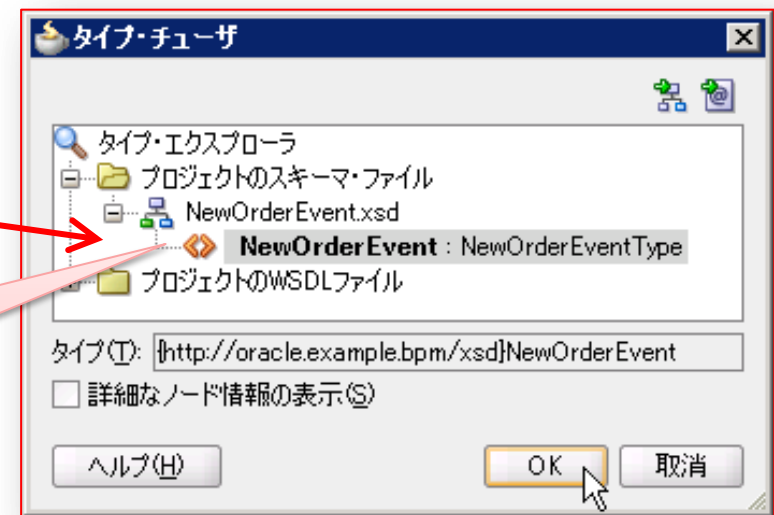
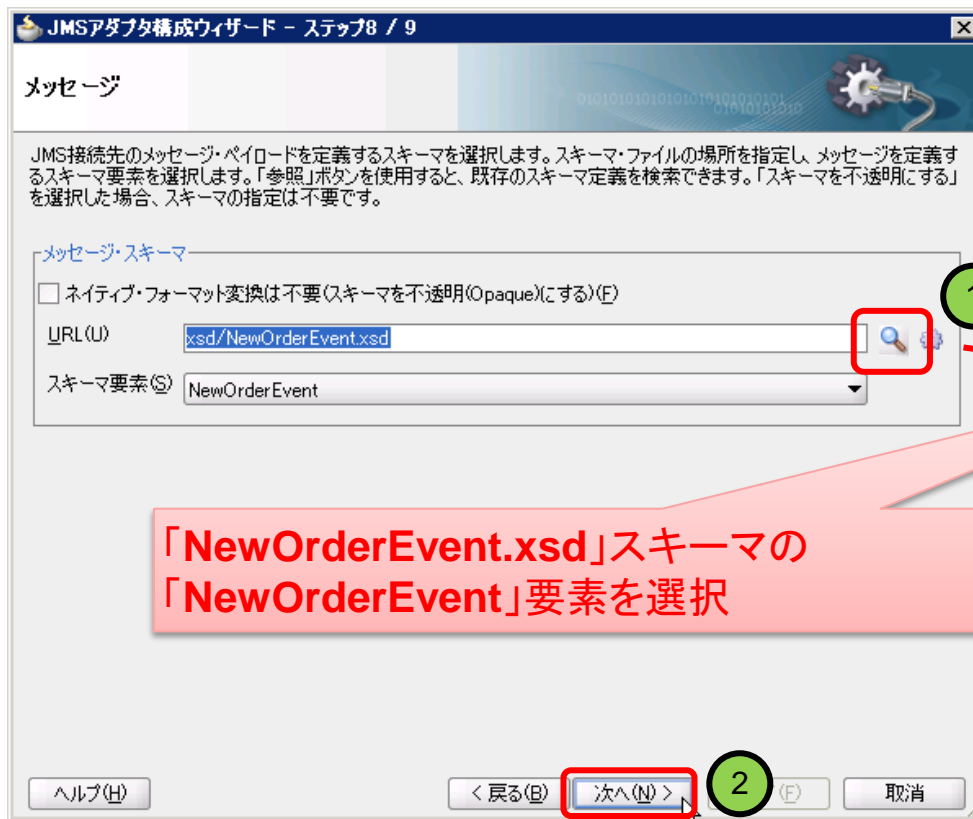
メッセージ・セレクタ:  
msgType='ORDER\_EVENT'

JNDI名が「jms/sampleQueue」のJMSキュー  
「JmsSampleQueue」を選択

# 各要素の実装 ~ シナリオ2

## [受信プロセス] JMSキューから受信 (6/17)

- メッセージ・スキーマを選択して「次へ」

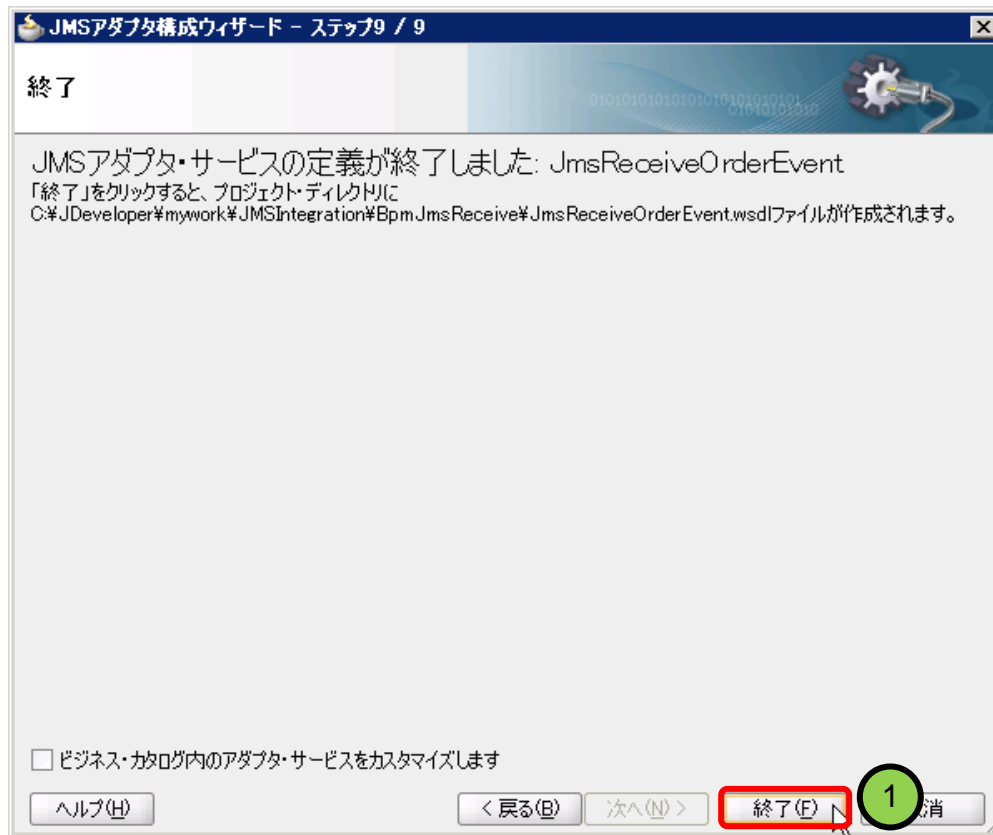


「NewOrderEvent.xsd」スキーマの  
「NewOrderEvent」要素を選択

# 各要素の実装 ～ シナリオ2

## [受信プロセス] JMSキューから受信 (7/17)

- 「終了」をクリックし、「すべて保存」をする





# 各要素の実装 ～ シナリオ2

## [受信プロセス] JMSキューから受信 (8/17)

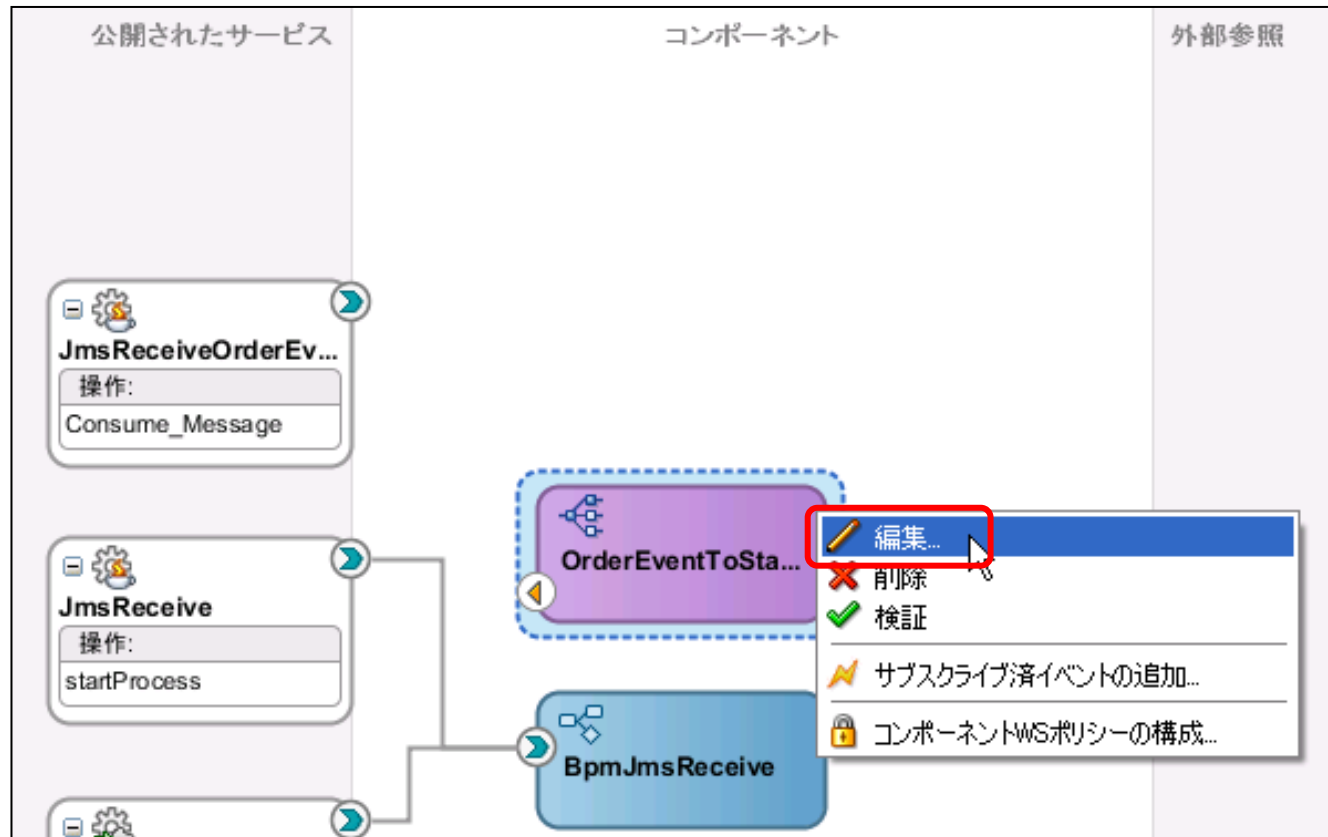
- 「BpmJmsReceive」プロセスのコンポジット・エディタの「コンポーネント」欄で、右クリックして「挿入 > メディエータ」を選択

The screenshot displays the Oracle BPM Suite Composite Editor interface. The main workspace shows a process diagram with two tasks: 'JmsReceiveOrderEv...' and 'JmsReceive'. The 'コンポーネント' (Component) palette is open, showing a context menu with '挿入' (Insert) > 'メディエータ' (Mediator) selected, indicated by a red box and a green circle with the number '1'. A dialog box titled 'メディエータの作成' (Create Mediator) is open, showing the '名前(N):' (Name) field set to 'OrderEventToStartMessage', highlighted with a red box and a green circle with the number '2'. The 'OK' button is also highlighted with a red box and a green circle with the number '3'. A red callout box points to the name field with the text '名前: OrderEventToStartMessage'. The background shows the 'BpmJmsReceive' process in the 'composite.xml' file.

# 各要素の実装 ～ シナリオ2

## [受信プロセス] JMSキューから受信 (9/17)

- 作成のメディエータを右クリックして、「編集」を選択



# 各要素の実装 ~ シナリオ2

## [受信プロセス] JMSキューから受信 (10/17)

- インターフェースのWSDLを設定

インターフェースのWSDLとしては  
受信のJMSアダプタ・サービスの  
「JmsReceiveOrderEvent.wsdl」を選択

# 各要素の実装 ～ シナリオ2

## [受信プロセス] JMSキューから受信 (11/17)

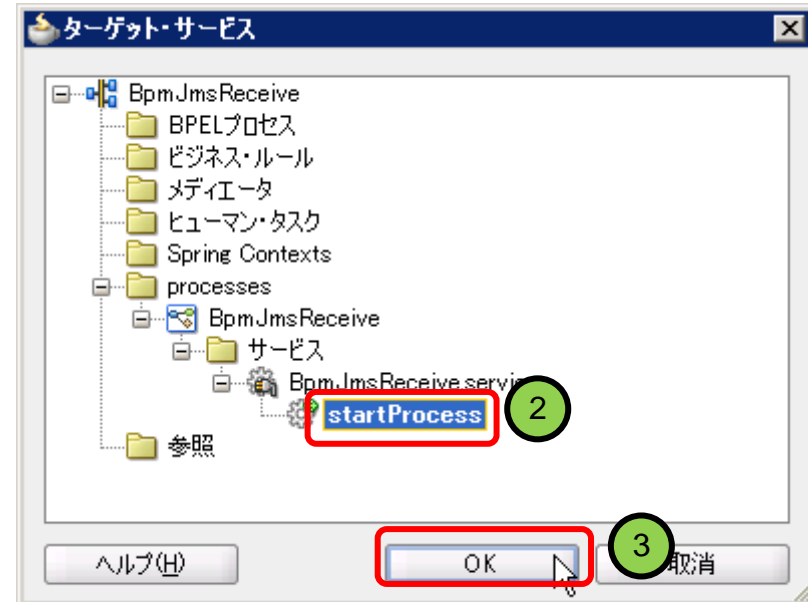
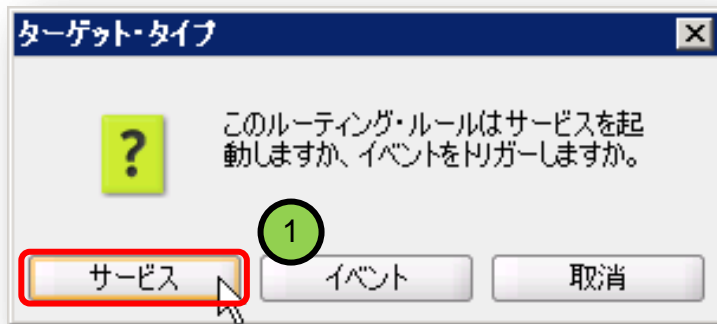
- 「静的ルーティング・ルール」を追加

The screenshot shows the Oracle BPM Studio interface. The main window displays the configuration for a Mediator named 'OrderEventToStartMessage'. The WSDL URL is 'JmsReceiveOrderEvent.wsdl' and the port type is 'Consume\_Message\_ptt'. The '静的ルーティング・ルール' (Static Routing Rule) section is expanded, showing a rule named 'Consume\_Message' with a priority of 4. The '静的ルーティング・ルール' (Static Routing Rule) button is highlighted with a red box, indicating the next step in the process.

# 各要素の実装 ～ シナリオ2

## [受信プロセス] JMSキューから受信 (12/17)

- ターゲット・タイプとして「サービス」を選択し、ターゲット・サービスに「BpmJmsReceive > サービス > BpmJmsReceive.service > startProcess」を選択して、「OK」をクリック



# 各要素の実装 ～ シナリオ2

## [受信プロセス] JMSキューから受信 (13/17)

- データ変換用のマッパー・ファイルを新規作成

Consume\_Message 優先度(P) 4  有効な構文(XSD)の検証(V)

コールアウト先 <<Javaコールアウト・クラス>>

再順序 オフ

静的ルーティング

<<フィルタ式>> BpmJmsReceive/BpmJmsReceive/service::startProcess 順序

セマンティックの検証

次を使用して変換 parameters : xsl/NewOrderEvent\_To\_startProcess.xsl

値の割当て

既存のマッパー・ファイル

# 各要素の実装 ~ シナリオ2

## [受信プロセス] JMSキューから受信 (14/17)

- 「新規マッパー・ファイルの作成」を選択し、マッパー・ファイルを新規作成

The image shows two screenshots from Oracle JDeveloper. The top screenshot is a dialog box titled 'リクエスト・トランスフォーメーション・マップ' (Request Transformation Map). It contains the text 'リクエスト・メッセージ Consume\_Message\_msg からメッセージ startProcess へのトランスフォーメーション。' (Transformation from request message Consume\_Message\_msg to message startProcess). Below this, there is a section 'パートに対するトランスフォーメーション: parameters' (Transformation for part: parameters). There are two radio buttons: '既存のマッパー・ファイルの使用' (Use existing mapper file) and '新規マッパー・ファイルの作成' (Create new mapper file). The second option is selected and circled in red with a green circle containing the number '1'. The text 'NewOrderEvent\_To\_startProcess.xsl' is entered in the adjacent text field. At the bottom of the dialog, the 'OK' button is circled in red with a green circle containing the number '2'. A red callout bubble with a green circle containing the number '3' points to the 'OK' button and contains the text 'ドラッグ・アンド・ドロップで 図のようにマッピングを追加' (Add mapping as shown in the figure by drag-and-drop). The bottom screenshot shows the XSLT mapping editor for the file 'NewOrderEvent\_To\_startProcess.xsl'. The left pane shows the source schema 'JmsReceiveOrderEvent.wsdl' with a tree view containing 'imp1:NewOrderEvent' and its elements 'imp1:orderId' and 'imp1:orderAmount'. The right pane shows the target schema 'BpmJmsReceive.wsdl' with a tree view containing 'tns:startProcess' and its elements 'imp1:NewOrderEvent', 'imp1:orderId', and 'imp1:orderAmount'. Blue lines connect the source elements to the target elements, indicating the mapping.

# 各要素の実装 ~ シナリオ2

## [受信プロセス] JMSキューから受信 (15/17)

- 「値の割当て」アイコンをクリックして、サービス・プロパティのマッピングを設定

静的ルーティング

静的ルーティング: BpmJmsReceive/BpmJmsReceive/service::startProcess

セマンティックの検証: [ ]

次を使用して変換: parameters : xsli/NewOrderEvent\_To\_startProcess.xsl

値の割当て: jca.jms.JMSProperty.msgType := jca.jms.JMSProperty.msgType

値の割当て

値の割当て

元	先
プロパティ: jca.jms.JMSProperty.msgType	プロパティ: jca.jms.JMSProperty.msgType

元: プロパティ jca.jms.JMSProperty.msgType  
先: プロパティ jca.jms.JMSProperty.msgType

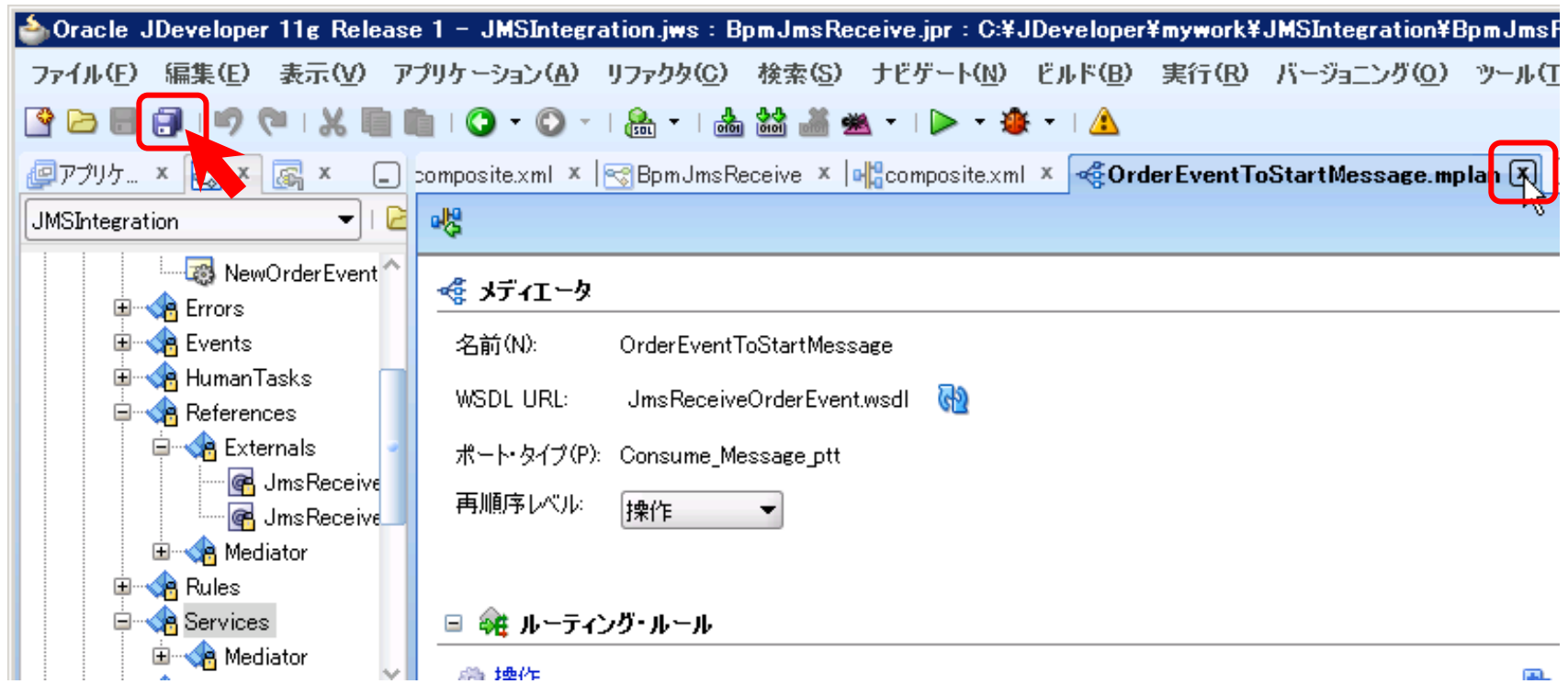
元	先
jca.jms.JMSProperty.msgType	jca.jms.JMSProperty.msgType
apps.context.header	apps.context.header
async.request.reply.to.element	async.request.reply.to.element
b2b.conversationId	b2b.conversationId
b2b.document.DefinitionName	b2b.document.DefinitionName
b2b.document.ProtocolName	b2b.document.ProtocolName
b2b.document.ProtocolVersion	b2b.document.ProtocolVersion
b2b.document.TypeName	b2b.document.TypeName
b2b.from.TradingPartnerId	b2b.from.TradingPartnerId
b2b.from.TradingPartnerIdType	b2b.from.TradingPartnerIdType
b2b.messageId	b2b.messageId
b2b.messageType	b2b.messageType
b2b.reply.To.MessageId	b2b.reply.To.MessageId
b2b.to.TradingPartnerId	b2b.to.TradingPartnerId
b2b.to.TradingPartnerIdType	b2b.to.TradingPartnerIdType
bpel.auditLevel	bpel.auditLevel
bpel.callback.PayloadPref	bpel.callback.PayloadPref
bpel.completion.PersistLevel	bpel.completion.PersistLevel
bpel.completion.PersistPolicy	bpel.completion.PersistPolicy
bpel.conversationId	bpel.conversationId
bpel.creator	bpel.creator
bpel.delivery.PersistPolicy	bpel.delivery.PersistPolicy



# 各要素の実装 ～ シナリオ2

## [受信プロセス] JMSキューから受信 (16/17)

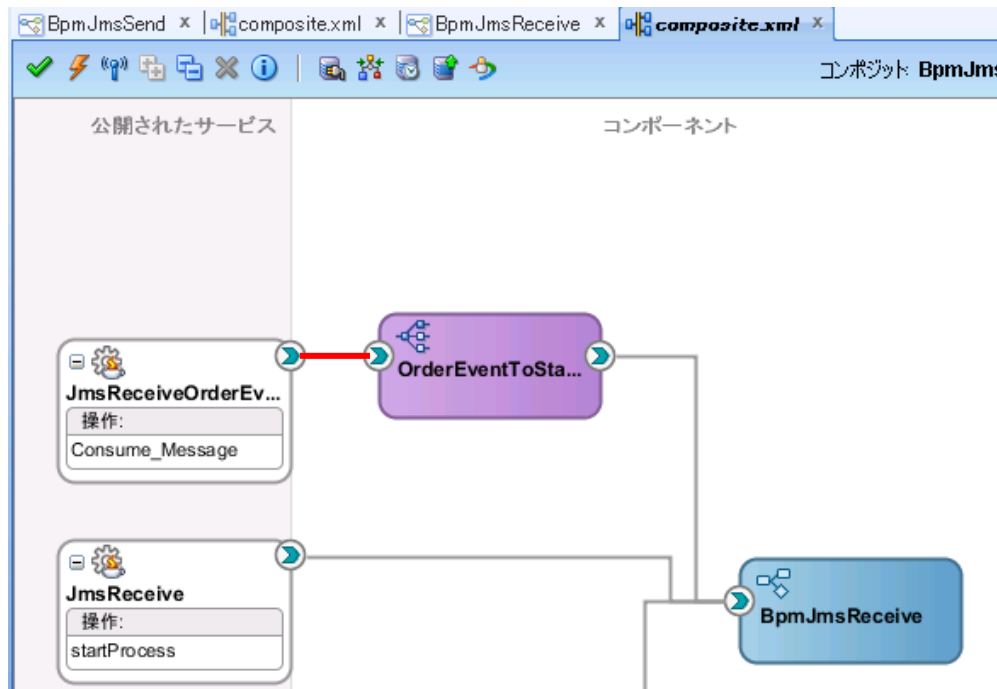
- 「すべて保存」をして、「OrderEventToStartMessage.mplan」タブを閉じる



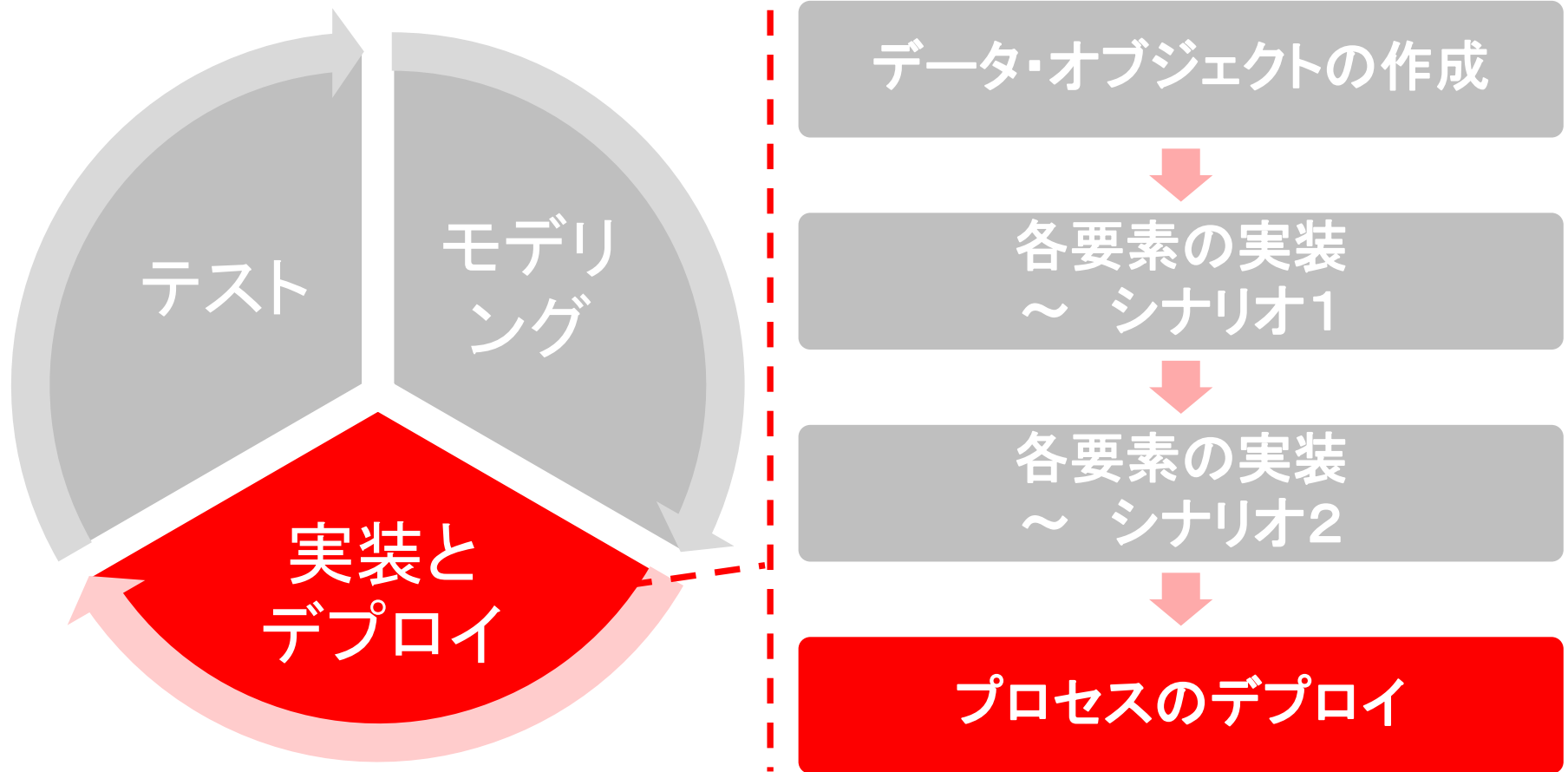
# 各要素の実装 ～ シナリオ2

## [受信プロセス] JMSキューから受信 (17/17)

- 「JmsReceiveOrderEvent」サービスからメディエータ「OrderEventToStartMessage」までのワイヤーを追加
- 「すべて保存」をして、「シナリオ2」の実装が完了

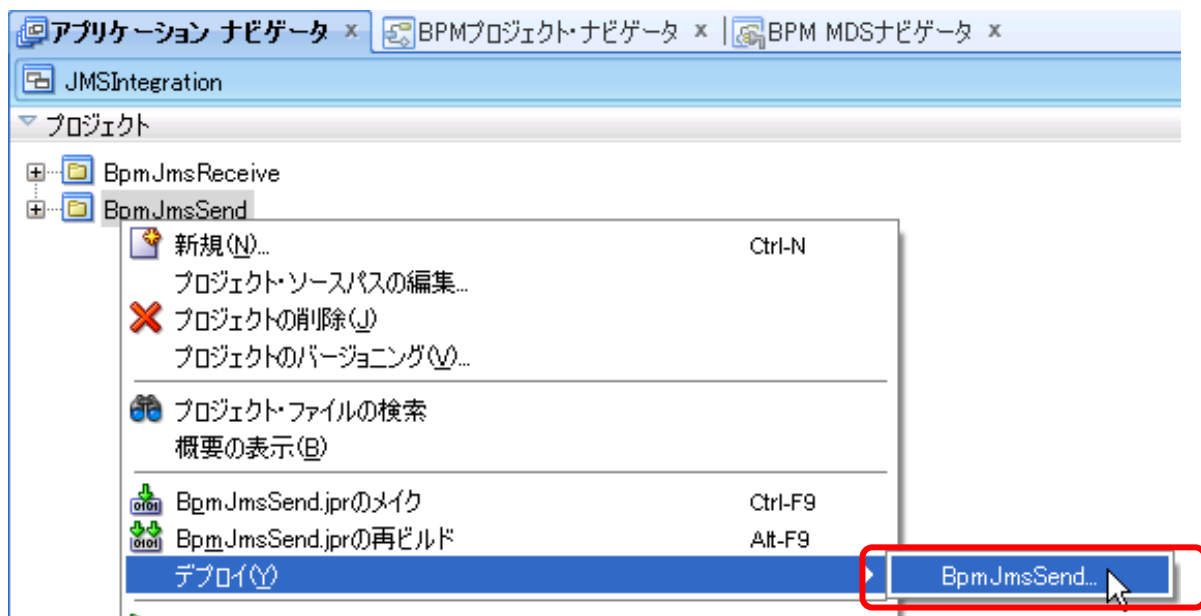


# JMSIntegrationアプリケーションの作成



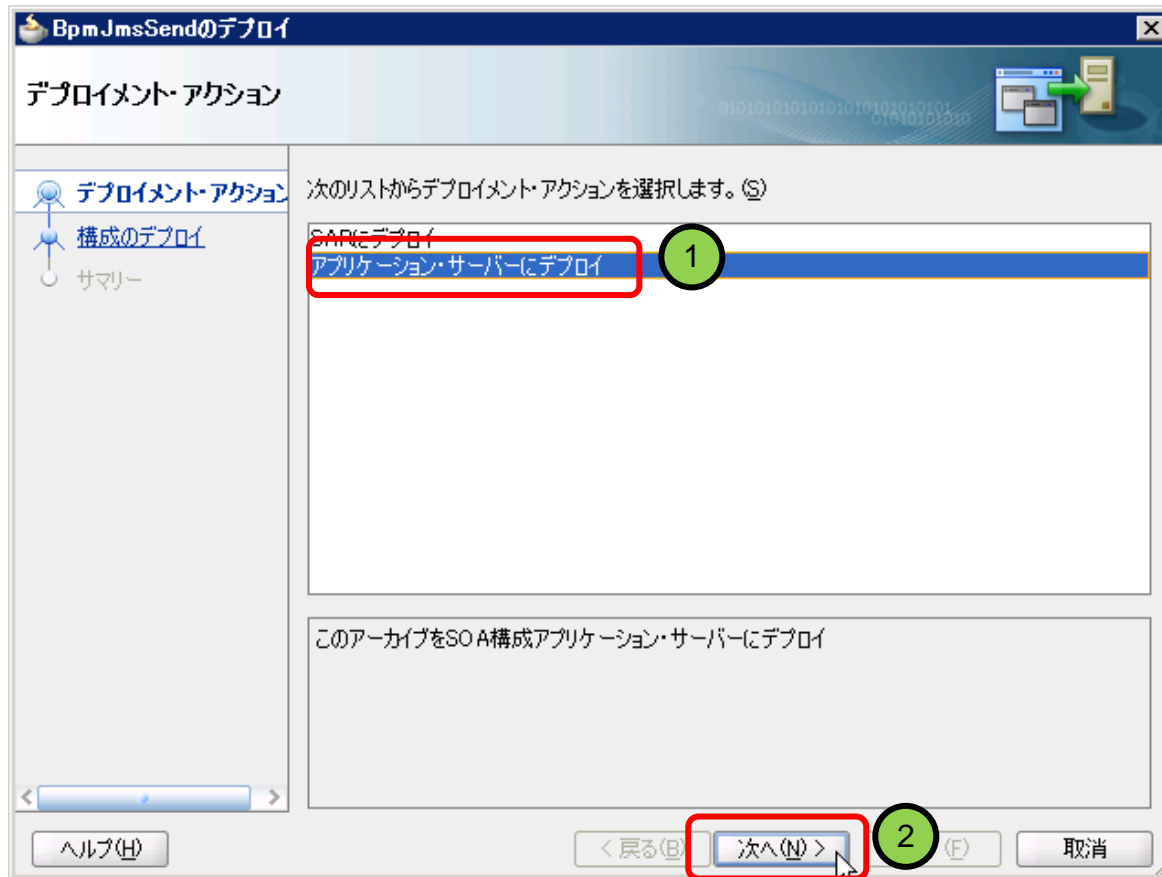
# プロセスのデプロイ (1/8)

- 「アプリケーション・ナビゲータ」で、プロジェクト「BpmJmsSend」を右クリックし、「デプロイ > BpmJmsSend」を選択



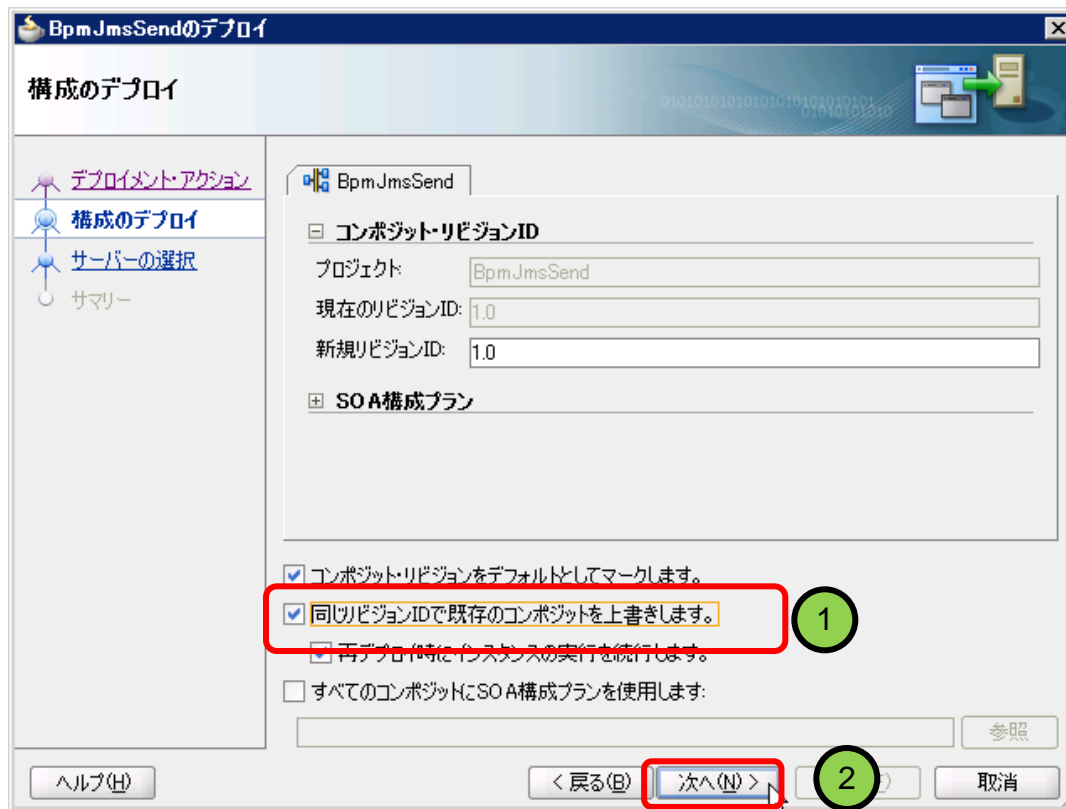
# プロセスのデプロイ (2/8)

- 「アプリケーション・サーバーにデプロイ」を選択して「次へ」



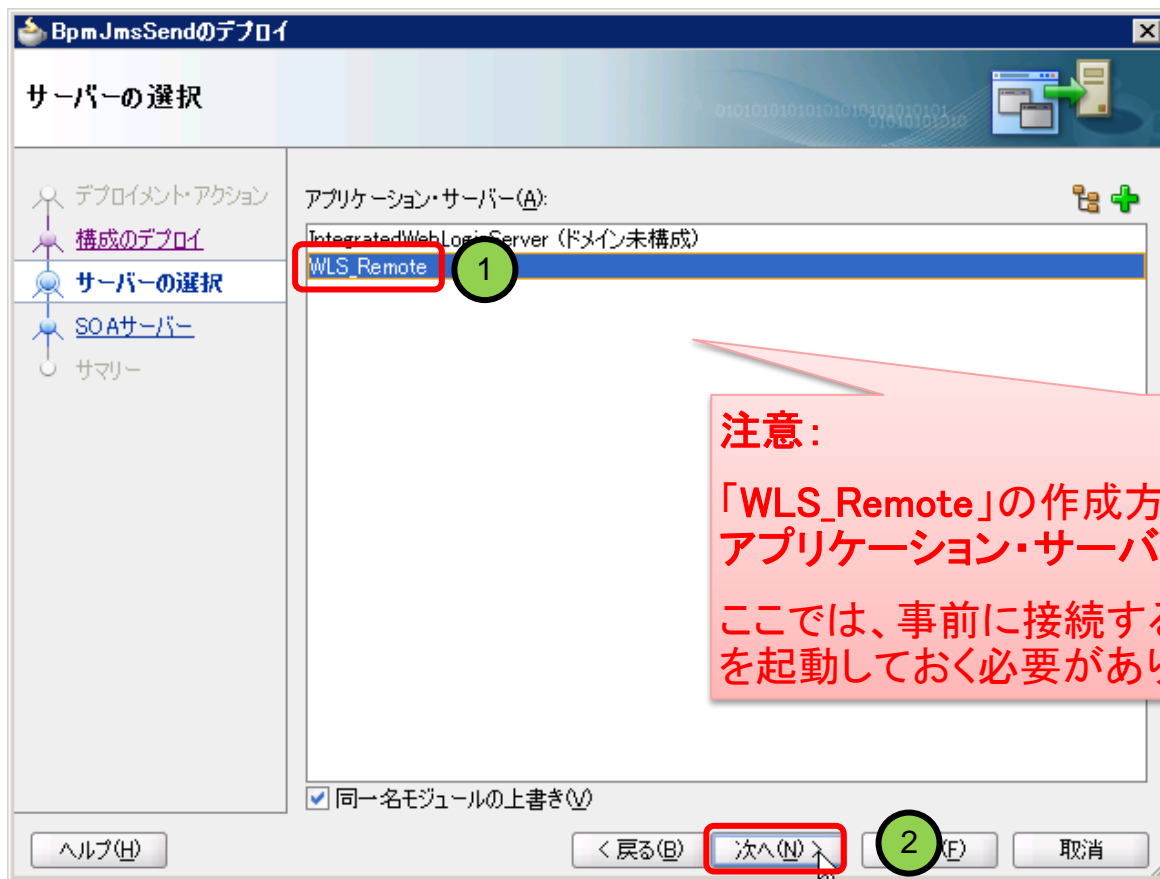
# プロセスのデプロイ (3/8)

- 「同じビジョンIDで既存のコンポジットを上書きします。」をチェックして「次へ」



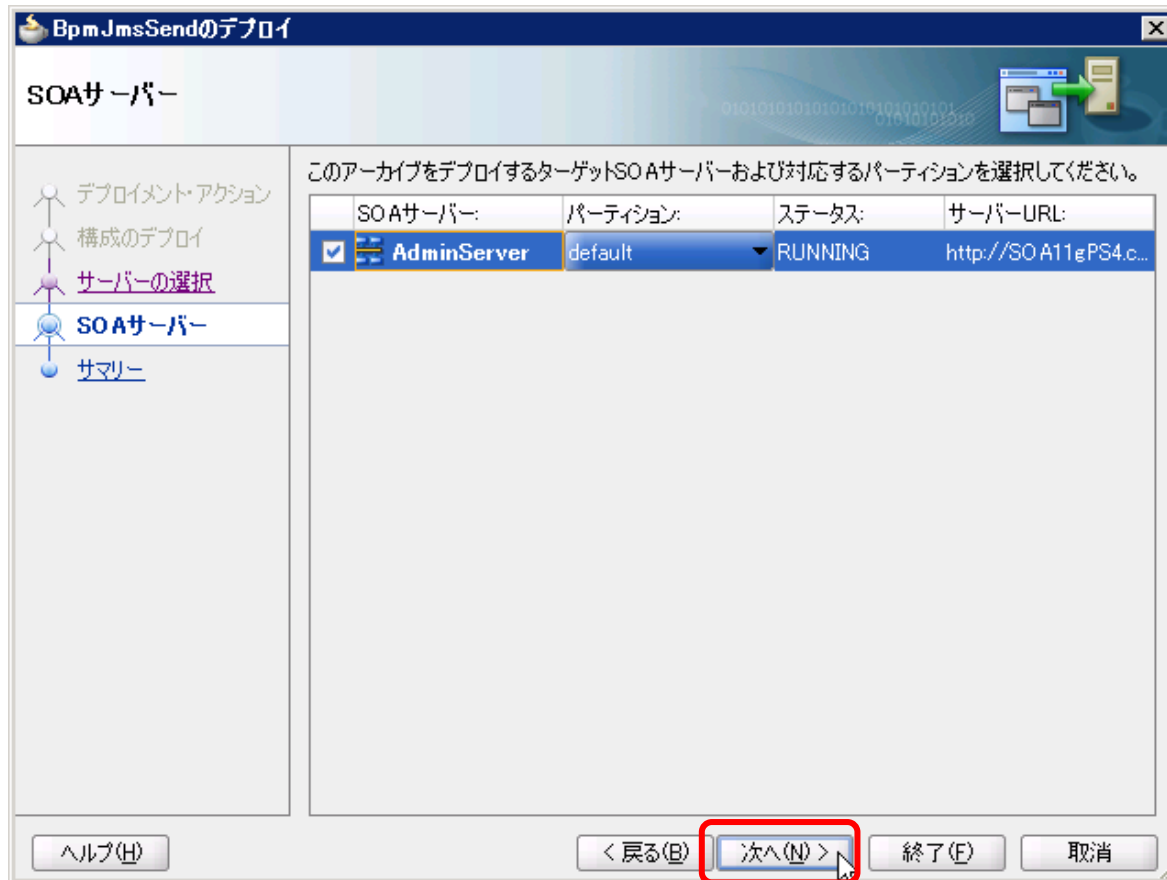
# プロセスのデプロイ (4/8)

- 「WLS\_Remote」を選択して「次へ」



# プロセスのデプロイ (5/8)

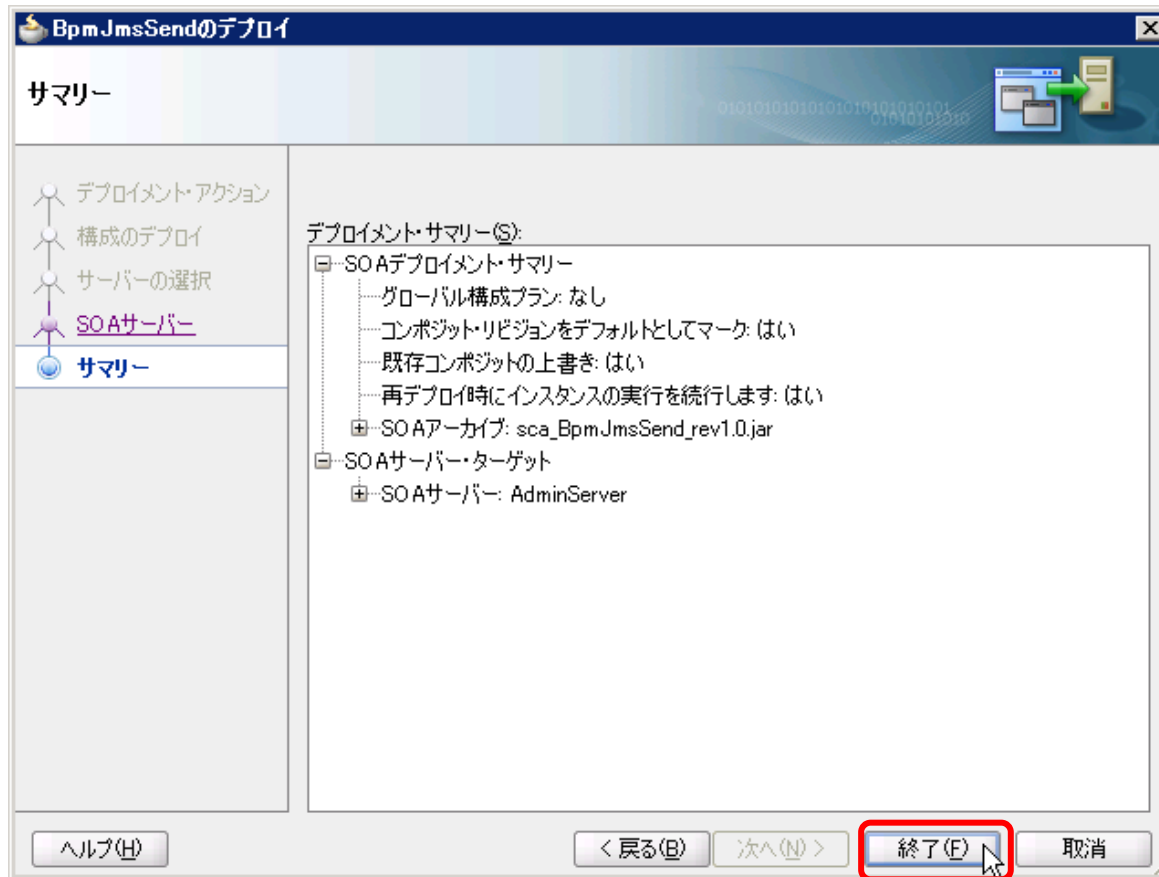
- 検出されたSOAサーバーを選択したまま「次へ」





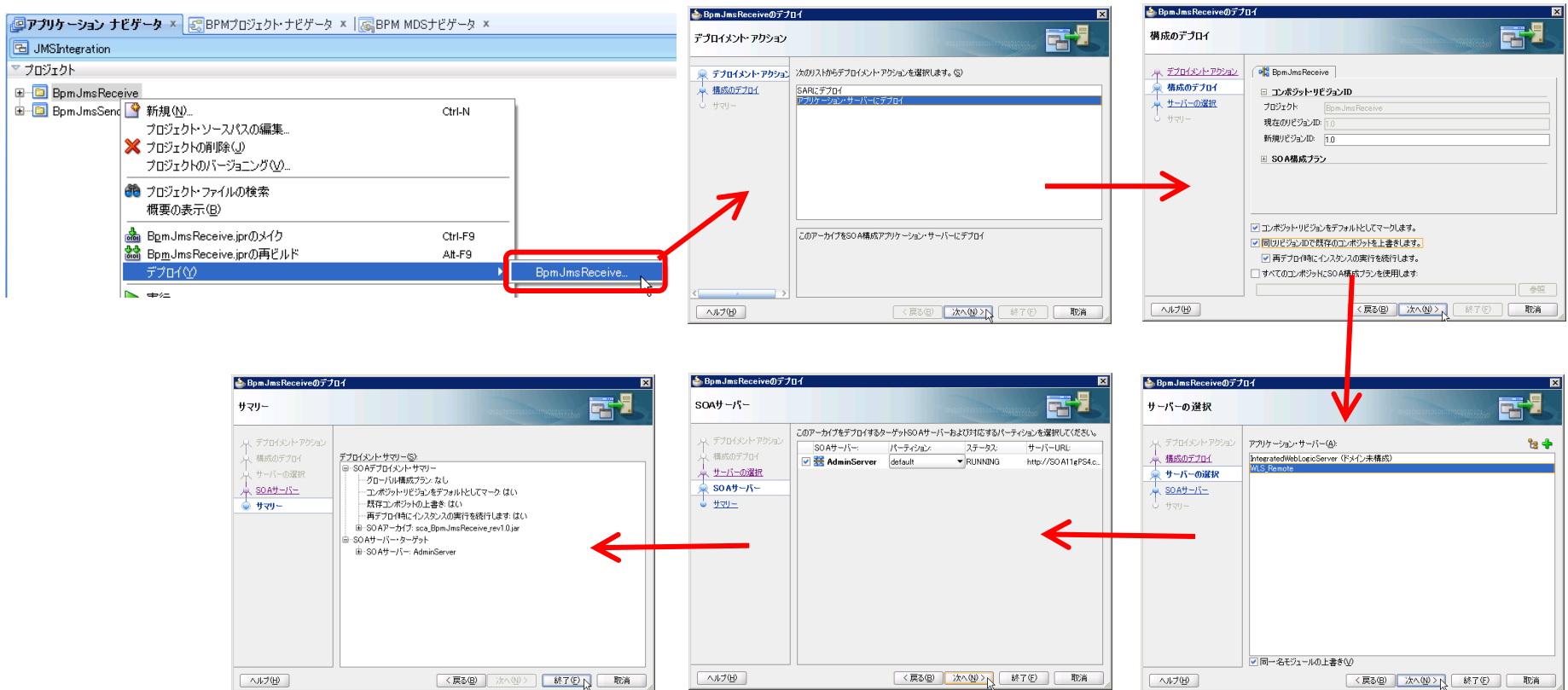
# プロセスのデプロイ (6/8)

- 「終了」をクリックし、デプロイを開始する



# プロセスのデプロイ (7/8)

- 同様に「BpmJmsReceive」をデプロイ

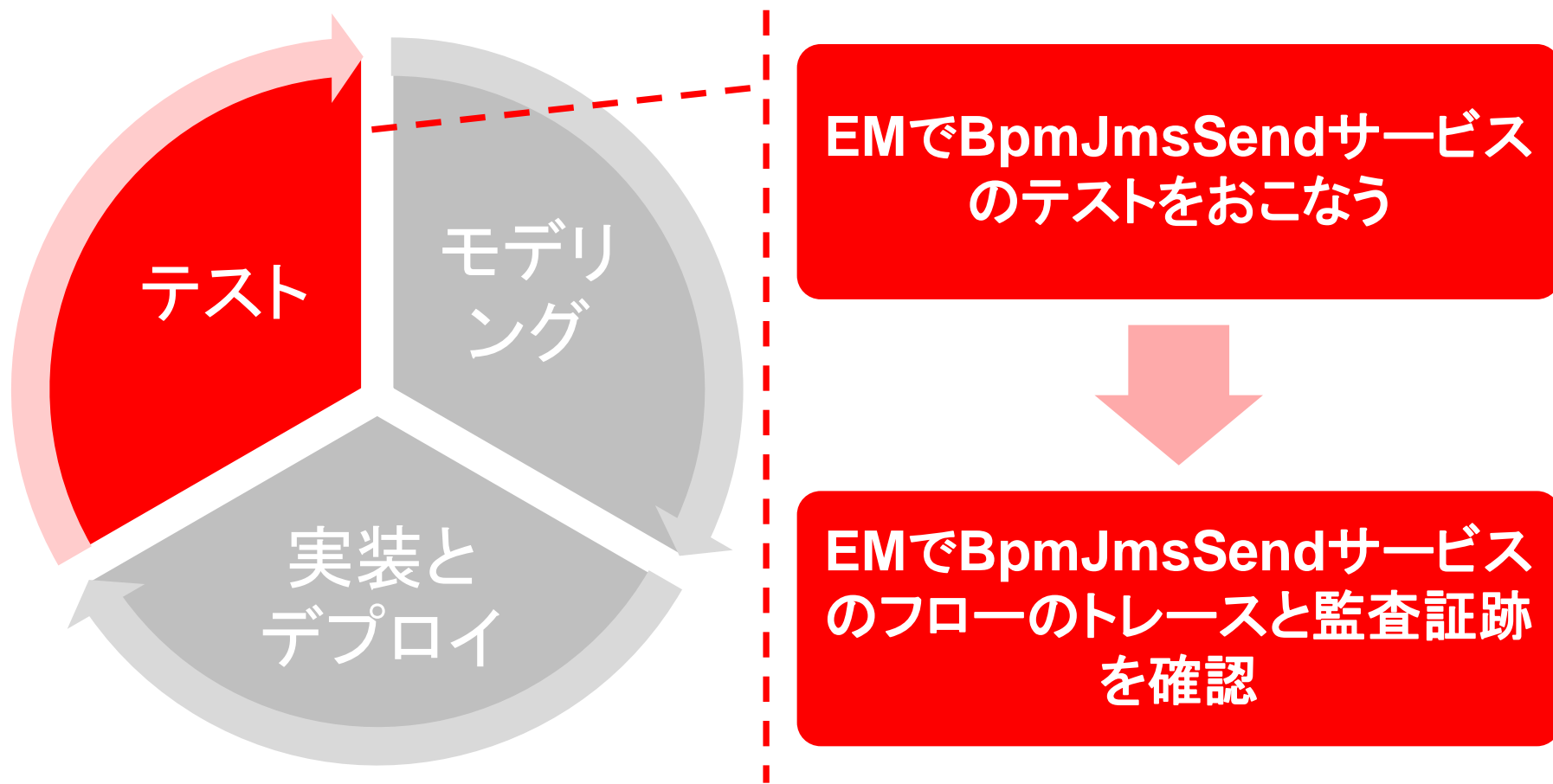


# プロセスのデプロイ (8/8)

- デプロイ完了した後のログは、以下となる

```
デプロイメント - ログ
[02:46:53 午後] ---- デプロイ開始。 ----
[02:46:53 午後] ターゲット・プラットフォームは (Weblogic 10.3)です。
[02:46:53 午後] 依存性分析を実行中...
[02:46:54 午後] ビルド中...
[02:49:01 午後] プロファイルのデプロイ中...
[02:49:01 午後] SOAプロジェクト'BpmJmsSend.jpr'のリビジョンIDを'1.0'に更新中..
[02:49:03 午後] アーカイブ・モジュールをC:\JDeveloper\mywork\JMSIntegration\BpmJmsSend\deploy\sca_BpmJmsSend_rev1.0.jarに作成しました
[02:49:04 午後] サーバーAdminServer [http://SOAllgPS4.cn.oracle.com:7001]のパーティション"default"にsca_BpmJmsSend_rev1.0.jarをデプロイしています
[02:49:04 午後] Processing sar=/C:/JDeveloper/mywork/JMSIntegration/BpmJmsSend/deploy/sca_BpmJmsSend_rev1.0.jar
[02:49:04 午後] Adding sar file - C:\JDeveloper\mywork\JMSIntegration\BpmJmsSend\deploy\sca_BpmJmsSend_rev1.0.jar
[02:49:04 午後] Preparing to send HTTP request for deployment
[02:49:06 午後] Creating HTTP connection to host:SOAllgPS4.cn.oracle.com, port:7001
[02:49:07 午後] Sending internal deployment descriptor
[02:49:07 午後] Sending archive - sca_BpmJmsSend_rev1.0.jar
[02:50:06 午後] Received HTTP response from the server, response code=200
[02:50:06 午後] サーバーAdminServer [http://SOAllgPS4.cn.oracle.com:7001]のパーティション"default"にアーカイブsca_BpmJmsSend_rev1.0.jarが正常にデプロイされました
[02:50:06 午後] デプロイメントの経過時間: 3分, 13秒
[02:50:06 午後] ---- デプロイ終了。 ----
[02:51:38 午後] ---- デプロイ開始。 ----
[02:51:38 午後] ターゲット・プラットフォームは (Weblogic 10.3)です。
[02:51:38 午後] 依存性分析を実行中...
[02:51:38 午後] ビルド中...
[02:51:45 午後] プロファイルのデプロイ中...
[02:51:45 午後] SOAプロジェクト'BpmJmsReceive.jpr'のリビジョンIDを'1.0'に更新中..
[02:51:45 午後] アーカイブ・モジュールをC:\JDeveloper\mywork\JMSIntegration\BpmJmsReceive\deploy\sca_BpmJmsReceive_rev1.0.jarに作成しました
[02:51:45 午後] サーバーAdminServer [http://SOAllgPS4.cn.oracle.com:7001]のパーティション"default"にsca_BpmJmsReceive_rev1.0.jarをデプロイしています
[02:51:45 午後] Processing sar=/C:/JDeveloper/mywork/JMSIntegration/BpmJmsReceive/deploy/sca_BpmJmsReceive_rev1.0.jar
[02:51:45 午後] Adding sar file - C:\JDeveloper\mywork\JMSIntegration\BpmJmsReceive\deploy\sca_BpmJmsReceive_rev1.0.jar
[02:51:45 午後] Preparing to send HTTP request for deployment
[02:51:45 午後] Creating HTTP connection to host:SOAllgPS4.cn.oracle.com, port:7001
[02:51:45 午後] Sending internal deployment descriptor
[02:51:45 午後] Sending archive - sca_BpmJmsReceive_rev1.0.jar
[02:52:19 午後] Received HTTP response from the server, response code=200
[02:52:19 午後] サーバーAdminServer [http://SOAllgPS4.cn.oracle.com:7001]のパーティション"default"にアーカイブsca_BpmJmsReceive_rev1.0.jarが正常にデプロイされま
[02:52:19 午後] デプロイメントの経過時間: 41秒
[02:52:19 午後] ---- デプロイ終了。 ----
```

# JMSIntegrationアプリケーションの作成



# プロセスのテスト (1/5)

- 「weblogic」ユーザーでEMにログインし、左側のパネルから「SOA > soa-infra > default > BpmJmsSend」を選択し、右側の「テスト」をクリック

The screenshot shows the Oracle Enterprise Manager 11g Fusion Middleware Control interface. The left-hand navigation tree is expanded to show the path: Farm\_base\_domain > アプリケーション・デプロイメント > SOA > soa-infra (AdminServer) > default > BpmJmsSend [1.0]. A red box highlights the 'BpmJmsSend [1.0]' component, with a red arrow pointing to it and a green circle containing the number '1'. The right-hand pane displays the details for 'BpmJmsSend [1.0]'. At the top, there are buttons for '実行中インスタンス 0', '合計 2', 'アクティブ', 'リタイア...', '停止...', and 'テスト'. The 'テスト' button is highlighted with a red box and a green circle containing the number '2'. Below the buttons, there are tabs for 'ダッシュボード', 'インスタンス', 'フォルトと拒否メッセージ', 'ユニット・テスト', and 'サマリー'. The '最新のインスタンス' section shows a table with columns for 'インスタンス ID', '名前', '対話ID', and '状態'. The table contains two rows of data, both with a status of '失効' (Failed).

インスタンス ID	名前	対話ID	状態
200003			失効
200001			失効

# プロセスのテスト (2/5)

- 引数に任意値を入力して、「Webサービスのテスト」をクリック

## Webサービスのテスト

このページを使用して、ファーム内には、WSDLを含む任意のWSDLをテストします。Webサービスをテストするには、WSDLを入力し、「WSDL解析」をクリックします。WSDLの詳細を使用してページがリフレッシュされた後、最初にサービスを選択してから、ポートを選択し、テスト対象の操作を選択します。任意の入力パラメータを指定し、「Webサービスのテスト」をクリックします。

Webサービスのテスト 

WSDL

WSDL解析

WSDLアクセスのためのHTTP Basic認証オプション

サービス BpmJmsSend.service

ポート BpmJmsSendPort

操作 operation

エンドポイントURL  編集 エンドポイントURL

リクエスト

レスポンス

セキュリティ

パラメータ

HTTP

追記

引数

ツリー表示

名前	タイプ	値
* parameters	parameters	
* orderId	string	<input type="text" value="orderId123"/>
* orderAmount	int	<input type="text" value="123456"/>

orderId123 

123456

# プロセスのテスト (3/5)

- フロー・トレースの起動して、ターゲット・プロセス「BpmJmsReceive」の2つのインスタンスが起動されていることを確認

The screenshot displays the Oracle BPM Suite interface. At the top, there are tabs for 'リクエスト' (Request) and 'レスポンス' (Response). The 'レスポンス' tab is active, showing a message: 'テストのステータス リクエストは正常に受信されました。' (Test status: Request received normally). Below this, it shows 'レスポンス時間(ミリ秒) 171' (Response time: 171 ms) and a 'ツリー表示' (Tree view) button. A red box highlights the text '新しいコンポジット・インスタンスが生成されました。' (New composite instance generated) and a button labeled 'フロー・トレースの起動' (Start flow trace).

Below the message, a 'トレース' (Trace) window is open, displaying a table of process instances. The table has columns for 'インスタンス' (Instance), 'タイプ' (Type), '使用状況' (Usage), '状態' (Status), and 'コンポジット・インスタンス' (Composite Instance). Two instances of 'BpmJmsReceive' are highlighted with a red box, indicating they are the target of the test.

インスタンス	タイプ	使用状況	状態	コンポジット・インスタンス
BpmJmsSend.service	Webサービス	サービス	完了	200007のBpmJmsSend
BpmJmsSend	BPMNコンポー...		完了	200007のBpmJmsSend
JmsSend	JCAアダプタ	参照	完了	200007のBpmJmsSend
JmsSendNewOrderEvent	JCAアダプタ	参照	完了	200007のBpmJmsSend
JmsReceive	JCAアダプタ	サービス	完了	200008のBpmJmsReceive
JmsReceiveOrderEvent	JCAアダプタ	サービス	完了	200009のBpmJmsReceive
OrderEventToStartMessage	メディエータ・コ...		完了	200009のBpmJmsReceive
BpmJmsReceive	BPMNコンポー...		完了	200009のBpmJmsReceive
BpmJmsReceive	BPMNコンポー...		完了	200008のBpmJmsReceive

# プロセスのテスト (4/5)

- シナリオ1で直接呼び出されたターゲット・プロセス・インスタンスの監査証跡を確認

## トレース

コンポーネント・インスタンスをクリックして監査アクティビティIDの表示

### インスタンス

- BpmJmsSend.service
  - BpmJmsSend
    - JmsSend
    - JmsSendNewOrderEvent
    - JmsReceive
    - JmsReceiveOrderEvent
    - OrderEventToStartMessage
      - BpmJmsReceive
      - BpmJmsReceive** ①

監査証跡 フロー フォルト 現在の監

アクティビティ	ループ件数	イベント
Start	0	インスタンス作成済 アクティビティ完了
DebugOutput	0	アクティビティ完了 インスタンスがアクティビティに入りました <b>インスタンスがアクティビティを離れました</b> ②
End	0	アクティビティ完了 インスタンス終了

バイロードXML

```
<? [msgType:START_MESSAGE] id:orderID123][amt:123456]<
```



# プロセスのテスト (5/5)

- シナリオ2で、メディエータ経由で呼び出されたターゲット・プロセス・インスタンスの監査証跡を確認

インスタンス

- BpmJmsSend.service
  - BpmJmsSend
    - JmsSend
    - JmsSendNewOrderEvent
    - JmsReceive
    - JmsReceiveOrderEvent
    - OrderEventToStartMessage
      - BpmJmsReceive** (1)
      - BpmJmsReceive

監査証跡

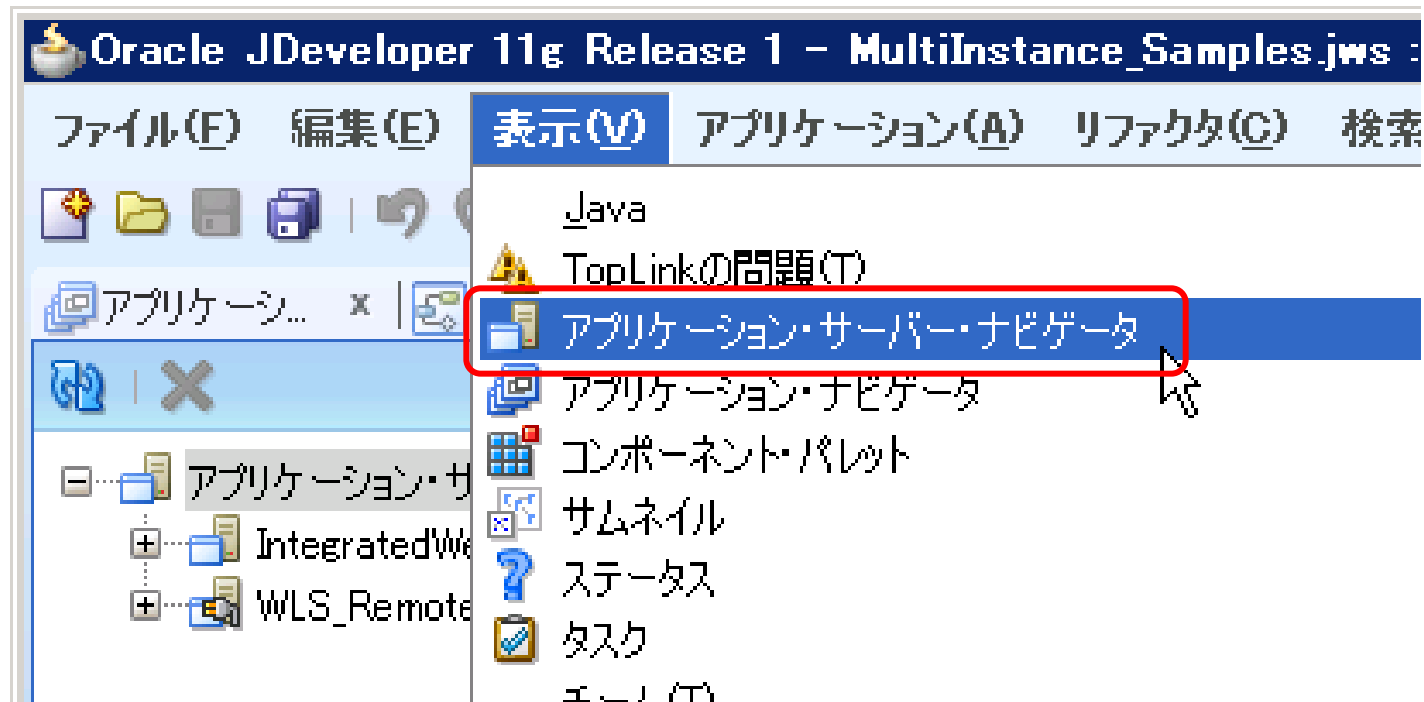
アクティビティ	ループ件数	イベント
		インスタンス作成済
Start	0	アクティビティ完了
DebugOutput	0	アクティビティ完了
		インスタンスがアクティビティに入りました
		<b>インスタンスがアクティビティを離れました</b> (2)
End	0	アクティビティ完了
		インスタンス終了

ペイロードXML

```
e">[msgType:ORDER_EVENT] id:orderID123][amt:123456]</
```

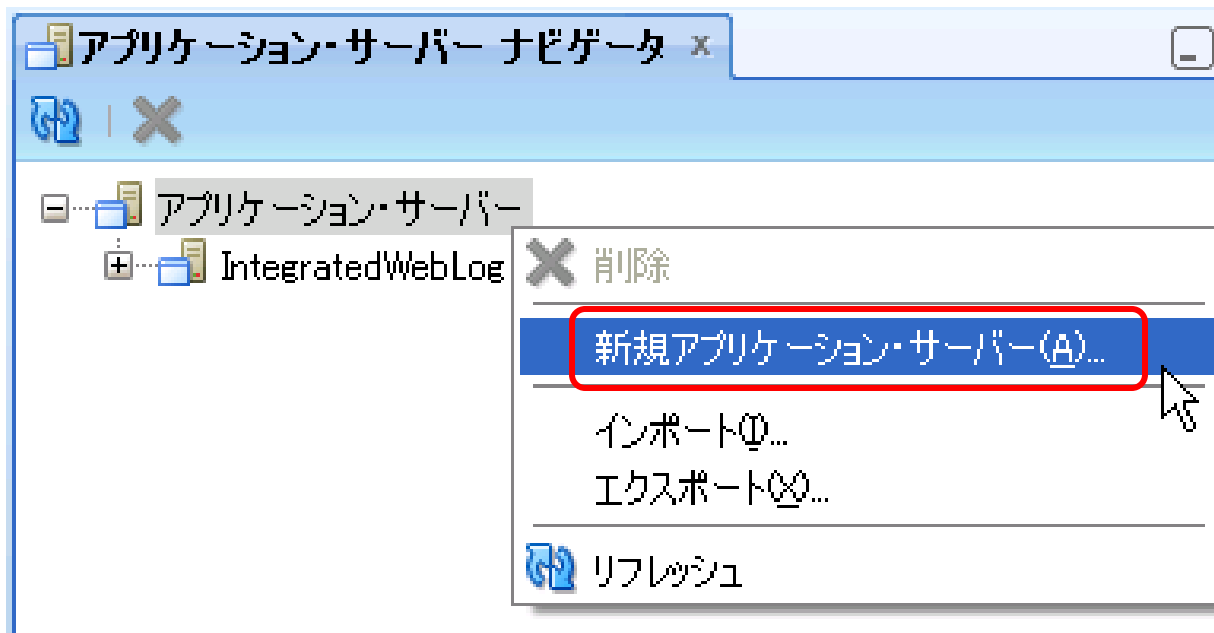
# [補足] アプリケーション・サーバー接続の作成 (1/7)

- メニューから「表示 > アプリケーション・サーバー・ナビゲータ」を選択して、アプリケーション・サーバー・ナビゲータを表示させる



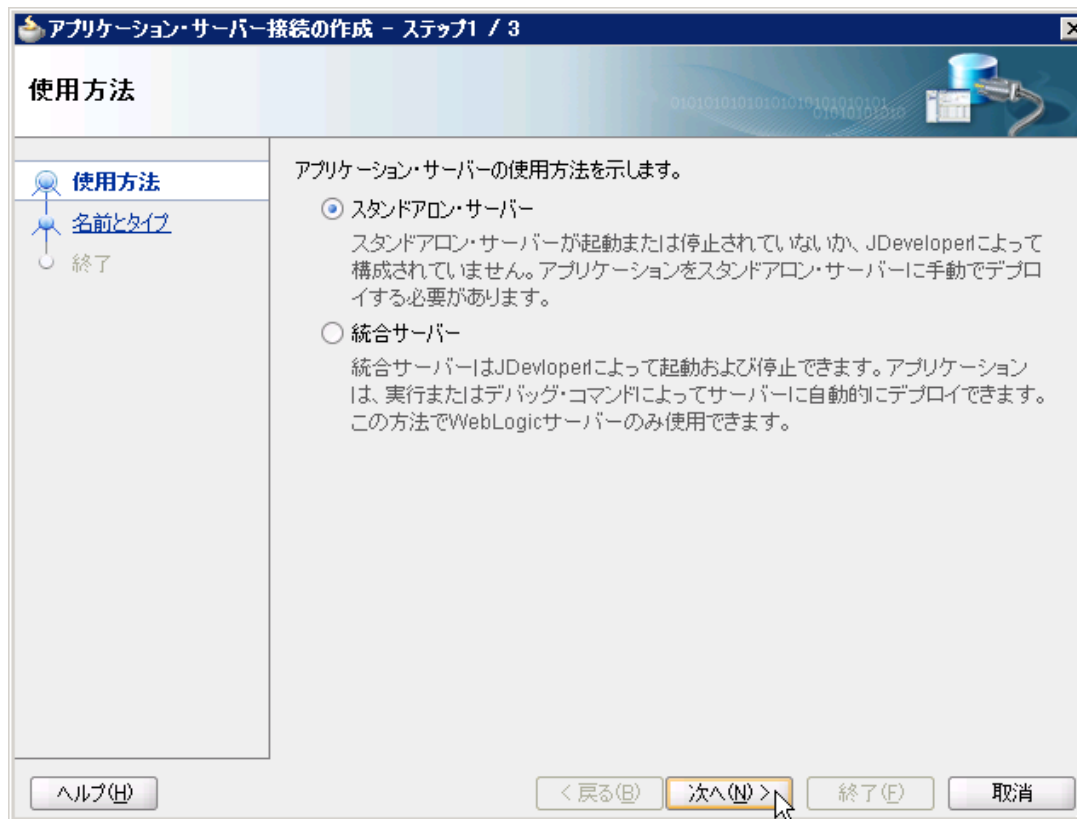
# [補足] アプリケーション・サーバ接続の作成 (2/7)

- アプリケーション・サーバー・ナビゲータで、ルート「アプリケーション・サーバー」を右クリックし、「新規アプリケーション・サーバー」をクリック



# [補足] アプリケーション・サーバ接続の作成 (3/7)

- 作成ウィザードが表示され、「スタンドアロン・サーバ」を選択し、「次へ」をクリック



# [補足] アプリケーション・サーバ接続の作成 (4/7)

- 「接続名」に「WLS\_Remote」を入力し、「次へ」をクリック

アプリケーション・サーバ接続の作成 - ステップ2 / 6

名前とタイプ

使用方法  
名前とタイプ  
認証  
構成  
テスト  
終了

接続について一意の名前とタイプを指定します。名前は有効なJava識別子である必要があります。

接続の作成場所: リソース・パレット(P)

接続名(N):  
WLS\_Remote

接続タイプ(T):  
WebLogic 10.3

ヘルプ(H) < 戻る(B) 次へ(N) > 終了(F) 取消

# [補足] アプリケーション・サーバ接続の作成 (5/7)

- 管理ユーザーの「weblogic」とパスワードを入力し、「次へ」をクリック

アプリケーション・サーバ接続の作成 - ステップ3 / 6

認証

接続を認証するためのユーザー名およびパスワードを指定します。

使用方法  
名前とタイプ  
認証  
構成  
テスト  
終了

ユーザー名(U):  
weblogic

パスワード(P):  
.....

ヘルプ(H) <戻る(B) **次へ(N) >** 終了(E) 取消

# [補足] アプリケーション・サーバ接続の作成 (6/7)

- 接続するWeblogicホスト名(管理サーバー)の情報を入力し、「次へ」をクリック

アプリケーション・サーバ接続の作成 - ステップ4 / 6

構成

WebLogic Server接続では接続を確立するためにホスト名とポートを使用します。ターゲットのドメインが検証されます

Weblogicホスト名(管理サーバー)(O):  
soa11eps4

ポート(P): 7001      SSLポート(S): 7002

常にSSLを使用(A)

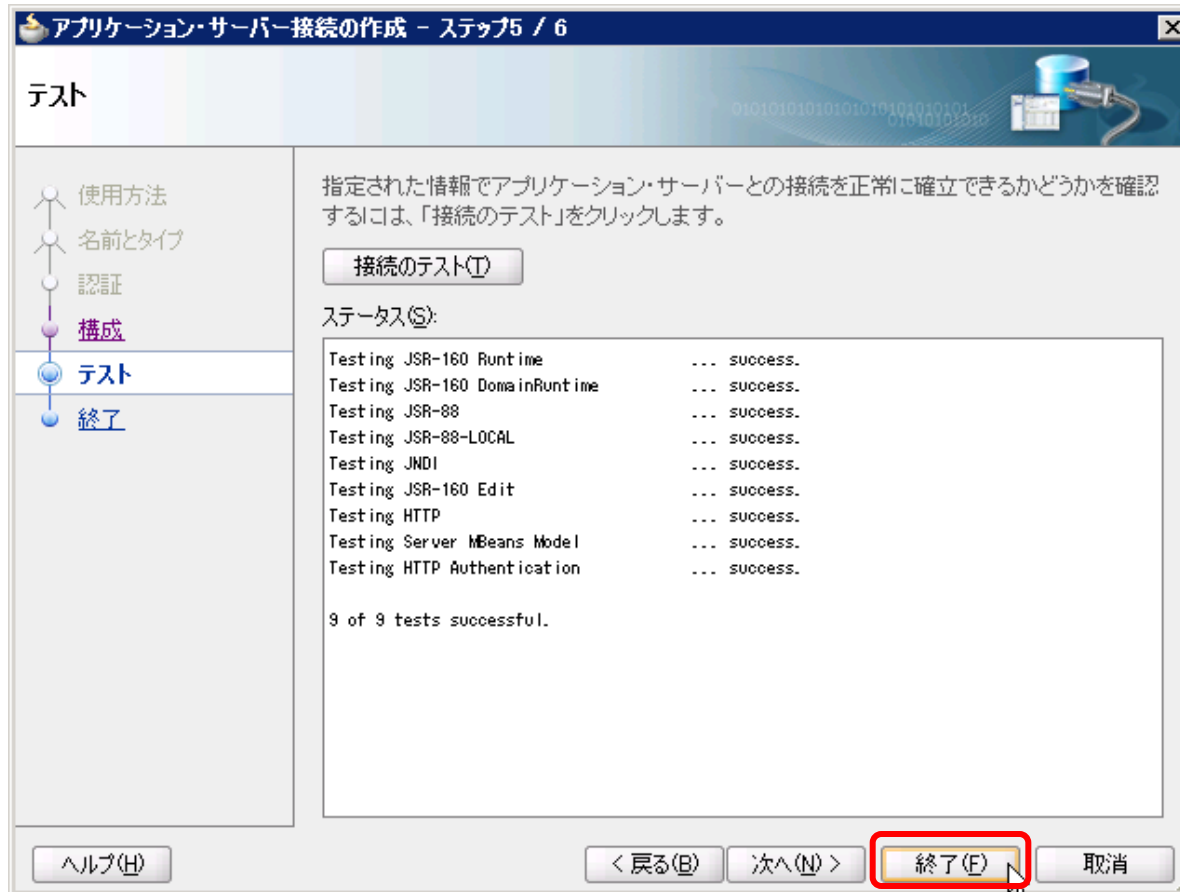
WebLogicドメイン(D):  
base\_domain

ヘルプ(H)      < 戻る(B)      次へ(N) >      終了(F)      取消

項目	値	
Weblogic ホスト名	localhost	環境に合わせて 変更必要
ポート	7001	環境に合わせて 変更必要
Weblogic ドメイン	base_domain	環境に合わせて 変更必要

# [補足] アプリケーション・サーバ接続の作成 (7/7)

- 接続テストをおこない、正常の場合、「終了」をクリック





# リファレンス

- 英語版のJMSIntegrationサンプル・アプリケーション・ガイド  
<http://java.net/projects/oraclebpmsuite11g/downloads/download/Samples/bpm-int-101-JMS-integration/bpm-int-101-JMS-integration.pdf>
- JMSアダプタの詳細情報について  
[http://docs.oracle.com/cd/E24001\\_01/integration.1111/b55918/adptr\\_jms.htm#CJACBCHJ](http://docs.oracle.com/cd/E24001_01/integration.1111/b55918/adptr_jms.htm#CJACBCHJ)
- JMSアダプタのプロパティの詳細情報について  
[http://docs.oracle.com/cd/E24001\\_01/integration.1111/b55918/adptr\\_properties.htm#CIHFDCCF](http://docs.oracle.com/cd/E24001_01/integration.1111/b55918/adptr_properties.htm#CIHFDCCF)

# リファレンス

- メディエータの詳細情報について

[http://docs.oracle.com/cd/E24001\\_01/integration.1111/b56238/partpage\\_i\\_i.htm#BABDDIDA](http://docs.oracle.com/cd/E24001_01/integration.1111/b56238/partpage_i_i.htm#BABDDIDA)

- JMSキューの構成について

[http://docs.oracle.com/cd/E24001\\_01/web.1111/b61636/basic\\_config.htm#i1129323](http://docs.oracle.com/cd/E24001_01/web.1111/b61636/basic_config.htm#i1129323)

# **Hardware and Software Engineered to Work Together**

**ORACLE®**