

PowerCenter Web Services Provider ガイド-copyright.....	3
はじめに.....	6
Informatica のリソース.....	6
Web サービスの概念.....	8
Web サービスの概念の概要.....	8
Simple Object Access Protocol (SOAP)	9
Web サービス記述言語 (WSDL)	9
Web Services Provider について.....	10
Web Services Provider についての概要.....	10
Web Services Provider のアーキテクチャ.....	12
パフォーマンスおよび拡張性.....	13
Web Services Hub のセキュリティ.....	14
Web Services Hub ログ.....	15
SOAP フォールトの処理.....	15
Web Services Hub Console の使い方.....	17
Web Services Hub Console の使い方の概要.....	17
Web Services Hub Console への接続.....	18
Web Services Hub Console について.....	18
Web サービスのテスト.....	21
バッチ Web サービスの操作.....	24
バッチ Web サービスの操作の概要.....	24
メタデータ Web サービスの操作.....	25
Data Integration Web サービスの操作.....	27
クライアントアプリケーションの作成.....	54
クライアントアプリケーションの作成の概要.....	54
バッチ Web サービスのクライアントアプリケーション.....	54
バッチ Web サービスの Java クライアントアプリケーション.....	57
バッチ Web サービスの C#クライアントアプリケーション.....	60
リアルタイム Web サービスのクライアントアプリケーション.....	63
リアルタイム Web サービスの Java クライアントアプリケーション.....	64
パラメータ配列の使用.....	66

クライアント要求へのセキュリティの追加.....	69
Web サービスのソースおよびターゲットに関する作業.....	73
Web サービスのソースおよびターゲットの概要.....	73
Web サービスのソースおよびターゲットの理解.....	74
Web サービスのソース定義またはターゲット定義のインポート.....	77
ソース定義またはターゲット定義の作成.....	81
Web サービスのソースおよびターゲットの編集.....	84
Web サービスのソースおよびターゲットの編集の概要.....	84
Designer ワークスペースでの定義の編集.....	84
WSDL ワークスペースでの定義の編集.....	85
Web サービスマッピングの使用.....	87
Web サービスマッピングの使用の概要.....	87
Web サービスマッピングのタイプ.....	87
WSDL からのマッピングの生成.....	89
リレーショナルまたはフラットファイルのソースまたはターゲットからのマッピングの生成.....	90
トランスフォーメーションまたはマップレットからのマッピングの生成.....	91
Web サービスマッピングでのターゲットインスタンスの編集.....	93
アタッチメント.....	94
Web サービスワークフローの使用.....	95
Web サービスワークフローの使用の概要.....	95
Web サービスワークフローの作成および設定.....	96
Web Services Provider Reader および Writer の設定.....	99
Web サービスセッションのパーティションの設定.....	103
Web サービスワークフローのトラブルシューティング.....	103
Web サービスのサンプルクライアントアプリケーション.....	104
Web サービスのサンプルクライアントアプリケーションの概要.....	104
バッチ Web サービスサンプルプログラムの使用.....	105
バッチ Web サービスの例.....	107
リアルタイム Web サービスサンプルプログラムの使用.....	112
リアルタイム Web サービスの例.....	116
Web ブラウザの設定.....	117
Web ブラウザの設定.....	117

PowerCenter Web Services Provider ガイド-copyright

本ソフトウェアおよびマニュアルは、使用および開示の制限を定めた個別の使用許諾契約のもとでのみ提供されています。本マニュアルのいかなる部分も、いかなる手段（電子的複製、写真複製、録音など）によっても、Informatica LLC の事前の承諾なしに複製または転載することは禁じられています。

Informatica、Informatica ロゴ、および PowerCenter は、米国およびその他の国における Informatica LLC の商標または登録商標です。Informatica の商標の最新リストは、Web (<https://www.informatica.com/trademarks.html>) にあります。その他の企業名および製品名は、それぞれの企業の商標または登録商標です。

本ソフトウェアまたはドキュメントの一部は、次のサードパーティが有する著作権に従います（ただし、これらに限定されません）。Copyright DataDirect Technologies. All rights reserved. Copyright (C) Sun Microsystems. All rights reserved. Copyright (C) RSA Security Inc. All rights reserved. Copyright (C) Ordinal Technology Corp. All rights reserved. Copyright (C) Aandacht c.v. All rights reserved. Copyright Genivia, Inc. All rights reserved. Copyright Isomorphic Software. All rights reserved. Copyright (C) Meta Integration Technology, Inc. All rights reserved. Copyright (C) Intalio. All rights reserved. Copyright (C) Oracle. All rights reserved. Copyright (C) Adobe Systems Incorporated. All rights reserved. Copyright (C) DataArt, Inc. All rights reserved. Copyright (C) ComponentSource. All rights reserved. Copyright (C) Microsoft Corporation. All rights reserved. Copyright (C) Rogue Wave Software, Inc. All rights reserved. Copyright (C) Teradata Corporation. All rights reserved. Copyright (C) Yahoo! Inc. All rights reserved. Copyright (C) Glyph & Cog, LLC. All rights reserved. Copyright (C) Thinkmap, Inc. All rights reserved. Copyright (C) Clearpace Software Limited. All rights reserved. Copyright (C) Information Builders, Inc. All rights reserved. Copyright (C) OSS Nokalva, Inc. All rights reserved. Copyright Edifecs, Inc. All rights reserved. Copyright Cleo Communications, Inc. All rights reserved. Copyright (C) International Organization for Standardization 1986. All rights reserved. Copyright (C) ej-technologies GmbH. All rights reserved. Copyright (C) Jaspersoft Corporation. All rights reserved. Copyright (C) International Business Machines Corporation. All rights reserved. Copyright (C) yWorks GmbH. All rights reserved. Copyright (C) Lucent Technologies. All rights reserved. Copyright (C) University of Toronto. All rights reserved. Copyright (C) Daniel Veillard. All rights reserved. Copyright (C) Unicode, Inc. Copyright IBM Corp. All rights reserved. Copyright (C) MicroQuill Software Publishing, Inc. All rights reserved. Copyright (C) PassMark Software Pty Ltd. All rights reserved. Copyright (C) LogiXML, Inc. All rights reserved. Copyright (C) 2003-2010 Lorenzi Davide, All rights reserved. Copyright (C) Red Hat, Inc. All rights reserved. Copyright (C) The Board of Trustees of the Leland Stanford Junior University. All rights reserved. Copyright (C) EMC Corporation. All rights reserved. Copyright (C) Flexera Software. All rights reserved. Copyright (C) Jinfonet Software. All rights reserved. Copyright (C) Apple Inc. All rights reserved. Copyright (C) Telerik Inc. All rights reserved. Copyright (C) BEA Systems. All rights reserved. Copyright (C) PDFlib GmbH. All rights reserved. Copyright (C) Orientation in Objects GmbH. All rights reserved. Copyright (C) Tanuki Software, Ltd. All rights reserved. Copyright (C) Ricebridge. All rights reserved. Copyright (C) Sencha, Inc. All rights reserved. Copyright (C) Scalable Systems, Inc. All rights reserved. Copyright (C) jQWidgets. All rights reserved. Copyright (C) Tableau Software, Inc. All rights reserved. Copyright (C) MaxMind, Inc. All rights reserved. Copyright (C) TMate Software s.r.o. All rights reserved. Copyright (C) MapR Technologies Inc. All rights reserved. Copyright (C) Amazon Corporate LLC. All rights reserved. Copyright (C) Highsoft. All rights reserved. Copyright (C) Python Software Foundation. All rights reserved. Copyright (C) BeOpen.com. All rights reserved. Copyright (C) CNRI. All rights reserved.

本製品には、Apache Software Foundation (<http://www.apache.org/>) によって開発されたソフトウェア、およびさまざまなバージョンの Apache License（まとめて「License」と呼んでいます）の下に許諾された他のソフトウェアが含まれます。これらのライセンスのコピーは、<http://www.apache.org/licenses/> で入手できます。適用法にて要求されないか書面にて合意されない限り、ライセンスの下に配布されるソフトウェアは「現状のまま」で配布され、明示的あるいは黙示的かを問わず、いかなる種類の保証や条件も付帯することはありません。ライセンス下での許諾および制限を定める具体的文言については、ライセンスを参照してください。

本製品には、Mozilla (<http://www.mozilla.org/>) によって開発されたソフトウェア、ソフトウェア Copyright (c) The JBoss Group, LLC, all rights reserved、ソフトウェア Copyright (c) 1999-2006 by Bruno Lowagie and Paulo Soares および GNU Lesser General Public License Agreement のさまざまなバージョン (<http://www.gnu.org/licenses/lgpl.html> で参照できる場合がある) に基づいて許諾されたその他のソフトウェアが含まれています。資料は、Informatica が無料で提供しており、一切の保証を伴わない「現状渡し」で提供されるものとし、Informatica LLC は市場性および特定の目的の適合性の黙示の保証などを含めて、一切の明示的及び黙示的保証の責任を負いません。

製品には、ワシントン大学、カリフォルニア大学アーバイン校、およびバンダービルト大学の Douglas C.Schmidt および同氏のリサーチグループが著作権を持つ ACE (TM) および TAO (TM) ソフトウェアが含まれています。Copyright (C) 1993-2006, All rights reserved.

本製品には、OpenSSL Toolkit を使用するために OpenSSL Project が開発したソフトウェア (copyright The OpenSSL Project.All Rights Reserved) が含まれています。また、このソフトウェアの再配布は、<http://www.openssl.org> および <http://www.openssl.org/source/license.html> にある使用条件に従います。

本製品には、Curl ソフトウェア Copyright 1996-2013, Daniel Stenberg, <daniel@haxx.se>が含まれます。All rights reserved. 本ソフトウェアに関する許諾および制限は、<http://curl.haxx.se/docs/copyright.html> にある使用条件に従います。すべてのコピーに上記の著作権情報とこの許諾情報が記載されている場合、目的に応じて、本ソフトウェアの使用、コピー、変更、ならびに配布が有償または無償で許可されます。

本製品には、MetaStuff, Ltd.のソフトウェアが含まれます。Copyright 2001-2005 (C) MetaStuff, Ltd. All Rights Reserved.本ソフトウェアに関する許諾および制限は、<http://www.dom4j.org/license.html> にある使用条件に従います。

本製品には、Per Bothner のソフトウェアが含まれます。Copyright (C) 1996-2006.All rights reserved. お客様がこのようなソフトウェアを使用するための権利は、ライセンスで規定されています。<http://www.gnu.org/software/kawa/Software-License.html> を参照してください。

本製品には、OSSP UUID ソフトウェアが含まれます。Copyright (C) 2002 Ralf S. Engelschall, Copyright (C) 2002 The OSSP Project Copyright (C) 2002 Cable & Wireless Deutschland.本ソフトウェアに関する許諾および制限は、<http://www.opensource.org/licenses/mit-license.php> にある使用条件に従います。

本製品には、Boost (<http://www.boost.org/>) によって開発されたソフトウェア、または Boost ソフトウェアライセンスの下で開発されたソフトウェアが含まれます。本ソフトウェアに関する許諾および制限は、http://www.boost.org/LICENSE_1_0.txt にある使用条件に従います。

本製品には、University of Cambridge のが含まれます。Copyright (C) 1997-2007.本ソフトウェアに関する許諾および制限は、<http://www.pcre.org/license.txt> にある使用条件に従います。

本製品には、The Eclipse Foundation のソフトウェアが含まれます。Copyright (C) 2007.All rights reserved. 本ソフトウェアに関する許諾および制限は、<http://www.eclipse.org/org/documents/epl-v10.php> および <http://www.eclipse.org/org/documents/edl-v10.php> にある使用条件に従います。

本製品には、<http://www.tcl.tk/software/tcltk/license.html>、<http://www.bosrup.com/web/overlib/?License>、<http://www.stlport.org/doc/license.html>、<http://www.asm.ow2.org/license.html>、<http://www.cryptix.org/LICENSE.TXT>、<http://hsqldb.org/web/hsqLicense.html>、<http://httpunit.sourceforge.net/doc/license.html>、<http://jung.sourceforge.net/license.txt>、http://www.gzip.org/zlib/zlib_license.html、<http://www.openldap.org/software/release/license.html>、<http://www.libssh2.org>、<http://slf4j.org/license.html>、<http://www.sente.ch/software/OpenSourceLicense.html>、<http://fusesource.com/downloads/license-agreements/fuse-message-broker-v-5-3-license-agreement>、<http://antlr.org/license.html>、<http://aopalliance.sourceforge.net/>、<http://www.bouncycastle.org/licence.html>、<http://www.jgraph.com/jgraphdownload.html>、<http://www.jcraft.com/jsch/LICENSE.txt>、http://jotm.objectweb.org/bsd_license.html に基づいて許諾されたソフトウェアが含まれています。
<http://www.w3.org/Consortium/Legal/2002/copyright-software-20021231>、<http://www.slf4j.org/license.html>、<http://nanoxml.sourceforge.net/orig/copyright.html>、<http://www.json.org/license.html>、<http://forge.ow2.org/projects/javaservice/>、<http://www.postgresql.org/about/licence.html>、<http://www.sqlite.org/copyright.html>、<http://www.tcl.tk/software/tcltk/license.html>、<http://www.jaxen.org/faq.html>、<http://www.jdom.org/docs/faq.html>、<http://www.slf4j.org/license.html>、<http://www.iodbc.org/dataspace/iodbc/wiki/IODBC/License>、<http://www.keplerproject.org/md5/license.html>、<http://www.toedter.com/en/jcalendar/license.html>、<http://www.edankert.com/bounce/index.html>、<http://www.net-snmp.org/about/license.html>、<http://www.openmdx.org/#FAQ>、http://www.php.net/license/3_01.txt、<http://srp.stanford.edu/license.txt>、<http://www.schneier.com/blowfish.html>、<http://www.jmock.org/license.html>、<http://xsom.java.net>、<http://benalman.com/about/license/>、<https://github.com/CreateJS/EaselJS/blob/master/src/easeljs/display/Bitmap.js>、<http://www.h2database.com/html/license.html#summary>、<http://jsoncpp.sourceforge.net/LICENSE>、<http://jdbc.postgresql.org/license.html>、<http://protobuf.googlecode.com/svn/trunk/src/google/protobuf/descriptor.proto>、<https://github.com/rantav/hector/blob/master/LICENSE>、<http://web.mit.edu/Kerberos/krb5-current/doc/mitK5license.html>、<http://jibx.sourceforge.net/jibx-license.html>、<https://github.com/lyokato/libgeohash/blob/master/LICENSE>、<https://github.com/hjiang/jsonxx/blob/master/LICENSE>、<https://code.google.com/p/lz4/>、<https://github.com/jedisct1/libsodium/blob/master/LICENSE>、<http://one-jar.sourceforge.net/index.php?page=documents&file=license>、<https://github.com/EsotericSoftware/kryo/blob/master/license.txt>、<http://www.scala-lang.org/license.html>、<https://github.com/tinkerpop/blueprints/blob/master/LICENSE.txt>、<http://gee.cs.oswego.edu/dl/classes/EDU/oswego/cs/dl/util/concurrent/intro.html>、<https://aws.amazon.com/asl/>、<https://github.com/twbs/bootstrap/blob/master/LICENSE>、および <https://sourceforge.net/p/xmlunit/code/HEAD/tree/trunk/LICENSE.txt>。

本製品には、Academic Free License (<http://www.opensource.org/licenses/afl-3.0.php>)、Common Development and Distribution License (<http://www.opensource.org/licenses/cddl1.php>)、Common Public License (<http://www.opensource.org/licenses/cpl1.0.php>)、Sun Binary Code License Agreement Supplemental License Terms、BSD License (<http://www.opensource.org/licenses/bsd-license.php>)、BSD License (<http://opensource.org/licenses/BSD-3-Clause>)、MIT License (<http://www.opensource.org/licenses/mit-license.php>)、Artistic License (<http://www.opensource.org/licenses/artistic-license-1.0>)、Initial Developer's Public License Version 1.0 (<http://www.firebirdsql.org/en/initial-developer-s-public-license-version-1-0/>) に基づいて許諾されたソフトウェアが含まれています。

本製品には、ソフトウェア copyright (C) 2003-2006 Joe Walnes, 2006-2007 XStream Committers が含まれています。All rights reserved. 本ソフトウェアに関する許諾および制限は、<http://j.org/license.html> にある使用条件に従います。本製品には、Indiana University Extreme! Lab によって開発

されたソフトウェアが含まれています。詳細については、<http://www.extreme.indiana.edu/>を参照してください。

本製品には、ソフトウェア Copyright (C) 2013 Frank Balluffi and Markus Moeller が含まれています。All rights reserved. 本ソフトウェアに関する許諾および制限は、MIT ライセンスの使用条件に従います。

特許については、<https://www.informatica.com/legal/patents.html> を参照してください。

免責: 本文書は、一切の保証を伴わない「現状渡し」で提供されるものとし、Informatica LLC は他社の権利の非侵害、市場性および特定の目的への適合性の黙示の保証などを含めて、一切の明示的および黙示的保証の責任を負いません。Informatica LLC では、本ソフトウェアまたはドキュメントに誤りのないことを保証していません。本ソフトウェアまたはドキュメントに記載されている情報には、技術的に不正確な記述や誤植が含まれる場合があります。本ソフトウェアまたはドキュメントの情報は、予告なしに変更されることがあります。

NOTICES

この Informatica 製品（以下「ソフトウェア」）には、Progress Software Corporation（以下「DataDirect」）の事業子会社である DataDirect Technologies からの特定のドライバ（以下「DataDirect ドライバ」）が含まれています。DataDirect ドライバには、次の用語および条件が適用されます。

1. DataDirect ドライバは、特定物として現存するままの状態を提供され、商品性の保証、特定目的適合性の保証および法律上の瑕疵担保責任を含むすべての明示もしくは黙示の保証責任を負わないものとします。国または地域によっては、法律の強行規定により、保証責任の制限が禁じられる場合、強行規定の制限を受けるものとします。
2. DataDirect または第三者は、予見の有無を問わず発生した ODBC ドライバの使用に関するいかなる直接的、間接的、偶発的、特別、あるいは結果的損害に対して責任を負わないものとします。本制限事項は、すべての訴訟原因に適用されます。訴訟原因には、契約違反、保証違反、過失、厳格責任、詐称、その他の不法行為を含みますが、これらに限るものではありません。

本マニュアルの情報は、予告なしに変更されることがあります。このドキュメントで問題が見つかった場合は、infa_documentation@informatica.com までご報告ください。

Informatica 製品は、それらが提供される契約の条件に従って保証されます。Informatica は、商品性、特定目的への適合性、非侵害性の保証等を含めて、明示的または黙示的ないかなる種類の保証をせず、本マニュアルの情報を「現状のまま」提供するものとします。

はじめに

Web Services Hub をホストとする Web Services Provider および PowerCenter Web サービスに関する情報については、『*PowerCenter(R) Web Services Provider ガイド*』を参照してください。PowerCenter ワークフローを Web サービスにする方法を確認し、Web Services Hub で利用可能な Web サービスを使用するクライアントアプリケーションを作成する例を参照するには、このガイドを使用します。

Informatica のリソース

Informatica は、Informatica Network やその他のオンラインポータルを通じてさまざまな製品リソースを提供しています。リソースを使用して Informatica 製品とソリューションを最大限に活用し、その他の Informatica ユーザーや各分野の専門家から知見を得ることができます。

Informatica Network

Informatica Network は、Informatica ナレッジベースや Informatica グローバルカスタマサポートなど、多くのリソースへの入口です。Informatica Network を利用するには、<https://network.informatica.com> にアクセスしてください。

Informatica Network メンバーは、次のオプションを利用できます。

- ナレッジベースで製品リソースを検索できます。
- 製品の提供情報を表示できます。
- サポートケースを作成して確認できます。
- 最寄りの Informatica ユーザーグループネットワークを検索して、他のユーザーと共同作業を行えます。

Informatica ナレッジベース

Informatica ナレッジベースを使用して、ハウツー記事、ベストプラクティス、よくある質問に対する回答など、製品リソースを見つけることができます。

ナレッジベースを検索するには、<https://search.informatica.com> にアクセスしてください。ナレッジベースに関する質問、コメント、ご意見の連絡先は、Informatica ナレッジベースチーム (KB_Feedback@informatica.com) です。

Informatica マニュアル

Informatica マニュアルポータルでは、最新および最近の製品リリースに関するドキュメントの膨大なライブラリを参照できます。マニュアルポータルを利用するには、<https://docs.informatica.com> にアクセスしてください。

製品マニュアルに関する質問、コメント、ご意見については、Informatica マニュアルチーム (infa_documentation@informatica.com) までご連絡ください。

Informatica 製品可用性マトリックス

製品可用性マトリックス (PAM) には、製品リリースでサポートされるオペレーティングシステム、データベースなどのデータソースおよびターゲットが示されています。Informatica PAM は、<https://network.informatica.com/community/informatica-network/product-availability-matrices> で参照できます。

Informatica Velocity

Informatica Velocity は、Informatica プロフェッショナルサービスが開発したヒントとベストプラクティスのコレクションで、多数のデータ管理プロジェクトから得た実体験に基づいています。Informatica Velocity には、世界中の組織と連携してデータ管理ソリューションを計画、開発、デプロイ、管理する Informatica コンサルタントによる集合知を表しています。

Informatica Velocity リソースには、<http://velocity.informatica.com> からアクセスしてください。Informatica Velocity についての質問、コメント、またはアイデアがある場合は、ips@informatica.com から Informatica プロフェッショナルサービスにお問い合わせください。

Informatica Marketplace

Informatica Marketplace は、お使いの Informatica 製品を拡張したり強化したりするソリューションを検索できるフォーラムです。Marketplace で、Informatica デベロッパーやパートナーからの多数のソリューションを活用すれば、生産性を向上したり、プロジェクトでの実装時間を短縮したりできます。Informatica Marketplace は、<https://marketplace.informatica.com> からアクセスしてください。

Informatica グローバルカスタマサポート

電話または Informatica Network からグローバルサポートセンターに連絡できます。

各地域の Informatica グローバルカスタマサポートの電話番号は、Informatica Web サイト (<https://www.informatica.com/services-and-training/customer-success-services/contact-us.html>) を参照してください。

Informatica Network でオンラインサポートリソースを見つけるには、<https://network.informatica.com> にアクセスし、eSupport オプションを選択します。

Web サービスの概念

Web サービスの概念の概要

Web サービスは、Web 上で運用できる業務機能です。Web サービスには、標準の XML メッセージを媒体としてネットワークアクセスできる一連の動作が定義されています。PowerCenter Web Services Provider は、PowerCenter メタデータとデータ統合機能を統合し、Web サービスとして公開します。Integration Services と通信できるアプリケーションをあらゆる言語またはプラットフォームで作成できます。また、これらのアプリケーションを既存のコンポーネントや製品に組み込むこともできます。

Web サービスは XML、SOAP、WSDL といった、広く公開された規格に準拠しているため、メーカーが独自に開発したアプリケーションよりも優れた相互運用性を提供します。

Web サービスの例としては、株価、航空機の運航スケジュール、信用調査などの業務サービスが挙げられます。

Web サービスを実現するコンポーネントは、以下のとおりです。

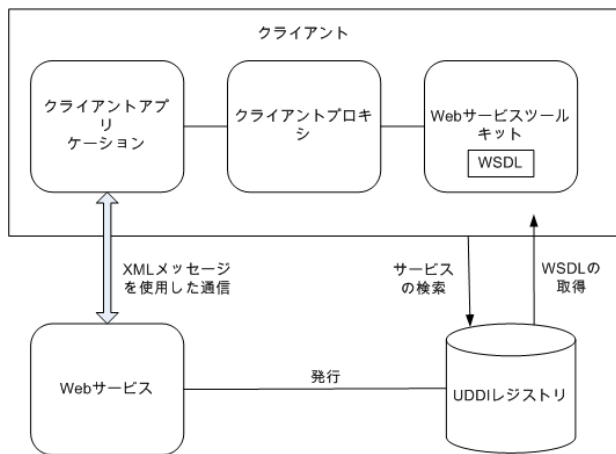
- **Simple Object Access Protocol (SOAP)**。SOAP は、Web サービス用の通信プロトコルです。これは、XML フォーマットを Web サービスメッセージ用に定義した仕様と言えます。
- **Web サービス記述言語 (WSDL)**。WSDL は、Web サービス操作を記述する XML 文書です。
- **レジストリ**。公開された Web サービスのディレクトリです。一部の Web サービスのプロバイダは、Universal Description, Discovery, and Integration (UDDI) のサービスを公開しています。すべての Web サービスが UDDI に登録されるとは限りません。

注: PowerCenter Web Services Provider では、UDDI レジストリは使用されません。

PowerCenter Web Services Provider の Web サービスクライアントを作成するには、インタフェースとする Web サービスを選択し、選択した Web サービスに対応した WSDL を取得します。Axis などの Web サービスツールキットを使用して、クライアントプロキシを生成します。クライアントプロキシには、Web サービスと接続して動作するのに必要なすべての関数呼び出しが含まれています。

Web サービスが提供する関数、Web サービスに必要なデータ、サービスの場所は、WSDL を調べれば分かります。WSDL は、サービスで使用する Web サービスインタフェースや演算について定義します。WSDL の情報を使用して、サービスを利用するためのクライアントアプリケーションを作成してください。

以下の図に、Web サービスの基本単位を示します。



Simple Object Access Protocol (SOAP)

SOAP は、Web サービス用の通信プロトコルです。Web サービスメッセージのフォーマットを定義しています。Java などのデータ構造を SOAP XML へ変換する方法は、SOAP エンコードによって SOAP 実行環境に通知されます。Web サービスとそのクライアント間の通信は、SOAP および WSDL によって定義されています。

SOAP メッセージは、次のセクションを含んでいます。

- **SOAP エンベロープ**。エンベロープは、メッセージの内容が含まれ、それを誰があるいは何が処理するか、あるいは、その処理を省略できるか否かといった、メッセージのフレームワークを定義します。
- **SOAP ヘッダ**。ヘッダは SOAP エンベロープの要素であり、SOAP メッセージに対し、分散化された形で機能を追加できるようにします。
- **SOAP ボディ**。SOAP ボディには、意図した受信者と情報を交換するための必須の情報を格納します。

ヘッダのエントリとして実装される拡張機能の代表的な例としては、認証およびトランザクション管理が挙げられます。SOAP ヘッダは、SOAP メッセージの本文に格納されたデータを処理する上での助けとなるものです。通常、ヘッダには認証またはトランザクションに関連した情報が格納されます。この情報がヘッダに格納されるのは、SOAP メッセージの本文を送信した団体、または、メッセージの処理対象に含まれるコンテキストはこの情報によって識別されるためです。

SOAP メッセージを作成したり解析したりする際には、SOAP ツールキットを使用します。このツールキットによって、ほかの言語の関数呼び出しが SOAP メッセージに変換されます。たとえば、Apache Axis ツールキットでは、Java の関数呼び出しが SOAP に変換されます。

SOAP を使用して、Web サービスを組織の内外にある各種プラットフォームに実装します。サポートされる関数呼び出しとパラメータは、SOAP の実装方法によって異なります。したがって、特定のツールキットで動作する関数が、必ずしも別のツールキットで動作するとは限りません。

Web サービス記述言語 (WSDL)

WSDL は Web サービスで使用するプロトコルやフォーマットについて記述した XML 文書です。

WSDL には、サービス要求の送信者にも受信者にも交換されるデータが理解できるように、Web サービスに渡されるデータが記述されています。WSDL 要素には、このほか、データ上で実行される操作の詳細も含まれているため、メッセージの受信者は処理方法を理解できます。また、プロトコルまたはトランスポートのバインドも含まれているので、送信者は送信方法を理解できます。

Web Services Hub Console 上の PowerCenter Web Services Provider でホストされる Web サービスの WSDL ファイルを表示し、ダウンロードすることができます。

関連項目：

- [「Web Services Hub Console の使い方」 \(ページ 17\)](#)

Web Services Provider について

Web Services Provider についての概要

Web Services Provider は、PowerCenter のワークフローおよびデータ統合機能に外部クライアントから Web サービス経由でアクセスできるようにする、PowerCenter Web サービスフレームワークのプロバイダエンティティです。

Web Services Provider は、以下のコンポーネントで構成されます。

- **Web Services Hub。** SOAP 標準を使用して要求の受信および Web サービスクライアントへの応答の送信を行う、PowerCenter ドメイン内のアプリケーションサービス。Web Services Hub は Integration Service および Repository Service とやり取りし、要求を処理して応答を生成します。
- **バッチ Web サービス。** Web Services Provider には、Integration Service プロセスおよびリポジトリメタデータにアクセスできる Web サービス操作のセットが用意されています。
- **リアルタイム Web サービス。** PowerCenter ワークフローを Web サービスとして有効にする場合は、リアルタイム Web サービスを作成します。PowerCenter ワークフローを Web サービスに変換すると、Web サービスクライアントからワークフローを実行できます。

Web Services Hub

Web Services Hub は、PowerCenter ドメイン内の Web サービスゲートウェイです。このゲートウェイを使用すると、クライアントアプリケーションは Web サービスの標準およびプロトコルを使用して機能にアクセスできます。Web Services Hub を使用すると、PowerCenter ワークフローを Web サービスとして有効にすることができます。また、PowerCenter プロセスのモニタリングおよびリポジトリ情報の取得も可能です。

Web Services Hub を使用すると、PowerCenter フレームワークの範囲内でデータ統合プロセスを実行できますが、要求および応答の処理には Web サービステクノロジーを使用します。Web Services Hub では、Web サービスクライアントから SOAP メッセージの形式で要求を受信し、そのメッセージを Integration Service に渡します。Integration Service は Repository Service と連携して要求を処理し、結果を Web Services Hub に送信します。Web Services Hub によって、Web サービスクライアントに SOAP メッセージの形式で応答が返信されます。

Web Services Hub には、Web サービスの管理および Web サービス用 WSDL ファイルの表示とダウンロードが可能な Web Services Hub Console が用意されています。WSDL ファイルを使用して、Web サービスにアクセスするクライアントアプリケーションを作成できます。

PowerCenter のインストールには Web Services Hub が含まれています。PowerCenter をインストールした後、ドメイン内のその他のアプリケーションサービスを有効にする場合と同様に、Informatica Administrator を使用して Web Services Hub を作成し、これを有効化します。

バッチ Web サービス

Web Services Provider に用意されている Web サービス操作を使用すると、ワークフローを実行およびモニタリングし、メタデータ情報にアクセスできます。この Web サービス操作は、総称してバッチ Web サービスと呼ばれます。バッチ Web サービス操作では、Web Services Hub に関連付けられたリポジトリ内のオブジェクトに関する情報を取得できます。また、Integration Service に接続して、ワークフローおよびタスクの実行を管理したり、ワークフローおよびセッションに関する情報を取得したりできます。

バッチ Web サービスは、以下のカテゴリに分類されます。

- **Data Integration Web サービス。**Data Integration Web サービスは、Integration Service に接続して and run or monitor PowerCenter ワークフローを実行またはモニタリングするために使用します。Data Integration Web Service には、Integration Service に関する詳細の取得、ワークフローのスケジュール設定および実行、ワークフローでのタスクの開始および停止、またはセッションに関する統計のモニタリングおよび取得が可能な操作が用意されています。
- **Metadata Web サービス。**Metadata Web Service には、PowerCenter リポジトリからメタデータを取得する操作が用意されています。Metadata Web Service を使用して、リポジトリでワークフローを実行およびモニタリングするのに必要なフォルダ、ワークフロー、ワークフロータスクなどのリポジトリオブジェクトに関する情報を取得します。

リアルタイム Web サービス

インストール後に初めて Web Services Hub を開始するときには、リアルタイム Web サービスは使用できません。リアルタイム Web サービスは、PowerCenter ワークフローを Web サービスとして公開する場合に作成します。Web サービスワークフローを実行するクライアントを作成し、ワークフロープロセスの結果を取得できます。Web サービスでは、1 つの SOAP メッセージ要求を受信して 1 つの SOAP メッセージ応答を生成します。

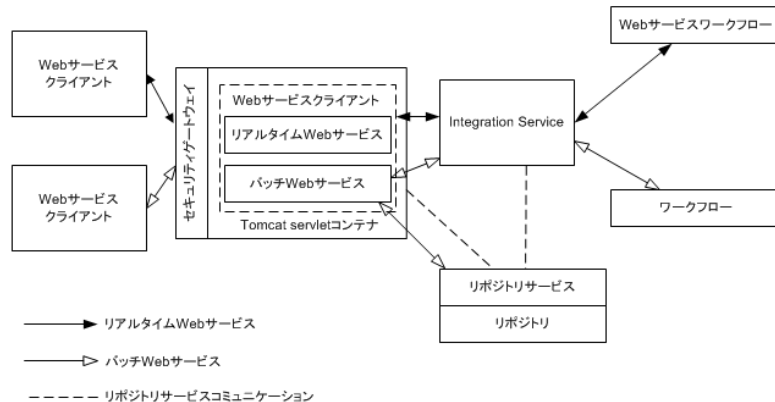
Web サービスクライアントからメッセージを受信してから変換し、それを PowerCenter がサポートする任意のターゲットに書き込むためのサービスマッピングを作成できます。また、Web サービスクライアントからメッセージ要求を受信し、データを変換して Web サービスクライアントに応答を返信するための、Web サービスのソース定義およびターゲット定義を持つ Web サービスマッピングを作成できます。ソース定義とターゲット定義はサービスの操作を表し、ソースはユーザー要求を、ターゲットは応答を定義します。

マッピングを作成した後は、Web サービスワークフローを作成して、Web サービスマッピングで定義されたプロセスを実行できます。Web サービスワークフローとは、Web サービスとして有効にされたワークフローです。Web サービスワークフローを設定し、ワークフローにセッションを追加します。ワークフローを保存すると、Web Services Hub によって Web Services Hub Console で Web サービスが発行されます。Integration Service はリクエストベースおよびワンウェイサービスの両方をパラレル処理することが可能です。

Web Services Provider のアーキテクチャ

Web Services Provider は、Web Services Hub と、Web Services Hub によってホストされるバッチ Web サービスおよびリアルタイム Web サービスで構成されます。Web Services Hub は Integration Service および Repository Service と連携し、Web サービス要求を処理します。

以下の図に、Web Services Provider のアーキテクチャを示します。



Web Services Hub は、リアルタイム Web サービスとバッチ Web サービスに対して、同様の方法で要求を処理します。

以下のプロセスは、Web Services Hub が Web サービス要求を処理する方法を示しています。

1. Web サービスクライアントでは、SOAP メッセージを Web Services Hub に送信して Web サービスを実行します。
2. バッチ Web サービスの場合、Web Services Hub はログイン中に生成されたセッション ID に基づいて Web サービスクライアントを認証します。

保護されたリアルタイム Web サービスの場合、Web Services Hub はユーザー名トークンに基づいて Web サービスクライアントを認証します。

3. Web Services Hub は、要求に対してメッセージ ID を生成します。

リアルタイム Web サービスに対する要求の場合、Web Services Hub は Integration Service にメッセージを送信します。

バッチ Web サービス操作に対する要求の場合、Web Services Hub は要求のタイプに基づいて Integration Service または Repository Service にメッセージを送信します。例えば、要求がワークフローの実行、または Integration Service の開始または停止の場合、Web Services Hub は Integration Service に処理のメッセージを送信します。要求がリポジトリ内にあるワークフローリストの取得の場合、Web Services Hub は Repository Service に処理のためにメッセージを送信します。

4. Integration Service または Repository Service が要求を処理します。

リアルタイム Web サービスに対する要求の場合、Integration Service は処理したデータを Web Services Hub に送信し、メッセージ ID を使用して要求を応答に関連付けます。

5. Web Services Hub は、SOAP 応答を Web サービスクライアントに送信します。

Integration Service と Web Services Hub は、プロセス全体を通じて Repository Service と通信します。

パフォーマンスおよび拡張性

1つのノード上で複数の Web Services Hub を実行できます。1つのノード上で複数の Web Services Hub を実行する場合、ノード上で実行できる Web サービスの数を増やして、リソースを最大限に活用します。

ドメイン内で Web サービスを実行するように Web Services Hub を設定すると、パフォーマンスを強化して柔軟性と拡張性を提供するための以下のオプションを使用できます。

- 複数のリポジトリと Web Services Hub との関連付け
- 複数の Web Services Hub とリポジトリとの関連付け
- Web サービスワークフローの複数インスタンスの実行
- グリッド上での Web サービスセッションまたはワークフローの実行

複数のリポジトリと Web Services Hub との関連付け

複数のリポジトリを Web Services Hub に関連付けることができます。複数のリポジトリを Web Services Hub に関連付ける場合、Web Services Hub は関連付けられたリポジトリのいずれかにある Web サービスワークフローを実行できます。これにより、同じ Web Services Hub を使用して異なるユーザーによって異なる時にアクセスされる Web サービスが実行できるようになり、Web Services Hub を最大限に活用できます。

リポジトリと複数の Web Services Hub Services との関連付け

リポジトリを複数の Web Services Hub に関連付けることができます。1つのリポジトリを複数の Web Services Hub Service に関連付けると、複数の Web Services Hub Service で同じ Web サービスを実行することができます。

サードパーティ製のロードバランサを使用して、サービス要求ロードが Web Services Hub Service 間でバランスをとるように Web Services Hub Service への要求を管理および分配します。プロダクション環境で使用されるハードウェアロードバランサは、Web サービスのパフォーマンスを最適化できます。管理者ツール内に Web Services Hub Service を作成する場合は、ロードバランサの URL を設定します。

バッチ Web サービスを使用する場合、ロードバランサは複数の Web Services Hub Service 間に要求を分配できません。Web Services Hub が Web サービスクライアントの認証に使用するセッション ID は、ログイン操作を起動する Web Services Hub に対してのみ有効です。同一または別のノード上の複数の Web Services Hub Service でバッチ Web サービス操作の同じセッション ID を使用することはできません。

Web サービスワークフローの複数インスタンスの実行

複数インスタンスで実行するようにワークフローを設定する場合、Web Services Hub は Web サービスの新規インスタンスを動的に開始して、できるだけ多くの Web サービス要求を処理できます。Web Services Hub は、Web サービス使用をモニタリングし、リソース使用状況および Web サービス処理時間を算出します。Web Services Hub が Web サービスへの要求を処理するためにかかる最大時間のしきい値を設定できます。処理時間がしきい値を上回ると、Web Services Hub は、Web サービスワークフローの別のインスタンスを開始して、新しい要求を処理します。

サービス要求の数が減ると、Web Services Hub は Web サービスインスタンスを動的にシャットダウンしてリソース使用を削減します。

関連項目：

- [「Web サービスワークフローの同時実行」 \(ページ 98\)](#)

グリッド上での Web サービスまたはワークフローの実行

PowerCenter ドメインにグリッドが含まれている場合、グリッド上で Web サービスワークフローを実行できます。グリッドを作成し、管理者ツールで Integration Service をグリッドに関連付けます。次に、Web サービスワークフローを実行する Integration Service を割り当てます。

クライアントアプリケーションから、グリッド上の Web サービスワークフローを実行するには、グリッドに関連付けられた Integration Service で Web サービスワークフローを実行します。

セッションをグリッド上で実行できるようにすることもできます。セッションをグリッド上で実行すると、Integration Service がセッションスレッドをグリッド内のノード間に分配します。セッションをグリッド上で実行するには、メッセージ ID を Web サービスのソースおよびターゲット定義に追加します。Integration Service は、メッセージ ID を使用してノード間で Web サービスの入力および出力メッセージを関連付けます。

Web Services Hub のセキュリティ

Web Services Hub には、以下のセキュリティレベルがあります。

- **暗号化。** Web Services Hub は、リポジトリへの接続に使用する環境設定ファイル内のリポジトリログイン情報を暗号化します。また、Web Services Hub をセキュアモードで実行し、Web サービスクライアント要求の暗号化に SSL プロトコルを使用することもできます。
- **認証。** バッチ Web サービスの場合、Web サービスクライアントアプリケーションは、Login 操作を呼び出してから他の操作を呼び出す必要があります。Web Services Hub はセッション ID に基づいて要求の認証を行います。

保護されたリアルタイム Web サービスの場合、Web Services Hub はユーザー名トークンに基づいて Web サービスクライアントを認証します。Web サービスクライアントでは、Web Services Hub に送信されるすべての SOAP 要求にユーザー名トークンを含める必要があります。ユーザー名トークンには、プレーンテキストパスワード、ハッシュパスワード、またはダイジェストパスワードを使用できます。

Web Services Hub では、パブリックなリアルタイム Web サービスの Web サービス要求を認証しません。

- **権限付与。** リポジトリにアクセスする Web サービスクライアントは、サービスを実行するためにフォルダに権限が必要です。保護されたリアルタイム Web サービスの場合、フォルダに適切な権限を持つ Web サービスクライアントは、サービス設定に基づいてそのフォルダでサービスを実行できます。例えば、サービスが実行できない場合、Web サービスクライアントはサービスを開始できませんが、Web サービスワークフローが稼働中の場合はサービスを起動できます。

関連項目：

- [「クライアント要求へのセキュリティの追加」 \(ページ 69\)](#)

Web Services Hub ログ

Web Services Hub は、サービスの初期化、タスクの実行、接続状態といったタスク関連の状態メッセージおよびエラーメッセージについてログを作成します。ログには、クライアントの IP アドレス、クライアントが起動するサービス、関連付けられたワークフローが含まれます。このログのエラーメッセージを調べることによって、問題を解決することができます。

管理者ツールでは、Web Services Hub のログを表示および設定できます。

注: Web Services Hub が要求を処理できない場合、SOAP 応答の fault 要素にもメッセージを書き込みます。

関連項目：

- [「SOAP フォールトの処理」 \(ページ 15\)](#)

ログの設定

PowerCenter ドメインのログマネージャは、Web Services Hub を含む、ドメイン内のすべてのサービスのすべてのログ機能を扱います。

管理者ツールでは、Web Services Hub ログのサイズや場所、ログに含めるエラーのレベルを設定できます。

ログの表示

Web Services Hub ログイベントは、Administration Console のログビューアで表示できます。ログイベントをフィルタして、Web Services Hub に対応するログイベントのみのリストを取得できます。ログビューアでログイベントを表示すると、ログマネージャはドメイン管理者が設定したログディレクトリに生成されたファイルからログイベントを表示します。

SOAP フォールトの処理

Web Services Hub は、SOAP フォールトメッセージとしてエラー応答を送信します。Web Services Hub は、以下のタイプのフォールト応答を生成できます。

- ユーザー定義フォールト
- システムフォールト

ユーザー定義フォールト

エラーデータをターゲットに送信するには、ターゲット定義でフォルトビューを定義します。Web サービスマッピング内の変換ロジックによりエラーデータがターゲットに送信される場合、Integration Service はフォールトターゲットにメッセージを書き込みます。特定のエラーを捕捉および解決するには、エラーデータをターゲットに送信します。たとえば、応答のデータタイプが文字列であることを期待しているとします。Web サービスワークフローから数値の応答が送信されたら、応答をフォールトターゲットに送信することができます。次に、応答を評価してエラーを解決できます。

システムフォールト

Web Services Hub にシステムエラーが発生した場合、エラーのタイプに基づいてフォールトメッセージが生成され、応答が Web サービスクライアントに送信されます。フォールトメッセージは、Web Services Hub がエラー発生時に実行するタスクに基づいています。

- Web Services Hub が SOAP 要求メッセージの header 要素を処理できない場合は、SOAP response header 要素の子要素内の、SOAP 要求メッセージのヘッダエントリに関するエラー情報が返されます。
- Web Services Hub は、SOAP 要求の header 要素にエラーを検出した場合、body 要素の処理を行いません。この要求に対する SOAP 応答では、SOAP ヘッダに header fault 要素が格納されるほか、detail 要素を除く SOAP fault 要素が返されます。
- Web Services Hub が body 要素の内容を処理できない場合、SOAP 応答メッセージの SOAP fault 要素にはエラー情報を含む detail 要素が格納されます。
- Web Services Hub は、以下のいずれかのシステムエラーが発生した場合に、detail 要素内にエラー情報と共に SOAP フォールト応答を生成します。
 - Integration Service が実行されていないため、Web Services Hub が入力メッセージを処理できません。
 - Web Services Hub がタイムアウトしました。
 - 保護された Web サービスでは、有効なユーザー名トークンは得られません。
- Web Services Hub は、次の場合に Web サービス要求に対して応答を返しません。
 - サービス要求の内容は、形式が不正であるか、構文エラーを生成します。
 - ワークフローが要求を除外します。

システムフォールトスキーマ

デフォルトでは、システムフォールトメッセージの内容には、プレフィックスとコード番号で構成されるメッセージコードだけでなく、メッセージテキストがあります。例えば、メッセージコード WSH_95002 は、空のワークフロー名を含む無効な要求に関連付けられています。

メッセージコードは、SOAP フォールトの detail 要素内の ErrorCode 要素であり、メッセージテキストは SOAP フォールトの faultstring 要素です。

SOAP フォールトヘッダ

Web Services Hub は、SOAP 応答ヘッダの header fault 要素でヘッダ関連のエラーを報告します。

この要素のスキーマを次に示します。

```
<ns1:HeaderFault xmlns:ns1="http://www.informatica.com/wsh">
  <ErrorCode>
    error code
  </ErrorCode>
  <ErrorMessage>
    error message
  </ErrorMessage>
</ns1:HeaderFault>
```

SOAP フォールトボディ

SOAP フォールトボディには、次のような下位要素が保持されています。

- **faultcode**。faultcode は、Web サービスクライアント側で発生したエラーか、Integration Service 側で発生したエラーかを示します。エラーは、メッセージの構造が正しくない場合に、Web サービスクライアントで発生します。
- **faultstring**。faultstring は、エラーの説明をします。faultstring の値は、Integration Service、Web Services Hub、または Repository Service のうちどこでエラーが発生したかを示します。
- **detail**。detail 要素はエラーコードを含むエラー情報を保持し、エラー明細要素は faultstring が Web Services Hub または Repository Service のエラーである場合の詳細エラー情報を提供します。

Web Services Hub は次の SOAP フォールトスキーマを使用します。

```
<SOAP-ENV: Fault>
  <faultcode> Client/Server </faultcode>
  <faultstring>Brief Description of Error</faultstring>
  <detail>
    <ns:WSHFaultDetails xmlns:ns="www.informatica.com/wsh">
      <ErrorCode>
        Error Code
      </ ErrorCode >
      <ExtendedDetails>
        Actual Error
      </ ExtendedDetails >
    </ns:WSHFaultDetails>
  </detail>
</SOAP-ENV: Fault>
```

Web Services Hub Console の使い方

Web Services Hub Console の使い方の概要

Web Services Hub Console は、Web Services Hub で使用可能なリアルタイム Web サービスおよびバッチ Web サービスの操作を表示してテストするために使用する PowerCenter アプリケーションです。Web Services Hub Console を使用して、以下の作業を実行します。

- **リアルタイム Web サービスのプロパティを表示します。** Web サービスが保護されているかどうかなど、Web サービスおよびプロパティの説明を表示できます。また、Web サービスを含むリポジトリおよびフォルダも確認できます。
- **リアルタイム Web サービスの WSDL を表示します。** WSDL をダウンロードするには、ハードディスク上のファイルに WSDL を保存します。
- **リアルタイム Web サービスをテストします。** 試行クライアントアプリケーションを使用し、有効な Web サービスを実行して Web Services Hub Console に応答を表示します。バッチ Web サービスの操作をテストすることもできます。
- **バッチ Web サービス操作の説明を表示します。** バッチ Web サービスの操作の説明を表示できます。試行アプリケーションを使用し、操作のパラメータを表示できます。
- **Data Integration および Metadata Web サービスの WSDL を表示します。** WSDL をダウンロードするには、ハードディスク上のファイルに WSDL を保存します。

- **バッチ Web サービス操作をテストします。** 試行アプリケーションを使用し、バッチ Web サービス操作を実行して Web Services Hub Console に応答を表示します。

注: Web Services Hub Console は認証の必要がありません。Web Services Hub Console にはログインなしでアクセスできます。セキュリティを確保するために、Web Services Hub はセキュリティ保護されたネットワーク環境内で実行してください。

Web Services Hub Console への接続

Web Services Hub Console には任意のブラウザから接続できます。

以下の URL のいずれかを使用して、Web Services Hub Console に接続します。

`http://<WebServicesHubHostName:PortNumber>/wsh`
`http://<WebServicesHubHostName:PortNumber>/PowerCenter`

コンテキスト名/wsh および/PowerCenter では、大文字と小文字が区別されます。

HTTP 上で実行する Web Services Hub のデフォルトのポート番号は 7333 です。HTTPS によるセキュリティ保護された接続を使用するように Web Services Hub を設定することもできます。HTTPS 上で実行する Web Services Hub のデフォルトのポート番号は 7343 です。管理者ツールで Web Services Hub を作成する場合は、ポート番号を設定できます。

管理者ツールから Web Services Hub Console に接続することもできます。Web Services Hub の詳細を表示して、サービス URL をクリックします。Web Services Hub Console に接続するには、Web Services Hub を有効にする必要があります。

Web Services Hub Console について

Web Services Hub Console は、以下の節で構成されています。

- **ナビゲータ。**ナビゲータには、Web Services Hub Console 上に表示できるサービスのタイプが表示されます。
- **Web Services または操作。**リアルタイム Web サービスの場合、[Web サービス] セクションに有効または無効な Web サービスが表示されます。バッチ Web サービスの場合、Metadata Web Service および Data Integration Web Service で使用可能な演算が [操作] セクションに表示されます。
[Web サービス] セクションでは、Web サービスのテストまたは Web サービスの WSDL の表示を実行できます。
[操作] セクションでは、バッチ Web サービス操作のテスト、またはバッチ Web サービス WSDL の表示および保存を実行できます。
- **説明。**[説明] セクションには、ナビゲータで選択した Web サービスのタイプに関する情報が表示されます。
- **プロパティ。**[プロパティ] セクションには、[Web サービス] または [操作] セクションで選択した Web サービスまたは Web サービス操作のプロパティが表示されます。

ナビゲータ

ナビゲータでは、スクロールして、情報を表示する Web サービスのタイプを選択できます。コンソールの他のセクションに表示される情報は、ナビゲータで選択した Web サービスのタイプに基づいて異なります。

[Web サービス] および [操作] セクション

Web Services Hub Console には、ナビゲータで選択した Web サービスのタイプに応じて [Web サービス] セクションまたは [操作] セクションが表示されます。

ナビゲータで [有効な Web サービス] または [無効な Web サービス] を選択すると、[Web サービス] セクションに Web Services Hub 上で実行するリアルタイム Web サービスに関する情報が表示されます。

ナビゲータで [メタデータ Web サービス] または [データ統合 Web サービス] を選択すると、[操作] セクションに Web Services Hub 上で使用可能なバッチ Web サービス操作が表示されます。

Web サービスまたは操作のリストは、ソートできます。Web サービスまたは操作のリストをソートするには、ソートするカラムのラベルをクリックします。Web Services Hub Console には、Web サービスまたは操作がクリックしたカラムに基づいてアルファベット順に一覧表示されます。カラムラベルの横の矢印は、リストのソート順（昇順または降順）を示します。

試行アプリケーションを使用し、[Web サービス] および [操作] セクションに表示された Web サービス操作をテストできます。Web サービス操作をテストするには、Web サービス操作の入力メッセージにパラメータの値を入力して応答を表示します。

[Web サービス] セクションには、Web サービスの WSDL を表示できます。[操作] セクションには、バッチ Web サービスの WSDL を表示できます。WSDL は、操作ではなくメタデータ Web サービスまたはデータ統合 Web サービスに対して発行されます。操作を選択して WSDL をクリックすると、Web Services Hub にはメタデータ Web サービスまたはデータ統合 Web サービスの WSDL が表示されます。WSDL を使用して、リアルタイム Web サービスまたはバッチ Web サービス操作を呼び出すクライアントアプリケーションを記述します。

[Web サービス] セクション

Web Services Hub Console にリアルタイム Web サービスを表示するには、Web サービスワークフローを作成する必要があります。Web サービスワークフローを表示できるように設定すると、Web Services Hub が Web Services Hub Console 上で Web サービスおよび WSDL を発行します。

ナビゲータで [有効な Web サービス] または [無効な Web サービス] を選択すると、[Web サービス] セクションに、Web Services Hub Console に表示するよう設定されたリアルタイム Web サービスのリストが表示されます。

リポジトリ内のオブジェクトを管理する特権を持っている場合は、リポジトリに関連付けられているすべての Web サービスを表示できます。他のユーザーが作成した Web サービスは、表示できますが実行できません。例えば、TestRepo リポジトリのランタイムオブジェクトの作成、編集、および削除特権を持っているとします。この場合、TestRepo リポジトリに関連付けられている Web Services Hub の Web Services Hub Console では、TestRepo リポジトリ内のすべての Web サービスを表示できます。他のユーザーが作成した TestRepo リポジトリ内の Web サービスは、表示できますが実行できません。

次の表に、[Web サービス] セクションで使用できるオプションを示します。

ラベル	説明
Try-It (試行)	選択した Web サービスのテストに使用できるクライアントアプリケーション。クリックすると、選択した Web サービスを実行できます。 無効な Web サービスには使用できません。
WSDL	選択した Web サービスの WSDL。クリックすると、選択した Web サービスの WSDL ファイルを表示できます。セクションの一番上、または選択した Web サービスと同じ行にある WSDL ボタンをクリックできます。WSDL をダウンロードするには、WSDL をローカルマシンに表示および保存します。 無効な Web サービスには使用できません。
検索	Web サービスを検索します。検索したいテキストの文字列を入力して、[実行] をクリックします。[Web Services (Web サービス)] セクションに、そのテキストを含む Web サービス名、リポジトリ名、またはワークフロー名が一覧表示されます。
サービス名	Web Services Hub 上で実行できる Web サービスの名前。
リポジトリ名	Web サービスに関連付けられたリポジトリの名前。
ワークフロー名	Web サービスを構成するワークフローの名前。

[操作] セクション

[操作] セクションには、Web Services Hub で使用可能なバッチ Web サービス操作が一覧表示されます。これらの操作をクライアントアプリケーションから呼び出して、ワークフローの実行およびモニタリング、PowerCenter メタデータへのアクセスを実行できます。

ナビゲータで Metadata Web Service を選択すると、[操作] セクションに使用可能な Metadata Web Service 操作のリストが表示されます。ナビゲータで Data Integration Web Service を選択すると、[操作] セクションに使用可能な Data Integration Web Service 操作のリストが表示されます。

以下の表に、[操作] セクションのアイコンと情報を示します。

ラベル	説明
試行	選択した操作をテストするアプリケーション。選択した Metadata Web Service または Data Integration Web Services 操作をクリックして実行します。
WSDL	Metadata または Data Integration Web Service の WSDL。クリックすると、Web サービスの選択したタイプの Web Services Hub によって発行される WSDL を表示できます。ナビゲータで Metadata Web Service を選択して [WSDL] をクリックすると、Metadata Web Service の WSDL が表示されます。ナビゲータで Data Integration Web Service を選択して [WSDL] をクリックすると、Data Integration Web Service の WSDL が表示されます。WSDL をダウンロードするには、WSDL をローカルマシンに表示および保存します。
検索	対象操作を検索します。検索したいテキストの文字列を入力して、[実行] をクリックします。[Operations (操作)] セクションに、テキストを含む操作名または説明が一覧表示されます。
操作名	Metadata Web Service または Data Integration Web Services 操作の名前。
説明	Metadata Web Service または Data Integration Web Services 操作の説明。

【プロパティ】 セクション

【プロパティ】 セクションには、【Web サービス】 または 【操作】 セクションで選択した Web サービスまたは Web サービス操作に関する情報が表示されます。

リアルタイム Web サービスの【プロパティ】 セクション

【Web サービス】 セクションで有効または無効なリアルタイム Web サービスを選択すると、選択した Web サービスのプロパティが【プロパティ】 セクションに表示されます。

以下の表に、リアルタイム Web サービスのプロパティを示します。

プロパティ	説明
サービス名	Web サービスの名前。
ドメイン名	Web Services Hub を含む PowerCenter ドメインの名前。
リポジトリ名	Web サービスワークフローを含むリポジトリ。
フォルダー名	Web サービスワークフローを含むフォルダの名前。
ワークフロー名	Web サービスに関連付けられたワークフローの名前。
説明	Web サービスの説明。
実行可能	Web サービスをクライアントアプリケーションによって開始できるかどうかを指定します。 True の場合、Web サービスクライアントは、Web サービスワークフローの開始、またはワークフロー実行中の Web サービスの起動を実行できます。 False の場合、Web サービスクライアントはワークフロー実行中に Web サービスを起動できますが、ワークフローの開始はできません。
保護	Web サービスが保護されているか公開されているかを示します。 True の場合、Web サービスクライアント要求が認証を渡す必要があります。SOAP 要求では、有効なユーザー名トークンがヘッダに含まれている必要があります。 False の場合、いずれの Web サービスクライアントも認証なしで Web サービス要求を実行できます。
一方向	Web サービスが一方向マッピングであるか要求/応答マッピングであるかを指定します。

バッチ Web サービスの【プロパティ】 セクション

【操作】 セクションで Metadata Web Service 操作または Data Integration Web Service 操作を選択すると、選択した Web サービス操作の名前および説明が【プロパティ】 セクションに表示されます。

Web サービスのテスト

試行アプリケーションは、Web Services Hub Console に表示されたリアルタイムまたはバッチ Web サービス操作を実行するために使用できるクライアントアプリケーションです。試行アプリケーションを使用すると、有効な Web サービス操作をテストし、Web Services Hub Console 上に結果を表示できます。

入力メッセージに必要なパラメータが確かでない場合、または特定の入力メッセージへの応答を表示する場合は、試行アプリケーションを使用できます。

試行アプリケーションを使用すると、WSDL をダウンロードしてクライアントアプリケーションのクライアントプロキシクラスを生成することなく、Web サービスを実行するか操作を呼び出すことができます。コンソール上に応答を表示して、クライアントアプリケーションで Web サービスから応答をどのように処理すべきかを判断できます。

有効なリアルタイム Web サービスアプリケーションまたはバッチ Web サービス操作をテストできます。SOAP アタッチメントが含まれる WSDL を使用した Web サービスのテストには、試行アプリケーションを使用できません。

保護されたリアルタイム Web サービスには、認証が必要です。保護された Web サービス操作をテストするには、PowerCenter リポジトリにログインする有効なユーザー名トークンを指定します。

入力メッセージ

試行アプリケーションには、Web サービス要求を作成するための 2 つの方法が用意されています。

- XML 入力
- フォーム入力

要求の要件に最適な方法を使用します。例えば、複数回出現要素が要求に含まれている場合は、XML 入力を使用して要求メッセージを作成します。

XML 入力

[XML 入力] タブを選択すると、Web Services Hub にはサービス要求操作の実行に必要な要素を含む SOAP 入力メッセージが表示されます。SOAP メッセージの要素の値を入力します。また、Web Services Hub Console の外で SOAP メッセージを作成して、[XML 入力] セクションにペーストすることができます。

Web Services Hub では、SOAP 入力メッセージを使用して Web サービスを実行します。応答を SOAP 出力メッセージとして表示します。

フォーム入力

[フォーム入力] タブを選択すると、Web Services Hub に Web サービス要求のパラメータリストが表示されます。パラメータの値を入力します。Web サービス要求に複合タイプの要素が含まれる場合は、[フォーム入力] タブに入力パラメータが正しい階層で表示されます。

Web Services Hub では、入力されたパラメータ値を使用して SOAP 入力メッセージを作成し、Web サービスを実行します。応答を SOAP 出力メッセージとして表示します。

パブリック Web サービスまたはバッチ Web サービス操作のテスト

パブリックまたはバッチ Web サービスをテストするには、Web サービス操作を選択して、Web サービス操作の入力メッセージにパラメータの値を入力します。

パブリック Web サービスまたはバッチ Web サービス操作をテストする手順

1. [Web サービス] または [操作] セクションで、有効なリアルタイム Web サービスまたは操作を選択します。

2. [試行] をクリックします。

試行アプリケーションウィンドウに、テスト可能な Web サービス操作のリストと、試行アプリケーションの実行方法に関する手順が表示されます。

3. テストする操作を選択します。

試行アプリケーションウィンドウに、入力メッセージのパラメータが表示されます。

4. [XML 入力] タブをクリックし、SOAP メッセージフォーマットで入力メッセージを入力します。

または、[フォーム入力] タブをクリックし、パラメータ入力形式で入力パラメータを入力します。

5. パラメータの値を入力します。

WSDL には、ユーザー定義データタイプを指定できます。フォールト応答を回避するために、データタイプに従ってパラメータの値を入力してください。

6. [送信] をクリックします。

Web Services Hub は、Web サービス操作を実行して SOAP メッセージ応答およびメッセージを表示し、成功または失敗を示します。

7. パラメータをクリアして新しい値を入力するには、[リセット] をクリックします。

8. Web ブラウザの [閉じる] ボタンをクリックして、試行アプリケーションウィンドウを終了し、Web Services Hub Console のメインページに戻ります。

保護されたリアルタイム Web サービスのテスト

保護されたリアルタイム Web サービスをテストするには、SOAP ヘッダに有効なユーザー名トークンを含めます。[フォーム入力] タブにユーザー名およびパスワードを入力するか、[XML 入力] タブでユーザー名トークンのすべての要素が含まれるように SOAP メッセージを変更します。

[フォーム入力] タブまたは [XML 入力] タブで、プレーンテキストパスワードまたはハッシュパスワードを使用して、保護された Web サービスをテストできます。保護された Web サービスをハッシュパスワードでテストするには、Web サービスをテストする前に、MD5 または SHA-1 ハッシュ関数でパスワードを暗号化します。暗号化は Base64 で符号化する必要があります。取得されたハッシュ値を Web サービスのパスワードとして使用します。

[XML 入力] タブでダイジェストパスワードを使用して、保護された Web サービスをテストできます。保護された Web サービスをダイジェストパスワードを使用してテストするには、ダイジェストパスワードに対して UsernameToken 要素で要求されるパスワード属性および要素を追加します。

保護された Web サービスをテストする手順

1. リアルタイム Web サービスの [Web サービス] セクションで、実行対象の保護された Web サービスを選択して、[試行] をクリックします。
2. 試行アプリケーションウィンドウで、保護された Web サービスの操作を選択します。
3. [フォーム入力] を使用して Web サービス操作をテストするには、[フォーム入力] タブを選択します。

[SOAP ヘッダ] セクションで、ユーザー名およびプレーンテキストパスワードまたはハッシュパスワードを入力します。

[SOAP ボディ] セクションで、保護された Web サービスが要求するパラメータの値を入力します。
または

XML 入力を使用して Web サービス操作をテストするには、[XML 入力] タブを選択して、UsernameToken 要素を更新します。

プレーンテキストパスワードまたはハッシュパスワードを使用する保護された Web サービスをテストするには、Username およびパスワード子要素内の値 *[string]* を有効なユーザー名とパスワードに置き換えます。

```
<UsernameToken>
  <Username>[string]</Username>
  <Password>[string]</Password>
</UsernameToken>
```

ダイジェストパスワードを使用する保護された Web サービスをテストするには、Username 要素内の値 *[string]* を有効なユーザー名に置き換えます。パスワード要素を更新して、適切な値の Nonce および Created 要素を追加します。

```
<UsernameToken>
  <Username>[string]</Username>
  <Password Type="PasswordDigest">[string]</Password>
  <Nonce>[NonceValue]</Nonce>
  <Created>[RequestCreationTimestamp]</Created>
</UsernameToken>
```

UsernameToken 要素の詳細については、[「SOAP リクエストの UsernameToken」 \(ページ 69\)](#)を参照してください。

[SOAP ボディ] セクションで、保護された Web サービスが要求するパラメータの値を入力します。

4. [送信] をクリックします。

Web Services Hub は、保護された Web サービスを実行して、コンソール上に SOAP メッセージ応答を表示します。

5. Web ブラウザの [閉じる] ボタンをクリックして、試行アプリケーションウィンドウを終了し、Web Services Hub Console のメインページに戻ります。

バッチ Web サービスの操作

バッチ Web サービスの操作の概要

Batch Web Services の演算を使用すると、既存のワークフローやタスクのスケジュール、開始、停止を設定できます。セッション統計やパフォーマンスデータを取得できます。ワークフローやセッションのログを取得できます。

バッチ Web サービスは、別々の WSDL に定義された以下のグループのサービスで構成されます。

- **Metadata Web サービス。** Metadata Web サービスの操作は、Web Services Hub Console の [バッチ Web サービス] ページにある Metadata WSDL に定義されています。
- **Data Integration Web サービス。** Data Integration Web サービスの操作は、Web Services Hub Console の [バッチ Web サービス] ページにある Data Integration WSDL に定義されています。

この章では、バッチ Web サービスで可能な操作について説明します。この操作に関する要求と応答の XML ドキュメントについて詳しい情報が必要な場合は、WSDL ファイルを参照してください。

注: バッチ Web サービス操作の呼び出しで取得されたログセグメントは、Integration Service コードページまたは UTF-16LE で表示されます。

メタデータ Web サービスの操作

Metadata Web サービスの操作を使用して、Web Services Hub に関連付けられた PowerCenter リポジトリからメタデータを取得します。

この操作を利用して、リポジトリにログインして以下のリポジトリオブジェクトのリストを取得できます。

- Web Services Hub に関連付けられているリポジトリ内のすべてのフォルダ
- フォルダ内のすべてのワークフロー
- ワークフロー内のすべてのワークレットおよびセッションタスク
- リポジトリに関連付けられているすべての Integration Service
- Web Services Hub に関連付けられているすべてのリポジトリ

ここでは、Metadata Web サービスで可能なすべての操作を示します。

getAllServers

この操作を使用して、リポジトリに関連付けられたすべての Integration Service の名前を取得します。

1 つ以上の Integration Service をリポジトリに関連付けて、ワークフローやセッションを実行することができます。複数の Integration Service の環境では、ユーザーが Integration Service を簡単に識別できるように、関連するサービスにそれぞれわかりやすいサービス名を付けることが重要です。リポジトリに関連付けられた各 Integration Service のサービス名と、ホスト名およびポート番号の組み合わせは、リポジトリに関連付けられたサービスに対して一意であることが必要です。

この操作は、指定されたリポジトリに関連付けられたすべての Integration Service の名前を返します。

SessionID パラメータを指定して getAllServers 操作を呼び出します。リポジトリにログインした後にセッション ID が生成されます。

GetAllFolders

この演算を使用してリポジトリ内のすべてのフォルダを取得します。

SessionID パラメータを指定して getAllFolders 操作を呼び出します。リポジトリにログインした後にセッション ID が生成されます。

getAllRepositories

この演算を使用して、Web Services Hub に関連するすべてのリポジトリを表示します。

getAllRepositories 操作にはパラメータがありません。

Web Services Hub クライアントアプリケーションでリポジトリを使用するには、Web Services Hub にリポジトリを関連付ける必要があります。管理者ツールを使用して、Web Services Hub にリポジトリを関連付けます。

注: getAllRepositories 演算は特定のリポジトリと関連付けられていないため、演算を使用するためにリポジトリにログインする必要はありません。getAllRepositories 演算は、ログイン演算を呼び出さなくても呼び出すことができます。

getAllTaskInstances

この操作を使用して、指定した深さのワークフローに含まれるすべてのワークレットおよびセッションタスクのインスタンスについての情報を取得します。リポジトリにログインしてから、この操作を呼び出す必要があります。

以下の表に、この操作のパラメータを示します。

パラメータ名	必須/オプション	説明
SessionID	必須	ログイン後に生成されたセッション ID。
Depth	必須	タスクインスタンスに関する情報を取得するワークフロータスク階層内のレベル数。
WorkflowName	必須	ワークレットおよびセッションが含まれるワークフローの名前。
FolderName	必須	ワークフローが格納されているフォルダの名前。
IsValid	オプション	有効または無効なタスクインスタンスを取得するかどうかを指定します。有効なタスクインスタンスを取得するには True に設定します。

getAllWorkflows

この操作を使用して、フォルダ内のすべてのワークフローに関する情報を取得します。ワークフローとは、セッション、メール通知、およびシェルコマンドといったタスクをどのように実行するかを Integration Service に示す指示のセットです。ワークフローの情報には、ワークフロー名やワークフローが存在するフォルダ名のほか、ワークフローが有効かどうかの情報が含まれます。

リポジトリにログインしてから、この操作を呼び出す必要があります。

以下の表に、この操作のパラメータを示します。

パラメータ名	必須/オプション	説明
SessionID	必須	ログイン後に生成されたセッション ID。
名前	必須	ワークフローが格納されているフォルダの名前。

ログイン

この操作を使用して、リポジトリにログインします。ログイン操作では、指定したリポジトリのユーザー名およびパスワードを認証します。クライアントアプリケーションは、他の演算を呼び出す前に、この演算を呼び出す必要があります。ログイン演算を呼び出した後、Web サービスのクライアントアプリケーションは任意の Batch Web Services の演算を呼び出すことができます。

ログイン演算には、リポジトリ名、ユーザー名、およびパスワードが必要です。それによって、暗号化されたセッション ID が返されます。リポジトリは、Web Services Hub と同じドメインにある必要があります。

以下の表に、この操作のパラメータを示します。

パラメータ名	必須/オプション	説明
RepositoryName	必須	ログインするリポジトリの名前。
UserName	必須	リポジトリにログインするために使用するユーザー名。
パスワード	必須	リポジトリにログインするために使用するユーザーアカウントのパスワード。
RepositoryDomainName	オプション	リポジトリサービスのドメイン名。
UserNameSpace	条件付	リポジトリにログインするために使用するユーザーアカウントのセキュリティドメイン。PowerCenter ドメインに複数のセキュリティドメインが存在する場合は必須。

セッションの有効期限

セッション ID は、非アクティブ状態の後で期限切れになります。Web Services Hub の **[SessionExpiryPeriod]** 詳細プロパティを設定し、セッションが期限切れとなるまでの非アクティブな状態の時間の長さを指定します。Web Services Hub の詳細プロパティは、管理者ツールで設定できます。

Web Services Hub は、セッションの開始時にユーザー名とパスワードをキャッシュします。Web Services Hub は、セッションが期限切れとなるまで、キャッシュされたユーザー名とパスワードをログイン認証に使用します。ユーザーアカウントを変更しても、セッションが期限切れになるまで認証には影響しません。ユーザーアカウントを削除または無効化しても、セッションが期限切れになるまで、ユーザーアカウントは削除または無効化されません。ユーザーアカウントがロックされた場合、ロックされたユーザーは、セッションの期限が切れるまで要求を送信できます。

注: **SessionExpiryPeriod** の値が大きい場合は、ユーザーアカウントの変更が有効になるまでしばらく時間がかかります。セキュリティ違反を防止するため、セッション内で使用中のユーザーアカウントを削除または変更しないでください。

ログアウト

この操作を使用して、リポジトリからログアウトします。Logout 操作を使用して、リポジトリおよび Integration Service からユーザーを切断します。クライアントアプリケーション実行の最後にこの操作を呼び出してリソースを解放し、Repository Service と Web Services Hub プロセスで起こりうるメモリリークを防ぎます。

SessionID パラメータを指定して Logout 操作を呼び出します。リポジトリにログインした後にセッション ID が生成されます。

Data Integration Web サービスの操作

Data Integration Web サービス操作で、以下の作業を実行できます。

- **Repository Service への接続。** リポジトリにログインおよびログアウトするには、次の操作を使用します。
 - ログイン

- ログアウト
- **Integration Service に関する詳細への接続および取得。** 次の操作を使用して、Integration Service が実行されていることを確認し、Integration Service に関する情報を取得できます。
 - pingDIServer
 - getDIServerProperties
 - initializeDIServerConnection (廃止)
 - deinitializeDIServerConnection (廃止)
- **ワークフローのスケジュール設定および実行。** 以下の操作を使用して、ワークフローの実行を管理できます。
 - startWorkflow
 - startWorkflowEx
 - stopWorkflow
 - scheduleWorkflow
 - startWorkflowFromTask
 - unscheduleWorkflow
 - waitTillWorkflowComplete
- **ワークフロー内のタスクの開始および停止。** 以下の操作を使用して、ワークフロー内のタスクを管理できます。
 - recoverWorkflow
 - resumeWorkflow (廃止)
 - startTask
 - stopTask
 - waitTillTaskComplete
- **セッションの統計の取得。** 次の操作を使用して、セッションまたはワークフローの実行に関する詳細を取得できます。
 - getNextLogSegment (廃止)
 - getSessionLog
 - getSessionPerformanceData
 - getSessionStatistics
 - getTaskDetails
 - getTaskDetailsEx
 - getWorkflowDetails
 - getWorkflowDetailsEx
 - getWorkflowLog
 - monitorDIServer
 - startSessionLogFetch (廃止)

- startWorkflowLogFetch（廃止）

Data Integration Web サービスの操作は、di.wsdl で定義されています。

deinitializeDIserverConnection

廃止された演算。クライアントアプリケーションと Integration Service の接続を明示的に解除する必要はありません。Logout 操作を使用して、クライアントアプリケーションによって取得された Integration Service への接続を開放し、cleanup 操作を実行します。

この操作には、initializeDIserverConnection SessionID パラメータが必要です。リポジトリにログインした後にセッション ID が生成されます。

getDIserverProperties

この演算を使用して Integration Service のプロパティを取得します。リポジトリにログインしてから、この操作を呼び出す必要があります。

Integration Service のプロパティには、以下の情報が含まれています。

- Integration Service 名
- Integration Service のバージョン
- 製品名
- Integration Service の起動時刻
- Integration Service に関連付けられているリポジトリの名前。
- データ移動モード（ASCII または Unicode）
- Integration Service がマッピングをデバッグできるかどうか

以下の表に、この操作のパラメータを示します。

パラメータ名	必須/オプション	説明
SessionID	必須	ログイン後に生成されたセッション ID。
ServiceName	必須	プロパティを取得する Integration Service の名前。
DomainName	オプション	Integration Service のドメイン名。

getNextLogSegment

廃止された演算。この操作では、セッションログまたはワークフローログの情報を単位で取得します。

getNextLogSegment 演算は、セッションログまたはワークフローログの一部を返します。リポジトリにログインしてから、この操作を呼び出す必要があります。

この演算は、startSessionLogFetch 演算または startWorkflowLogFetch 演算と共に使用します。ログの最後に達するまで、startSessionLogFetch 演算または startWorkflowLogFetch 演算によって生成されるログハンドルを使用して、getNextLogSegment 演算を呼び出します。

1 つの操作のセッションログ情報を取得するには、getSessionLog 操作を使用します。1 つの操作のワークフローログ情報を取得するには、getWorkflowLog 操作を使用します。

以下の表に、この操作のパラメータを示します。

パラメータ名	必須/オプション	説明
SessionID	必須	ログイン後に生成されたセッション ID。
LogHandle	必須	startSessionLogFetch 操作または startWorkflowLogFetch 操作によって生成されたログ ID。startSessionLogFetch 操作または startWorkflowLogFetch 操作を呼び出した後、この操作を呼び出す必要があります。
タイムアウト	必須	この操作中に、クライアント要求が Web Services Hub への接続を保持する時間。タイムアウト期間内に操作が完了しない場合、この操作は失敗します。タイムアウト期間は、startSessionLogFetch 操作または startWorkflowLogFetch 操作を呼び出したときから開始されます。

getSessionLog

この操作を使用して、セッションログ内のすべての情報を 1 回の操作で取得します。リポジトリにログインしてから、この操作を呼び出す必要があります。

サービスセッションが実行されると、Integration Service はセッションログにプロセスの初期化処理、セッションの検査、reader スレッドおよび writer スレッドの SQL コマンドの作成、検出したエラー、ロード要約などの情報を書き込みます。セッションログ内の詳細情報の量は、設定されているトレースレベルによって決まります。getSessionLog 演算は、セッションログの情報を返します。

セッションログ情報を単位で取得するには、getNextLogSegment 操作を使用します。

以下の表に、この操作のパラメータを示します。

パラメータ名	必須/オプション	説明
SessionID	必須	ログイン後に生成されたセッション ID。
FolderName	必須	ワークフローが格納されているフォルダの名前。
WorkflowName	必須	セッションを含むワークフローの名前。
TaskInstancePath	必須	タスクの場所を指定するパス。タスクがワークフローの直下にある場合には、タスク名のみを入力します。タスクがワークレット内にある場合は、「 <i>WorkletName.TaskName</i> 」と入力します。taskInstancePath を完全修飾文字列で入力してください。
タイムアウト	必須	この操作中に、クライアント要求が Web Services Hub への接続を保持する時間。タイムアウト期間内に操作が完了しない場合、この操作は失敗します。
ServiceName	必須	ワークフローを実行する Integration Service の名前。
DomainName	オプション	Integration Service のドメイン名。

getSessionPerformanceData

この演算を使用して、Integration Service 上で実行中のセッションのパフォーマンスデータを取得します。パフォーマンス詳細には、セッションとマッピングの効率を把握するうえで役立つカウンタが含まれています。

リポジトリにログインしてから、この操作を呼び出す必要があります。

以下の表に、この操作のパラメータを示します。

パラメータ名	必須/オプション	説明
SessionID	必須	ログイン後に生成されたセッション ID。
FolderName	必須	ワークフローが格納されているフォルダの名前。
WorkflowName	必須	セッションを含むワークフローの名前。
TaskInstancePath	必須	セッションの場所を指定するパス。セッションがワークフロー内にある場合は、セッション名のみを入力します。セッションがワークレット内にある場合は、「 <i>WorkletName.SessionName</i> 」と入力します。 <i>taskInstancePath</i> を完全修飾文字列で入力してください。
ServiceName	必須	ワークフローを実行する Integration Service の名前。
DomainName	オプション	Integration Service のドメイン名。

getSessionStatistics

この演算を使用して、Integration Service 上で実行中のセッションの統計を取得します。セッションが実行されていない場合、一番最近に実行されたセッションについての統計情報が返されます。

リポジトリにログインしてから、この操作を呼び出す必要があります。

セッションの統計情報には、フォルダ名、ワークフロー名、セッションの状態、タスクの実行状態、エラー情報、ソースとターゲットの成功行数と失敗行数、適用された行数、影響を受けた行数、拒否された行数などが含まれます。

以下の表に、この操作のパラメータを示します。

パラメータ名	必須/オプション	説明
SessionID	必須	ログイン後に生成されたセッション ID。
FolderName	必須	ワークフローが格納されているフォルダの名前。
WorkflowName	必須	セッションタスクが含まれるワークフローの名前。
TaskInstancePath	必須	セッションの場所を指定するパス。セッションがワークフロー内にある場合は、セッション名のみを入力します。セッションがワークレット内にある場合は、「 <i>WorkletName.SessionName</i> 」と入力します。 <i>taskInstancePath</i> を完全修飾文字列で入力してください。

パラメータ名	必須/オプション	説明
ServiceName	必須	ワークフローを実行する Integration Service の名前。
DomainName	オプション	Integration Service のドメイン名。

getTaskDetails

この演算を使用して、Integration Service からタスクの詳細を取得します。親ワークフローが実行中であり、そこに含まれるタスクが既に実行されている場合、この演算は実行中のワークフローにおける現在のタスクの詳細情報を返します。親ワークフローが実行中でない場合、この演算は最後に実行したワークフローのタスクの詳細を返します。

リポジトリにログインしてから、この操作を呼び出す必要があります。

タスクの詳細情報には、フォルダ名、ワークフロー名、タスク名、タスクのタイプ、開始時刻、実行状態、実行エラーコード、実行エラーメッセージなどが含まれます。

以下の表に、この操作のパラメータを示します。

パラメータ名	必須/オプション	説明
SessionID	必須	ログイン後に生成されたセッション ID。
FolderName	必須	ワークフローが格納されているフォルダの名前。
WorkflowName	必須	タスクを含むワークフローの名前。
TaskInstancePath	必須	タスクの場所を指定するパス。タスクがワークフローの直下にある場合には、タスク名のみを入力します。タスクがワークレット内にある場合は、「<WorkletName>.<TaskName>」と入力します。 taskInstancePath を完全修飾文字列で入力してください。
ServiceName	必須	ワークフローを実行する Integration Service の名前。
WorkflowRunId	オプション	タスクが格納されているワークフロー実行インスタンスの ID。このパラメータは使用できません。
WorkflowRunInstanceName	オプション	タスクが含まれるワークフロー実行インスタンスの名前。
ParameterFileName	オプション	ワークフロー実行時に使用されるパラメータファイルの名前。
RequestMode	オプション	セッションタスクのリカバリ方法を以下から指定。 - Normal。リカバリせずにセッションを再開します。 - RECOVERY。セッションをリカバリします。
IsAbort	オプション	タスクを強制終了するかどうかを指定します。この操作には適用できません。
DomainName	オプション	Integration Service のドメイン名。

パラメータ名	必須/オプション	説明
ParameterScope	オプション	パラメータ配列定義内のパラメータのスコープ。
ParameterName	オプション	パラメータ配列定義内のパラメータの名前。
ParameterValue	オプション	パラメータ配列定義内のパラメータの値。

getTaskDetailsEx

この操作を使用して、ワークフローの複数インスタンスが同時に実行されるときに Integration Service からタスクの詳細を取得します。

getTaskDetailsEx 操作は、getTaskDetails 操作に似ていますが、タスクのインスタンスすべてに関する情報を返します。親ワークフローが実行中であり、そこに含まれるタスクが既に実行されている場合、この操作は実行中のワークフローにおけるタスクのインスタンスすべての詳細情報を返します。親ワークフローが実行中でない場合、この操作は最後に実行したワークフローのタスクインスタンスの詳細を返します。タスクの詳細は、操作によって返された Integration Service の詳細に含まれます。

リポジトリにログインしてから、この操作を呼び出す必要があります。

以下の表に、この操作のパラメータを示します。

パラメータ名	必須/オプション	説明
SessionID	必須	ログイン後に生成されたセッション ID。
FolderName	必須	ワークフローが格納されているフォルダの名前。
WorkflowName	必須	タスクを含むワークフローの名前。
TaskInstancePath	必須	タスクの場所を指定するパス。タスクがワークフローの直下にある場合には、タスク名のみを入力します。タスクがワークレット内にある場合は、「<WorkletName>.<TaskName>」と入力します。taskInstancePath を完全修飾文字列で入力してください。
ServiceName	必須	ワークフローを実行する Integration Service の名前。
WorkflowRunInstanceName	オプション	ワークフロー実行インスタンスの名前。
DomainName	オプション	Integration Service のドメイン名。

getWorkflowDetails

この操作を使用して、指定されたワークフローの詳細を取得できます。ワークフローが実行中の場合、この演算は実行中のワークフローの詳細を返します。ワークフローが実行中でない場合、この演算は最後に実行したこのワークフローの詳細を返します。

ワークフローの詳細情報には、フォルダ名、ワークフロー名、ワークフローのログファイル名のほか、ワークフローを実行するユーザー名が含まれます。ワークフローの実行タイプ、ログファイルのコードページ、開始時刻、終了時刻、実行状態、実行エラーコード、実行エラーメッセージも含まれています。

リポジトリにログインしてから、この操作を呼び出す必要があります。

以下の表に、この操作のパラメータを示します。

パラメータ名	必須/オプション	説明
SessionID	必須	ログイン後に生成されたセッション ID。
FolderName	必須	ワークフローが格納されているフォルダの名前。
WorkflowName	必須	ワークフロー名。
RequestMode	必須	セッションタスクのリカバリ方法を以下から指定。 - Normal。リカバリせずにセッションを再開します。 - RECOVERY。セッションをリカバリします。
ServiceName	必須	ワークフローを実行する Integration Service の名前。
WorkflowRunId	オプション	ワークフロー実行インスタンスの ID。
WorkflowRunInstanceName	オプション	ワークフロー実行インスタンスの名前。
理由	オプション	ワークフローまたはタスクを開始する理由の記述。
ParameterFileName	オプション	ワークフロー実行時に使用されるパラメータファイルの名前。
TaskInstancePath	オプション	タスクの場所を指定するパス。タスクがワークフローの直下にある場合には、タスク名のみを入力します。タスクがワークレット内にある場合は、「<WorkletName>.<TaskName>」と入力します。 taskInstancePath を完全修飾文字列で入力してください。
IsAbort	オプション	タスクを強制終了するかどうかを指定します。この操作には適用できません。
OSUser	オプション	ワークフローに割り当てられたオペレーティングシステムプロファイルを指定します。
DomainName	オプション	Integration Service のドメイン名。
AttributeName	オプション	ワークフローまたはタスクを開始またはスケジュール設定するために使用する、属性名および属性値ペアのうちの属性名。
AttributeValue	オプション	ワークフローまたはタスクを開始またはスケジュール設定するために使用する、属性名および属性値ペアのうちの属性値。
キー	オプション	ワークフローまたはタスクを開始するために使用するキー。

パラメータ名	必須/オプション	説明
MustUse	オプション	ワークフローまたはタスクを開始するためにキーを使用する必要があるかどうかを指定します。
ParameterScope	オプション	パラメータ配列定義内のパラメータの範囲。
ParameterName	オプション	パラメータ配列定義内のパラメータの名前。
ParameterValue	オプション	パラメータ配列定義内のパラメータの値。

getWorkflowDetailsEx

この操作を使用して、ワークフローの複数インスタンスが同時に実行されるときにワークフローの詳細を取得します。

この操作は、getWorkflowDetails 操作に似ていますが、ワークフローのインスタンスに関する情報を返します。ワークフローが実行中の場合、この操作は実行中のワークフローのインスタンスすべての詳細を返します。ワークフローが実行中でない場合、この操作は最後に実行したワークフローの詳細を返します。ワークフローの詳細は、操作によって返された Integration Service の詳細に含まれます。

リポジトリにログインしてから、この操作を呼び出す必要があります。

以下の表に、この操作のパラメータを示します。

パラメータ名	必須/オプション	説明
SessionID	必須	ログイン後に生成されたセッション ID。
FolderName	必須	ワークフローが格納されているフォルダの名前。
WorkflowName	必須	ワークフロー名。
ServiceName	必須	ワークフローを実行する Integration Service の名前。
WorkflowRunId	オプション	ワークフロー実行インスタンスの ID。
WorkflowRunInstanceName	オプション	ワークフロー実行インスタンスの名前。
DomainName	オプション	Integration Service のドメイン名。

getWorkflowLog

この操作を使用して、ワークフローログ内のすべての情報を 1 回の操作で取得します。リポジトリにログインしてから、この操作を呼び出す必要があります。

Web サービスワークフローが実行されると、ワークフローログにはプロセスの初期化処理、ワークフロータスクの実行情報、検出したエラー、およびワークフローの実行要約などの情報が Integration Service によって書き込まれます。ワークフローログ内の詳細情報の量は、トレースレベルによって決まります。getWorkflowLog 演算は、ワークフローログの情報を返します。

ワークフローログ情報を単位で取得するには、getNextLogSegment 操作を使用します。

以下の表に、この操作のパラメータを示します。

パラメータ名	必須/オプション	説明
SessionID	必須	ログイン後に生成されたセッション ID。
FolderName	必須	ワークフローが格納されているフォルダの名前。
WorkflowName	必須	ワークフロー名。
タイムアウト	必須	この操作中に、クライアント要求が Web Services Hub への接続を保持する時間。タイムアウト期間内に操作が完了しない場合、この操作は失敗します。
ServiceName	必須	ワークフローを実行する Integration Service の名前。
WorkflowRunId	オプション	ワークフロー実行インスタンスの ID。
WorkflowRunInstanceName	オプション	ワークフロー実行インスタンスの名前。
DomainName	オプション	Integration Service のドメイン名。

initializeDIserverConnection

廃止された演算。Integration Service への接続を初期化する必要はありません。

リポジトリにログインしてから、この操作を呼び出す必要があります。

以下の表に、この操作のパラメータを示します。

パラメータ名	必須/オプション	説明
SessionID	必須	ログイン後に生成されたセッション ID。
DIserverName	必須	ワークフローが格納されているフォルダの名前。
LoginHandle	オプション	セッション ID と同じ。
DIserverDomain	オプション	Integration Service のドメイン名。

ログイン

Login 演算は、Data Integration Web サービスと Metadata Web サービスに含まれています。

この操作を使用して、リポジトリにログインします。ログイン操作では、指定したリポジトリのユーザー名およびパスワードを認証します。クライアントアプリケーションは、他の演算を呼び出す前に、この演算を呼び出す必要があります。ログイン演算を呼び出した後、Web サービスのクライアントアプリケーションは任意の Batch Web Services の演算を呼び出すことができます。

ログイン演算には、リポジトリ名、ユーザー名、およびパスワードが必要です。それによって、暗号化されたセッション ID が返されます。リポジトリは、Web Services Hub と同じドメインにある必要があります。

以下の表に、この操作のパラメータを示します。

パラメータ名	必須/オプション	説明
RepositoryName	必須	ログインするリポジトリの名前。
UserName	必須	リポジトリにログインするために使用するユーザー名。
パスワード	必須	リポジトリにログインするために使用するユーザーアカウントのパスワード。
RepositoryDomainName	オプション	リポジトリサービスのドメイン名。
UserNameSpace	条件付	リポジトリにログインするために使用するユーザーアカウントのセキュリティドメイン。PowerCenter ドメインに複数のセキュリティドメインが存在する場合は必須。

セッションの有効期限

セッション ID は、非アクティブ状態の後で期限切れになります。Web Services Hub の **[SessionExpiryPeriod]** 詳細プロパティを設定し、セッションが期限切れとなるまでの非アクティブな状態の時間の長さを指定します。Web Services Hub の詳細プロパティは、管理者ツールで設定できます。

Web Services Hub は、セッションの開始時にユーザー名とパスワードをキャッシュします。Web Services Hub は、セッションが期限切れとなるまで、キャッシュされたユーザー名とパスワードをログイン認証に使用します。ユーザーアカウントを変更しても、セッションが期限切れになるまで認証には影響しません。ユーザーアカウントを削除または無効化しても、セッションが期限切れになるまで、ユーザーアカウントは削除または無効化されません。ユーザーアカウントがロックされた場合、ロックされたユーザーは、セッションの期限が切れるまで要求を送信できます。

注: SessionExpiryPeriod の値が大きい場合は、ユーザーアカウントの変更が有効になるまでしばらく時間がかかります。セキュリティ違反を防止するため、セッション内で使用中のユーザーアカウントを削除または変更しないでください。

ログアウト

Logout 演算は、Data Integration Web サービスと Metadata Web サービスに含まれています。

この操作を使用して、リポジトリからログアウトします。Logout 操作を使用して、リポジトリおよび Integration Service からユーザーを切断します。クライアントアプリケーション実行の最後にこの操作を呼び出してリソースを解放し、Repository Service と Web Services Hub プロセスで起こりうるメモリリークを防ぎます。

SessionID パラメータを指定して Logout 操作を呼び出します。リポジトリにログインした後にセッション ID が生成されます。

monitorDIserver

この操作を使用して、Integration Service の状態、スケジュールされたアクティブなワークフローの詳細、タスクの詳細、およびワークフロー内のリンクを取得します。

リポジトリにログインしてから、この操作を呼び出す必要があります。

以下の表に、この操作のパラメータを示します。

パラメータ名	必須/オプション	説明
SessionID	必須	ログイン後に生成されたセッション ID。
MonitorMode	必須	モニタリングするワークフローのモード。 <ul style="list-style-type: none">- RUNNING。アクティブなワークフローの状態の詳細情報が返されます。アクティブなワークフローには、実行中、サスペンド状態、サスペンド中のワークフローが含まれます。- SCHEDULED。スケジュールが設定されたワークフローの状態に関する詳細情報が返されます。- ALL。スケジュールが設定された、アクティブなすべてのワークフローの状態に関する詳細情報が返されます。
ServiceName	必須	モニタリングする Integration Service の名前。
DomainName	オプション	Integration Service のドメイン名。

pingDIserver

この演算を使用して、Integration Service が起動しているかどうかを判断します。戻り値は ALIVE または FAIL になります。リポジトリにログインしてから、この操作を呼び出す必要があります。

以下の表に、この操作のパラメータを示します。

パラメータ名	必須/オプション	説明
SessionID	必須	ログイン後に生成されたセッション ID。
タイムアウト	必須	この操作中に、クライアント要求が Web Services Hub への接続を保持する時間。タイムアウト期間内に操作が完了しない場合、この操作は失敗します。
ServiceName	必須	起動を確認する Integration Service の名前。
DomainName	オプション	Integration Service のドメイン名。

recoverWorkflow

この操作を使用して、一時停止したワークフローをリカバリします。Integration Service は、すべての一時停止または失敗したワークレット、およびすべての一時停止または失敗した [コマンド]、[メール]、[セッション] タスクから、ワークフローをリカバリします。

リポジトリにログインしてから、この操作を呼び出す必要があります。

以下の表に、この操作のパラメータを示します。

パラメータ名	必須/オプション	説明
SessionID	必須	ログイン後に生成されたセッション ID。
FolderName	必須	ワークフローが格納されているフォルダの名前。
WorkflowName	必須	リカバリするワークフローの名前。
RequestMode	必須	セッションタスクのリカバリ方法を以下から指定。 - Normal。リカバリせずにセッションを再開します。 - RECOVERY。セッションをリカバリします。
ServiceName	必須	ワークフローを実行する Integration Service の名前。
WorkflowRunId	オプション	ワークフロー実行インスタンスの ID。
WorkflowRunInstanceName	オプション	ワークフロー実行インスタンスの名前。
理由	オプション	ワークフローまたはタスクを開始する理由の記述。
ParameterFileName	オプション	ワークフロー実行時に使用されるパラメータファイルの名前。
TaskInstancePath	オプション	タスクの場所を指定するパス。タスクがワークフローの直下にある場合には、タスク名のみを入力します。タスクがワークレット内にある場合は、「<WorkletName>.<TaskName>」と入力します。taskInstancePath を完全修飾文字列で入力してください。
IsAbort	オプション	タスクを強制終了するかどうかを指定します。この操作には適用できません。
OSUser	オプション	ワークフローに割り当てられたオペレーティングシステムプロファイルを指定します。
DomainName	オプション	Integration Service のドメイン名。
AttributeName	オプション	ワークフローまたはタスクを開始またはスケジュール設定するために使用する、属性名および属性値ペアのうちの属性名。
AttributeValue	オプション	ワークフローまたはタスクを開始またはスケジュール設定するために使用する、属性名および属性値ペアのうちの属性値。
キー	オプション	ワークフローまたはタスクを開始するために使用するキー。
MustUse	オプション	ワークフローまたはタスクを開始するためにキーを使用する必要があるかどうかを指定します。
ParameterScope	オプション	パラメータ配列定義内のパラメータのスコープ。

パラメータ名	必須/オプション	説明
ParameterName	オプション	パラメータ配列定義内のパラメータの名前。
ParameterValue	オプション	パラメータ配列定義内のパラメータの値。

resumeWorkflow

廃止された演算。代わりに recoverWorkflow 操作を使用します。

リポジトリにログインしてから、この操作を呼び出す必要があります。

以下の表に、この操作のパラメータを示します。

パラメータ名	必須/オプション	説明
SessionID	必須	ログイン後に生成されたセッション ID。
FolderName	必須	ワークフローが格納されているフォルダの名前。
WorkflowName	必須	再開するワークフローの名前。
RequestMode	必須	セッションタスクのリカバリ方法を以下から指定。 - Normal。リカバリせずにセッションを再開します。 - RECOVERY。セッションをリカバリします。
ServiceName	必須	ワークフローを実行する Integration Service の名前。
WorkflowRunId	オプション	ワークフロー実行インスタンスの ID。
WorkflowRunInstanceName	オプション	ワークフロー実行インスタンスの名前。
理由	オプション	ワークフローまたはタスクを開始する理由の記述。
ParameterFileName	オプション	ワークフロー実行時に使用されるパラメータファイルの名前。
TaskInstancePath	オプション	タスクの場所を指定するパス。タスクがワークフローの直下にある場合には、タスク名のみを入力します。タスクがワークレット内にある場合は、「<WorkletName>.<TaskName>」と入力します。taskInstancePath を完全修飾文字列で入力してください。
IsAbort	オプション	タスクを強制終了するかどうかを指定します。この操作には適用できません。
OSUser	オプション	ワークフローに割り当てられたオペレーティングシステムプロファイルを指定します。
DomainName	オプション	Integration Service のドメイン名。

パラメータ名	必須/オプション	説明
AttributeName	オプション	ワークフローまたはタスクを開始またはスケジュール設定するために使用する、属性名および属性値ペアのうちの属性名。
AttributeValue	オプション	ワークフローまたはタスクを開始またはスケジュール設定するために使用する、属性名および属性値ペアのうちの属性値。
キー	オプション	ワークフローまたはタスクを開始するために使用するキー。
MustUse	オプション	ワークフローまたはタスクを開始するためにキーを使用する必要があるかどうかを指定します。
ParameterScope	オプション	パラメータ配列定義内のパラメータの範囲。
ParameterName	オプション	パラメータ配列定義内のパラメータの名前。
ParameterValue	オプション	パラメータ配列定義内のパラメータの値。

scheduleWorkflow

この操作を使用して、ワークフローのスケジュールを設定できます。オンデマンドでは実行されないすべてのワークフローに対してスケジュールを設定できます。

リポジトリにログインしてから、この操作を呼び出す必要があります。

以下の表に、この操作のパラメータを示します。

パラメータ名	必須/オプション	説明
SessionID	必須	ログイン後に生成されたセッション ID。
FolderName	必須	ワークフローが格納されているフォルダの名前。
WorkflowName	必須	スケジュールを設定するワークフローの名前。
ServiceName	必須	ワークフローを実行する Integration Service の名前。
WorkflowRunId	オプション	ワークフロー実行インスタンスの ID。
WorkflowRunInstanceName	オプション	スケジュールを設定するワークフロー実行インスタンスの名前。
理由	オプション	ワークフローまたはタスクを開始する理由の記述。
ParameterFileName	オプション	ワークフロー実行時に使用されるパラメータファイルの名前。
RequestMode	オプション	セッションタスクのリカバリ方法を以下から指定。 <ul style="list-style-type: none"> - Normal。リカバリせずにセッションを再開します。 - RECOVERY。セッションをリカバリします。

パラメータ名	必須/オプション	説明
TaskInstancePath	オプション	タスクの場所を指定するパス。タスクがワークフローの直下にある場合には、タスク名のみを入力します。タスクがワークレット内にある場合は、「<WorkletName>.<TaskName>」と入力します。taskInstancePath を完全修飾文字列で入力してください。
IsAbort	オプション	タスクを強制終了するかどうかを指定します。この操作には適用できません。
OSUser	オプション	ワークフローに割り当てられたオペレーティングシステムプロファイルを指定します。
DomainName	オプション	Integration Service のドメイン名。
AttributeName	オプション	ワークフローのスケジュールを設定するために使用する属性名および属性値ペアのうちの属性名。
AttributeValue	オプション	ワークフローのスケジュールを設定するために使用する属性名および属性値ペアのうちの属性値。
キー	オプション	ワークフローまたはタスクを開始するために使用するキー。
MustUse	オプション	ワークフローまたはタスクを開始するためにキーを使用する必要があるかどうかを指定します。
ParameterScope	オプション	パラメータ配列定義内のパラメータのスコープ。
ParameterName	オプション	パラメータ配列定義内のパラメータの名前。
ParameterValue	オプション	パラメータ配列定義内のパラメータの値。

startSessionLogFetch

廃止された演算。この演算は、セッションログ内の情報の単位取得を開始します。

startSessionLogFetch では、getNextLogSegment 演算と共に使用するためのログハンドルが生成されます。startSessionLogFetch 演算を呼び出した後は、ログの最後に達するまで、startSessionLogFetch によって生成されたログハンドルを使用して getNextLogSegment 演算を呼び出します。

リポジトリにログインしてから、この操作を呼び出す必要があります。

以下の表に、この操作のパラメータを示します。

パラメータ名	必須/オプション	説明
SessionID	必須	ログイン後に生成されたセッション ID。
FolderName	必須	ワークフローが格納されているフォルダの名前。
WorkflowName	必須	セッションが格納されているワークフローの名前。
ServiceName	必須	ワークフローを実行する Integration Service の名前。

startTask

この演算を使用して、ワークフロー内の特定のタスクを開始できます。リポジトリにログインしてから、この操作を呼び出す必要があります。

以下の表に、この操作のパラメータを示します。

パラメータ名	必須/オプション	説明
SessionID	必須	ログイン後に生成されたセッション ID。
FolderName	必須	ワークフローが格納されているフォルダの名前。
WorkflowName	必須	タスクが含まれるワークフローの名前。
RequestMode	必須	セッションタスクのリカバリ方法を以下から指定。 - Normal。リカバリせずにセッションを再開します。 - RECOVERY。セッションをリカバリします。
TaskInstancePath	必須	タスクの場所を指定するパス。タスクがワークフローの直下にある場合には、タスク名のみを入力します。タスクがワークレット内にある場合は、「<WorkletName>.<TaskName>」と入力します。taskInstancePath を完全修飾文字列で入力してください。
ServiceName	必須	ワークフローを実行する Integration Service の名前。
WorkflowRunId	オプション	タスクが格納されているワークフロー実行インスタンスの ID。
WorkflowRunInstanceName	オプション	タスクが含まれるワークフロー実行インスタンスの名前。
ParameterFileName	オプション	ワークフロー実行時に使用されるパラメータファイルの名前。
IsAbort	オプション	タスクを強制終了するかどうかを指定します。この操作には適用できません。
DomainName	オプション	Integration Service のドメイン名。
ParameterScope	オプション	パラメータ配列定義内のパラメータの範囲。
ParameterName	オプション	パラメータ配列定義内のパラメータの名前。
ParameterValue	オプション	パラメータ配列定義内のパラメータの値。

startWorkflow

この演算を使用して、ワークフローを開始できます。リポジトリにログインしてから、この操作を呼び出す必要があります。

以下の表に、この操作のパラメータを示します。

パラメータ名	必須/オプション	説明
SessionID	必須	ログイン後に生成されたセッション ID。
FolderName	必須	ワークフローが格納されているフォルダの名前。
WorkflowName	必須	実行するワークフローの名前。
RequestMode	必須	セッションタスクのリカバリ方法を以下から指定。 - Normal。リカバリせずにセッションを再開します。 - RECOVERY。セッションをリカバリします。
ServiceName	必須	ワークフローを実行する Integration Service の名前。
WorkflowRunId	オプション	ワークフロー実行インスタンスの ID。
WorkflowRunInstanceName	オプション	実行するワークフロー実行インスタンスの名前。
理由	オプション	ワークフローを開始する理由の記述。
ParameterFileName	オプション	ワークフロー実行時に使用されるパラメータファイルの名前。
TaskInstancePath	オプション	タスクの場所を指定するパス。タスクがワークフローの直下にある場合には、タスク名のみを入力します。タスクがワークレット内にある場合は、「<WorkletName>.<TaskName>」と入力します。taskInstancePath を完全修飾文字列で入力してください。
IsAbort	オプション	タスクを強制終了するかどうかを指定します。この操作には適用できません。
OSUser	オプション	ワークフローに割り当てられたオペレーティングシステムプロファイルを指定します。
DomainName	オプション	Integration Service のドメイン名。
AttributeName	オプション	ワークフローを開始するために使用する属性名および属性値ペアのうちの属性名。
AttributeValue	オプション	ワークフローを開始するために使用する属性名および属性値ペアのうちの属性値。
キー	オプション	ワークフローまたはタスクを開始するために使用するキー。
MustUse	オプション	ワークフローまたはタスクを開始するためにキーを使用する必要があるかどうかを指定します。
ParameterScope	オプション	パラメータ配列定義内のパラメータの範囲。
ParameterName	オプション	パラメータ配列定義内のパラメータの名前。
ParameterValue	オプション	パラメータ配列定義内のパラメータの値。

startWorkflowEx

この演算を使用して、ワークフローを開始できます。startWorkflowEx 操作は、ワークフローの実行インスタンス ID を返します。この操作によって開始されるワークフローの実行 ID を取得するには、startWorkflow 操作ではなく startWorkflowEx 操作を使用します。

リポジトリにログインしてから、この操作を呼び出す必要があります。

以下の表に、この操作のパラメータを示します。

パラメータ名	必須/オプション	説明
SessionID	必須	ログイン後に生成されたセッション ID。
FolderName	必須	ワークフローが格納されているフォルダの名前。
WorkflowName	必須	実行するワークフローの名前。
RequestMode	必須	セッションタスクのリカバリ方法を以下から指定。 - Normal。リカバリせずにセッションを再開します。 - RECOVERY。セッションをリカバリします。
ServiceName	必須	ワークフローを実行する Integration Service の名前。
WorkflowRunInstanceName	オプション	実行するワークフロー実行インスタンスの名前。
理由	オプション	ワークフローを開始する理由の記述。
ParameterFileName	オプション	ワークフロー実行時に使用されるパラメータファイルの名前。
TaskInstancePath	オプション	タスクの場所を指定するパス。タスクがワークフローの直下にある場合には、タスク名のみを入力します。タスクがワークレット内にある場合は、「<WorkletName>.<TaskName>」と入力します。taskInstancePath を完全修飾文字列で入力してください。
OSUser	オプション	ワークフローに割り当てられたオペレーティングシステムプロファイルを指定します。
DomainName	オプション	Integration Service のドメイン名。
AttributeName	オプション	ワークフローを開始するために使用する属性名および属性値ペアのうちの属性名。
AttributeValue	オプション	ワークフローを開始するために使用する属性名および属性値ペアのうちの属性値。
キー	オプション	ワークフローまたはタスクを開始するために使用するキー。
MustUse	オプション	ワークフローまたはタスクを開始するためにキーを使用する必要があるかどうかを指定します。
ParameterScope	オプション	パラメータ配列定義内のパラメータの範囲。

パラメータ名	必須/オプション	説明
ParameterName	オプション	パラメータ配列定義内のパラメータの名前。
ParameterValue	オプション	パラメータ配列定義内のパラメータの値。

startWorkflowFromTask

この演算を使用してタスクからワークフローを開始します。ワークフローをタスクから開始した場合、Integration Service は、選択されたタスクからワークフローの最後までワークフローを実行します。

開始するタスクのタスクインスタンスパスを指定する必要があります。タスクインスタンスパスによって、ワークフロー内のタスクインスタンスが識別されます。ワークフロー内のタスクは、タスク名だけで識別されます。ワークレット内のタスクは、ピリオドで区切られたワークレット名およびタスク名 `<WorkletName>.<TaskName>` で識別されます。たとえば、ワークフローにはワークレット B を含むワークレット A があるとしします。タスク C はワークレット B 内部のタスクです。タスク C のタスクインスタンスパスは A.B.C です。

リポジトリにログインしてから、この操作を呼び出す必要があります。

以下の表に、この操作のパラメータを示します。

パラメータ名	必須/オプション	説明
SessionID	必須	ログイン後に生成されたセッション ID。
FolderName	必須	ワークフローが格納されているフォルダの名前。
WorkflowName	必須	実行するワークフローの名前。
RequestMode	必須	セッションタスクのリカバリ方法を以下から指定。 - Normal。リカバリせずにセッションを再開します。 - RECOVERY。セッションをリカバリします。
TaskInstancePath	必須	タスクの場所を指定するパス。タスクがワークフローの直下にある場合には、タスク名のみを入力します。タスクがワークレット内にある場合は、「 <code><WorkletName>.<TaskName></code> 」と入力します。 taskInstancePath を完全修飾文字列で入力してください。
ServiceName	必須	ワークフローを実行する Integration Service の名前。
WorkflowRunId	オプション	ワークフロー実行インスタンスの ID。
WorkflowRunInstanceName	オプション	開始するワークフロー実行インスタンスの名前。
理由	オプション	ワークフローを開始する理由の記述。
ParameterFileName	オプション	ワークフロー実行時に使用されるパラメータファイルの名前。

パラメータ名	必須/オプション	説明
IsAbort	オプション	タスクを強制終了するかどうかを指定します。この操作には適用できません。
OSUser	オプション	ワークフローに割り当てられたオペレーティングシステムプロファイルを指定します。
DomainName	オプション	Integration Service のドメイン名。
AttributeName	オプション	ワークフローを開始するために使用する属性名および属性値ペアのうちの属性名。
AttributeValue	オプション	ワークフローを開始するために使用する属性名および属性値ペアのうちの属性値。
キー	オプション	ワークフローまたはタスクを開始するために使用するキー。
MustUse	オプション	ワークフローまたはタスクを開始するためにキーを使用する必要があるかどうかを指定します。
ParameterScope	オプション	パラメータ配列定義内のパラメータのスコープ。
ParameterName	オプション	パラメータ配列定義内のパラメータの名前。
ParameterValue	オプション	パラメータ配列定義内のパラメータの値。

startWorkflowLogFetch

廃止された演算。この演算は、ワークフローログ内の情報の単位取得を開始します。

startWorkflowLogFetch では、getNextLogSegment 演算と共に使用するためのログハンドルが生成されます。startWorkflowLogFetch 演算を呼び出した後は、ログの最後に達するまで、startWorkflowLogFetch によって生成されたログハンドルを使用して getNextLogSegment 演算を呼び出します。

リポジトリにログインしてから、この操作を呼び出す必要があります。

以下の表に、この操作のパラメータを示します。

パラメータ名	必須/オプション	説明
SessionID	必須	ログイン後に生成されたセッション ID。
FolderName	必須	ワークフローが格納されているフォルダの名前。
WorkflowName	必須	ワークフロー名。
ServiceName	必須	ワークフローを実行する Integration Service の名前。
WorkflowRunId	オプション	ワークフロー実行インスタンスの ID。

パラメータ名	必須/オプション	説明
WorkflowRunInstanceName	オプション	ワークフロー実行インスタンスの名前。
DomainName	オプション	Integration Service のドメイン名。

stopTask

この演算を使用して、Integration Service 上で実行中のタスクを停止します。タスク、ワークフロー、ワークレットは、いつでも停止または強制終了することができます。ワークフロー内のタスクを停止する場合、Integration Service はそのタスクとそのパスにある他のすべてのタスクの実行を停止します。

また、isAbort パラメータを true に設定することで、実行中のタスクを強制終了することもできます。強制終了するタスクのタスクインスタンスパスを指定します。通常、Integration Service がタスクの停止に失敗した場合のみタスクを強制終了します。

リポジトリにログインしてから、この操作を呼び出す必要があります。

以下の表に、この操作のパラメータを示します。

パラメータ名	必須/オプション	説明
SessionID	必須	ログイン後に生成されたセッション ID。
FolderName	必須	ワークフローが格納されているフォルダの名前。
WorkflowName	必須	タスクが含まれるワークフローの名前。
TaskInstancePath	必須	タスクの場所を指定するパス。タスクがワークフローの直下にある場合には、タスク名のみを入力します。タスクがワークレット内にある場合は、「<WorkletName>.<TaskName>」と入力します。taskInstancePath を完全修飾文字列で入力してください。
ServiceName	必須	ワークフローを実行する Integration Service の名前。
WorkflowRunId	オプション	タスクが格納されているワークフロー実行インスタンスの ID。
WorkflowRunInstanceName	オプション	タスクが含まれるワークフロー実行インスタンスの名前。
ParameterFileName	オプション	ワークフロー実行時に使用されるパラメータファイルの名前。
RequestMode	オプション	セッションタスクのリカバリ方法を以下から指定。 - Normal。リカバリせずにセッションを再開します。 - RECOVERY。セッションをリカバリします。
IsAbort	オプション	タスクを強制終了するかどうかを指定します。タスクを強制終了するには、このパラメータを True に設定します。

パラメータ名	必須/オプション	説明
DomainName	オプション	Integration Service のドメイン名。
ParameterScope	オプション	パラメータ配列定義内のパラメータのスコープ。
ParameterName	オプション	パラメータ配列定義内のパラメータの名前。
ParameterValue	オプション	パラメータ配列定義内のパラメータの値。

関連項目：

- [「startWorkflowFromTask」 \(ページ 46\)](#)

stopWorkflow

この操作を使用して、ワークフローの実行を停止できます。ワークフローを停止すると、Integration Service は、ワークフローで現在実行中のタスクをすべて停止しようとします。ワークフローにワークレットが含まれている場合、Integration Service は、ワークレット内の現在実行中のタスクもすべて停止しようとします。

ワークフローを停止するほかに、isAbort のパラメータを true に設定することで、実行中のタスクを強制終了することもできます。通常、Integration Service がワークフローの停止に失敗した場合のみワークフローを強制終了します。

リポジトリにログインしてから、この操作を呼び出す必要があります。

以下の表に、この操作のパラメータを示します。

パラメータ名	必須/オプション	説明
SessionID	必須	ログイン後に生成されたセッション ID。
FolderName	必須	ワークフローが格納されているフォルダの名前。
WorkflowName	必須	停止するワークフローの名前。
ServiceName	必須	ワークフローを実行する Integration Service の名前。
WorkflowRunId	オプション	停止するワークフロー実行インスタンスの ID。
WorkflowRunInstanceName	オプション	停止するワークフロー実行インスタンスの名前。
理由	オプション	ワークフローまたはタスクを開始する理由の記述。
ParameterFileName	オプション	ワークフロー実行時に使用されるパラメータファイルの名前。
RequestMode	オプション	セッションタスクのリカバリ方法を以下から指定。 - Normal。リカバリせずにセッションを再開します。 - RECOVERY。セッションをリカバリします。

パラメータ名	必須/オプション	説明
TaskInstancePath	オプション	タスクの場所を指定するパス。タスクがワークフローの直下にある場合には、タスク名のみを入力します。タスクがワークレット内にある場合は、「<WorkletName>.<TaskName>」と入力します。taskInstancePath を完全修飾文字列で入力してください。
IsAbort	オプション	ワークフローを強制終了するかどうかを指定します。ワークフローを強制終了するには、このパラメータを True に設定します。
OSUser	オプション	ワークフローに割り当てられたオペレーティングシステムプロファイルを指定します。
DomainName	オプション	Integration Service のドメイン名。
AttributeName	オプション	ワークフローまたはタスクを開始またはスケジュール設定するために使用する、属性名および属性値ペアのうちの属性名。
AttributeValue	オプション	ワークフローまたはタスクを開始またはスケジュール設定するために使用する、属性名および属性値ペアのうちの属性値。
キー	オプション	ワークフローまたはタスクを開始するために使用するキー。
MustUse	オプション	ワークフローまたはタスクを開始するためにキーを使用する必要があるかどうかを指定します。
ParameterScope	オプション	パラメータ配列定義内のパラメータのスコープ。
ParameterName	オプション	パラメータ配列定義内のパラメータの名前。
ParameterValue	オプション	パラメータ配列定義内のパラメータの値。

unscheduleWorkflow

この演算を使用して、ワークフローのスケジュールを解除できます。リポジトリにログインしてから、この操作を呼び出す必要があります。

以下の表に、この操作のパラメータを示します。

パラメータ名	必須/オプション	説明
SessionID	必須	ログイン後に生成されたセッション ID。
FolderName	必須	ワークフローが格納されているフォルダの名前。
WorkflowName	必須	スケジュール設定を解除するワークフローの名前。
ServiceName	必須	ワークフローを実行する Integration Service の名前。

パラメータ名	必須/オプション	説明
WorkflowRunId	オプション	スケジュール設定を解除するワークフロー実行インスタンスの ID。
WorkflowRunInstanceName	オプション	スケジュール設定を解除するワークフロー実行インスタンスの名前。
理由	オプション	ワークフローまたはタスクを開始する理由の記述。
ParameterFileName	オプション	ワークフロー実行時に使用されるパラメータファイルの名前。
RequestMode	オプション	セッションタスクのリカバリ方法を以下から指定。 - Normal。リカバリせずにセッションを再開します。 - RECOVERY。セッションをリカバリします。
TaskInstancePath	オプション	タスクの場所を指定するパス。タスクがワークフローの直下にある場合には、タスク名のみを入力します。タスクがワークレット内にある場合は、「<WorkletName>.<TaskName>」と入力します。taskInstancePath を完全修飾文字列で入力してください。
IsAbort	オプション	タスクを強制終了するかどうかを指定します。この操作には適用できません。
OSUser	オプション	ワークフローに割り当てられたオペレーティングシステムプロファイルを指定します。
DomainName	オプション	Integration Service のドメイン名。
AttributeName	オプション	ワークフローまたはタスクを開始またはスケジュール設定するために使用する、属性名および属性値ペアのうちの属性名。
AttributeValue	オプション	ワークフローまたはタスクを開始またはスケジュール設定するために使用する、属性名および属性値ペアのうちの属性値。
キー	オプション	ワークフローまたはタスクを開始するために使用するキー。
MustUse	オプション	ワークフローまたはタスクを開始するためにキーを使用する必要があるかどうかを指定します。
ParameterScope	オプション	パラメータ配列定義内のパラメータのスコープ。
ParameterName	オプション	パラメータ配列定義内のパラメータの名前。
ParameterValue	オプション	パラメータ配列定義内のパラメータの値。

waitTillTaskComplete

この演算を使用して、Integration Service 上で実行中のタスクが完了するのを待ちます。リポジトリにログインしてから、この操作を呼び出す必要があります。

以下の表に、この操作のパラメータを示します。

パラメータ名	必須/オプション	説明
SessionID	必須	ログイン後に生成されたセッション ID。
FolderName	必須	ワークフローが格納されているフォルダの名前。
WorkflowName	必須	タスクが含まれるワークフローの名前。
TaskInstancePath	必須	タスクの場所を指定するパス。タスクがワークフローの直下にある場合には、タスク名のみを入力します。タスクがワークレット内にある場合は、「<WorkletName>.<TaskName>」と入力します。 taskInstancePath を完全修飾文字列で入力してください。
RequestMode	必須	セッションタスクのリカバリ方法を以下から指定。 - Normal。リカバリせずにセッションを再開します。 - RECOVERY。セッションをリカバリします。
ServiceName	必須	ワークフローを実行する Integration Service の名前。
WorkflowRunId	オプション	タスクが格納されているワークフロー実行インスタンスの ID。
WorkflowRunInstanceName	オプション	タスクが含まれるワークフロー実行インスタンスの名前。
理由	オプション	ワークフローまたはタスクを開始する理由の記述。
ParameterFileName	オプション	ワークフロー実行時に使用されるパラメータファイルの名前。
IsAbort	オプション	タスクを強制終了するかどうかを指定します。この操作には適用できません。
OSUser	オプション	ワークフローに割り当てられたオペレーティングシステムプロファイルを指定します。
DomainName	オプション	Integration Service のドメイン名。
AttributeName	オプション	ワークフローまたはタスクを開始またはスケジュール設定するために使用する、属性名および属性値ペアのうちの属性名。
AttributeValue	オプション	ワークフローまたはタスクを開始またはスケジュール設定するために使用する、属性名および属性値ペアのうちの属性値。
キー	オプション	ワークフローまたはタスクを開始するために使用するキー。
MustUse	オプション	ワークフローまたはタスクを開始するためにキーを使用する必要があるかどうかを指定します。
ParameterScope	オプション	パラメータ配列定義内のパラメータのスコープ。

パラメータ名	必須/オプション	説明
ParameterName	オプション	パラメータ配列定義内のパラメータの名前。
ParameterValue	オプション	パラメータ配列定義内のパラメータの値。

waitTillWorkflowComplete

別のワークフローが完了した後にのみ操作またはワークフローを実行するには、この操作を使用して、他のワークフローが完了したときに通知を受け取るようにします。リポジトリにログインしてから、この操作を呼び出す必要があります。

以下の表に、この操作のパラメータを示します。

パラメータ名	必須/オプション	説明
SessionID	必須	ログイン後に生成されたセッション ID。
FolderName	必須	ワークフローが格納されているフォルダの名前。
WorkflowName	必須	ワークフロー名。
RequestMode	必須	セッションタスクのリカバリ方法を以下から指定。 - Normal。リカバリせずにセッションを再開します。 - RECOVERY。セッションをリカバリします。
ServiceName	必須	ワークフローを実行する Integration Service の名前。
WorkflowRunId	オプション	ワークフロー実行インスタンスの ID。
WorkflowRunInstanceName	オプション	ワークフロー実行インスタンスの名前。
理由	オプション	ワークフローまたはタスクを開始する理由の記述。
ParameterFileName	オプション	ワークフロー実行時に使用されるパラメータファイルの名前。
TaskInstancePath	オプション	タスクの場所を指定するパス。タスクがワークフローの直下にある場合には、タスク名のみを入力します。タスクがワークレット内にある場合は、「<WorkletName>.<TaskName>」と入力します。taskInstancePath を完全修飾文字列で入力してください。
IsAbort	オプション	タスクを強制終了するかどうかを指定します。この操作には適用できません。
OSUser	オプション	ワークフローに割り当てられたオペレーティングシステムプロファイルを指定します。
DomainName	オプション	Integration Service のドメイン名。

パラメータ名	必須/オプション	説明
AttributeName	オプション	ワークフローまたはタスクを開始またはスケジュール設定するために使用する、属性名および属性値ペアのうちの属性名。
AttributeValue	オプション	ワークフローまたはタスクを開始またはスケジュール設定するために使用する、属性名および属性値ペアのうちの属性値。
キー	オプション	ワークフローまたはタスクを開始するために使用するキー。
MustUse	オプション	ワークフローまたはタスクを開始するためにキーを使用する必要があるかどうかを指定します。
ParameterScope	オプション	パラメータ配列定義内のパラメータのスコープ。
ParameterName	オプション	パラメータ配列定義内のパラメータの名前。
ParameterValue	オプション	パラメータ配列定義内のパラメータの値。

クライアントアプリケーションの作成

クライアントアプリケーションの作成の概要

この章では、PowerCenter Web Services Provider が提供する Web サービスを使用するためのクライアントアプリケーションを作成する方法について概要を説明します。クライアントアプリケーションを作成する一般的な手順を解説してから、Java および .NET フレームワークにおけるクライアントアプリケーションの作成方法の例を説明します。

PowerCenter Web サービスのクライアントアプリケーションを作成するには、Web サービス WSDL および Web サービスツールキットが必要になります。Web サービスツールキットでは、Web サービス WSDL からクライアント側プロキシクラスを生成することで、クライアントアプリケーションを簡単に作成することができます。Microsoft .NET および Apache Axis Web サービスツールキットを使用することによって、PowerCenter Web サービスのクライアントアプリケーションを作成できます。

PowerCenter バッチ Web サービスまたはリアルタイム Web サービスを実行するクライアントアプリケーションを作成できます。アプリケーション開発は、同じ基本ステップに従います。

注: Web Services Hub では、チャンクメッセージを処理できます。クライアント要求でチャンク化された転送エンコードを有効にするには、SOAP メッセージに以下のヘッダを追加します。

TRANSFER_ENCODING=chunked

バッチ Web サービスのクライアントアプリケーション

PowerCenter Web Services Provider で使用可能なバッチ Web サービスにアクセスするクライアントアプリケーションの開発には、次の要素が含まれます。

- クライアントプロキシクラス

- 初期化
- セッションメンテナンス
- 演算の呼び出し
- リソースのクリーンアップ
- エラー処理
- プロキシオブジェクト

クライアントプロキシクラスの生成

Web Services Hub で使用可能なバッチ Web サービス操作を使用するには、Web サービスツールキットを使用して、Web サービス WSDL のクライアントプロキシクラスを生成する必要があります。

クライアントプロキシクラスを生成するには、次の手順を完了します。

1. 開発に使用するプラットフォームと言語に対応した Web サービスツールキットを選択します。
2. Web Services Hub Console から、Metadata Web サービスおよび Data Integration Web サービスの WSDL ファイルをダウンロードします。デフォルトでは、Web Services Hub Console から WSDL ファイルをダウンロードすると、Web Services Hub のホスト名およびポート番号にエンドポイント URL が設定されます。プロキシクラスを生成する前に、WSDL ファイルに正しいエンドポイント URL が含まれていることを確認してください。
3. Web サービスツールキットを使用し、WSDL ファイルからクライアント側プロキシクラスを生成します。プロキシクラス生成の詳細については、Web サービスツールキットのドキュメントを参照してください。各ツールキットは特定の方法でクライアントプロキシクラスを生成します。

初期化

クライアントアプリケーションは、Metadata および Data Integration Web サービス操作に対する呼び出しを実行する前に、初期化の手順を実行する必要があります。

初期化を実行するには、以下の手順を実行します。

1. Metadata API のプロキシクラスをインスタンス化します。
この例で、Metadata API プロキシオブジェクトの名前を *MWSPProxy* とします。ここでは、*MWSPProxy* という名前を使用して Metadata API プロキシオブジェクトを参照します。
2. Data Integration API のプロキシクラスをインスタンス化します。
この例で、Data Integration API プロキシオブジェクトの名前を *DIWSPProxy* とします。ここでは、*DIWSPProxy* という名前を使用して Data Integration API プロキシオブジェクトを参照します。
3. *MWSPProxy* オブジェクトを使用して、ログイン操作を呼び出します。
ログイン演算には、リポジトリ名、ユーザー名、およびパスワードが必要です。この演算はセッション ID を返します。この演算を呼び出すと、*MWSPProxy* オブジェクトがリポジトリ名とユーザー名のペアに関連付けられます。*MWSPProxy* オブジェクトを使用して実行されるバッチ Web サービス操作に対するすべての後続要求で、これらのリポジトリ名およびユーザー名が使用されます。

セッションメンテナンス

Web Services Hub では、キャッシュリソースに対するセッションメンテナンスが必要です。SOAP メッセージの SOAP ヘッダには、セッションメンテナンスを容易に行うためのセッション情報が保持されます。

セッションメンテナンスを設定および実行するには、以下の手順を実行します。

1. ログイン操作の呼び出しに対する応答から、ルート要素名 Context および名前空間 `http://www.informatica.com/wsh` を含むヘッダを抽出します。この SOAP ヘッダには、Web Services Hub によって送信されたセッション ID が含まれます。
2. ログイン操作を呼び出した後に、MWSProxy オブジェクトの SOAP ヘッダを設定します。これにより、MWSProxy オブジェクトを使用したすべての後続要求に対して、SOAP ヘッダのセッション ID が送信されます。
3. DIWSProxy オブジェクトを使用して実行されたすべての後続要求に対して、同じセッション ID が送信されるように、同じセッション ID で DIWSProxy オブジェクトの SOAP ヘッダを設定します。

操作の呼び出し

MWSProxy オブジェクトと DIWSProxy オブジェクトを使用して、Metadata Web Service の演算と Data Integration Web Service の演算を呼び出す準備が整いました。MWSProxy オブジェクトを使用して、Metadata Web Service の操作を呼び出します。DIWSProxy オブジェクトを使用して、Data Integration Web Service の演算を呼び出します。

リソースのクリーンアップ

Web Services Hub では、パフォーマンスおよびリソースのクリーンアップに対して、セッションの有効期限が指定されます。Logout 操作は、クライアントアプリケーションによって取得された Web Services Hub のリソースを開放して、クリーンアップ操作を実行します。

リソースを開放するには、MWSProxy オブジェクトを使用して Logout 操作を呼び出します。

リポジトリにログインはしても、Login 操作は呼び出さない場合は、セッションの有効期限後に、Web Services Hub でリソースのクリーンアップが実行されます。

エラー処理

SOAP 応答の SOAP Fault 要素には、Web サービスを呼び出す際に発生するエラーが含まれます。

バッチ Web サービス操作を呼び出している間、SOAP Fault を取得するための適切なエラー処理方法が実装されている必要があります。この方法は、ツールキットによって異なります。

Web サービスツールキットには、Fault 要素から `faultcode` および `faultstring` フィールドを取得するための（例外）処理方法が用意されています。ただし、エラーコードおよびエラー明細を取得するために、詳細要素フィールドを解析するための XML パーサーが必要になる場合があります。

プロキシオブジェクト

ログイン演算を呼び出すと、指定したリポジトリ名およびユーザー名に対するセッションが作成されます。このセッションは、Login 演算呼び出しによって取得したセッション ID によって識別されます。セッション ID は Metadata プロキシオブジェクトに対応します。Metadata プロキシオブジェクトは、セッション ID が有効な限り有効です。Login 操作を呼び出した後は、対応する Metadata プロキシオブジェクトおよび Data Integration プロキシオブジェクトと共に、セッション ID が無効になります。

バッチ Web サービスの Java クライアントアプリケーション

この節では、Axis Web サービスツールキットを使用し、Java でクライアントアプリケーションを作成する手順について説明します。

注: 次の節にあるサンプルコードの一部は、Web Services Hub に付属のバッチ Web サービスのサンプルプログラムから取り出したものです。PowerCenter Web サービスのサンプルプログラムは、次のディレクトリで確認できます。

<PowerCenterInstallationDir>/server/samples/WebServices/samples/BatchWebServices/axis

Axis でのクライアントプロキシクラスの生成

Java によるクライアントプロキシクラスは、Axis Web サービスツールキットを使用して生成できます。このツールキットを使用するには、CLASSPATH 環境変数に axis.jar ファイルを追加します。

Java によるクライアントプロキシクラスを生成する手順

1. Web Services Hub Console から、Metadata Web サービスおよび Data Integration Web サービスの WSDL ファイルをダウンロードします。

エンドポイント URL の Web Services Hub で、WSDL にの正しいホスト名およびポート番号が設定されていることを確認します。エンドポイント URL が正しくない場合は、WSDL の definitions \service\port 階層にある address 要素を更新します。

2. 次のコマンドを使用して、クライアントプロキシクラスを生成します。

```
java org.apache.axis.wsdl.WSDL2Java --NStoPkg  
http://www.informatica.com/wsh=ProxyClasses -W <WSDLFile>
```

-W オプションは、ラップされたドキュメントリテラルサービスのためのサポートを無効にします。

たとえば、WSDL ファイルの名前が Metadata.wsdl および DataIntegration.wsdl の場合は、次のコマンドを実行します。

```
java org.apache.axis.wsdl.WSDL2Java --NStoPkg  
http://www.informatica.com/wsh=ProxyClasses -W Metadata.wsdl  
java org.apache.axis.wsdl.WSDL2Java --NStoPkg  
http://www.informatica.com/wsh=ProxyClasses -W DataIntegration.wsdl
```

これらのコマンドによって、ProxyClasses パッケージにクライアントプロキシクラスが生成されます。コマンドが生成するプロキシクラスは、以下のとおりです。

- **MetadataInterface.java**。Metadata Web サービスのインタフェースが含まれています。
- **DataIntegrationInterface.java**。Data Integration Web サービスのインタフェースが含まれています。

Axis での初期化

クライアントアプリケーションは、Metadata Web サービスおよび Data Integration Web サービスに対する呼び出しを実行する前に、初期化の手順を実行する必要があります。

初期化を実行するには、以下の手順を実行します。

1. MetadataService オブジェクトおよび DataIntegrationService オブジェクトは、サービスロケータクラスをインスタンス化して作成します。

```
MetadataService mdService = new MetadataServiceLocator();  
DataIntegrationService diService = new DataIntegrationServiceLocator();
```

2. MetadataInterface オブジェクト (MWSPProxy) を手順 1 で作成した MetadataService オブジェクトから取得します。

Metadata.wsdl にある Metadata Service のエンドポイント URL が正しい場合、次のように MWSPProxy オブジェクトを取得します。

```
MWSPProxy=mdService.getMetadata();
```

それ以外の場合は、次のように MWSPProxy オブジェクトを取得します。

```
MWSPProxy=mdService.getMetadata(new java.net.URL(MWS_URL));
```

MWS_URL は、Metadata Web サービスのエンドポイント URL を含む変数です。

MWSPProxy オブジェクトを使用して、Metadata Web Service の操作を呼び出します。

3. DataIntegrationInterface オブジェクト (DIWSPProxy) を手順 1 で作成した DataIntegrationService オブジェクトから取得します。

DataIntegration.wsdl にあるサービスのエンドポイント URL が正しい URL であれば、次の手順で DIWSPProxy オブジェクトを取得します。

```
DIWSPProxy=diService.getDataIntegration();
```

それ以外の場合は、次のようにして DIWSPProxy オブジェクトを取得します。

```
DIWSPProxy=diService.getDataIntegration(new java.net.URL(DIWS_URL));
```

DIWS_URL は、Data Integration Web サービスのエンドポイント URL を含む変数です。

DIWSPProxy オブジェクトを使用して、Data Integration Web Service の演算を呼び出します。

4. MWSPProxy オブジェクトを使用してログイン演算を呼び出して、クライアントアプリケーションのユーザーアカウントに対するセッション ID を作成します。ログイン演算では、オブジェクト LoginRequest でラップされたドメイン、リポジトリ、ユーザー名、およびパスワードが処理され、セッション ID が返されます。

```
LoginRequest loginReq = new LoginRequest();  
loginReq.setRepositoryDomainName(REPO_DOMAIN_NAME);  
loginReq.setRepositoryName(REPO_NAME);  
loginReq.setUserName(USER_NAME);  
loginReq.setPassword(PASSWORD);  
String sessionId = MWSPProxy.login(loginReq);
```

REPO_DOMAIN_NAME は PowerCenter ドメイン名を含む文字列であり、REPO_NAME はドメイン内のリポジトリ名を含む文字列であり、USER_NAME はリポジトリに対して有効なユーザー名を含む文字列であり、PASSWORD はユーザーがリポジトリにログインするためのパスワードを含む文字列です。

5. MWSPProxy オブジェクトおよび DIWSPProxy オブジェクトを、セッション ID のリポジトリ名およびユーザー名と関連付けます。MWSPProxy または DIWSPProxy オブジェクトを使用してバッチ Web サービスに対して行われる後続の要求はすべて、セッション ID 内のリポジトリおよびユーザー名を使用します。

```
((org.apache.axis.client.Stub)MWSPProxy).setHeader(createSessionHeader(sessionID));  
((org.apache.axis.client.Stub)DIWSPProxy).setHeader(createSessionHeader(sessionID));
```

Axis でのセッションメンテナンス

Web Services Hub では、キャッシュリソースに対するセッションメンテナンスが必要です。SOAP メッセージの SOAP ヘッダには、セッションメンテナンスを容易に行うためのセッション情報が保持されません。

セッションメンテナンスを実行するには、以下の手順を実行します。

1. MWSPProxy オブジェクトを使用したログイン操作の呼び出しに対する応答から、ルート要素名 Context および名前空間 `http://www.informatica.com/wsh` が含まれる SOAP ヘッダを抽出します。この SOAP ヘッダには、Web Services Hub によって送信されたセッション ID が含まれます。

```
/** Create session ID in the Soap message header */
public static SOAPHeaderElement createSessionHeader(String sessID) throws SOAPException
{
    String WSSE_NS = "http://www.informatica.com/";
    String WSSE_PREFIX = "infa";
    Name hdrname = SOAPFactory.newInstance().createName("Context", WSSE_PREFIX, WSSE_NS);
    SOAPHeaderElement header = new SOAPHeaderElement(hdrname);
    SOAPElement token = header.addChildElement("SessionId");
    token.addTextNode(sessID);
    return header;
}
```

2. MWSPProxy オブジェクトを使用したすべての後続要求に対して、SOAP ヘッダ内のこのセッション ID を送信します。ログイン操作を呼び出した後に、以下のように MWSPProxy オブジェクトで SOAP ヘッダを 1 回設定します。

```
((org.apache.axis.client.Stub) MWSPProxy).setHeader(createSessionHeader(sessionID));
```

3. 以下のように、同じ SOAP ヘッダを持つ DIWSPProxy オブジェクトの SOAP ヘッダを設定します。

```
((org.apache.axis.client.Stub) DIWSPProxy).setHeader(createSessionHeader(sessionID));
```

Axis における操作の呼び出し

MWSPProxy オブジェクトと DIWSPProxy オブジェクトを使用して、Metadata Web Service の演算と Data Integration Web Service の演算を呼び出す準備が整いました。

例えば、`getAllDIServers` 操作を呼び出して、Integration Services のリストを取得できます。

```
DIServerInfoArray servers = MWSPProxy.getAllDIServers(null);
if (servers.getDIServerInfo() != null) {
    for(int i=0; i < servers.getDIServerInfo().length; i++) {
        System.out.println("("+(i+1)+") "+servers.getDIServerInfo(i).getName());
    }
}
```

`pingDIServer` 操作を呼び出して、Integration Service の状態を確認できます。

```
DIServiceInfo diInfo = new DIServiceInfo();
diInfo.setDomainName(DI_DOMAIN_NAME);
diInfo.setServiceName(SERVICE_NAME);
PingDIServerRequest pingReq = new PingDIServerRequest();
pingReq.setDIServiceInfo(diInfo);
pingReq.setTimeout(100);
EPingState eps = DIWSPProxy.pingDIServer(pingReq);
```

`DI_DOMAIN_NAME` は、Integration Service を含むドメインの名前を含む変数です。 `SERVICE_NAME` は、Integration Service 名を含む変数です。

Axis でのクリーンアップ

クリーンアップ操作により、クライアントアプリケーションによって取得された Web Services Hub のリソースが開放されます。クリーンアップしてリソースを開放するには、MWSPProxy オブジェクトを使用して `Logout` 操作を呼び出します。

```
MWSPProxy.logout(null);
```

Axis におけるエラー処理

Axis でクライアントアプリケーションにエラー処理を実装するには、try ブロックに FaultDetails オブジェクトを捕獲するためのコードを記述します。FaultDetails クラスはクライアントプロキシの一部として生成されます。

try ブロックにある次のコードを使用して、FaultDetails オブジェクトを取得できます。

```
try {  
    // Code for steps explained above.  
}  
catch(FaultDetails fault) {  
    // Display fault code  
    System.out.println("fault code : " + fault.getFaultCode());  
    // Display fault string  
    System.out.println("fault string : " + fault.getFaultString());  
    // Display error code  
    System.out.println("error code is : " + fault.getErrorCode());  
    // Display extended details  
    System.out.println("extended detail is : " + fault.getExtendedDetails());  
}
```

バッチ Web サービスの C#クライアントアプリケーション

この節では、.NET Web サービスツールキットを使用して、C#でクライアントアプリケーションを作成する手順について説明します。

注: 次のセクションにあるサンプルコードの一部は、バッチ Web サービスのサンプルプログラムから取り出したものです。サンプルプログラムは、次のディレクトリで確認できます。

```
<PowerCenterInstallationDir>\server\samples\WebServices\samples\BatchWebServices\dotnet\csharp
```

.NET でのクライアントプロキシクラスの生成

Microsoft の .NET Web サービスツールキットを使用すると、C#での Web Services Hub のクライアントプロキシクラスを作成できます。

クライアントプロキシクラスを C#で生成する手順

1. Web Services Hub Console から、Metadata Web サービスおよび Data Integration Web サービスの WSDL ファイルをダウンロードします。エンドポイント URL の Web Services Hub で、WSDL にの正しいホスト名およびポート番号が設定されていることを確認します。エンドポイント URL が正しくない場合は、WSDL の definitions\service\port 階層にある address 要素を更新します。
2. 次のコマンドを使用して、クライアントプロキシクラスを生成します。

```
wsdl <WSDLFile>
```

たとえば、WSDL ファイルの名前が Metadata.wsdl および DataIntegration.wsdl の場合は、次のコマンドを実行します。

```
wsdl Metadata.wsdl
```

```
wsdl DataIntegration.wsdl
```

コマンドが生成するプロキシクラスは、以下のとおりです。

- **MetadataService.cs**。Metadata Web サービスのインタフェースが含まれています。
- **DataIntegrationService.cs**。Data Integration Web サービスのインタフェースが含まれています。

.NET での初期化

クライアントアプリケーションは、Metadata Web サービスおよび Data Integration Web サービスに対する呼び出しを実行する前に、初期化の手順を実行する必要があります。

初期化を実行するには、以下の手順を実行します。

1. MetadataService クラスのオブジェクト (MWSPProxy) をインスタンス化します。

```
MWSPProxy= new MetadataService();
```

Metadata.wsdl にある Metadata Service のエンドポイント URL が間違っていれば、次のコードで URL を設定することができます。

```
MWSPProxy.Url = MWS_URL;
```

MWS_URL は、Metadata Web サービスのエンドポイント URL を含む変数です。

MWSPProxy オブジェクトを使用して、Metadata Web Service の操作を呼び出します。

2. DataIntegrationService クラスのオブジェクト (DIWSPProxy) をインスタンス化します。

```
DIWSPProxy= new DataIntegrationService ();
```

DataIntegration.wsdl にある Data Integration Service のエンドポイント URL が間違っていれば、次のコードで URL を設定することができます。

```
DIWSPProxy.Url = DIWS_URL;
```

DIWS_URL は、Data Integration Web サービスのエンドポイント URL を含む文字列です。

DIWSPProxy オブジェクトを使用して、Data Integration Web Service の操作を呼び出します。

3. MWSPProxy オブジェクトを使用してログイン演算を呼び出して、クライアントアプリケーションのユーザーアカウントに対するセッション ID を作成します。ログイン演算では、オブジェクト LoginRequest でラップされたドメイン、リポジトリ、ユーザー名、およびパスワードが処理され、セッション ID が返されます。

```
LoginRequest loginReq = new LoginRequest();  
loginReq.RepositoryDomainName = REPO_DOMAIN_NAME;  
loginReq.RepositoryName = REPO_NAME;  
loginReq.UserName = USER_NAME;  
loginReq.Password = PASSWORD;  
String sessID = MWSPProxy.Login(loginReq);
```

REPO_DOMAIN_NAME は PowerCenter ドメイン名を含む文字列であり、REPO_NAME はドメイン内のリポジトリ名を含む文字列であり、USER_NAME はリポジトリに対して有効なユーザー名を含む文字列であり、PASSWORD はユーザーがリポジトリにログインするためのパスワードを含む文字列です。

4. MWSPProxy オブジェクトおよび DIWSPProxy オブジェクトを、セッション ID のリポジトリ名およびユーザー名と関連付けます。MWSPProxy または DIWSPProxy オブジェクトを使用してバッチ Web サービスに対して行われる後続の要求はすべて、セッション ID 内のリポジトリおよびユーザー名を使用します。

```
MWSPProxy.Context.SessionId = sessID;  
DIWSPProxy.Context.SessionId = sessID;
```

.NET でのセッションメンテナンス

Web Services Hub では、キャッシュリソースに対するセッションメンテナンスが必要です。SOAP メッセージの SOAP ヘッダには、セッションメンテナンスを容易に行うためのセッション情報が保持されません。

追加手順を実行する必要はありません。.NET のクライアントプロキシクラスで、セッションメンテナンスを処理します。

.NET における演算の呼び出し

MWSPProxy オブジェクトと DIWSPProxy オブジェクトを使用して、Metadata Web Service の操作と Data Integration Web Service の操作を呼び出す準備が整いました。

例えば、getAllDIServers 操作を呼び出して、Integration Services のリストを取得できます。

```
DIServerInfo[] servers = MWSPProxy.GetAllDIServers(null);
if (servers != null) {
    for(int i=0; i < servers.Length ; i++) {
        Console.WriteLine("("+(i+1)+") "+servers[i].Name);
    }
}
```

pingDIServer 操作を呼び出して、Integration Service の状態を確認できます。

```
PingDIServerRequest pingReq = new PingDIServerRequest();
pingReq.Timeout = (PING_TIME_OUT);
DIServiceInfo diInfo1 = new DIServiceInfo();
diInfo1.DomainName = DI_DOMAIN_NAME;
diInfo1.ServiceName = DI_SERVICE_NAME1;
pingReq.DIServiceInfo = diInfo1;
EPingState pingResult = DIWSPProxy1.pingDIServer(pingReq);
```

DI_DOMAIN_NAME は、Integration Service を含むドメインの名前を含む変数です。DI_SERVICE_NAME は、Integration Service 名を含む変数です。

.NET におけるエラー処理

.NET でクライアントアプリケーションにエラー処理を実装するには、try ブロックに SOAP Exception オブジェクトを捕獲するためのコードを記述します。SOAP Exception クラスは.NET framework SDK に含まれています。

try ブロックにある次のコードを使用して、SOAP Exception オブジェクトを取得できます。

```
try {
    //Code for steps explained above.
}
catch(SoapException fault) {
    // Display fault code
    Console.WriteLine("fault code is : " + fault.Code);
    // Display fault string
    Console.WriteLine("fault string is : " + fault.Message);
    // Parsing detail element
    XmlNode detail = fault.Detail;
    XmlElement WSHFaultDetails = detail["WSHFaultDetails", "http://www.informatica.com/PowerCenter"];
    XmlElement ErrorCode= WSHFaultDetails ["ErrorCode"];
    XmlElement ExtendedDetails= WSHFaultDetails ["ExtendedDetails"];
    // Display error code
    Console.WriteLine ("error code is : " + ErrorCode.InnerText);
    // Display extended details
    Console.WriteLine ("extended detail is : " + ExtendedDetails.InnerText);
}
```

リアルタイム Web サービスのクライアントアプリケーション

リアルタイム Web サービスのクライアントアプリケーションには、次の要素が含まれています。

- Web サービスワークフロー
- クライアントプロキシクラス
- 初期化
- 演算の呼び出し
- エラー処理

Web サービスワークフロー

リアルタイム Web サービスのクライアントアプリケーションを作成して、Web サービスワークフローを実行します。クライアントアプリケーションを作成する前に、PowerCenter でマッピングとワークフローを作成する必要があります。クライアントアプリケーションがワークフローを実行できるようにするには、ワークフローで次のオプションを有効にします。

- **Web サービス。**ワークフローを Web サービスワークフローにするためには、[Web サービス] オプションを有効にします。
- **実行可能。**クライアントアプリケーションが Web サービスワークフローを実行できるようにするには、[実行可能] オプションを有効にします。
- **ビジブル。**Web Services Hub が Web Services Hub Console の Web サービスに対して WSDL を公開するように、[ビジブル] オプションを有効にします。

クライアントプロキシクラスの生成

PowerCenter で作成するリアルタイム Web サービスを使用するには、アクセスする Web サービスの WSDL からクライアントプロキシクラスを生成する必要があります。

クライアントプロキシクラスを生成するには、次の手順を完了します。

1. 開発に使用するプラットフォームと言語に対応した Web サービスツールキットを選択します。
2. Web Services Hub Console からリアルタイム Web サービス用の WSDL をダウンロードします。
3. Web サービスツールキットを使用し、WSDL からクライアント側プロキシクラスを生成します。プロキシクラス生成の詳細については、Web サービスツールキットのドキュメントを参照してください。各ツールキットは特定の方法でクライアントプロキシクラスを生成します。

初期化

クライアントアプリケーションは、Web サービス操作への呼び出しを作成できるようにするために、クライアントプロキシクラスの Web サービスオブジェクトをインスタンス化して、Web サービスのポートを取得する必要があります。

操作の呼び出し

Web サービス操作を呼び出すために、クライアントアプリケーションは要求オブジェクトを作成して、ポート操作に渡す必要があります。Web サービスが応答を送り返す場合、クライアントアプリケーションは必要に応じて応答を処理する必要があります。

エラー処理

リアルタイム Web サービスのクライアントアプリケーションでのエラー処理は、バッチ Web サービスのクライアントアプリケーションの場合と同じです。SOAP 応答の SOAP Fault 要素には、Web サービスを呼び出す際に発生するエラーが含まれます。クライアントアプリケーションは、SOAP Fault を取得するための適切なエラー処理方法を実装する必要があります。

リアルタイム Web サービスの Java クライアントアプリケーション

この節では、Axis Web サービスツールキットを使用して PowerCenter リアルタイム Web サービスを呼び出す Java クライアントアプリケーションを作成する手順について説明します。Axis Web サービスツールキットの使用法の詳細については、以下の Apache Web サイト上のドキュメントを参照してください。

<http://ws.apache.org/axis/java/user-guide.html>

PowerCenter Web サービスワークフローを呼び出すクライアントアプリケーションを作成できるようにするには、最初に Web サービスワークフローを作成して Web サービスの WSDL を生成する必要があります。Web サービス WSDL に基づいて、クライアントアプリケーションを作成します。

PowerCenter Web サービスを作成して WSDL を生成する手順

1. Web サービスワークフローのマッピングを作成します。Web サービスクライアントからメッセージを受信し、データを変換した後で、Web サービスクライアントに応答を送信するか、または PowerCenter でサポートされている任意のターゲットに書き込むようにするため、マッピングを作成することができます。
2. ワークフローを作成し、それを Web サービスとして有効にします。マッピングを実行するワークフローを作成して、ワークフロープロパティで [Web サービス] オプションを有効にします。PowerCenter の外部のクライアントアプリケーションがワークフローを実行できるようにするために、[実行可能] オプションを選択します。
3. Web サービスワークフローの WSDL を特定してダウンロードします。Web サービスワークフローを作成する場合、PowerCenter は Web サービスの WSDL を生成します。Web サービスを表示可能に設定すると、Web サービスに関連付けられた Web Services Hub のコンソール上に WSDL を表示できます。

Web サービスを作成した後、クライアントアプリケーションを開発して Web サービスワークフローを実行できます。

関連項目：

- [「Web サービスワークフローの作成および設定」 \(ページ 96\)](#)
- [「\[Web サービス\] セクション」 \(ページ 19\)](#)

リアルタイム Web サービスのクライアントアプリケーションの作成

リアルタイム Web サービスを呼び出すクライアントアプリケーションを作成する手順

1. Web サービスのクライアントプロキシクラスを生成します。
プロキシクラスを作成した後に、Java アプリケーションを作成して Web サービスを呼び出します。Java アプリケーション内部で次の手順を実行します。
2. Web サービスオブジェクトを初期化します。

3. 要求オブジェクトを作成します。
4. 要求オブジェクトをポート操作に渡して、応答を処理します。

注: 次の節にあるサンプルコードの一部は、複数の行ルックアップのためにリアルタイム Web サービスから取り出されたものです。例は、次のディレクトリで確認できます。

```
<PowerCenterInstallationDir>/server/samples/WebServices/samples/RealTimeWebServices/  
UnprotectedWebServices/axis/CustomLookup_MULTIPLEROW
```

手順 1. Axis でのクライアントプロキシクラスの生成

Axis Web サービスツールキットを使用することによって、Web サービス WSDL の Java クライアントプロキシクラスを生成できます。具体的には、WSDL2Java ツールを実行すると、Java プロキシクラスファイルを生成できます。

エンドポイント URL の Web サービスに対して、WSDL に正しいホスト名およびポート番号が設定されていることを確認します。エンドポイント URL が正しくない場合は、WSDL の\definitions\service\port 階層にある address 要素を更新します。

次のコマンドを使用して、クライアントプロキシクラスを生成します。

```
java org.apache.axis.wsdl.WSDL2Java -W <WSDLFile>
```

例えば、SampleWS.wsdl という名前の WSDL では次のコマンドを実行します。

```
java org.apache.axis.wsdl.WSDL2Java -W SampleWS.wsdl
```

-W オプションは、ラップされたドキュメントリテラルサービスのためのサポートを無効にします。

WSDL2Java は、WSDL に定義された各データタイプのクラスを生成します。デフォルトで、WSDL2Java は、WSDL 内の名前空間に基づいてパッケージ名を生成します。通常、名前空間が *http://x.y.com* または *urn:x.y.com* の形式の場合、該当するパッケージは *com.y.x* になります。

手順 2. Web サービスオブジェクトの初期化

Web サービス操作を呼び出すためには、クライアントプロキシクラスに Web サービスオブジェクトを作成して、Web サービスのポートを取得します。

Web サービスオブジェクトを作成するには、サービスロケータクラスをインスタンス化します。サンプルプログラムでは、次のコードでサービスロケータをインスタンス化します。

```
CustomerLookup_MULTIPLEROW service = new CustomerLookup_MULTIPLEROWLocator();
```

Web サービスのポートを取得するには、ポートタイプに作成されたプロキシクラスを使用します。サンプルプログラムでは、次のコードで Web サービスのポートを取得します。

```
CustomerLookup_MULTIPLEROWPort port =  
    service.getCustomerLookup_MULTIPLEROWPort(new java.net.URL(END_POINT_URL));
```

変数 END_POINT_URL には、WSDL の URL が含まれます。

手順 3. 要求オブジェクトの作成

Web サービスに渡すために要求オブジェクトと必須パラメータを作成する必要があります。サンプルクライアントアプリケーションでは、次のコードでルックアップ要求オブジェクトを作成します。

```
CustomerLookupRequest request = new CustomerLookupRequest();  
request.setCustomerID_in(CustomerID);
```

手順 4. 要求の送信と応答の処理

要求オブジェクトを作成した後、それをポート操作に渡します。Web サービスが応答を送り返します。ユーザーの要件に基づき、応答を処理できます。

サンプルクライアントアプリケーションで、次のコードが要求オブジェクトをポートに渡して応答を表示します。

```
CustomerLookupResponse[] response = port.customerLookup_MULTIPLEROWOperation(requestOperation);
System.out.println();
if (response[0].getCustomerID_out() == 0)
{
    System.out.println("Customer(s) with the ID as " + CustomerID + " does not exist!!!");
}
else
{
    System.out.println("***** Customer(s) that matches with the Customer ID is/are ...");
    for (int i = 0; i < response.length; i++)
    {
        System.out.println("***** Customer ID: " + response[i].getCustomerID_out());
        System.out.println("***** Customer Name: " + response[i].getCustomerName_out());
        System.out.println("***** Customer Age: " + response[i].getCustomerAge_out());
        System.out.println("***** Customer Gender: " +
            response[i].getCustomerGender_out());
        System.out.println("***** Customer Address: " +
            response[i].getCustomerAddress_out());
        if (i < response.length - 1) System.out.println ();
    }
}
```

パラメータ配列の使用

PowerCenter では、パラメータはセッション間で変更できる値（データベース接続、ソースファイルまたはターゲットファイルなど）を表します。ワークフローまたはセッションに関連付けられたパラメータを作成すると、ワークフローまたはセッションを実行するたびに柔軟に対応できます。

Web サービスクライアントアプリケーションの場合は、パラメータファイルまたはパラメータ配列内に、ワークフローまたはセッションに関連付けられたパラメータの値を定義できます。パラメータファイルでパラメータを使用するには、クライアントアプリケーションでパラメータファイル名を指定します。パラメータファイルは、Integration Service にアクセス可能でなくてはなりません。パラメータ配列を使用するには、クライアントアプリケーションでパラメータ配列の要素にパラメータ値を指定します。

たとえば、ワークフローまたはタスクを開始する要求によって、パラメータファイルの名前、またはパラメータ配列のパラメータおよび値のリストを含むワークフローまたはタスクに関連付けられたパラメータを指定できます。

パラメータ配列の定義

SOAP 要求のパラメータ定義は、パラメータの範囲、名前、および値で構成されます。Integration Service でワークフローまたはタスクが実行されている場合、パラメータファイルのパラメータの場合と同様に、配列のパラメータが使用されます。

WSDL には、以下のようなパラメータ配列要素の定義が含まれます。

```
<complexType name="Parameter">
  <sequence>
    <element name="Scope" type="xsd:string" />
```



```

        <element name="Name" type="xsd:string" />
        <element name="Value" type="xsd:string" />
    </sequence>
</complexType>

<complexType name="ParameterArray">
    <sequence>
        <element maxOccurs="unbounded" minOccurs="0" name="Parameters"
            nillable="true" type="impl:Parameter" />
    </sequence>
</complexType>

```

例えば、パラメータファイルには以下のパラメータが含まれます。

```

[s_m_A]
$a=1
$b=2
$c=3
[WSH_Folder.s_m_B]
$d=4

```

パラメータ配列に同じパラメータを持つ StartWorkflow 操作の Web サービス呼び出しに対する SOAP 要求には、以下の要素が含まれます。

```

<StartWorkflow>
...
    <Parameters>
        <Parameter>
            <Scope>s_m_A</Scope>
            <Name>$a</Name>
            <Value>1</Value>
        </Parameter>
        <Parameter>
            <Scope>s_m_A</Scope>
            <Name>$b</Name>
            <Value>2</Value>
        </Parameter>
        <Parameter>
            <Scope>s_m_A</Scope>
            <Name>$c</Name>
            <Value>3</Value>
        </Parameter>
        <Parameter>
            <Scope>WSH_Folder.s_m_B</Scope>
            <Name>$d</Name>
            <Value>4</Value>
        </Parameter>
    ...
</StartWorkflow>

```

WorkflowRequest タイプおよび TaskRequest タイプには、ParameterArray 要素が含まれます。パラメータ配列には任意の数のパラメータを指定できます。

Axis での Web サービスクライアントアプリケーションの以下のサンプルコードは、WorkflowRequest でのパラメータ配列の作成方法を示しています。

```

Parameter[] parameters = new Parameter[4];

Parameter param1 = new Parameter();
Param1.setScope("s_m_A");
Param1.setName("$a");

```

```

Param1.setValue("1");
Parameters[0] = param1;

Parameter param2 = new Parameter();
Param2.setScope("s_m_A");
Param2.setName("$b");
Param2.setValue("2");
Parameters[1] = param2;

Parameter param3 = new Parameter();
Param3.setScope("s_m_A");
Param3.setName("$c");
Param3.setValue("3");
Parameters[2] = param3;

Parameter param4 = new Parameter();
Param4.setScope("WSH_Folder.s_m_B");
Param4.setName("$d");
Param4.setValue("4");
Parameters[3] = param4;

WorkflowRequest wfReq = new WorkflowRequest();
wfReq.setParameters(parameters);

```

以下の操作で、パラメータ配列を使用できます。

- startWorkflow
- startWorkflowFromTask
- recoverWorkflow
- startTask

パラメータ配列の使用に関する規則とガイドライン

Web サービス要求でパラメータ配列を使用する場合は、以下の規則とガイドラインを使用します。

- **パラメータファイルまたはパラメータ配列を使用する。** Web サービス操作の呼び出しを実行する場合には、SOAP リクエストでパラメータファイル名とパラメータ配列を指定しないでください。 SOAP リクエストでパラメータファイルとパラメータ配列の両方を指定すると、Web Services Hub は、要求がパラメータリストとパラメータファイルを指定していることを示すフォールトメッセージ警告を返します。
- **パラメータファイルとパラメータ配列が定義されている場合、Integration Service は SOAP リクエストのパラメータ配列のパラメータ値を使用します。** 以下の条件が該当する場合、Integration Service は、パラメータ配列で定義されているパラメータ値を使用します。
 - Web サービス要求でパラメータ配列を指定してワークフローを開始している。
 - ワークフロープロパティで、ワークフローに関連付けられたパラメータファイルが定義されている。

クライアント要求へのセキュリティの追加

Web Services Hub では、Web サービスに対して以下のタイプのセキュリティが使用されます。

- **ユーザークレデンシャル。** クライアントが要求に対するクレデンシャルを取得するには、実行する Web サービスを含む PowerCenter リポジトリにログインする必要があります。 ログインによって、Web サービスクライアントが SOAP リクエストに含める必要があるセッション ID が生成されます。
Web Services Hub では、バッチ Web サービスに対してこのセキュリティオプションが使用されます。 バッチ Web サービスを呼び出すクライアントアプリケーションは、他の操作を呼び出す前にリポジトリにログインする必要があります。

- **ユーザー名トークン。** ユーザー名トークンを使用し、OASIS Web サービスセキュリティ標準に基づく Web サービスセキュリティ。これには、SOAP メッセージの内容の整合性とセキュリティを確保する SOAP 拡張のセットが含まれています。

ユーザー名トークンは、保護されている Web サービスのデフォルトのセキュリティオプションです。デフォルトでは、保護されている Web サービスの Web Services Hub で生成された WSDL には、UsernameToken 要素を持つセキュリティヘッダが含まれています。

OASIS Web サービスセキュリティ標準の詳細については、OASIS Web サイトにある Web サービスセキュリティ仕様を参照してください。

http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=ws-sx

注: クライアントアプリケーションが Web サービスを実行するためにログイン要求を送信した場合は、応答が受信された後で明示的にログアウト要求を送信します。ログイン要求に対応するログアウト要求がないと、Repository Service と Web Services Hub プロセスでのメモリリークの原因になります。

SOAP リクエストの UsernameToken

Web Services Hub で生成された WSDL に基づいてクライアントアプリケーションを作成すると、要求オブジェクトではデフォルトでヘッダに UsernameToken 要素が含まれます。

SOAP リクエスト内の UsernameToken 要素には、以下のパスワードセキュリティのいずれかを指定することができます。

- **プレーンテキストパスワード。** プレーンテキストのパスワードが含まれます。
- **ハッシュパスワード。** ハッシュ関数 MD5 または SHA-1 を使用してハッシュされた暗号化パスワードが含まれます。
- **ダイジェストパスワード。** ナンス値とタイムスタンプによってハッシュされた暗号化パスワードが含まれます。

UsernameToken のパスワード要素にユーザーパスワードが含まれます。

パスワード要素には、使用されるパスワードセキュリティのタイプを示すタイプ属性が指定されます。タイプ属性を省略すると、パスワードタイプはデフォルトで *PasswordText* になります。

注: Informatica ドメインで Kerberos ネットワーク認証が使用されている場合、SOAP リクエストでハッシュパスワードやダイジェストパスワードを使用することはできません。

プレーンテキストパスワード

UsernameToken 要素には、次の子要素が含まれます。

- **Username 要素。**PowerCenter のネイティブセキュリティドメインまたは任意の LDAP セキュリティドメインのユーザー名が含まれます。デフォルトのセキュリティドメインはネイティブセキュリティドメインです。ユーザー名がネイティブセキュリティドメインに属する場合、Username 要素ではセキュリティドメインの名前を必要としません。ユーザー名が LDAP セキュリティドメインに属する場合は、ユーザー名の前にセキュリティドメインの名前とスラッシュ (/) を付ける必要があります。

以下の Username 要素の例の表は、ユーザーアカウントのセキュリティドメインを示すために使用する形式を示しています。

Username 要素の値	セキュリティドメイン
<code><UsernameToken> <Username>Native/Administrator</Username> <Password>Administrator</Password> </UsernameToken></code>	ネイティブ
<code><UsernameToken> <Username>/Administrator</Username> <Password>Administrator</Password> </UsernameToken></code>	ネイティブ
<code><UsernameToken> <Username>Administrator</Username> <Password>Administrator</Password> </UsernameToken></code>	ネイティブ
<code><UsernameToken> <Username>LDAPAdm/Administrator</Username> <Password>Administrator</Password> </UsernameToken></code>	LDAPAdm という名前の LDAP セキュリティドメイン

- **パスワード要素。**プレーンテキストのパスワードが含まれます。パスワード要素のタイプ属性は、省略するか *PasswordText* に設定できます。

ハッシュパスワード

UsernameToken 要素には、次の子要素が含まれます。

- **Username 要素。**PowerCenter のネイティブセキュリティドメインのユーザー名が含まれます。
- **パスワード要素。**ハッシュパスワードが含まれます。このパスワードは、MD5 ハッシュ関数または SHA-1 ハッシュ関数でハッシュ化して Base64 にエンコードする必要があります。パスワード要素のタイプ属性は、省略するか“PasswordText”に設定できます。

以下のコードは、ハッシュパスワードを使用する要求のセキュリティヘッダの例を示しています。

```
<soapenv:Header>  
  <!-- UsernameTokens -->  
  <inf:Security>  
    <UsernameToken>  
      <Username>Native/Administrator</Username>
```

```

        <Password>Ntm58Cxf7SB0QAz30lsTq1nv-D7</Password>
    </UsernameToken>
</inf:Security>
</soapenv:Header>

```

サードパーティツールを使用したハッシュパスワードの作成

OpenSSL などのサードパーティツールと Java MessageDigest クラスを使用して、ハッシュパスワードを作成できます。

Unix での OpenSSL

OpenSSL を使用して UNIX マシンにハッシュパスワードを作成するには、メッセージダイジェストコマンド dgst を使用して OpenSSL を実行します。

次の例は、MD5 ハッシュ関数を使用し、Base64 でエンコードされた、パスワード文字列 Administrator のハッシュパスワードを作成する方法を示します。

```
echo -n "Administrator" | openssl dgst -md5 -binary | openssl base64
```

次の例は、SHA-1 ハッシュ関数を使用し、Base64 でエンコードされた、パスワード文字列 Administrator のハッシュパスワードを作成する方法を示します。

```
echo -n "Administrator" | openssl dgst -sha1 -binary | openssl base64
```

注: これらの例の echo コマンドでは、文字列に改行文字が追加されます。コマンド中の -n オプションにより、改行文字が削除されます。

Windows での OpenSSL

OpenSSL を使用して Windows マシンにハッシュパスワードを作成するには、メッセージダイジェストコマンド dgst を使用して OpenSSL を実行します。

次の例は、SHA-1 ハッシュ関数を使用し、Base64 でエンコードされた、ハッシュパスワードを作成する方法を示します。

```
openssl dgst -sha1 -binary -out <output file name> <input file name>
openssl enc -base64 -in <output file name>
```

入力ファイルはハッシュするパスワード文字列を含みます。OpenSSL はハッシュパスワードを出力ファイルに書き込みます。

Java MessageDigest

次の例は、Java MessageDigest クラスを使用して、MD5 ハッシュ関数を使用し、Base64 でエンコードされたハッシュパスワードを作成する方法を示します。

```

public static String md5hash(String password) throws Exception{
    MessageDigest digest = java.security.MessageDigest.getInstance("MD5");
    digest.reset();
    digest.update(password.getBytes());
    byte[] hash = digest.digest();
    return new String(org.apache.commons.codec.binary.Base64.encodeBase64(hash));
}

```

次の例は、Java MessageDigest クラスを使用して、SHA-1 ハッシュ関数を使用し、Base64 でエンコードされたハッシュパスワードを作成する方法を示します。

```

public static String sha1hash(String password) throws Exception{
    MessageDigest digest = java.security.MessageDigest.getInstance("SHA-1");

```

```

    digest.reset();
    digest.update(password.getBytes());
    byte[] hash = digest.digest();
    return new String(org.apache.commons.codec.binary.Base64.encodeBase64(hash));
}

```

ダイジェストパスワード

UsernameToken 要素には、次の子要素が含まれます。

- **Username 要素。**PowerCenter のネイティブセキュリティドメインのユーザー名が含まれます。
- **パスワード要素。**ダイジェストパスワードが含まれます。このパスワードは、Nonce 要素のナンス値および Created 要素のタイムスタンプと連結されているパスワードをハッシュ化して生成された値です。このパスワードは、SHA-1 ハッシュ関数でハッシュ化して Base64 にエンコードする必要があります。

ダイジェストパスワードのセキュリティのために、パスワード要素のタイプ属性は *PasswordDigest* に設定する必要があります。

- **Nonce 要素。**1 回しか使用できないランダムな値であるナンス値が含まれます。
- **Created 要素。**要求が作成された時刻を示すタイムスタンプ値が含まれます。タイムスタンプでは UTC 形式、yyyy-MM-dd'T'HH:mm:ss.SSS'Z'を使用します。例: 2008-08-11T18:06:32.425Z。

SOAP リクエストに含めるナンス値は 1 回しか使用できません。デフォルトでは、Created 要素の値が示すように、要求の作成後 300 秒（5 分）間有効です。クライアントアプリケーションはナンス値が有効である時間内に要求を送信する必要があります。例えば、Created 値が、要求が午前 10:00 に作成されたことを示しているとします。この要求は午前 10:00 から午前 10:05 まで有効です。クライアントアプリケーションが午前 10:00 より前または午前 10:05 より後に Web Services Hub に要求を送信すると、要求とナンス値は無効になり、要求は失敗します。

ダイジェストパスワードでは標準の OASIS パスワードダイジェストアルゴリズムが使用されます。

Password_Digest = Base64 (SHA-1 (nonce + created + password))

ナンス値、タイムスタンプ、およびダイジェストパスワードは、任意のツールを使用して生成できます。

以下のコードは、ダイジェストパスワードを使用する要求のセキュリティヘッダの例を示しています。

```

<soapenv:Header>
  <!-- UsernameTokens -->
  <inf:Security>
    <UsernameToken>
      <Username>Administrator</Username>
      <Password Type="PasswordDigest"> Xty5lCAf5SV00AY30tsYq7nv/DI=</Password>
      <Nonce>KjsaeiuDFKJEwkr4332rL=</Nonce>
      <Created>2008-08-12T01:11:47.013Z</Created>
    </UsernameToken>
  </inf:Security>
</soapenv:Header>

```

サードパーティツールを使用したダイジェストパスワードの作成

Java MessageDigest クラスなどのサードパーティツールを使用してダイジェストパスワードを作成できます。

次の例は、Java MessageDigest クラスを使用し、タイムスタンプとナンス値を使用して、Base64 でエンコードされたダイジェストパスワードを作成する方法を示します。

```
public static String oasisDigest(String password, String nonce) throws Exception{
    SimpleDateFormat sdf = new SimpleDateFormat("yyyy-MM-dd'T'HH:mm:ss.SSS'Z'");
    String created = sdf.format(new Date());
    System.out.println("Created : " + created);
    System.out.println("Nonce : " + new
String(org.apache.commons.codec.binary.Base64.encodeBase64(nonce.getBytes())));
    String toDigest = nonce + created + password;
    MessageDigest digest = java.security.MessageDigest.getInstance("SHA-1");
    digest.reset();
    digest.update(toDigest.getBytes());
    byte[] hash = digest.digest();
    return new String(org.apache.commons.codec.binary.Base64.encodeBase64(hash));
}
```

soapUI などの Web サービステストツールを使用してクライアントアプリケーションをテストする場合、そのツールを使用してクライアント要求用のダイジェストパスワードを生成できます。

Web サービスのソースおよびターゲットに関する作業

Web サービスのソースおよびターゲットの概要

Web サービスワークフロー内で使用するマッピングを作成する場合、マッピングのソースおよびターゲットでは、Web サービスの入力メッセージおよび出力メッセージを定義する必要があります。Web サービスソースは、Web サービス操作の入力メッセージを定義し、Web サービス SOAP 要求のメタデータを表します。Web サービスターゲットは、Web サービス操作の出力メッセージを定義し、Web サービス SOAP 要求のメタデータを表します。

Web サービス記述言語 (WSDL) は、Web サービスの入力および出力メッセージを記述します。作成する Web サービスワークフローの WSDL がある場合は、その WSDL からソース定義およびターゲット定義をインポートできます。WSDL がない場合は、リレーショナルおよびフラットファイルソースまたはターゲット内のカラムに基づいて入力および出力メッセージを作成できます。または、定義するカラムから入力および出力メッセージを作成できます。

Designer を使用して、Web サービスのソース定義およびターゲット定義を作成します。Web サービスのソース定義およびターゲット定義は、以下の方法で作成できます。

- **WSDL から定義をインポートする。** WSDL で定義される操作のソースまたはターゲット定義をインポートします。ソース定義のインポート時に、Designer によって入力メッセージの定義がインポートされます。ターゲット定義のインポート時に、Designer によって出力メッセージおよびフォールトメッセージの定義がインポートされます。
- **定義をリレーショナルまたはフラットファイルのソースおよびターゲットから作成する。** フォルダで定義された、リレーショナルファイルまたはフラットファイルのソースおよびターゲット定義に基づいて、Web サービスのソース定義およびターゲット定義を作成できます。さらに、カラムを手動で定義してデータタイプおよびカラムサイズを指定することもできます。Web サービスのソースとターゲット定義をリレーショナルまたはフラットファイルソースとターゲットから作成する場合、またはカラムを手動で定義する場合、WSDL は必要ありません。

Web サービスのソースおよびターゲットの理解

XML のソースおよびターゲットと同様に、Web サービスのソース定義およびターゲット定義は XML ビューにまとめられています。XML ビューとは、入力メッセージおよび出力メッセージに定義された要素および属性を表すカラムのグループです。

ソース定義およびターゲット定義をインポートすると、Designer によって、WSDL で定義された入力メッセージまたは出力メッセージの要素に基づいて XML ビューが生成されます。また、Designer は入力メッセージまたは出力メッセージへの MIME アタッチメントのためのビューも生成します。

Web サービスのソース定義およびターゲット定義を作成すると、Designer によって、リレーショナルまたはフラットファイルのソースまたはターゲットで定義されたカラム、または手動で定義するカラムに基づいて XML ビューが作成されます。

XML ビューおよびグループ

Web サービスのソース定義とターゲット定義には次に示すビューが含まれます。

- **エンベロープ**。SOAP エンベロープおよび body 要素に対応する XML ビュー。エンベロープビューは、入力メッセージまたは出力メッセージのプライマリキーとポートを含むメインビューです。

body メッセージ部が単純であれば、Designer がエンベロープビューを生成します。

body メッセージ部が複雑であれば、Designer が追加のボディビューを生成できます。

- **要素**。入力メッセージまたは出力メッセージに複数回出現要素が含まれる場合に生成されるビュー。Designer は、入力または出力メッセージの各複数回出現要素について、要素ビューを生成します。要素ビューとエンベロープビューのリレーションは n 対 1 です。

- **タイプ**。入力メッセージまたは出力メッセージに複合タイプの定義が含まれる場合に生成されるビュー。Designer は、入力または出力メッセージの複合タイプの各要素について、タイプビューを生成します。タイプビューとエンベロープビューのリレーションは n 対 1 です。

Designer は、エンティティ関係モードでインポートされた Web サービスのソース定義とターゲット定義について、タイプビューを生成します。

- **ヘッダ**。SOAP ヘッダ要素に対応する XML ビュー。ヘッダメッセージ部が複雑であれば、Designer がヘッダビューを個別の要素およびタイプのビューに分割できます。
- **フォルト**。操作の出力メッセージについてフォルトメッセージを定義する場合に作成されるビュー。Designer は、操作に対して定義される各フォルトメッセージについてフォルトビューを生成します。フォルトビューとエンベロープビューのリレーションは n 対 1 です。Web サービスのターゲット定義にのみフォルトビューが含まれています。

Designer は、正規化された階層関係モードまたはエンティティ関係モードでインポートされた Web サービスのターゲット定義について、フォルトビューを生成します。

- **添付**。MIME 添付を含む WSDL に対して生成される添付ビュー。添付ビューとエンベロープビューのリレーションは n 対 1 です。

Designer は、MIME 添付の要素定義を含む WSDL に基づいて、Web サービスのソース定義またはターゲット定義に対する添付ビューを生成します。

関連項目：

- [「WSDL アタッチメント」 \(ページ 94\)](#)

ソース定義

Designer は、入力メッセージの定義に基づいて、Web サービスのソース定義に対する XML ビューを生成します。

次の表に、Designer が Web サービスソース定義に生成できる XML ビューを示します。

インポートモード	エンベロープ	要素	タイプ	フォールト	添付
エンティティ関係	はい	はい	はい	いいえ	はい
正規化された階層関係	はい	はい	いいえ	いいえ	はい

ターゲット定義

WSDL 内の操作に対するターゲット定義を作成する場合、Designer はその操作に関連付けられている出力メッセージとあらゆるフォールトメッセージをインポートします。操作結果内部の関数が別のフォールトを起こす場合、Designer がターゲット定義内に複数のフォールトビューを作成します。フォールトメッセージは要求処理のエラーを表します。

次の表に、Designer が Web サービスターゲット定義に生成できる XML ビューを示します。

インポートモード	エンベロープ	要素	タイプ	フォールト	添付
エンティティ関係	はい	はい	はい	はい	はい
正規化された階層関係	はい	はい	いいえ	はい	はい

注: フォールトメッセージに別々のターゲット定義を作成するには、PowerCenter Designer で Web サービスオプションを設定します。

Web サービスのソースおよびターゲットのインポートまた作成のための規則およびガイドライン

Web サービスのソースとターゲットをインポートまたは作成するとき、次の規則およびガイドラインに従ってください。

- **複雑なリレーションの要素に対しては WSDL を使用します。** 複雑な要素のリレーションを持つ Web サービスのソースまたはターゲットを作成するには、まず WSDL を作成して要素の階層を定義してから、WSDL からソースまたはターゲットをインポートします。複数回出現する要素を含む、あるいは複合タイプの要素を含む Web サービスのソースまたはターゲットを作成するには、WSDL を使用します。
- **単純な Web サービスのソースまたはターゲット定義を手動で定義します。** 単純なカラムのセットで入れ子の要素なしで Web サービスのソースまたはターゲットを作成するには、定義を手動で作成するかリレーショナルまたはフラットファイルのソース定義またはターゲット定義を使用します。Web サービスのソース定義またはターゲット定義のすべてのカラムが複数回出現することを指定できます。

- **同じ方法でソース定義およびターゲット定義を作成します。** ソース定義の作成と同時に、リレーショナルまたはフラットファイルソースまたはターゲットから Web サービスのターゲット定義を作成します。 Web サービスマッピングのソースとターゲットを同時に作成するには、[Web サービス定義の作成] ウィンドウで [ソースの作成] オプションおよび [ターゲットの作成] オプションが選択されていることを確認します。例えば、Source Analyzer で、[ソース] - [Web Services Provider] - [Web サービス定義の作成] を選択します。 [Web サービス定義の作成] ウィンドウで、[ターゲットの作成] オプションを選択します。
- **同じ方法を使用して要求／応答マッピングのソース定義とターゲット定義を作成します。** 要求／応答 Web サービスマッピングを作成する場合は、同じ方法を使用してソース定義とターゲット定義を作成します。たとえば、WSDL からソース定義をインポートする場合、WSDL 内の同じ操作からターゲット定義をインポートします。ソース定義の作成をカラムの定義によって行うか、リレーショナルまたはフラットファイルのソースおよびターゲットによって行う場合は、同じ方法でターゲット定義を作成します。
- **WSDL を使用してフォルトビューを持つターゲットを作成します。** ターゲット定義に特定のデータエラーについてのフォルトビューを持たせたい場合、WSDL を使用して Web サービスのターゲット定義を作成します。フラットファイルもしくはリレーショナルソースまたはリレーショナルターゲットにもとづいてターゲット定義を作成する場合、ターゲット定義でフォルトビューを定義できません。フラットファイルもしくはリレーショナルソースまたはリレーショナルターゲットに基づいて Web サービスのターゲット定義を作成する場合、Web Services Hub はシステム障害についてのみフォルトメッセージを生成できます。
- **WSDL では、入力メッセージと出力メッセージは同じエンコードスタイルを使用する必要があります。** Web サービスのソースおよびターゲットを WSDL からインポートする場合は、入力メッセージと出力メッセージのエンコードスタイルを同じにする必要があります。入力メッセージのスタイルが RPC/SOAP Encoded の場合、出力メッセージも RPC/SOAP Encoded スタイルを使用する必要があります。入力メッセージが Document/Literal スタイルを使用する場合、出力メッセージも Document/Literal スタイルを使用する必要があります。

Web サービスのソース定義またはターゲット定義の作成を手動で行うか、リレーショナルまたはフラットファイルのソースおよびターゲットに基づいて行う場合、Designer は入力および出力メッセージのドキュメント/リテラルエンコードスタイルを使用します。
- **WSDL 内の要素は標準の W3C XML スキーマを参照することができません。** 標準の W3C XML スキーマを参照する要素を含む WSDL から、Web サービスのソース定義およびターゲット定義をインポートすることはできません。
- **エンティティ関係モードで空の complexType 要素を持つ WSDL をインポートします。** Web サービス要求の入力メッセージに値がない complexType 要素が WSDL に含まれる場合は、エンティティ関係モードで WSDL からソース定義およびターゲット定義をインポートします。正規化された階層モードで WSDL からソース定義およびターゲット定義がインポートされる場合、complexType 要素が空の要求を送信すると、Web サービスはフォールト応答を生成します。
- **有効な XML 構文を使用して WSDL からソースおよびターゲットをインポートします。** 無効な WSDL からインポートした場合、WSDL 定義は Web サービスウィザードに正しく表示されません。Designer によってエラーメッセージが生成されず、WSDL が部分的に解析され、解析に成功したサービスおよび操作のみが表示されることがあります。WSDL からインポートしたとき、Web サービスウィザードに正しい WSDL 定義が表示されない場合は、その WSDL を XML ファイルとして開き、構文が正しいことを確認してください。

- **正しい構文で 2 次元配列を定義します。** WSDL で 2 次元配列の文字列として complexType 要素を定義する場合は、以下の構文を使用します。

```
wsdl:arrayType="xsd:string[][]"
```

異なる構文を使用して定義された 2 次元配列を含む WSDL から、Web サービスのソース定義およびターゲット定義をインポートすることはできません。

- **多数の XML ビューを生成する WSDL から Web サービスのソースおよびターゲットをインポートすることはできません。** WSDL ファイルから生成可能な XML ビューの最大数は 400 です。XML ビューの数が 400 を超える Web サービスのソースまたはターゲットを作成するには、WSDL ワークスペース内でグループを手動で作成します。

関連項目：

- [「SOAP フォールトの処理」 \(ページ 15\)](#)

Web サービスのソース定義またはターゲット定義のインポート

作成対象の Web サービスの入力および出力メッセージを定義する WSDL がある場合は、WSDL からソース定義およびターゲット定義をインポートできます。Web Services ウィザードを使用することによって、WSDL から Web サービスのソース定義またはターゲット定義をインポートできます。同じ方法でソース定義およびターゲット定義をインポートできます。

インポートモード

WSDL をインポートするモードに基づいて、Web サービスのソース定義またはターゲット定義の XML ビューを作成できます。

- **エンティティ関係。**これは、WSDL からインポートされるソース定義またはターゲット定義のデフォルトのインポートモードです。このインポートモードを使用して、1 つの大きな階層を作成する代わりにビュー間の関係を作成します。エンティティ関係を含む Web サービスのソースまたはターゲットを作成するとき、Designer は複数回出現する要素および複合型に対して個別のビューを生成します。Designer は、すべての派生複合型のビューを含みます。
- **正規化された階層関係。**正規化された階層ビューでは、すべての要素または属性が一度に表示されます。1 対多関係は、ビューを関連付けるキーを持つ個別の XML ビューになります。
- **XML ビュー生成のスキップ。**このインポートモードを使用して、XML ビューを定義せずにソース定義またはターゲット定義を作成します。WSDL ワークスペースを使用することによって、XML ビューおよびポートを追加できます。

関連項目：

- [「XML ビューを作成しない WSDL からのインポート」 \(ページ 79\)](#)

メッセージ ID

段階的マッピングで Web サービスのソース定義およびターゲット定義を使用するには、Web ソース定義およびターゲット定義にメッセージ ID を入れる必要があります。Web Services Hub は、メッセージ ID をプライマリキーとして使用し、Web サービスの要求および応答をバインドします。たとえば、第 1 セッションが Web サービスソースから読み取り、リレーショナルターゲットに書き込みます。第 2 セッションは、リレーショナルターゲットをソースとして使用し、Web サービスターゲットに書き込みます。

Web Services Hub は、メッセージ ID を使用して、第 1 セッション内の要求の入力メッセージを第 2 セッション内の応答の出力メッセージに関連付けます。

また、グリッド上の Web サービスセッションを実行する時に、Web ソース定義およびターゲット定義内にメッセージ ID を入れる必要があります。グリッド上で Web サービスセッションを実行する場合、Integration Service は、グリッド内のノード上で実行されている複数の DTM プロセスにセッションスレッドを分散します。Integration Service は、メッセージ ID を使用してノード間で Web サービスの入力および出力メッセージに関連付けます。

Web サービスのソース定義またはターゲット定義を作成するときにメッセージポートを追加する場合、Designer はメッセージ ID とクライアントポートをエンベロープビューに追加します。

次の表に、エンベロープビューに追加されたメッセージ ID とクライアントポートを示します。

ポート名	説明
MessageID	Web Services Hub は、要求を受信するとメッセージ ID を生成します。この ID を使用して、入ってくる要求と出ていく応答の相関関係を設定します。
ClientIP	Web サービスクライアントの TCP/IP アドレス。

詳細オプション

WSDL から Web サービスのソースまたはターゲットをインポートする場合、長さが定義されていないフィールドの長さとして XML カラムの命名規則を指定できます。

次の表に、WSDL から Web サービスのソースまたはターゲットをインポートする時に設定できる詳細オプションを示します。

オプション	説明
すべての長さの infinite 指定を以下の値でオーバーライドする	長さの定義がないフィールド（文字列など）に長さのデフォルトを指定できます。デフォルトでは、このオプションは選択されています。
XMLColumn 名を生成するときのオプションを指定してください	<p>XML カラムに名前を付ける場合、シーケンス番号を使用するか、あるいはスキーマの要素名または属性名を使用するかを選択できます。名前を使用する場合は、以下のオプションから選択します。</p> <ul style="list-style-type: none"> - XML カラムが属性を参照する場合、その前に要素名を付けます。 PowerCenter では XML カラムの名前に次の形式を使用します。 <i>NameOfElement_NameOfAttribute</i> - すべての XML カラムの XML ビュー名に接頭語を付けます。 PowerCenter では、XML カラム名に次の形式が使用されます： <i>NameOfView_NameOfElement</i> - すべての外部キーカラムの XML ビュー名に接頭語を付けます。 PowerCenter では、生成した外部キーカラムの名前に次の形式が使用されます： <i>FK_NameOfView_NameOfParentView_NameOfPKColumn</i> <p>カラム名の最大長は 80 文字です。81 文字を超えるカラム名は、PowerCenter によって切り詰められます。カラム名が一意でない場合は、名前を一意にするために PowerCenter によって数字の接尾語が追加されます。</p>
文字列にマップされた anyType 要素のデフォルトの長さ	<p>タイプ anyType 要素に作成された文字列ポートのデフォルトの長さ。タイプ anyType 要素のタイプ文字列のポートを作成できます。デフォルトで、文字列の長さは、ここに設定した値になります。</p> <p>文字列の長さを変更するには、WSDL ワークスペースで Web サービスのソース定義またはターゲット定義を編集します。デフォルトは 10,000 です。</p>

XML ビューを作成しない WSDL からのインポート

WSDL からソース定義またはターゲット定義をインポートし、手動で XML ビューおよびポートを定義する場合は、空のソース定義またはターゲット定義を作成できます。

たとえば、入力メッセージ中の 10 の要素を定義する WSDL があり、その中の 2 つだけをソース定義に入れるとします。この場合、空のソース定義を作成し、その 2 つの要素を手動で定義できます。ターゲットの定義には影響を与えません。ターゲット定義をインポートして XML ビューを作成できます。

XML を作成せずに WSDL からソース定義またはターゲット定義をインポートするには、インポートプロセスの手順 2 で、[XML View 作成のスキップ] を選択します。空のソース定義またはターゲット定義の作成後、WSDL ワークスペースを使用して、XML のビューおよびポート、ビュー間のリレーションを定義します。ソース定義またはターゲット定義のタイトルを右クリックして、[WSDL ワークスペース] を選択します。

関連項目：

- [「WSDL ワークスペースでの定義の編集」 \(ページ 85\)](#)

WSDL からの Web サービスのソース定義またはターゲット定義のインポート

WSDL から Web サービスのソース定義またはターゲット定義をインポートするには、同じ手順に従います。WSDL ではソース定義とターゲット定義は異なる要素を表すので、Designer によって作成されるソース定義はターゲット定義とは異なります。

ローカルにまたは URL によって利用できる WSDL から、Web サービスのソースまたはターゲットをインポートできます。RPC/SOAP Encoded スタイルまたはドキュメント/リテラルエンコードスタイルを使った WSDL から定義をインポートできます。

注: Designer は Web サービスのターゲット定義をインポートするときに、動作とターゲットタイプ（出力やターゲットなど）を基に定義に名前を付けます。インポート後に定義名を変更すると、[メタデータエクステンション] タブでターゲットタイプを確認できます。

WSDL から Web サービスのソース定義またはターゲット定義をインポートする手順

1. ソース定義をインポートするには、Source Analyzer で [ソース] - [Web Services Provider] - [WSDL からのインポート] の順にクリックします。ターゲット定義をインポートするには、Target Designer で [ターゲット] - [Web Services Provider] - [WSDL からのインポート] の順にクリックします。
2. [詳細オプション] をクリックします。
[XML ビュー作成および命名オプション] ウィンドウが表示されます。
3. 長さが定義されていないフィールドにデフォルトの長さを指定して、XML カラムの名前を生成する方法を選択します。
4. ローカルファイルからインポートするか、URL からインポートするかを選択します。
URL からインポートする場合は、URL を入力するかまたは [アドレス] リストから URL を選択し、[開く] をクリックします。
ローカルファイルからインポートする場合は、ローカルフォルダから WSDL ファイルを選択し、[開く] をクリックします。
5. ソース定義またはターゲット定義作成の対象となる、WSDL で定義する操作を選択します。
注: エラーが含まれている WSDL からインポートした場合は、WSDL で定義されたサービス、バインド、ポートまたは操作のリストが [Web サービスウィザード] ウィンドウ（手順 1）に正しく表示されません。空または部分的な WSDL 定義ツリーがウィンドウに表示されます。たとえば、WSDL に型定義のエラーがある場合は、空の WSDL 定義ツリーがウィンドウに表示されます。
6. [次へ] をクリックします。
[Web Services 定義作成オプション] ダイアログボックスが表示されます。
7. 院歩とモードを選択します。
インポートモードは、生成する XML ビューのタイプを特定します。XML ビューは、エンティティ関係として生成することも、階層関係として生成することもできます。デフォルトのインポートモードは、エンティティ関係です。
8. ビューまたはポートを定義せずにソース定義またはターゲット定義を作成するには、[XML ビュー生成のスキップ] を選択します。

XML ビューを生成しない場合、Designer は空のソース定義またはターゲット定義を作成します。ソース定義またはターゲット定義には、ビューまたはポートが含まれません。WSDL ワークスペースを使用して、ソース定義およびターゲット定義のビューおよびポートを手動で追加できます。

9. ソース定義またはターゲット定義にメッセージポートおよびヘッダポートを追加するかどうかを選択します。

詳細については、「[メッセージ ID](#)」 (ページ 77) を参照してください。

10. 同じインポートプロセスでソース定義およびターゲット定義を生成するには、[ソースの作成] と [ターゲットの作成] の両方を選択します。

Designer は、選択されたオプションに基づいてソース定義およびターゲット定義を作成します。

11. [終了] をクリックします。

Web サービスのソース定義またはターゲット定義がワークスペースに表示されます。

ソース定義またはターゲット定義の作成

Web サービスのソース定義またはターゲット定義のインポート元となる WSDL がない場合は、リレーショナルまたはフラットファイルのソースまたはターゲットから定義を作成できます。また、ソースまたはターゲットのポートを手動で定義して、データタイプおよび出現回数を指定することもできます。

他のソースおよびターゲットから、または手動で定義されたカラムから Web サービスのソース定義およびターゲット定義を作成する場合は、Designer によってエンティティ関係モードのビューが作成されます。階層関係モードでは、他のソースまたはターゲットから Web サービスのソースまたはターゲットを作成できません。

以下のリレーショナルデータベースから作成されたリレーショナルなソースおよびターゲットから、Web サービスのソース定義およびターゲット定義を作成できます。

- Oracle
- DB2
- Informix
- Teradata
- Microsoft SQL Server
- Sybase

リレーショナルまたはフラットファイルのソースまたはターゲットから Web サービスのソース定義またはターゲット定義を作成する場合は、ソースおよびターゲットへのショートカットなど、フォルダ内で使用可能なソースおよびターゲットが一覧表示されます。

同じ方法でソース定義およびターゲット定義を作成します。リレーショナルまたはフラットファイルのソースまたはターゲットに基づいて Web サービスのソース定義またはターゲット定義を作成した後、Designer のワークスペースでソース定義またはターゲット定義のカラムを編集できます。

WSDL から Web サービスのソース定義またはターゲット定義をインポートする場合は、以下のオプションを選択できます。

- 複数回出現する要素
- メッセージポート

関連項目：

- [「Designer ワークスペースでの定義の編集」 \(ページ 84\)](#)

複数回出現要素

Web サービスのソース定義またはターゲット定義をリレーショナルまたはフラットファイルのソースまたはターゲットから作成する場合は、ソース定義またはターゲット定義内のカラムが複数回出現するかどうかを指定する必要があります。[複数回出現要素] オプションを選択すると、グループとしてのカラムが複数回出現することを指定します。リスト内のカラムは、配列を表します。

このオプションで作成される Web サービスのソースまたはターゲットを編集する場合、ソース定義またはターゲット定義のカラムの複数回出現プロパティを変更することができません。

メッセージポート

Web ソース定義およびターゲット定義にメッセージ ID を入れることができます。Web Services Hub は、メッセージ ID をプライマリキーとして使用し、Web サービスの要求および応答をバインドします。

注: Web サービスのソース定義またはターゲット定義を Designer ワークスペースで編集する場合、メッセージポートとクライアントポートは [Web サービス定義] タブに表示されません。メッセージポートとクライアントポートは変更できません。

関連項目：

- [「メッセージ ID」 \(ページ 77\)](#)
- [「\[Web サービス定義\] タブ」 \(ページ 85\)](#)

リレーショナルまたはフラットファイルのソースまたはターゲットからのソースまたはターゲットの作成

この手順に従って、以下のいずれかの方法で Web サービスのソースまたはターゲットを作成します。

- フラットファイルのソースまたはターゲットから
- リレーショナルなソースまたはターゲットから
- 手動でカラムを追加して名前、データタイプおよび精度を指定

リレーショナルまたはフラットファイルのソースおよびターゲットから Web サービスのソースまたはターゲットを作成する手順

1. Web サービスのソースを作成するには、Source Analyzer で [ソース] - [Web Services Provider] - [Web Service 定義の作成] の順にクリックします。Web サービスのターゲットを作成するには、Target Designer で [ターゲット] - [Web Services Provider] - [Web Service 定義の作成] の順にクリックします。
2. ソース定義およびターゲット定義を使用する予定の Web サービスマッピングの名前を入力します。

Designer は、Web サービス定義名をソース定義およびターゲット定義の名前として使用します。ソース定義名には接尾辞 *_input* を追加し、ターゲット定義名には *_output* を追加します。

Web サービスのソース定義またはターゲット定義に入れるカラムが分かっている場合は、それらをソース定義またはターゲット定義に直接追加できます。フォルダ内にリレーショナルまたはフラットファイルのソースまたはターゲットがある場合は、リレーショナルまたはフラットファイルのカラムに基づいてソース定義またはターゲット定義を作成できます。

3. リストにカラムを追加するには、[追加] ボタンをクリックしてカラム名、データタイプおよび精度を指定します。

複数回出現要素を使用して Web サービスのソースを作成する場合、または Web サービスのターゲットを作成する場合、Web Services Hub は Not Null プロパティを無視します。

4. リレーショナルまたはフラットファイルのソースまたはターゲット内のカラムに基づいて Web サービスのソースまたはターゲットを作成するには、[ソース/ターゲットからのインポート] をクリックしてソース定義またはターゲット定義を選択します。

Designer は、選択されたソースまたはターゲット内に見つかったカラムを一覧表示します。

5. [OK] をクリックします。

6. 必要であれば、カラム名と文字列カラムの精度を編集します。

カラムは定義に追加して、それらのデータタイプおよびプロパティを定義できます。使用しないカラムは、削除できます。

ソース定義またはターゲット定義で定義するカラムが 1 回だけ出現するか複数回出現するかを指定できます。

7. カラムが複数回出現する場合は、[複数回出現要素] をオンにします。

このオプションは、グループとしてのカラムが複数回出現することを指定します。このオプションを選択すると、Designer はすべてのカラムを含む要素ビューを生成します。

8. ソース定義を作成中でありターゲット定義も作成する場合は、[ターゲットの作成] をクリックして、手順 [3~7](#) を繰り返してポートをターゲット定義に追加します。

ターゲット定義を作成中のときにソース定義も作成する場合は、[ソースの作成] をクリックして手順 [3~7](#) を繰り返し、ソース定義にポートを追加します。

9. メッセージポートおよびクライアントポートをソース定義またはターゲット定義に追加するには、[メッセージポートの追加] をクリックします。

Designer は、メッセージポートおよびクライアントポートをソース定義またはターゲット定義のエンベロープビューに追加します。ソース定義およびターゲット定義を同じプロセスで作成している場合、Designer はメッセージポートおよびクライアントポートをソース定義およびターゲット定義のエンベロープビューに追加します。

10. [OK] をクリックします。

Designer は、Web サービスのソース定義またはターゲット定義を作成します。

ソース定義およびターゲット定義内の XML ビューを見直して、ビューとポートが Web サービスマッピング要件に一致することを確認します。カラムを追加、削除、または変更するには、Designer ワークスペースのソース定義またはターゲット定義を編集します。

関連項目：

- [「複数回出現要素」 \(ページ 82\)](#)
- [「メッセージ ID」 \(ページ 77\)](#)
- [「Designer ワークスペースでの定義の編集」 \(ページ 84\)](#)

Web サービスのソースおよびターゲットの編集

Web サービスのソースおよびターゲットの編集の概要

Web サービスのソース定義またはターゲット定義は、その作成方法に基づいて編集できます。

- **WSDL からインポートされたソース定義またはターゲット定義。** Web サービスのソース定義またはターゲット定義を WSDL からインポートする場合、WSDL のワークスペースでソース定義またはターゲット定義を編集できます。WSDL ワークスペースを使用することによって、ソース定義またはターゲット定義内でビューの追加、変更、または削除を実行できます。

Designer ワークスペースに定義を表示し、一部のプロパティを編集できます。

- **リレーショナルまたはフラットファイルのソースおよびターゲットから作成されたソース定義またはターゲット定義。** Web サービスのソース定義またはターゲット定義をリレーショナルまたはフラットファイルのターゲットに基づいて作成する場合、Designer のワークスペースでカラムを編集できます。

WSDL ワークスペースにソース定義またはターゲット定義を表示できます。WSDL ワークスペースではソース定義またはターゲット定義を編集できません。

注: PowerCenter version 9.0.1 以降で作成された Web サービスのソースとターゲットは編集できます。Web サービスワークフローを実行したときにエラーが発生するのを防止するために、以前のバージョンからアップグレードされた Web サービスのソースとターゲットは編集しないでください。以前のバージョンの PowerCenter から Web サービスのソースまたはターゲットを更新するには、PowerCenter version 9.0.1 以降でソースまたはターゲットを再作成します。

Designer ワークスペースでの定義の編集

Designer ワークスペースで、いつでも Web サービスのソース定義またはターゲット定義に対して説明を追加したり、ビジネス文書へのリンクを指定したりすることができます。ソース定義またはターゲット定義の作成を手動で行うか、リレーショナルまたはフラットファイルのターゲットに基づいて行う場合は、ソース定義またはターゲット定義のカラムのリストを変更できます。カラムに変更を加えると、変更が XML ビュー内で直ちに反映されます。

ソース定義またはターゲット定義のプロパティを表示または編集するには、Source Analyzer でソース定義をダブルクリックするか、Target Designer でターゲット定義をダブルクリックします。あるいは、ソース定義またはターゲット定義のタイトルバーを右クリックして、[編集] を選択します。

次のタブで、Web サービスのソース定義またはターゲット定義を表示または編集できます。

- **テーブル。** [テーブル] タブには、所有者名と説明を指定できます。また、定義名の変更もできます。テーブルタイプは変更できません。
- **カラム。** [カラム] タブでは、String データタイプの精度を編集できます。また、ビジネス名とカラムの説明も追加できます。
- **属性。** [属性] タブには、ソースまたはターゲット定義の各カラムの属性値を表示できます。
- **メタデータエクステンション。** [メタデータエクステンション] タブには、Web Services Domain のメタデータエクステンションを表示できます。また、ユーザー定義のメタデータドメインにメタデータエクステンションを追加することもできます。

- **Web サービス定義。** このタブは、リレーショナルまたはフラットファイルのターゲットから作成されたソース定義またはターゲット定義を編集する場合に表示されます。ソース定義またはターゲット定義のカラムを追加、編集、または削除できます。加えた変更は、[カラム] タブに直ちに表示されます。

[テーブル] タブ

[テーブル] タブには、ソース定義またはターゲット定義のテーブル情報が表示されます。ソース定義またはターゲット定義の名前は変更できます。ソース定義またはターゲット定義のオーナーおよび説明を変更することもできます。

[カラム] タブ

[カラム] タブには、Web サービスのソース定義またはターゲット定義の XML ビューが表示されます。文字列およびバイナリデータタイプの精度の編集と、ビジネス名とカラムの説明の追加が可能です。

文字列データタイプのデフォルト精度は、WSDL インポートプロセス時に無限長データの精度が設定される値です。文字列データタイプの精度は、ソース定義またはターゲット定義を WSDL からインポートするときに設定できます。個々のカラムの精度は、定義の編集時に設定できます。

注: 総カラム長が 500 MB より大きいソースおよびターゲット Web サービス定義を使用したマッピングは、Mapping Designer によって無効化されます。

[属性] タブ

[属性] タブは、読み込み専用タブであり、Web サービスのソース定義またはターゲット定義の各フィールドの XPath 値と XMLDataType 値を表示します。この定義にアタッチメントグループがある場合、[属性] タブはデータフィールドに MIME 型を表示します。

[メタデータエクステンション] タブ

メタデータエクステンションは [メタデータエクステンション] タブで作成できます。ベンダ定義エクステンションの内容を Web Services Provider Domain に表示することもできます。これらのメタデータエクステンションは、メッセージタイプ（入力、出力、フォールト）を識別します。

メタデータエクステンションの詳細については、『PowerCenter リポジトリガイド』を参照してください。

[Web サービス定義] タブ

[Web サービス定義] タブは、手動で定義するかリレーショナルまたはフラットファイルのソース/ターゲットに基づいている Web サービスのソース/ターゲット定義に対して表示されます。ソース定義またはターゲット定義のカラムを追加または削除できます。カラムの名前およびデータタイプを変更し、特定のデータタイプの精度および位取りを変更できます。また、カラムが 1 回出現するか複数回出現するかを指定することもできます。

[Web サービス定義] タブのカラムに変更を加えると、変更が [カラム] タブに反映されます。

WSDL ワークスペースでの定義の編集

WSDL からソース定義またはターゲット定義をインポートして XML ビューを作成する場合、WSDL ワークスペースで XML ビュー、ポート、およびリレーションを編集できます。ソース定義またはターゲット定義

を WSDL からインポートするが、XML ビューを生成しない場合、WSDL ワークスペースを使用して、ビューの作成、コンポーネントの変更、カラムの追加、ワークスペース内のビューのリレーションの維持が可能です。ソース定義やターゲット定義を更新すると、Designer はそのソースまたはターゲットを含んでいるすべてのマッピングに変更を反映します。

ソース定義またはターゲット定義を WSDL ワークスペースで表示または編集するには、Source Analyzer でそのソース定義のタイトルを右クリックするか、あるいは Target Designer でターゲット定義のタイトルを右クリックします。続いて、[WSDL ワークスペース] を選択します。

WSDL ワークスペースは、XML エディタに相当します。WSDL ワークスペースの使用方法は、XML エディタの使用方法と同じです。ただし、WSDL ワークスペースでは、Web サービスのソース定義とターゲット定義に特有のビューの変更が検査されます。また、XML ワークスペースで許可されている WSDL ワークスペース内の一部の作業を実行できません。

次の作業は、WSDL ワークスペース内で実行できません。

- Web サービスのソースまたはターゲット定義に XML ビューを追加または編集する時に、カラムをピボットする。
- XML ビュー内で要素または属性をフィルタ選択するために XPath クエリ述語を作成する。
- XML データをプレビューする。
- FileName カラムを XML ビューを追加する。
- 参照ポートを追加する。
- エンティティ関係を再作成する。
- [カラム] ウィンドウで XML ビューオプションをセットする。

WSDL ワークスペースの規則およびガイドライン

WSDL ワークスペース内で Web サービスのソース定義またはターゲット定義に対して XML ビューの追加または編集を行う場合は、以下の規則およびガイドラインを使用します。

- Web サービスのマッピングのソース定義とターゲット定義には、Web サービスリクエスト、レスポンスおよびフォルトメッセージの SOAP:エンベロープに相当するエンベロープビューが含まれています。
- ソース定義は、入力メッセージのビューを定義する必要があります。ソース定義は、出力メッセージまたはフォルトメッセージのビューを定義できません。
- ソース定義またはターゲット定義のルートグループおよびルートグループのプライマリキーの名前には、次の命名規則を使用しなくてはなりません。<NameString>は英数字の文字列です。
 - ルートグループは、Message または X_<NameString>_Envelope という名前にする必要があります。
 - ルートグループのプライマリキーは、PK_Message または PK_<NameString>_Envelope という名前にする必要があります。
 - ルートグループおよびそのプライマリキーの<NameString>を同じにする必要があります。
- ターゲット定義は、出力メッセージまたはフォルトメッセージのビューを定義する必要があります。入力メッセージのビューを定義できません。
- タイプが anytype または any の要素を定義できます。タイプ anyType の要素の文字列ポートを作成するか、それをタイプ complexType の要素にマッピングすることができます。
- ターゲット定義内のエンベロープビューには、エンベロープノードとしてビュールートおよびビュー行を入れます。

- ソース定義またはターゲット定義の soap:Body 要素と soap:Header 要素のタイプ定義を変更できません。
- デフォルトの名前空間の設定と、ソース定義またはターゲット定義のビューで定義した名前空間のプレフィックスの変更が可能です。名前空間は変更できません。次のいずれの文字列も名前空間プレフィックスとして使用できません。
 - mime
 - wsdl
 - soap
 - soapenc
 - http

Web サービスマッピングの使用

Web サービスマッピングの使用の概要

Web サービスのソース定義およびターゲット定義を作成した後に、Integration Service が Web サービス要求で受信したデータを処理して Web サービス応答を送出する方法を特定するマッピングを作成します。Web サービスマッピングは、入力メッセージを SOAP 要求として受信し、データを変換し、出力メッセージを SOAP 応答として送信します。

Web サービスマッピングの作成は、Mapping Designer で他の PowerCenter マッピングの作成と同じ方法で実行できます。Web サービスのソース定義およびターゲット定義とトランスフォーメーションをマッピングに追加します。

また、Web サービスソース定義、ソース修飾子トランスフォーメーション、および Web サービスターゲット定義を含むマッピングも生成できます。PowerCenter Designer には、Web サービスマッピングを生成する各種の方法があります。

以下の方法で Web サービスマッピングを生成できます。

- **WSDL からソース定義およびターゲット定義をインポート。**WSDL では、Web サービスのソースまたはターゲットを作成するのと同じ方法でマッピングを作成できます。
- **リレーショナルまたはフラットファイルのソースあるいはターゲット定義から。**リレーショナルまたはフラットファイルのソースあるいはターゲットから Web サービスのソースまたはターゲットを作成するのと同じ方法でマッピングを作成できます。
- **トランスフォーメーションまたはマッピングレットから。**再利用可能なトランスフォーメーションまたはマッピングレットから、1 つの入力と 1 つの出力でマッピングを作成できます。

マッピングの生成後、Web サービスマッピングの完了に必要なトランスフォーメーション、リンク、およびその他のマッピングオブジェクトをさらに追加することができます。

Web サービスマッピングのタイプ

Web サービスクライアントからメッセージを受信し、データを変換した後で、Web サービスクライアントに応答を送信するか、または PowerCenter でサポートされている任意のターゲットに書き込むように

するため、マッピングを作成することができます。ソース定義とターゲット定義に従って、Integration Service はアタッチメントを SOAP 要求の一部として送受信できます。

フラットファイルまたは XML ソースとターゲットによってマッピングを作成し、Web サービスワークフローで使用することもできます。これにより、メッセージデータをファイルから読み取るのではなく、アタッチメントにより SOAP 呼び出しを介して受信できます。

作成するマッピングは、実行する Web サービスのタイプによって異なります。

- **要求/応答 Web サービス。** 要求/応答 Web サービスは、Web サービスクライアントから入ってくる要求を受信し、データを変換してから Web サービスクライアントに応答を返信します。要求/応答 Web サービスは、Web サービスのソースとターゲットの両方を使用します。

1 つの要求/応答 Web サービスを処理するのに、1 つまたは複数のマッピングを作成することができます。

- **1 つのマッピング。** Web サービスのソース定義とターゲット定義の両方を持ったマッピングを 1 つ作成します。Integration Service は入ってくる要求の受信、データの変換、応答の返信を 1 つのセッションで実行します。

- **複数のマッピング。** Web サービスクライアントに応答を返信する前にデータステージングを行うために、複数のマッピングを作成します。マッピングごとのセッションを含むワークフローを作成できます。

- **一方向 Web サービス。** 更新や通知を Web サービスクライアントから受信するが、応答を返信する必要がない場合は、一方向マッピングを作成できます。一方向マッピングはソースとして Web クライアントを使用します。Integration Service は、ほとんどの場合、Web サービス要求を介したリアルタイムイベントを契機としてデータをターゲットにロードします。

マッピングに入れる Web サービスのソース定義とターゲット定義は、作成するマッピングのタイプによって異なります。

次の表に、マッピングタイプを基に使用する Web サービスのソース定義とターゲット定義を示します。

マッピングタイプ	Web サービスのソース	Web サービスのターゲット
要求/応答	Web サービスで 1 つのソース定義のインスタンスが 1 つ必要です。	Web サービスで 1 つのターゲット定義のインスタンスが 1 つ必要です。 ターゲット定義に複数のフォールトビューを持つことができます。
一方向	Web サービスで 1 つのソース定義のインスタンスが 1 つ必要です。	Web サービスのターゲット定義を持ちません。

関連項目：

- [「アタッチメント」 \(ページ 94\)](#)

要求/応答マッピング

要求/応答マッピングは、Web サービスのソースと Web サービスのターゲットを使用します。

要求/応答マッピングを作成する場合、同じ方法で作成されたソース定義とターゲット定義を使用してください。WSDL からソース定義をインポートする場合、WSDL 内の同じ操作からターゲット定義をインポー

トします。ソース定義の作成をカラムの定義によって行うか、リレーショナルまたはフラットファイルのソースおよびターゲットによって行う場合は、同じ方法でターゲット定義を作成します。

Web サービスのソース定義とターゲット定義を確実に同じ方法で作成するには、ソース定義とターゲット定義を 1 つのプロセスで作成します。

注: ソース定義とターゲット定義を WSDL 内の同じ操作からインポートしない場合、またはそれらを同じ方法で作成しない場合、予期できない結果が発生することがあります。

SQL トランスフォーメーションを使用すると、要求/応答マッピング中にデータベースを更新したり、複数のデータベース行を取得したりできます。SQL トランスフォーメーションは、複数のデータベース行をターゲットに返すことができます。処理中にデータベースでエラーが発生した場合、SQL トランスフォーメーションはデータベースからエラーを受け取り、エラーテキストをターゲットに送信します。

例えば、SQL トランスフォーメーションを使用して複数行を取得する Web サービスの場合、PowerCenter に付属のリアルタイム Web サービスの例を参照してください。デフォルトでは、リアルタイム Web サービスのサンプルプログラムは、次のディレクトリにインストールされます。

```
/<PowerCenterInstallDir>/server/samples/WebServices/samples/RealTimeWebServices
```

関連項目：

- [「ソース定義またはターゲット定義の作成」 \(ページ 81\)](#)

段階的マッピング

要求/応答セッションを実行したいが、まずデータのステージングが必要という場合は、複数のマッピングを作成してデータを処理できます。

たとえば、処理したいメッセージデータを受信します。WebSphere MQ 経由で外部システムに対して非同期呼び出しを行う必要があります。次に示すマッピングを作成してください。

1. Web サービスのソース定義を持つ要求マッピングを作成します。このマッピングは、フラットファイルターゲットと WebSphere MQ ターゲットに書き込みます。両方のターゲットにすべてのメッセージデータを書き込みます。

外部アプリケーションは WebSphere MQ ターゲットからメッセージを受信し、処理してから別の WebSphere MQ キューにメッセージを送信します。

2. Web サービスのターゲット定義を持つ要求マッピングを作成します。このマッピングは、ソースとして最初のマッピング内のフラットファイルターゲットを使用します。また、処理済みデータを持つ WebSphere MQ キューもソースとして使用します。

Web Services Hub は、メッセージ ID を使用して要求および応答を段階的マッピングで接続します。段階的マッピングで Web サービスのソース定義およびターゲット定義を使用するには、Web ソース定義およびターゲット定義にメッセージ ID を入れる必要があります。

関連項目：

- [「メッセージ ID」 \(ページ 77\)](#)

WSDL からのマッピングの生成

Web サービスマッピングの生成は、ローカルで利用できるか URL によって利用できる WSDL から Web サービスのソースまたはターゲットをインポートすることによって可能です。

ソース定義およびターゲット定義をインポートすることによって Web サービスマッピングを生成する場合、Designer は選択した操作の入力メッセージからソース定義を作成します。また、選択する操作の出力メッセージからターゲット定義を作成します。

WSDL から生成された Web サービスマッピングには、次のオブジェクトが含まれます。

- Web サービスのソース定義
- ソース修飾子
- Web サービスのターゲット定義

Designer は、ソースインスタンスからターゲットインスタンスにポートをリンクします。マッピングを実行するには、作成する Web サービスに必要なトランスフォーメーションおよび他のマッピングコンポーネントを追加します。

WSDL からマッピングを作成するには：

1. PowerCenter Designer で Mapping Designer を開きます。
2. [マッピング] - [Web サービスマッピングの作成] - [WSDL からインポート] をクリックします。

WSDL からソースおよびターゲットをインポートすることによって Web サービスマッピングを生成する手順は、Web サービスのソース定義またはターゲット定義を WSDL から作成する手順と同じです。詳細については、[「Web サービスのソース定義またはターゲット定義のインポート」](#) (ページ 77) を参照してください。

3. マッピングをリポジトリに保存します。

リレーショナルまたはフラットファイルのソースまたはターゲットからのマッピングの生成

リレーショナルまたはフラットファイルのソースまたはターゲットに基づいて、Web サービスマッピングを生成できます。リレーショナルまたはフラットファイルのソースまたはターゲットを使用することによって、Web サービスのソース定義およびターゲット定義のカラムを定義します。

リレーショナルまたはフラットファイルのソースまたはターゲットからマッピングを生成すると、生成された Web サービスマッピングには次のオブジェクトが含まれます。

- Web サービスのソース定義
- ソース修飾子
- Web サービスのターゲット定義

Designer は、ソースインスタンスからターゲットインスタンスにポートをリンクします。マッピングを実行するには、作成する Web サービスに必要なトランスフォーメーションおよび他のマッピングコンポーネントを追加します。

注: リレーショナルまたはフラットファイルのソースまたはターゲットからマッピングを生成する場合は、Web サービスのソースおよびターゲットを同じプロセス内に作成します。Web サービスのソースおよびターゲットが異なる時点で作成されるマッピングを含むワークフローを実行すると、ワークフローが失敗する場合があります。

リレーショナルまたはフラットファイルのソースまたはターゲットからマッピングを生成する手順

1. PowerCenter Designer で Mapping Designer を開きます。

2. [マッピング] - [Web サービスマッピングの作成] - [ソース/ターゲット定義の使用] をクリックします。

リレーショナルまたはフラットファイルのソースまたはターゲットから Web サービスマッピングを生成する手順は、リレーショナルまたはフラットファイルのソースまたはターゲットから Web サービスのソースおよびターゲットを作成する手順と同じです。詳細については、[「ソース定義またはターゲット定義の作成」 \(ページ 81\)](#)を参照してください。

3. マッピングをリポジトリに保存します。

トランスフォーメーションまたはマップレットからのマッピングの生成

再利用可能なトランスフォーメーションまたはマップレットからマッピングを生成できます。Designer は、トランスフォーメーションまたはマップレットでポートを使用して、Web サービスのソース定義およびターゲット定義を生成します。

注: トランスフォーメーションまたはマップレットからマッピングを生成する場合は、Web サービスのソースおよびターゲットを同じプロセス内に作成します。Web サービスのソースおよびターゲットが異なる時点で作成されるマッピングを含むワークフローを実行すると、ワークフローが失敗する場合があります。

再利用可能なトランスフォーメーションからのマッピングの生成

以下の表に、Web サービスマッピングの生成が可能なトランスフォーメーションのタイプを示します。

トランスフォーメーション	タイプ	グループ
Expression	パッシブ	シングル
HTTP	パッシブ	1 つの入力および 1 つの出力
Java	アクティブまたはパッシブ	1 つの入力および 1 つの出力
ルックアップ	パッシブ	シングル
SQL	アクティブまたはパッシブ	1 つの入力および 1 つの出力
Stored Procedure	パッシブ	シングル

Web サービスマッピングの生成に使用するトランスフォーメーションは、再利用可能なトランスフォーメーションでなくてはなりません。トランスフォーメーションに基づいて Web サービスマッピングを生成する場合、Designer は、再利用可能なトランスフォーメーションとフォルダ内で使用できる再利用可能なトランスフォーメーションへのショートカットを一覧表示します。

Web サービスマッピングをトランスフォーメーションから生成する場合、Designer はトランスフォーメーションでポートを使用してソース定義およびターゲット定義のカラムを定義します。次に、トランスフォーメーション入力ポートを反映する XML ビューのソースと、トランスフォーメーション出力ポートを反映する XML ビューのターゲット定義を含むマッピングを作成します。

トランスフォーメーションから生成された Web サービスマッピングには、次のオブジェクトが含まれます。

- Web サービスのソース定義
- ソース修飾子
- マッピングの生成に使用するトランスフォーメーション
- Web サービスのターゲット定義

Designer は、ソースインスタンスからターゲットインスタンスにポートをリンクします。

マプレットからのマッピングの生成

以下のタイプのマプレットから Web サービスマッピングを生成できます。

- 1 つの入力トランスフォーメーションと 1 つの出力トランスフォーメーションを含むマプレット
- アクティブなトランスフォーメーションが含まれていないマプレット

マプレットから Web サービスマッピングを生成する場合、Designer はマプレットと、プロセスに許可されたマプレットへのショートカットを一覧表示します。

マプレットから Web サービスマッピングを生成する場合、Designer はマプレット内でポートを使用してソース定義およびターゲット定義のカラムを定義します。次に、マプレット入力ポートを反映する XML ビューのソースと、マプレット出力ポートを反映する XML ビューのターゲット定義を含むマッピングを作成します。

マプレットから生成された Web サービスマッピングには、次のオブジェクトが含まれます。

- Web サービスのソース定義
- ソース修飾子
- マッピングの生成に使用するマプレット
- Web サービスのターゲット定義

Designer は、ソースインスタンスからターゲットインスタンスにポートをリンクします。

再利用可能なトランスフォーメーションまたはマプレットからのマッピングの生成

同じ手順を使用して、再利用可能なトランスフォーメーションまたはマプレットから Web サービスマッピングを生成します。

再利用可能なトランスフォーメーションまたはマプレットから Web サービスマッピングを生成する手順

1. Mapping Designer で、[マッピング] - [Web サービスマッピングの作成] - [トランスフォーメーション/マプレット定義の使用] をクリックします。
2. Web サービスマッピングに使用するトランスフォーメーションまたはマプレットを選択します。
Designer は、入力ポートと、データタイプ、精度、およびスケールのリストを表示します。
マッピング内のソース定義とターゲット定義内のカラムが 1 回出現するか複数回出現するかを指定できます。
3. カラムが複数回出現するように指定する場合は、[ソースとターゲットを複数回出現するオブジェクトにする] を選択します。

このオプションは、ソースとターゲット内のカラムが配列であることを指定します。グループとしてのカラムが複数回出現します。

4. メッセージポートおよびクライアントポートをソース定義またはターゲット定義に追加するには、[メッセージポートの追加] をクリックします。

Designer は、メッセージポートおよびクライアントポートをソース定義またはターゲット定義のエンベロープビューに追加します。

5. [OK] をクリックします。

Designer は、Web サービスマッピングを作成して、マッピングが正しく作成されたことを示すメッセージを表示します。また、トランスフォーメーションまたはマプレットの名前を接頭辞 *m_* を付けたソース定義とターゲット定義の名前として使用します。ソース定義名には接尾辞 *_input* を追加し、ターゲット定義名には *_output* を追加します。

Web サービスマッピングでのターゲットインスタンスの編集

Web サービスマッピングを生成した後、マッピング内のターゲットインスタンスを編集できます。

Mapping Designer でターゲットインスタンスを編集する場合、Target Designer で使用できないプロパティを編集できます。

Web サービスマッピング内のターゲット定義を編集するには、Mapping Designer のターゲット定義インスタンスをダブルクリックします。

[プロパティ] タブ内の次のトランスフォーメーション属性は編集できます。

- ロード範囲
- 部分ロードリカバリ

ロード範囲

ロード範囲属性により、ターゲットのロード範囲を指定できます。Web サービスのターゲット定義のロード範囲は、トランスフォーメーション内のトランスフォーメーション範囲と似ています。

ロード範囲は、次の値に設定できます。

- トランザクション。ロード範囲を [トランザクション] に設定すると、Integration Service はトランザクションのすべてのデータを受信したときに、応答を生成します。ターゲットのグループは、必ず同一のトランザクションジェネレータからデータを受信する必要があります。
- All Input。ロード範囲を [すべての入力] に設定すると、Integration Service はすべての入力データを受信した後に応答を生成します。ターゲットのグループを変えれば、異なるトランザクションジェネレータからデータを受信することができます。ロード範囲が [すべての入力] の場合、Integration Service はコミットを無視します。

デフォルトは [トランザクション] です。トランスフォーメーション範囲の詳細については、『PowerCenter 上級ワークフローガイド』を参照してください。

部分ロードリカバリ

部分ロードリカバリ属性により、リカバリ中の以前の部分ロードの処理方法を指定できます。

Web サービスターゲットの場合、デフォルト値 None を使用します。Web サービスのリカバリは指定できません。

アタッチメント

以下の方法で、アタッチメントを使用するように PowerCenter Web サービスワークフローを設定できます。

- SOAP メッセージのアタッチメントとしてフラットファイルあるいは XML ソースまたはターゲットを使用する
- MIME アタッチメント付きの WSDL を使用する

フラットファイルまたは XML ソースおよびターゲットアタッチメント

SOAP メッセージ要求または応答のアタッチメントとしてデータを受信または送信できます。ソースまたはターゲットとしては、フラットファイルまたは XML ドキュメントを使用できます。例えば、Web サービスアプリケーションからメッセージが格納されたフラットファイルに、定期的に FTP を使ってアクセスします。FTP を使用する代わりに、SOAP 要求のアタッチメントとしてフラットファイルからデータを受信する Web サービスワークフローを作成できます。

SOAP メッセージ要求のアタッチメントとしてデータを受信するには、マッピング内でフラットファイルまたは XML ソース定義を使用します。Web サービスのソースとしてフラットファイルを使用するには、Web Services Provider Reader for Flat Files を使用するように Reader を設定します。マッピングを実行する Web サービスセッションを編集します。セッションプロパティで、[マッピング] タブをクリックして、ソースを選択します。リーダーを Flat File Reader から Web Services Provider Reader for Flat Files に変更します。

SOAP メッセージ応答のアタッチメントとしてデータを送信するには、マッピング内でフラットファイルまたは XML ターゲット定義を使用します。Web サービスのターゲットとしてフラットファイルを使用するには、Web Services Provider Reader for Flat Files を使用するように Writer を設定します。マッピングを実行する Web サービスセッションを編集します。セッションプロパティで、[マッピング] タブをクリックして、ターゲットを選択します。ライターを Flat File Writer から Web Services Provider Writer for Flat Files に変更します。

WSDL アタッチメント

ソース定義とターゲット定義に従って、アタッチメントを SOAP 要求の一部として送受信することができます。アタッチメントは、XML ドキュメントなどのテキストファイルである必要があります。JPEG、GIF、PDF ファイルなどのバイナリドキュメントをアタッチすることはできません。例えば、Oracle データベースから XML ドキュメントを抽出し、応答メッセージへのアタッチメントとしてこの XML 文書を Web サービスに渡すことができます。

ソースとしてバイナリファイルを使用するには、ファイルを hexbinary または base64binary に変換してから Web サービスソースに渡します。hexbinary または base64binary ファイルは、テキストファイルとして扱われます。同様に、Web サービスターゲットによって生成されたテキストファイル応答をバイナリファイルに変換できます。

次の表に、Web サービス定義で使用するアタッチメントグループのポートを示します。

ポート名	説明
FK_Att_Name	ルートグループの PK_Message を指示する生成外部キーです。
Att_Data_Name	アタッチメントを含んでいます。アタッチメントの MIME タイプを [属性] タブに表示することができます。
Att_Index_Name	メッセージ内のアタッチメントごとに割り当てられる一意の識別子です。
Att_Type_Name	アタッチメントのタイプ。

MIME アタッチメント付きの WSDL を使用する規則およびガイドライン

アタッチメントを扱う場合、下記の規則およびガイドラインに従ってください。

- 要求または応答に設定できるアタッチメントは 1 つだけです。
- アタッチメントはテキストファイルとし、UTF-16LE コードページ、または UTF-16LE コードページのサブセットであるコードページを使用する必要があります。
- 要求または応答によりアタッチメントを渡すには、アタッチメントグループ内のすべてのポートを接続する必要があります。
- マッピング内の定義にアタッチメントグループが存在してもアタッチメントの送受信を望まない場合、グループ内のポート間では、接続を一切実行しないでください。
- 他のソースからメッセージを受信し、各メッセージにアタッチメントが指定されている場合、応答時に送信する各アタッチメントに一意のインデックスを生成するときは、Sequence Generator のトランスフォーメーションが利用できます。
- アタッチメントを送信または受信するには、MIME アタッチメントをサポートするツールキットを使用してクライアントアプリケーションを作成する必要があります。

Web サービスワークフローの使用

Web サービスワークフローの使用の概要

Web サービスワークフローを作成するには、Workflow Manager を使用します。Web サービスワークフローを作成するには、ワークフローの [Web サービス] オプションを有効にしてから、Web サービスプロパティを設定します。

Web サービスワークフローにセッションを作成する場合、セッションは Web サービスセッションと呼ばれます。Web サービスセッションに次のタイプのマッピングを入れることができます。

- Web サービスマッピング
- フラットファイルマッピング
- XML マッピング

Web サービスセッションは、Web Services Provider Reader および Writer を使用します。Web サービススマッピングに Web サービスのソースおよびターゲットが含まれる場合、デフォルトではセッションが Web Services Provider Reader および Writer を使用します。Web サービスマッピングにフラットファイ

ルまたは XML ソースまたはターゲットが含まれる場合、Reader および Writer タイプを Web Services Provider Reader または Writer に変更する必要があります。

Web サービスセッションに XML ファイルまたはフラットファイルのソースまたはターゲットが含まれる場合、クライアントアプリケーションは SOAP メッセージへの MIME アタッチメントとして Web Services Hub に要求を送信します。アタッチメントを送信または受信するには、クライアントアプリケーションは MIME アタッチメントをサポートするツールキットによって作成する必要があります。

Web Services Hub は Web サービスワークフローを実行するための SOAP メッセージ要求を受信すると、要求を Integration Service に渡します。Integration Service は Web サービス要求を実行すると、応答を Web Services Hub に渡します。Web Services Hub は SOAP メッセージ応答を生成し、Web サービスクライアントに返信します。

Web サービスのソース定義およびターゲット定義を含むセッションに複数パーティションをセットアップすることが可能になりました。Integration Service はセッション内のソース数、ターゲット数、およびパーティション数を基に Web Services Hub への接続を作成します。

注: Web サービスワークフローを実行できるようにするには、Web Services Hub を管理者ツールに作成および設定し、実行する Web サービスワークフローを含むリポジトリと関連付ける必要があります。

Web サービスワークフローの作成および設定

Web サービスワークフローを作成するには、Web サービスマッピングを処理し、ワークフロープロパティで [Web サービス] オプションが有効になるようにワークフローを設定します。Web サービスを設定すると、Web サービスクライアントがワークフローを実行できるようになります。

Web サービスワークフローを作成および設定するには、以下の作業を実行します。

- Web サービスワークフローを作成します。
- Web サービスを設定します。

Web サービスワークフローの作成

Web サービスワークフローを作成するには、ワークフローの [Web サービス] オプションを有効にします。次に、Web サービスを設定し、Web サービスセッションをワークフローに追加します。Web サービスセッションは、Web サービスマッピングに基づいています。

ほとんどの場合、各 Web サービスワークフローには、1 つの Web サービス入力メッセージソースと必ず 1 タイプの Web サービス出力メッセージターゲットが含まれている必要があります。セッションは、ターゲット内の複数のフォールトビューに対して書き込みができます。一方向 Web サービスは、応答を送信せず、Web サービスのターゲットを必要としません。

Web サービスワークフローを作成する場合は、必ず Integration Service を指定してください。使用可能な Integration Services のリストから選択するには、[Integration Service のブラウズ] ボタンをクリックします。

Web サービスワークフローを作成した後に、Web サービスマッピングを実行するためにセッションを追加できます。セッションを作成してワークフローに追加する方法と同じ方法で、セッションを作成して Web サービスワークフローに追加します。

注: Web サービスワークフローを作成するために、ワークフローウィザードを使用しないでください。ワークフローウィザードの使用時に [Web サービス] オプションは選択できません。

Web サービスワークフローを作成するには：

1. Workflow Manager で、Workflow Designer を開いて、[ワークフロー] - [作成] をクリックします。
2. ワークフローの名前を入力します。
3. ワークフローを実行する Integration Service を選択するには、[Integration Service のブラウズ] ボタンをクリックしてリストから選択します。
4. [Web サービス] オプションを有効にして、[サービスの設定] をクリックして、Web サービスワークフローを設定します。

[Web サービス] オプションを有効にすると、[並列実行の設定] オプションがデフォルトで有効になります。Web サービスワークフローの設定プロパティには、Web サービスの同時実行の設定が含まれています。

5. 必要に応じて、Web サービスワークフロープロパティを設定します。
6. [OK] をクリックします。

Web サービスワークフローの設定

Web サービスワークフローを設定するときに、Web サービスワークフローを実行する Web Services Hub を指定し、Web サービスを実行および利用するためのオプションを設定できます。

以下の表に、Web サービスについて設定するプロパティを示します。

プロパティ	説明
サービス名	Web サービスの名前です。ワークフローのチェック時に、サービスが表示可能ならば Web Services Hub はこの名前を公開します。デフォルト名は、リポジトリ名、フォルダ名、ワークフロー名を結合したものです。この名前は一意でなければなりません。
タイムアウト（秒）	要求がタイムアウトするまで Web Services Hub が要求の処理と SOAP 応答の生成にかかる最大時間。Web Services Hub はタイムアウト期間内に応答を生成することができないと、Web サービスクライアントにフォールトメッセージを送信し、接続を切断します。 デフォルトは 60 秒です。この値を 0 にすれば、タイムアウト期間を無効にすることができます。
サービスタイムしきい値（ミリ秒）	Web Services Hub が、別のインスタンスを開始して次の要求の処理に移るまでに、要求の処理にかかる最大時間。サービス期間は、Web Services Hub が SOAP 要求を受信する時間から SOAP 応答を生成する時間までの期間です。Web Services Hub が要求の処理にかかる平均時間がサービス期間を超えると、Web Services Hub が Web サービスの新しいインスタンスを開始して新しい要求を処理します。 例えば、サービス時間が 1000 ミリ秒に設定されているとします。Web Services Hub が 1000 ミリ秒以内に要求を処理できないと、Web Services Hub が Web サービスの別のインスタンスを開始して次の SOAP 要求を処理します。 デフォルトは 1000 です。 注: パフォーマンス低下を防ぐために、サービスタイムしきい値は 100 ミリ秒未満に設定しないでください。

プロパティ	説明
Web Services Hub	<p>ワークフローを実行する Web Services Hub サービス。 [参照] ボタンをクリックし、Web サービスワークフローを実行するために 1 つ以上の Web Services Hub サービスを選択します。 デフォルトで、Web サービスワークフローは、リポジトリに関連付けられた任意の Web Services Hub サービス上で実行できます。</p> <p>注: ワークフローを手動で開始する予定であれば、Web Services Hub を選択してワークフローを実行します。 [すべての Hub で実行] は選択しないでください。 Web サービスワークフローを開始するためには、Web Services Hub が有効になっていることを確認してください。</p>
1 ハブあたりの最大実行数	<p>Web Services Hub で開始可能な Web サービスインスタンスの最大数。 インスタンスを動的に開始した場合も手動で開始した場合も、Web Services Hub 上で実行している Web サービスワークフローのすべてのインスタンスが回数に含まれます。 Web Services Hub は、最大数に達した場合、別の Web サービスインスタンスを開始できません。</p>
保護	<p>Web サービスを実行できるようにするために、認証が必要です。 Web Services Hub はユーザー名トークンに基づいて要求の認証を行います。 サービスを保護するか、公開するかを選択できます。</p> <p>ワークフローを実行できる PowerCenter ユーザーなら誰でも、Workflow Manager、<i>pmcmd</i>、または LMAPI を使用して、保護された Web サービスワークフローを実行できます。 Web サービスが保護されていない場合、Web サービスクライアントは認証なしでサービスを開始できます。</p> <p>詳細については、「クライアント要求へのセキュリティの追加」 (ページ 69) を参照してください。</p>
ビジブル	<p>Web サービスを Web Services Hub Console 内で表示可能にします。 サービスを表示可能にすると、Web Services Hub は Web Services Hub Console 上の Web サービスと WSDL を公開します。 Web サービスのテスト、WSDL の表示、および Web Services Hub Console からの WSDL のダウンロードが可能です。</p> <p>サービスが表示不可の場合、Web Services Hub は Web サービス WSDL を公開しません。</p>
実行可能	<p>Web Services Hub に要求を送信することによって、Web サービスクライアントにワークフローの開始を許可します。 Web サービスワークフローが実行可能である場合、Web サービスクライアント要求はワークフローの実行中にワークフローを開始するか Web サービスを実行できます。 Web サービスクライアントにワークフローを開始させなければ、オンデマンドで実行するようワークフローをスケジュールしてください。 Web サービスワークフローが実行不可能である場合、Web サービスクライアントはワークフローの実行中に Web サービスを起動できませんが、ワークフローは開始できません。 無効にすると、Workflow Manager、LMAPI、または <i>pmcmd</i> を介してワークフローの開始ができます。</p>

Web サービスワークフローの同時実行

Web Services Hub は、Web サービスのプロパティに設定したリソースおよび値の可用性に基づいて、Web サービスワークフローの新しいインスタンスをいつ開始するかを決定します。 Web Services Provider Reader のプロパティに設定した値に基づいて、Web サービスワークフローのインスタンスをいつシャットダウンするかを決定します。

新しいインスタンスの開始

Web Services Hub は、現在のリソース使用状況と Web サービスワークフローの以下のプロパティに基づいて、Web サービスワークフローの別のインスタンスをいつ開始するかを決定します。

- **サービスタイムしきい値。** Web Services Hub が Web サービスの処理にかかる平均時間がサービスタイムしきい値を超えると、Web Services Hub が Web サービスの別のインスタンスを開始します。
- **1 ハブあたりの最大実行数。** Web Services Hub は、インスタンス数がハブの最大実行回数に達するまで、Web サービスのインスタンスを開始します。最大実行回数に達すると、Web Services Hub は平均サービスタイムしきい値であっても Web サービスの新規インスタンスを開始することはできません。

インスタンスのシャットダウン

Web Services Hub は、現在のリソース使用状況と Web Services Provider Reader の以下のプロパティに基づいて、Web サービスワークフローインスタンスをシャットダウンします。

- **アイドル時間。** ワークフローインスタンスが要求をアイドル時間内に受信しないと、Web Services Hub はワークフローインスタンスをシャットダウンします。
- **メッセージカウント。** ワークフローインスタンスによって受信されたメッセージ数が Integration Service でセッション内で読み取るように設定されているメッセージ最大数に達すると、Web Services Hub がワークフローインスタンスをシャットダウンします。
- **Reader の時間制限。** Integration Service が Web Services Hub から入力メッセージを読み取ることができる最大時間に達すると、Integration Service は Web Services Hub からの入力メッセージの読み取りを停止します。Web Services Hub がワークフローインスタンスをシャットダウンします。

これらのプロパティのいずれかがワークフローに設定されたしきい値に到達すると、Web Services Hub は Web サービスワークフローインスタンスをシャットダウンします。

関連項目：

- [「Web サービスワークフローの設定」 \(ページ 97\)](#)
- [「Web Services Provider Reader の設定」 \(ページ 100\)](#)

Web Services Provider Reader および Writer の設定

Web サービスセッションを設定する場合、セッションの Reader および Writer を設定できます。デフォルトで、Web サービスのソースおよびターゲットがある Web サービスセッションは Web Services Provider Reader および Writer を使用します。

Web サービスセッションにフラットファイルまたは XML ソースまたはターゲットが含まれる場合、Web Services Provider Reader または Writer を使用するようセッションを設定する必要があります。Web Services Hub は、要求および応答を MIME アタッチメントとして SOAP メッセージに送信します。

Web サービスセッションの Reader の設定時には、アイドル時間やメッセージ数などの終了条件を設定します。

Web サービスセッションの Writer の設定時、Integration Service がターゲットデータをキャッシュするために使用するキャッシュ情報を設定します。このほか、ターゲットデータの出力形式も設定できます。

Web サービスセッションを設定するには、Workflow Manager を使用します。Workflow Designer で、Web サービスワークフローのセッションを編集します。Web Services Provider Reader を設定するに

は、[マッピング] タブをクリックしてソースを選択します。Web Services Provider Writer を設定するには、ターゲットを選択します。

関連項目：

- [「アタッチメント」](#) (ページ 94)

Web Services Provider Reader の設定

Web Services Provider Reader に設定するプロパティは、マッピングで使用するソースタイプによって異なります。

以下の表に、Web サービスセッションに設定するソースプロパティを示します。

プロパティ	Reader タイプ	説明
アイドル時間	<ul style="list-style-type: none">- Web サービス- Web Services Provider Reader フラットファイル- Web Services Provider Reader XML ファイル	Integration Service がメッセージの受信を待機した後にソースからの読み取りを停止し、Web Services Hub がワークフローインスタンスをシャットダウンするまでの時間の長さ（秒単位）です。 セッションは、このプロパティの条件を満たすと停止します。 デフォルトは 180 です。
メッセージカウント	<ul style="list-style-type: none">- Web サービス- Web Services Provider Reader フラットファイル- Web Services Provider Reader XML ファイル	Web Services Hub がワークフローインスタンスをシャットダウンするまでに Integration Service が読み取るメッセージの数。-1 の値は無限時間を示します。セッションにフラットファイルまたは XML のターゲットが使用されている場合、常にメッセージカウントを 1 に設定してください。詳細については、 「XML およびフラットファイルセッションの Reader および Writer の設定」 (ページ 102) を参照してください。 セッションは、このプロパティの条件を満たすと停止します。 デフォルトは-1 です。
Reader の時間制限	<ul style="list-style-type: none">- Web サービス- Web Services Provider Reader フラットファイル- Web Services Provider Reader XML ファイル	Integration Service が Web Services Hub からソースメッセージを読み込む時間（秒単位）です。例えば、Reader Time Limit に 10 を指定した場合、Integration Service は 10 秒後に Web Services Hub からの読み取りを停止します。 セッションは、このプロパティの条件を満たすと停止します。 デフォルトは 0 であり、これは無限時間を示します。
空のコンテンツを Null として扱う	Web Services Provider Reader XML ファイル	空の文字列を Null 値として扱います。デフォルトでは、空のコンテンツは Null ではありません。
リカバリキャッシュフォルダ	なし	このプロパティは Web Services Provider では使用されません。

Web Services Provider Writer の設定

Web Services Provider Writer のセッションプロパティの設定時に、キャッシュサイズとキャッシュディレクトリを設定します。

以下の表に、Web サービスセッションに設定するターゲットプロパティを示します。

プロパティ	Writer タイプ	説明
XML 日付フォーマット	Web Services Provider Writer XML ファイル	<p>サービスターゲットに渡されたデータの形式を決定します。精度はナノ秒。</p> <p>日付フォーマットを次から選択します。</p> <ul style="list-style-type: none"> - Local Time。Integration Service のタイムゾーンに従った時刻。 - Local Time with Time Zone。Integration Service のタイムゾーンとグリニッジ標準時との時差。 - UTC。グリニッジ標準時。
Null コンテンツの表現	Web Services Provider writer XML ファイル	<p>無効なコンテンツがどのようにターゲット内で表されるかを決定します。</p> <p>次のオプションから選択します。</p> <ul style="list-style-type: none"> - No Tag。タグを出力しません。 - 空のコンテンツを含むタグ。タグのみを出力します。 <p>デフォルトは [タグなし] です。</p>
空の文字列コンテンツの表現	Web Services Provider writer XML ファイル	<p>空の文字列がターゲット内でどのように表されるかを決定します。</p> <p>次のオプションから選択します。</p> <ul style="list-style-type: none"> - No Tag。タグを出力しません。 - 空のコンテンツを含むタグ。タグのみを出力します。 <p>デフォルトは [空のコンテンツを含むタグ] です。</p>
重複グループ行の処理	Web Services Provider writer XML ファイル	<p>セッション中に Integration Service が重複グループ行をどのように扱うかを決定します。</p> <p>次のオプションから選択します。</p> <ul style="list-style-type: none"> - First Row。Integration Service は、重複行のうち最初の行をターゲットに渡します。この行の後に処理された同じプライマリキーを持つ行は、Integration Service で拒否されます。 - Last Row。Integration Service は、重複行のうち最後の行をターゲットに渡します。 - エラー。Integration Service は、最初の行をターゲットに渡します。以降に重複するプライマリキーを持つ行が見つかったら、エラーカウントが 1 つ増やされます。エラーカウントがエラーしきい値を超えると、セッションが失敗します。 <p>デフォルトがエラー。</p>
孤立行の処理	Web Services Provider Writer XML ファイル	<p>セッション中に Integration Service が孤立した行をどのように扱うかを決定します。</p> <p>次のオプションから選択します。</p> <ul style="list-style-type: none"> - Ignore。Integration Service は孤立した行を無視します。 - エラー。エラーカウントがエラーしきい値を超えると、セッションが失敗します。

プロパティ	Writer タイプ	説明
キャッシュサイズ	<ul style="list-style-type: none"> - Web サービス - Web Services Provider Writer - XML ファイル 	<p>Writer が使用するメモリキャッシュの総サイズ（バイト）。</p> <p>これには、ターゲットインスタンス内の各グループのプライマリキーおよび外部キーインデックスキャッシュと、全グループ分の 1 つのデータキャッシュが含まれます。総キャッシュ所要量は、各ターゲットグループのデータキャッシュおよびインデックスキャッシュの所要量の合計です。</p> <p>デフォルトは 10,000,000 バイトです。</p>
キャッシュディレクトリ	<ul style="list-style-type: none"> - Web サービス - Web Services Provider Writer - XML ファイル 	<p>ターゲットキャッシュファイルのディレクトリ。デフォルトは、\$PMCacheDir サービスプロセス変数です。</p>

Writer のタイプを Web Services Provider Writer に変更する場合は、次に示す規則とガイドラインを使用します。

- フラットファイルターゲットの Writer のタイプを変更する場合、Integration Service はターゲットメッセージをキャッシュしません。
- フラットファイルまたは XML ターゲットの Writer タイプを変更する場合は、ターゲットを Web サービスの出力メッセージとして使用できますが、フォールトメッセージとしては使用できません。
- XML ターゲットの Writer のタイプを変更する場合は、XML Writer のプロパティを設定します。

XML およびフラットファイルセッションの Reader および Writer の設定

XML またはフラットファイルのソースおよびターゲットを含むマッピングに基づいて Web サービスセッションを作成するには、Reader または Writer タイプを Web Services Provider Reader または Writer に設定します。XML またはフラットファイルリーダーで Web サービスワークフローを実行するには、クライアントアプリケーションは SOAP メッセージへの MIME アタッチメントとして、要求を Web Services Hub に送信します。Web Services Hub が SOAP メッセージをアタッチメントと一緒に Integration Service に渡すと、アタッチメントが処理されます。

Web サービスワークフローが XML またはフラットファイルライタを使用して設定されている場合、Integration Service は応答を生成し、その応答を Web Services Hub に渡します。Web Services Hub は応答を Web サービスクライアントに SOAP メッセージへの MIME アタッチメントとして送り戻します。

フラットファイルまたは XML のソースまたはターゲットが存在する要求/応答 Web サービスセッションを設定する場合、次の規則およびガイドラインに従ってください。

- Reader プロパティでメッセージカウントを 1 に設定します。
- Reader または Writer のタイプを Web Services Provider に変更する場合、ワークフローには 1 つのセッションを含めることができます。
- セッションプロパティで Reader または Writer のタイプを Web Services Provider Reader または Writer に変更する場合は、MIME アタッチメントをサポートするツールキットを使用してクライアントアプリケーションを作成する必要があります。

Web サービスセッションのパーティションの設定

Web サービスのソース定義とターゲット定義が含まれるセッションで複数のパーティションを設定すると、Integration Service は、セッション内のソース数、ターゲット数、およびパーティション数に基づいて Web Services Hub への接続を作成します。例えば、ソースとターゲットがそれぞれ 1 つのセッションで 3 つのパーティションを設定すると、Integration Service は Web Services Hub に対して 6 つの接続を作成します。内訳はソースとターゲットがそれぞれ 3 つずつです。パーティションにより、Web サービス要求の同時実行が可能になります。

複数のパーティションがあるセッションを実行すると、Web Services Hub はソース接続を使って要求を Integration Service に渡します。Integration Service はターゲット接続を使用して、Web Services Hub に応答を送信します。Web Services Hub と Integration Service は、ソース接続とターゲット接続をラウンドロビン方式で使用します。

Web サービスマッピングのパーティションを設定すると、Web サービスのソースおよびターゲットに対するパススルーパーティションを設定できます。

Web サービスワークフローのトラブルシューティング

Web サービスセッションに対して Debugger を実行しようとしていますが、セッションが失敗して、セッションの実行にはワークフローコンテキストが必要であることを示すエラーメッセージがセッションログに表示されます。

Web サービスセッションをデバッグする場合、Web サービスワークフローに対して Debugger を実行する必要があります。ワークフローがなければ、Web サービスマッピングや再利用セッションに対して Debugger を実行することはできません。

ソースの WSDL を更新した後で、自分のソース定義とターゲット定義をインポートしなおしました。ワークフローは有効ですが、なぜかサービスの WSDL が更新されません。

マッピングを変更しても、Web Services Hub に動的に反映されることはありません。マッピングの変更を反映するために WSDL を生成する場合は、ワークフローを編集して保存する必要があります。ワークフローを保存すると、Web Services Hub がサービスの WSDL を生成します。

Web サービスワークフローが Workflow Manager で有効になっているにもかかわらず、Web Services Hub を開始したら無効になりました。

Web Services Hub を開始した後、Web Service Hub は各 Web サービスワークフローに対し、Workflow Manager の検査規則に加えて、独自の検査規則で検査します。

Web Services Hub は、次の規則に従って Web サービスワークフローを検査します。

- マッピングに複数の Web サービスのソース定義は存在しない。
- マッピングに複数の Web サービスのターゲット定義は存在しない。
- マッピングに Web サービスのターゲット定義が存在しない場合、Web Services Hub は Web サービスを一方向サービスとして扱う。
- Repository Service は、Web Services Hub に関連付けられている必要がある。
- Integration Service はワークフローに関連付けられている必要がある。

Workflow Manager の [検査] タブで Web Services Hub のエラーメッセージを確認し、メッセージに従い問題を修正する。

Web Services Hub 上のワークフローをフェッチしようとしたところ、サービスワークフローに対して Integration Service が指定されていないため、サービスワークフローが無効であることを示すエラーメッセージが表示されました。

Web サービスワークフローを作成する場合、Integration Service を割り当てる必要があります。詳細については、[「Web サービスワークフローの設定」 \(ページ 97\)](#)を参照してください。

Web Services Hub 上の複数のインスタンスを実行するように設定された Web サービスワークフローに要求を送信しました。要求の送信後に Web サービスワークフローを停止しました。フォールト応答を受信しました。

Web Services Hub がワークフローのステータスを定期的にチェックします。要求をワークフローに送信する際には、ワークフローが実行されていないことが登録される前にフォールト応答が生成されます。複数のインスタンスを実行するようにワークフローが設定されている場合、Web Services Hub はワークフローの別のインスタンスを開始します。ただし、Web Services Hub は要求をキャッシュに格納しないため、ワークフローの新しいインスタンスに要求を再送信することはできません。

バージョン管理されたリポジトリ内で、リアルタイム Web サービスワークフローを変更しました。ワークフローを実行しても、変更内容が有効になっていません。

バージョン管理されたリポジトリでリアルタイム Web サービスワークフローを変更した場合、変更内容を有効にするには、ワークフローをチェックインする必要があります。

例えば、リアルタイム Web サービスワークフローを変更し、異なる PowerCenter Integration Service に関連付けます。変更内容をチェックインすると、Web Services Hub は新しい Integration Service を使用してワークフローを実行します。変更内容をチェックインしないと、Web Services Hub を再起動しない限り、Web Services Hub は新しい Integration Service を使用しません。

Web サービスのサンプルクライアントアプリケーション

Web サービスのサンプルクライアントアプリケーションの概要

Informatica は、PowerCenter Web サービスの使用方法を説明するサンプルのクライアントアプリケーションプログラムを出荷しています。このサンプルには、Java および C# のプログラムが含まれます。Java サンプルプログラムでは、Axis Web サービスツールキットによって生成されるプロキシクラスが使用されます。C# サンプルプログラムでは、wsdl.exe ツールによって .NET プラットフォーム用に生成されたプロキシクラスが使用されます。サンプルプログラムは PowerCenter バッチ Web サービスとリアルタイム Web サービスで使用できます。

Web サービスのサンプルプログラムは、次のディレクトリにインストールされます。

```
/<PowerCenterInstallDir>/server/samples/WebServices
```

Web サービスサンプルプログラムを実行するためには、PowerCenter ドメインで Web Services Hub を作成および有効化します。管理者ツールを使用して、Web Services Hub の作成、設定、有効化を行います。

バッチ Web サービスサンプルプログラムの使用

バッチ Web サービスのサンプルプログラムを使用するには、PowerCenter をインストールして実行する必要があります。PowerCenter ドメインには、Repository Service に関連付けられている Web Services Hub が含まれている必要があります。

バッチ Web サービスのサンプルプログラムは、次のディレクトリにインストールされます。

`/<PowerCenterInstallDir>/server/samples/WebServices`

以下の表に、/WebServices ディレクトリ内のファイルとディレクトリを示します。

ディレクトリ	説明
/lib	サンプルプログラムの実行に必要なライブラリファイルが含まれます。
/ssl	セキュアモード (HTTPS) でクライアントアプリケーションを実行するためのサンプルキーストアが含まれます。
/samples/BatchWebServices/axis/ <SampleProgramDirectory>	Java サンプルプログラムが含まれます。各バッチ Web サービスのサンプルプログラムのソースファイルは、それぞれのディレクトリにあります。ディレクトリの名前は、サンプルプログラムで実演されるバッチ Web サービスの操作を示します。たとえば、/multiservers ディレクトリのサンプルプログラムは、Repository Service に関連付けられた複数の Integration Service へのログインを実演します。 このディレクトリには、サンプルプログラムをコンパイルおよび実行するためのバッチファイルとスクリプトファイルも含まれます。
/samples/BatchWebServices/axis/ proxyclasses	Java サンプルプログラムのプロキシクラスが含まれます。
/samples/BatchWebServices/dotnet/csharp/ <SampleProgramDirectory>	C# サンプルプログラムが含まれます。各バッチ Web サービスのサンプルプログラムのソースファイルは、それぞれのディレクトリにあります。ディレクトリの名前は、サンプルプログラムで実演されるバッチ Web サービスの操作を示します。たとえば、/multiservers ディレクトリのサンプルプログラムは、Repository Service に関連付けられた複数の Integration Service へのログインを実演します。 各サンプルプログラムのディレクトリには、サンプルプログラムをコンパイルするためのバッチファイルも含まれます。
/samples/BatchWebServices/dotnet/csharp/ proxyclasses	C# サンプルプログラムのプロキシクラスが含まれます。このディレクトリには、プロキシクラスをコンパイルするためのバッチファイルも含まれます。

バッチ Web サービスサンプルプログラムのコンパイル

バッチ Web サービスのサンプルプログラムをコンパイルする手順は、プログラミング言語によって決まります。

サンプル Java プログラムのコンパイル

サンプル Java プログラムをコンパイルするには、サンプルプログラムのディレクトリに移動し、コンパイルバッチまたはスクリプトファイルを実行します。コンパイルするサンプルプログラムの名前と一致するバッチまたはスクリプトファイルを実行します。

たとえば、/axis/multithreaded ディレクトリの Sample1.java プログラムをコンパイルするには、そのディレクトリに移動して、CompileSample1.bat (Windows) または CompileSample1.sh (UNIX) を実行します。コンパイルプロセスによって、同じディレクトリにサンプルプログラムの.class ファイルが作成されます。

サンプル C#プログラムのコンパイル

サンプル C#プログラムをコンパイルするには、次の手順を実行します。

1. /dotnet/csharp/proxyclasses ディレクトリに移動して、compile.bat を実行します。
コンパイルプロセスによって、次のディレクトリに WebServicesHub.dll という名前のダイナミックリンクライブラリが作成されます。
/dotnet/csharp/bin directory.
2. サンプルプログラムのディレクトリに移動して、コンパイルするサンプルプログラムの名前と一致するコンパイルバッチファイルを実行します。
コンパイルプロセスによって、コンパイル対象プログラムファイルの名前と.exe 拡張子が付けられた実行可能ファイルが作成されます。

バッチ Web サービスサンプルプログラムの実行

Web Services Hub は、クライアントアプリケーションの実行時に実行されている必要があります。

必要なパラメータを付けて、サンプルプログラムを実行します。バッチ Web サービスのサンプルプログラムを実行する手順は、プログラミング言語によって決まります。

サンプル Java プログラムの実行

サンプル Java プログラムを実行するには、サンプルプログラムのディレクトリに移動し、実行するサンプルプログラムの名前と一致するバッチまたはスクリプトファイルを実行します。たとえば、/axis/multithreaded ディレクトリの Sample1.java プログラムをコンパイルするには、そのディレクトリに移動して、RunSample1.bat (Windows) または RunSample1.sh (UNIX) を実行します。

サンプル C#プログラムの実行

サンプル C#プログラムを実行するには、サンプルプログラムのディレクトリに移動し、実行するサンプルプログラムの実行可能ファイルを実行します。

バッチ Web サービスの例

以下の表に、サンプルプログラムを含むディレクトリを示します。

プラットフォーム	ディレクトリ
Java	/WebServices/samples/BatchWebServices/axis/<SampleProgramDirectory>
C#	/WebServices/samples/BatchWebServices/dotnet/csharp/<SampleProgramDirectory>

Java および C#に対して、同じサンプルプログラムのセットが出荷されます。各プラットフォームには同じディレクトリが含まれ、各ディレクトリには Web サービスの異なる使用方法を実演するサンプルプログラムが含まれています。ここでは、Java および C#のサンプルプログラムについて説明します。

参照

/browsing ディレクトリのサンプルプログラムは、リポジトリから情報を取得する Web サービス操作の使用方法を実演します。

Sample1.java and Sample1.cs

このサンプルプログラムは、リポジトリにログインしてから、メタデータ Web サービス操作を使用して、リポジトリおよびリポジトリで登録された Integration Services 内のフォルダ、ワークフローおよびタスクに関する情報を取得します。

ディレクトリ: /browsing

Java および C#のサンプルをコンパイルするファイル: CompileSample1.bat または CompileSample1.sh

Java サンプルを実行するファイル: RunSample1.bat または RunSample1.sh

C#のサンプルを実行するファイル: Sample1.exe

以下の表は、Sample1 アプリケーションの実行に使用するパラメータについて説明します。

パラメータ	説明
セキュリティモード	アプリケーションを実行するセキュリティモードを指定します。引数 <i>-ns</i> を渡すことによって、アプリケーションを非セキュアモード (HTTP) で実行します。 例では、セキュアモード (HTTPS) をサポートしていません。
ホスト名	Web Services Hub を実行しているマシンの名前または IP アドレス。
ポート番号	Web Services Hub を実行しているポート番号。
リポジトリドメイン名	リポジトリサービスを含むドメインの名前。
リポジトリ名	リポジトリサービスの名前。

パラメータ	説明
ユーザー名	リポジトリにログインするユーザー名。
パスワード	リポジトリにログインするユーザー名のパスワード。

Sample2.java and Sample2.cs

このサンプルプログラムは、リポジトリにログインして、関連する Integration Service に接続します。また、メタデータおよびデータ統合 Web サービス操作を使用して、リポジトリ内のフォルダにアクセスし、フォルダ内の最初のワークフローを開始および停止します。

ディレクトリ: /browsing

Java および C#のサンプルをコンパイルするファイル: CompileSample2.bat または CompileSample2.sh

Java サンプルを実行するファイル: RunSample2.bat または RunSample2.sh

C#のサンプルを実行するファイル: Sample2.exe

以下の表に、Sample2 アプリケーションの実行に使用するパラメータを示します。

パラメータ	説明
セキュリティモード	アプリケーションを実行するセキュリティモードを指定します。引数 <i>-ns</i> を渡すことによって、アプリケーションを非セキュアモード (HTTP) で実行します。 例では、セキュアモード (HTTPS) をサポートしていません。
ホスト名	Web Services Hub を実行しているマシンの名前または IP アドレス。
ポート番号	Web Services Hub を実行しているポート番号。
リポジトリドメイン名	リポジトリサービスを含むドメインの名前。
リポジトリ名	リポジトリサービスの名前。
ユーザー名	リポジトリにログインするユーザー名。
パスワード	リポジトリにログインするユーザー名のパスワード。
Integration Service のドメイン名	Integration Service を含むドメインの名前。
Integration Service 名	Integration Service の名前。

データ統合

/dataintegration ディレクトリのサンプルプログラムは、Data Integration Web サービス内で使用可能なワークフローおよびタスクの使用方法を実演します。

Sample1.java and Sample1.cs

このサンプルプログラムは、リポジトリにログインして、関連する Integration Service に接続します。また、データ統合 Web サービス操作を使用して、Integration Service で実行するワークフローを開始および停止します。

ディレクトリ: /dataintegration

Java および C#のサンプルをコンパイルするファイル: CompileSample1.bat または CompileSample1.sh

Java サンプルを実行するファイル: RunSample1.bat または RunSample1.sh

C#のサンプルを実行するファイル: Sample1.exe

以下の表は、Sample1 アプリケーションの実行に使用するパラメータについて説明します。

パラメータ	説明
セキュリティモード	アプリケーションを実行するセキュリティモードを指定します。引数 <i>-ns</i> を渡すことによって、アプリケーションを非セキュアモード (HTTP) で実行します。 例では、セキュアモード (HTTPS) をサポートしていません。
ホスト名	Web Services Hub を実行しているマシンの名前または IP アドレス。
ポート番号	Web Services Hub を実行しているポート番号。
リポジトリドメイン名	リポジトリサービスを含むドメインの名前。
リポジトリ名	リポジトリサービスの名前。
ユーザー名	リポジトリにログインするユーザー名。
パスワード	リポジトリにログインするユーザー名のパスワード。
Integration Service のドメイン名	Integration Service を含むドメインの名前。
Integration Service 名	Integration Service の名前。
フォルダ名	リポジトリ内のフォルダの名前。
ワークフロー名	セッションを含むワークフローの名前。
タスク名	開始するタスクの名前。

複数の Integration Service

/multiservers ディレクトリのサンプルプログラムは、リポジトリサービスに関連付けられた複数の Integration Service へのログインを実演します。同じ方式を使用して、リポジトリサービスに関連付けられた任意の数の Integration Service に同時にアクセスできます。

Sample1.java and Sample1.cs

このサンプルプログラムは、リポジトリにログインして、リポジトリに関連付けられた 2 つの Integration Service に接続します。データ統合 Web サービス操作を使用して、2 つの Integration Service のプロパティを取得します。

注: 例に示すように、2 つの Integration Service にログインするには、Data Integration Web サービスに対して 2 つのプロキシオブジェクトを作成する必要があります。ログインする各 Integration Service に対して、1 つのプロキシオブジェクトを作成します。

ディレクトリ: /multiservers

Java および C#のサンプルをコンパイルするファイル: CompileSample1.bat または CompileSample1.sh

Java サンプルを実行するファイル: RunSample1.bat または RunSample1.sh

C#のサンプルを実行するファイル: Sample1.exe

以下の表は、Sample1 アプリケーションの実行に使用するパラメータについて説明します。

パラメータ	説明
セキュリティモード	アプリケーションを実行するセキュリティモードを指定します。引数 <i>-ns</i> を渡すことによって、アプリケーションを非セキュアモード (HTTP) で実行します。 例では、セキュアモード (HTTPS) をサポートしていません。
ホスト名	Web Services Hub を実行しているマシンの名前または IP アドレス。
ポート番号	Web Services Hub を実行しているポート番号。
リポジトリドメイン名	リポジトリサービスを含むドメインの名前。
リポジトリ名	リポジトリサービスの名前。
ユーザー名	リポジトリにログインするユーザー名。
パスワード	リポジトリにログインするユーザー名のパスワード。
Integration Service のドメイン名	Integration Service を含むドメインの名前。
Integration Service 名 1	リポジトリに関連付けられた Integration Service の名前。
Integration Service 名 2	リポジトリに関連付けられた 2 つ目の Integration Service の名前。

マルチスレッド

/multithreaded ディレクトリのサンプルプログラムは、操作を並行して実行するマルチスレッドでのプロキシオブジェクトの使用方法を実演します。同じ方式を使用して、クライアントアプリケーションが操作の完了を待機するときに、継続して他の操作を実行し呼び出すようにクライアントアプリケーションを有効化できます。たとえば、クライアントアプリケーションが WaitTillWorkflowComplete 操作をスレッドで呼び出す場合、アプリケーションは継続して他の操作を他のスレッドで実行できます。

Sample1.java and Sample1.cs

このサンプルプログラムは、リポジトリにログインして、関連する Integration Service に接続します。2つのスレッドを開始して、データ統合 Web サービスプロキシオブジェクトを両方のスレッドに渡します。一方のスレッドでは、Integration Service 上のワークフローを開始し、それが完了するまで待機します。残りのスレッドでは、Integration Service のプロパティを取得します。同様に、複数のスレッドでメタデータ Web サービスのプロキシオブジェクトを使用することもできます。

ディレクトリ: /multithreaded

Java および C#のサンプルをコンパイルするファイル: CompileSample1.bat または CompileSample1.sh

Java サンプルを実行するファイル: RunSample1.bat または RunSample1.sh

C#のサンプルを実行するファイル: Sample1.exe

以下の表は、Sample1 アプリケーションの実行に使用するパラメータについて説明します。

パラメータ	説明
セキュリティモード	アプリケーションを実行するセキュリティモードを指定します。引数 <i>-ns</i> を渡すことによって、アプリケーションを非セキュアモード (HTTP) で実行します。 例では、セキュアモード (HTTPS) をサポートしていません。
ホスト名	Web Services Hub を実行しているマシンの名前または IP アドレス。
ポート番号	Web Services Hub を実行しているポート番号。
リポジトリドメイン名	リポジトリサービスを含むドメインの名前。
リポジトリ名	リポジトリサービスの名前。
ユーザー名	リポジトリにログインするユーザー名。
パスワード	リポジトリにログインするユーザー名のパスワード。
Integration Service のドメイン名	Integration Service を含むドメインの名前。
Integration Service 名	Integration Service の名前。
フォルダ名	ワークフローを含むリポジトリ内のフォルダの名前。
ワークフロー名	リポジトリ内のワークフローの名前。

Web Services Hub のテスト

/testsamples ディレクトリのサンプルプログラムは、有効な Web Services Hub が PowerCenter ドメインで実行中であることを確認する方法を実演します。

Sample1.java and Sample1.cs

このサンプルプログラムは、リポジトリにログインして、関連する Integration Service に接続します。メタデータおよびデータ統合の Web サービス操作を使用して、Repository Service および Integration Service に関する情報を取得します。

ディレクトリ: /testsamples

Java および C#のサンプルをコンパイルするファイル: CompileSample1.bat または CompileSample1.sh

Java サンプルを実行するファイル: RunSample1.bat または RunSample1.sh

C#のサンプルを実行するファイル: Sample1.exe

以下の表は、Sample1 アプリケーションの実行に使用するパラメータについて説明します。

パラメータ	説明
セキュリティモード	アプリケーションを実行するセキュリティモードを指定します。引数 <i>-ns</i> を渡すことによって、アプリケーションを非セキュアモード (HTTP) で実行します。 例では、セキュアモード (HTTPS) をサポートしていません。
ホスト名	Web Services Hub を実行しているマシンの名前または IP アドレス。
ポート番号	Web Services Hub を実行しているポート番号。
リポジトリドメイン名	リポジトリサービスを含むドメインの名前。
リポジトリ名	リポジトリサービスの名前。
ユーザー名	リポジトリにログインするユーザー名。
パスワード	リポジトリにログインするユーザー名のパスワード。
Integration Service のドメイン名	Integration Service を含むドメインの名前。
Integration Service 名	Integration Service の名前。

リアルタイム Web サービスサンプルプログラムの使用

リアルタイム Web サービスのサンプルプログラムを使用するには、PowerCenter をインストールして実行する必要があります。PowerCenter ドメインには、Repository Service に関連付けられている Web Services Hub が含まれている必要があります。

リアルタイム Web サービスのサンプルプログラムは、次のディレクトリにインストールされます。

`/<PowerCenterInstallDir>/server/samples/WebServices`

リアルタイム Web サービスの例には、サンプルプログラムで使用するルックアップテーブルおよび Web サービスワークフローを作成するファイルが含まれます。

以下の表に、/RealTimeWebServices ディレクトリ内のファイルとディレクトリを示します。

ディレクトリ	説明
/samples/RealTimeWebServices/ImportXML	リアルタイム Web サービスのサンプルプログラムによって呼び出される Web サービスワークフローが含まれます。サンプルプログラムを使用するには、XML ファイルをリポジトリにインポートして、Web サービスワークフロー内の SQL および Lookup トランスフォーメーション用のデータベース接続をセットアップします。
/lib	サンプルプログラムの実行に必要なライブラリファイルが含まれます。
/samples/RealTimeWebServices/SQLScripts /SINGLEROWLOOKUP	1 行ルックアップのサンプルプログラムで使用されるルックアップテーブルを作成する SQL スクリプトが含まれます。SQL スクリプトを実行すると、選択したデータベース内にテーブルを作成します。
/samples/RealTimeWebServices/SQLScripts /MULTIPLEROWLOOKUP	複数行ルックアップのサンプルプログラムで使用されるルックアップテーブルを作成する SQL スクリプトが含まれます。SQL スクリプトを実行すると、選択したデータベース内にテーブルを作成します。
/samples/RealTimeWebServices/Unprotected WebServices/axis/<SampleProgramDirectory>	Java サンプルプログラムが含まれます。各リアルタイム Web サービスのサンプルプログラムのソースファイルは、それぞれのディレクトリにあります。各ディレクトリには、サンプルプログラムをコンパイルおよび実行するバッチおよびスクリプトファイルが含まれ、サンプルプログラムで使用されるプロキシクラスのサブフォルダが含まれます。

リアルタイム Web サービスの例を使用するには、以下の手順を実行する必要があります。

1. サンプルプログラムによってルックアップテーブルとして使用されるデータソーステーブルを作成します。
2. Web Services Hub に関連付けられたリポジトリにマッピングと Web サービスワークフローをインポートします。
3. m_CustomerLookup_MULTIPLEROW マッピングで、SQL トランスフォーメーションのデータベースおよびデータタイプを変更します。
4. サンプルワークフロー内のデータベース接続設定をセットアップします。
5. リアルタイム Web サービスのサンプルプログラムをコンパイルします。
6. リアルタイム Web サービスのサンプルプログラムを実行します。

手順 1。ルックアップテーブルを作成するには：

バッチ Web サービスのサンプルプログラムに付属の SQL スクリプトファイルを使用して、リレーショナルデータベースにルックアップテーブルを作成します。以下のデータベース内に、ルックアップテーブルを作成できます。

- IBM DB2
- Informix
- Microsoft SQL Server

- Oracle
- Sybase
- Teradata

注: Teradata にルックアップテーブルを作成する場合、データベースサーバーのデフォルトモードを ANSI に設定する必要があります。

以下の表に、SQL スクリプトを示します。

スクリプトファイル名	説明
CustomerLookup_SINGLEROW_<Database>.sql	1 行ルックアップのサンプルプログラムで使用する SINGLEROWLOOKUP という名前の顧客テーブルを作成します。
CustomerLookup_MULTIPLEROW_<Database>.sql	複数行ルックアップのサンプルプログラムで使用する MULTIPLEROWLOOKUP という名前の顧客テーブルを作成します。

注: データベース接続設定。 サンプルワークフローをリポジトリにインポートした後に、ワークフロー内のトランスフォーメーションのデータベース接続設定をデータベース設定に合うように変更する必要があります。

手順 2. マッピングとワークフローのインポート

リアルタイム Web サービスのサンプルプログラムは、サンプル Web サービスのワークフローを実行します。 サンプルプログラムを使用するには、Web Services Hub に関連付けられたリポジトリにサンプルのマッピングとワークフローをインポートします。

以下の表に、XML ファイルを示します。

スクリプトファイル名	説明
wf_CustomerLookup_SINGLEROW.XML	Web サービスワークフローと、1 行ルックアップのサンプルプログラムで使用する Lookup トランスフォーメーションが含まれます。
wf_CustomerLookup_MULTIPLEROW.XML	Web サービスワークフローと、複数行ルックアップのサンプルプログラムで使用する SQL トランスフォーメーションが含まれます。

手順 3. SQL トランスフォーメーションのデータベースおよびデータタイプの変更

複数行ルックアップを実演する Web サービスの例では、SQL トランスフォーメーションを使用します。使用するデータベースによって、SQL トランスフォーメーション内でポートに使用可能なネイティブのデータタイプが決まります。適切なデータベースを使用するように SQL トランスフォーメーションを設定し、適切なネイティブデータタイプを使用するようにポートを設定する必要があります。

SQL トランスフォーメーションのデータベースおよびデータタイプを変更するには、次の手順を実行してください。

1. PowerCenter Designer で、Mapping Designer 内の m_CustomerLookup_MULTIPLEROW マッピングを開いて、sql_Customer トランスフォーメーションインスタンスを編集します。

2. [トランスフォーメーションの編集] ウィンドウで、[SQL 設定] タブに移動して [データベースタイプ] 属性の値を例で使用しているデータベースに設定します。
3. [SQL ポート] タブに移動して、ポートのデータタイプが適切なネイティブデータタイプにマッピングされていることを確認します。
ほとんどのデータベースでは、デフォルトのネイティブデータタイプマッピングが適しています。Microsoft SQL Server および Sybase の場合は、string データタイプを varchar ネイティブデータタイプにマッピングします。
4. 変更は、m_CustomerLookup_MULTIPLEROW マッピングに保存します。
マッピングを変更した後に、マッピングを実行するワークフローを更新します。
5. PowerCenter Workflow Manager で、マッピングを実行するワークフローを開いて、マッピングを更新します。

手順 4. データベース接続設定の変更

インポートされたワークフロー内の SQL およびルックアップトランスフォーメーションは、手順 1 で作成されたサンプルルックアップテーブルに接続する必要があります。

インポートプロセスは、サンプルワークフロー内のトランスフォーメーションの接続オブジェクトをインポートしません。接続オブジェクトを作成して、セッション内で使用する必要があります。

トランスフォーメーションの接続設定を更新するには、以下の手順を実行します。

1. PowerCenter Workflow Manager で、サンプルテーブルに接続するために接続オブジェクトを作成します。
2. s_m_CustomerLookup_SINGLEROW セッションを編集して、lcp_Customer トランスフォーメーションのリレーショナル接続情報を更新します。
リレーショナル接続を新しい接続オブジェクトの名前に設定します。新しい設定のセッションを保存します。
3. s_m_CustomerLookup_MULTIPLEROW セッションを編集して、sql_Customer トランスフォーメーションのリレーショナル接続情報を更新します。
リレーショナル接続を接続オブジェクトの名前に設定します。新しい設定のセッションを保存します。

手順 5. リアルタイム Web サービスのサンプルプログラムのコンパイル

サンプル Java プログラムをコンパイルするには、サンプルプログラムのディレクトリに移動し、コンパイルバッチまたはスクリプトファイルを実行します。コンパイルするサンプルプログラムの名前と一致するバッチまたはスクリプトファイルを実行します。

たとえば、/axis/CustomerLookup_SINGLEROW ディレクトリの Sample.java プログラムをコンパイルするには、そのディレクトリに移動して、CompileSample.bat (Windows) または CompileSample.sh (UNIX) を実行します。

コンパイルプロセスによって、同じディレクトリにサンプルプログラムの.class ファイルが作成されます。

手順 6. リアルタイム Web サービスのサンプルプログラムの実行

サンプルプログラムを実行するマシンには、Java バージョン 1.5.0_11-b03 がインストールされている必要があります。Web Services Hub は、サンプルプログラムの実行時に実行されている必要があります。

サンプル Java プログラムを実行するには、サンプルプログラムのディレクトリに移動し、実行するサンプルプログラムのバッチまたはスクリプトファイルを実行します。たとえば、/axis/ CustomerLookup_MULTIPLEROW ディレクトリの Sample.java プログラムをコンパイルするには、そのディレクトリに移動して、RunSample.bat (Windows) または RunSample.sh (UNIX) を実行します。必要なパラメータを付けて、サンプルプログラムを実行します。

リアルタイム Web サービスの例

ここでは、リアルタイム Web サービスのサンプルプログラムについて説明します。各ディレクトリには、リアルタイム Web サービスを使用する異なる方法を実演するサンプルプログラムが含まれています。

複数行ルックアップ

/CustomerLookup_MULTIPLEROW ディレクトリのサンプルプログラムは、クライアントアプリケーションがルックアップを実行して複数行データの応答を処理する Web サービスワークフローを実行する方法を示します。

Sample.java

このサンプルプログラムは、データベース内で顧客 ID をルックアップして顧客情報を印刷する PowerCenter Web サービスワークフローを呼び出します。ワークフローは、SQL トランスフォーメーションを使用して、データベースから複数行を取得します。

ディレクトリ: /CustomerLookup_MULTIPLEROW

Java サンプルをコンパイルするファイル: CompileSample.bat または CompileSample.sh

Java サンプルを実行するファイル: RunSample.bat または RunSample.sh

以下の表は、Sample アプリケーションの実行に使用するパラメータについて説明します。

パラメータ	説明
顧客 ID	ルックアップする顧客の ID。顧客 ID は、整数として渡します。
エンドポイント URL	Web サービスを探す URL エンドポイント URL は、文字列として渡します。 リアルタイム Web サービスのエンドポイント URL は、Web サービス WSDL の service 要素の soap:address 位置要素に見つかります。サンプル Web サービスのデフォルトのエンドポイント URL は <code>http://<WSHHostName>:<WSHPort>/wsh/services/ts/CustomerLookup_MULTIPLEROW</code> です。 Web Services Hub を HTTPS で実行している場合、エンドポイント URL は HTTPS で始まります。

1 行ルックアップ

/CustomerLookup_SINGLEROW ディレクトリのサンプルプログラムは、クライアントアプリケーションがルックアップを実行して 1 行データの応答を処理する Web サービスワークフローを実行する方法を示します。

Sample.java

このサンプルプログラムは、データベース内で顧客 ID をルックアップして顧客情報を印刷する PowerCenter Web サービスワークフローを呼び出します。 マッピングは、ルックアップトランスフォーメーションを使用して、データベースから 1 行取得します。

ディレクトリ: /CustomerLookup_SINGLEROW

Java サンプルをコンパイルするファイル: CompileSample.bat または CompileSample.sh

Java サンプルを実行するファイル: RunSample.bat または RunSample.sh

以下の表は、Sample アプリケーションの実行に使用するパラメータについて説明します。

パラメータ	説明
顧客 ID	ルックアップする顧客の ID。顧客 ID は、整数として渡します。
エンドポイント URL	Web サービスを探す URL エンドポイント URL は、文字列として渡します。 リアルタイム Web サービスのエンドポイント URL は、Web サービス WSDL の service 要素の soap:address 位置要素に見つかります。サンプル Web サービスのデフォルトのエンドポイント URL は <code>http://<WSHHostName>:<WSHPort>/wsh/services/ts/CustomerLookup_SINGLEROW</code> です。 Web Services Hub を HTTPS で実行している場合、エンドポイント URL は HTTPS で始まります。

Web ブラウザの設定

Web ブラウザの設定

Microsoft Internet Explorer または Google Chrome を使用して、Informatica プラットフォームで Web サービス Hub コンソール起動することができます。

Web サービス Hub コンソールを実行するには、ブラウザで次のオプションを設定します。

スクリプトと ActiveX

Microsoft Internet Explorer で次のコントロールを有効にします。

- アクティブスクリプト
- プログラム的なクリップボード アクセスの許可
- ActiveX コントロールとプラグインの実行
- スクリプトを実行しても安全だとマークされている ActiveX コントロールのスクリプトの実行

コントロールを設定するには、[ツール] > [インターネットオプション] > [セキュリティ] > [レベルのカスタマイズ] をクリックします。

信頼できるサイト

Informatica ドメインが Kerberos 認証を使ったネットワーク上で実行されている場合、ブラウザを Informatica の Web アプリケーションにアクセスできるように設定する必要があります。Microsoft Internet Explorer と Google Chrome で、Informatica Web アプリケーションの URL を信頼できる

サイトのリストに追加します。Chrome 41 以降を使用している場合は、AuthServerWhitelist ポリシーと AuthNegotiateDelegateWhitelist ポリシーも設定する必要があります。