Oracle® Communications MetaSolv Solution

Web Services Developer's Guide

Release 6.3

E69851-05

April 2018

ORACLE®

Oracle Communications MetaSolv Solution Web Services Developer's Guide, Release 6.3

E69851-05

Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

This software and related documentation are provided under a license agreement containing restrictions on use and disclosure and are protected by intellectual property laws. Except as expressly permitted in your license agreement or allowed by law, you may not use, copy, reproduce, translate, broadcast, modify, license, transmit, distribute, exhibit, perform, publish, or display any part, in any form, or by any means. Reverse engineering, disassembly, or decompilation of this software, unless required by law for interoperability, is prohibited.

The information contained herein is subject to change without notice and is not warranted to be error-free. If you find any errors, please report them to us in writing.

If this is software or related documentation that is delivered to the U.S. Government or anyone licensing it on behalf of the U.S. Government, then the following notice is applicable:

U.S. GOVERNMENT END USERS: Oracle programs, including any operating system, integrated software, any programs installed on the hardware, and/or documentation, delivered to U.S. Government end users are "commercial computer software" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, use, duplication, disclosure, modification, and adaptation of the programs, including any operating system, integrated software, any programs installed on the hardware, and/or documentation, shall be subject to license terms and license restrictions applicable to the programs. No other rights are granted to the U.S. Government.

This software or hardware is developed for general use in a variety of information management applications. It is not developed or intended for use in any inherently dangerous applications, including applications that may create a risk of personal injury. If you use this software or hardware in dangerous applications, then you shall be responsible to take all appropriate fail-safe, backup, redundancy, and other measures to ensure its safe use. Oracle Corporation and its affiliates disclaim any liability for any damages caused by use of this software or hardware in dangerous applications.

Oracle and Java are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

Intel and Intel Xeon are trademarks or registered trademarks of Intel Corporation. All SPARC trademarks are used under license and are trademarks or registered trademarks of SPARC International, Inc. AMD, Opteron, the AMD logo, and the AMD Opteron logo are trademarks or registered trademarks of Advanced Micro Devices. UNIX is a registered trademark of The Open Group.

This software or hardware and documentation may provide access to or information about content, products, and services from third parties. Oracle Corporation and its affiliates are not responsible for and expressly disclaim all warranties of any kind with respect to third-party content, products, and services unless otherwise set forth in an applicable agreement between you and Oracle. Oracle Corporation and its affiliates will not be responsible for any loss, costs, or damages incurred due to your access to or use of third-party content, products, or services, except as set forth in an applicable agreement between you and Oracle.

Contents

Preface xi

Audience xi
Related Documents xi
Documentation Accessibility xii

1 Web Services Overview

Overview 1-1
About the MSS Web Services 1-1
Customer Web Service 1-1
Event Web Service 1-1
Inventory Web Service 1-2
Order Web Service 1-2
Activation Web Service 1-2
Service Order Activation (SOA) Web Service 1-2
About MSS Web Service Standards and Specifications 1-2
About the MSS Web Services Framework 1-3
About the MSS Web Service Module 1-4
About Transaction Handling 1-5
About Exception Stacktraces 1-5
Understanding How MSS Defines Web Services 1-5
About WAR and WSDL Files 1-6
About WSDL Definitions 1-6
About WSDL Locations 1-6
About WSDL Bindings 1-7
About Namespaces 1-7
Understanding MSS WSDL Operations 1-8
About Schema Files 1-10
Getting Information about MSS WSDL Operations 1-10
Using the Message Payload Documentation 1-11

2 Migrating XML APIs to Web Services

About MSS XML APIs 2-1
Migrating XML APIs to Web Services 2-1
Migrating XML APIs Developed Using Workshop Using Java 2-1
Migrating XML APIs Using Oracle SOA Suite 2-7

Migrating XML APIs Using a Java Class 2-7
Mapping Existing XML APIs to Web Services 2-9
Functional Differences from the XML APIs 2-14
Empty Structures on Responses 2-14
Get DLR Response Field Changes 2-14
Schema File Changes 2-14
Sample Build XML File 2-15

3 Deploying, Testing, and Invoking Secure Web Services

About Deploying MSS Web Services 3-1
Undeploying MSS Web Services 3-1
Deploying MSS Web Services 3-2
About Testing MSS Web Services 3-3
Testing MSS Web Services using SOAPUI 3-3
Testing MSS Web Services with a Java Test Client 3-5
About Invoking Secure MSS Web Services 3-5
Securing Web Services 3-5
About Policy Files 3-6
Modifying Web Service Security 3-6
Accessing Security 3-7
Associating a Policy File 3-7
Disassociating a Policy File 3-7
Enabling Transactions Transport Security Mode 3-8
Troubleshooting 3-8
Connecting to the WSDL Location 3-8
Redirecting Log Messages 3-9
Invoking MSS Web Services in an SSL Environment 3-9

4 Customer Web Service Reference

About the Customer Web Service 4-1
About the Customer Web Service Packaging 4-1
About the Customer WSDL, WAR, and Schema Files 4-1
About Customer Schema Files 4-1
Common Schemas 4-1
Entity Schemas 4-2
Data Schemas 4-2
API Schemas 4-2
deleteCustomerRequest Operation 4-2
deleteCustomerRequest 4-3
deleteCustomerRequestResponse 4-3
getCustomerAccountByKey Operation 4-4
getCustomerAccountByKey 4-4
getCustomerAccountByKeyResponse 4-5
importCustomerAccount Operation 4-5
importCustomerAccount 4-5
importCustomerAccountResponse 4-18

5 Event Web Service Reference

About the Event Web Service 5-1
About the Event Web Service Packaging 5-1
About the Event WSDL, WAR, and Schema Files 5-1
About Event Schema Files 5-1
Common Schemas 5-1
Entity Schemas 5-2
Data Schemas 5-2
API Schemas 5-2
getIntegrationEventData Operation 5-2
getIntegrationEventData 5-2
getIntegrationEventDataResponse 5-3
updateInboundEventStatus Operation 5-3
updateInboundEventStatus 5-4
updateInboundEventStatusResponse 5-4
updateIntegrationEventStatus Operation 5-5
updateIntegrationEventStatus 5-5
updateIntegrationEventStatusResponse 5-6
updateOutboundEventStatus Operation 5-7
updateOutboundEventStatus 5-7
updateOutboundEventStatusResponse 5-8

6 Inventory Web Service Reference

About the Inventory Web Service 6-1
About the Inventory Web Service Packaging 6-1
About the Inventory WSDL, WAR, and Schema Files 6-1
About Inventory Schema Files 6-2
Common Schemas 6-2
Entity Schemas 6-2
Data Schemas 6-2
API Schemas 6-2
auditTrailRecording Operation 6-2
auditTrailRecording 6-3
auditTrailRecordingResponse 6-4
createEntityByValueRequest Operation 6-4
createEntityByValueRequest 6-4
createEntityByValueResponse 6-6
createLocationRequest Operation 6-7
createLocationRequest 6-8
createLocationRequestResponse 6-12
createNewInventoryItemRequest Operation 6-13
createNewInventoryItemRequest 6-13
createNewInventoryItemResponse 6-14
getAvailablePhysicalPortsRequest Operation 6-15
getAvailablePhysicalPortsRequest 6-15
getAvailablePhysicalPortsRequestResponse 6-15

getDlrByKeyRequest Operation 6-16
getDlrByKeyRequest........ 6-16
getDlrByKeyRequestResponse........ 6-17
getDlrByOrderKey Operation 6-18
getDlrByOrderKey 6-19
getDlrByOrderKeyResponse 6-19
getEntityByKeyRequest Operation 6-20
getEntityByKeyRequest........ 6-20
getEntityByKeyRequestResponse........ 6-21
getIpAddressesRequest Operation 6-22
getIpAddressesRequest........ 6-22
getIpAddressesRequestResponse 6-23
getLocationRequest Operation 6-23
getLocationRequest........ 6-23
getLocationRequestResponse 6-24
getNetworkAreasByGeoAreaRequest Operation 6-25
getNetworkAreasByGeoAreaRequest 6-25
getNetworkAreasByGeoAreaRequestResponse........ 6-25
getNetworkComponentsRequest Operation 6-26
getNetworkComponentsRequest 6-26
getNetworkComponentsRequestResponse........ 6-27
inventoryAssociationRequest Operation 6-27
inventoryAssociationRequest........ 6-27
inventoryAssociationRequestResponse........ 6-28
processTNRecallRequestDocument Operation 6-28
processTNRecallRequest........ 6-28
processTNRecallResponse 6-29
queryEndUserLocation Operation 6-29
queryEndUserLocation 6-29
queryEndUserLocationResponse........ 6-30
queryInventoryManagementRequest Operation........ 6-31
queryInventoryManagementRequest 6-31
queryInventoryManagementRequestResponse........ 6-33
queryNetworkLocation Operation........ 6-35
queryNetworkLocation 6-35
queryNetworkLocationResponse 6-36
processtnValidationRequest Operation 6-36
processTNValidationRequest........ 6-36
processTNValidationResponse 6-37
updateEntityByValueRequest Operation........ 6-37
updateEntityByValueRequest 6-37
updateEntityByValueRequestResponse 6-39
updateLocationRequest Operation........ 6-40
updateLocationRequest........ 6-41
updateLocationRequestResponse........ 6-41
updateTNRequest Operation........ 6-42
updateTNRequest 6-42

updateTNRequestResponse 6-44

7 Order Web Service Reference

About the Order Web Service 7-1
About the Order Web Service Packaging 7-1
About the Order WSDL, WAR, and Schema Files 7-2
About Order Schema Files 7-2
Common Schemas 7-2
Entity Schemas 7-2
Data Schemas 7-2
API Schemas 7-2
addTaskRequest Operation 7-3
addTaskRequest 7-3
addTaskRequestResponse 7-3
addTaskJeopardyRequest Operation 7-4
addTaskJeopardyRequest 7-4
addTaskJeopardyResponse 7-5
assignProvPlanRequest Operation 7-5
assignProvPlanRequest 7-5
assignProvPlanRequestResponse 7-6
billingTelephoneNumberRequest Operation 7-7
billingTelephoneNumberRequest 7-7
billingTelephoneNumberRequestResponse 7-7
completeTaskRequest Operation 7-8
completeTaskRequest 7-8
completeTaskRequestResponse 7-9
createAttachment Operation 7-10
createAttachment 7-10
createAttachmentResponse 7-10
createISROrderRequest Operation 7-11
createISROrderRequest 7-11
createISROrderResponse 7-13
createOrderRelationship Operation 7-13
createOrderRelationship 7-14
createOrderRelationshipResponse 7-14
createOrderRequest Operation 7-14
createOrderRequest 7-15
createOrderRequestResponse 7-41
createPSROrderRequest Operation 7-41
createPSROrderRequest 7-41
createPSROrderRequestResponse 7-42
getCnamData Operation 7-43
getCnamData 7-43
getCnamDataResponse 7-43
getE911Data Operation 7-43
getE911Data 7-44
getE911DataResponse 7-44

getLidbData Operation 7-44
getLidbData 7-44
getLidbDataResponse 7-45
getOrderByKey Operation 7-45
getOrderByKey 7-45
getOrderByKeyResponse 7-46
getPSROrderByTN Operation 7-46
getPSROrderByTN 7-46
getPSROrderByTNResponse 7-47
processSuppOrderRequest Operation 7-47
processSuppOrderRequest 7-48
processSuppOrderRequestResponse 7-48
queryOrderManagementRequest Operation 7-49
queryOrderManagementRequest 7-49
queryOrderManagementRequestResponse 7-53
reopenTaskRequest Operation 7-57
reopenTaskRequest 7-57
reopenTaskRequestResponse 7-57
startOrderRequest Operation 7-58
startOrderRequest 7-58
startOrderRequestResponse 7-58
TaskJeopardyRequest Operation 7-59
TaskJeopardyRequest 7-59
TaskJeopardyRequestResponse 7-60
TaskRequest Operation 7-60
TaskRequest 7-61
TaskRequestResponse 7-61
transferTaskRequest Operation 7-61
transferTaskRequest 7-62
transferTaskRequestResponse 7-62
updateCnamData Operation 7-63
updateCnamData 7-63
updateCnamDataResponse 7-64
updateE911DataRequest Operation 7-64
updateE911DataRequest 7-64
updateE911DataRequestResponse 7-66
updateEstimationCompletedDateRequest Operation 7-66
updateEstimationCompletedDateRequest 7-66
updateEstimationCompletedDateRequestResponse 7-67
updateGWEventRequest Operation 7-67
updateGWEventRequest 7-67
updateGWEventRequestResponse 7-68
updateLidbData Operation 7-69
updateLidbData 7-69
updateLidbDataResponse 7-70
updateOrderRequest Operation 7-70
updateOrderRequest 7-71

updateOrderRequestResponse 7-72
updatePSROrderRequest Operation 7-74
updatePSROrderRequest 7-74
updatePSROrderRequestResponse 7-75

8 Activation Web Service Reference

About the Activation Web Service 8-1
About the Activation Web Service Packaging 8-1
About the Activation WSDL, WAR, and Schema Files 8-1
About Activation Schema Files 8-1
Common Schemas 8-1
Entity Schemas 8-2
Data Schemas 8-2
API Schemas 8-2
getActivationData Operation 8-2
getActivationData 8-2
getActivationDataResponse 8-3

9 SOA Web Service Reference

About the SOA Web Service 9-1
About the SOA Web Service Packaging 9-1
About the SOA WSDL, WAR, and Schema Files 9-1
About SOA Schema Files 9-1
Common Schemas 9-2
Entity Schemas 9-2
Data Schemas 9-2
API Schemas 9-2
createSoaMessageRequest Operation 9-2
createSoaMessageRequest 9-3
createSoaMessageRequestResponse 9-5
getSoaDefaultsRequest Operation 9-5
getSoaDefaultsRequest 9-5
getSoaDefaultsRequestResponse 9-6
getSoaInformationRequest Operation 9-6
getSoaInformationRequest 9-7
getSoaInformationRequestResponse 9-7
getSoaMessageToSendRequest Operation 9-8
getSoaMessageToSendRequest 9-8
getSoaMessageToSendRequestResponse 9-8
getSoaTnsForOrderRequest Operation 9-9
getSoaTnsForOrderRequest 9-9
getSoaTnsForOrderRequestResponse 9-9
setTnSoaCompleteRequest Operation 9-10
setTnSoaCompleteRequest 9-10
setTnSoaCompleteRequestResponse 9-10

10 Engineering Work Order Web Service Reference

About the Engineering Work Order Web Service 10-1
About the Engineering Work Order Web Service Packaging 10-1
About the Engineering Work Order WSDL, WAR, and Schema Files 10-1
About Engineering Work Order Schema Files 10-2
API Schemas 10-2
createWorkOrder Operation 10-2
createWorkOrder........ 10-2
createWorkOrderResponse........ 10-3
associateConnectionToWorkOrder Operation 10-3
associateConnectionToWorkOrder 10-3
associateConnectionToWorkOrderResponse 10-4
associateEquipmentToWorkOrder Operation 10-4
associateEquipmentToWorkOrder 10-4
associateEquipmentToWorkOrderResponse 10-5
createWorkOrderNote Operation 10-5
createWorkOrderNote 10-5
createWorkOrderNoteResponse 10-5
updateWorkOrderNote Operation 10-6
updateWorkOrderNote 10-6
updateWorkOrderNoteResponse 10-6
processDDChangeSupplement Operation 10-6
processDDChangeSupplement 10-6
processDDChangeSupplementResponse 10-7
updateWorkOrder Operation 10-7
updateWorkOrder........ 10-7
updateWorkOrderResponse........ 10-8
getWorkOrder Operation 10-8
getWorkOrder........ 10-8
getWorkOrderResponse........ 10-9

Preface

This guide describes the Oracle Communications MetaSolv Solution Web Services. The guide includes information about the MetaSolv Solution Web Service framework that supports web services, the various web services that are available, and how to migrate existing XML API interfaces to web service operations.

This guide includes examples based on particular situations. These examples may not be applicable in every situation.

For additional information about required third-party software, such as the Oracle WebLogic server or the database, consult the relevant documentation.

Audience

This guide is intended for developers who have a working knowledge of web services in general, and who understand XML and Java development, including standard Java practices and J2EE principles. This document is for developers who are integrating MetaSolv Solution with Oracle Products, or with other external systems.

Related Documents

For more information, see the following documents in Oracle Communications MetaSolv Solution 6.3 documentation set:

- *MSS Planning Guide:* Describes information you need to consider in planning your MSS environment before installation.
- *MSS Installation Guide*: Describes system requirements and installation procedures for installing MSS.
- *MSS System Administrator's Guide:* Describes postinstallation tasks and administrative tasks such as maintaining user security.
- *MSS Security Guide:* Provides guidelines and recommendations for setting up MSS in a secure configuration.
- *MSS Database Change Reference:* Provides information on the database changes in MSS releases.
- *MSS Network Grooming User's Guide:* Provides information about the MSS Network Grooming tool.
- *MSS Address Correction Utility User's Guide:* Provides information about the MSS Address Correction utility.
- *MSS Technology Module Guide:* Describes each of the MSS technology modules.

- *MSS Data Selection Tool How-to Guide:* Provides an overview of the Data Selection Tool, and procedures on how it used to migrate the product catalog, equipment specifications, and provisioning plans from one release of your environment to another.
- *MSS CORBA API Developer's Reference:* Describes how MSS APIs work, high-level information about each API, and instructions for using the APIs to perform specific tasks.
- *MSS Custom Extensions Developer's Reference:* Describes how to extend the MSS business logic with custom business logic through the use of custom extensions.
- *MSS EJB API Developer's Reference*: Provides an overview of the MetaSolv Solution EJB APIs and instructions for using the APIs to perform tasks.

For step-by-step instructions for tasks you perform in MetaSolv Solution, log in to the application to see the online Help.

Documentation Accessibility

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at `http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc`.

Access to Oracle Support

Oracle customers that have purchased support have access to electronic support through My Oracle Support. For information, visit `http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info` or visit `http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs` if you are hearing impaired.

1 Web Services Overview

This chapter provides introductory information about the Oracle Communications MetaSolv Solution (MSS) Web Services.

Overview

Web services support interoperable system-to-system interaction over a network. They are APIs that can be accessed over a network and run on a remote system hosting the requested services. Web services are described by the Web Service Definition Language (WSDL).

This document describes how to integrate MetaSolv Solution with other Oracle products or with external applications using web services. In this book, you can find migration information, deployment information, and a reference chapter for each web service.

About the MSS Web Services

The MSS Web Services include a group of web services that address customer, event, inventory, order, activation, and service order activation functionality. MSS Web Services replace the existing MSS XML API functionality that is deprecated. See Chapter 2, "Migrating XML APIs to Web Services" for more information about migrating XML API's to the web services.

Customer Web Service

You can use the Customer Web Service operations to retrieve, import, and delete customer account information. The customer account contains customer information such as address, service items, along with customer service sales information. The customer account information also helps you keep track of sales performance by individual or product type. Refer to Chapter 4, "Customer Web Service Reference" for more information about the Customer Web Service set of operations.

Event Web Service

You can use Event Web Service operations to retrieve and update gateway event details. Operations are separated for inbound and outbound gateway events. You can also use Event Web Service operations to update external system information, such as an external name and external key values. Refer to Chapter 5, "Event Web Service Reference" for more information about the Event Web Service set of operations.

Inventory Web Service

You can use Inventory Web Service operations to retrieve, create, and update the Inventory details. For example, you can use the Inventory Web Service for the following type of operations:

- Get DLR information
- Create, update, and retrieve network locations
- Create, update and retrieve end user locations
- New inventory creation
- Query inventory details

Refer to Chapter 6, "Inventory Web Service Reference" for more information about the Inventory Web Service set of operations.

Order Web Service

You can use the Order Web Service operations to perform operations on orders, provisioning plans, and tasks. Order Web Service contains the objects necessary to generate information about all types of service requests, such as PSR and ISR. It also contains related provisioning objects that support work management. Refer to Chapter 7, "Order Web Service Reference" for more information about the Order Web Service set of operations.

Activation Web Service

You can use the Activation Web Service operations to get the activation details. The Activation Web Service is called to get the activation details which are used for on-field activation. Refer to Chapter 8, "Activation Web Service Reference" for more information about the Activation Web Service set of operations.

Service Order Activation (SOA) Web Service

You can use the SOA Web Service to enable an external system to activate services for previously placed orders in MetaSolv Solution. You can use the SOA Web Service operations for the following operations:

- Create a service order activation message
- Get service order activation information and defaults
- Get service order activation telephone numbers for an order
- Set a telephone number for service activation order completion

Refer to Chapter 9, "SOA Web Service Reference" for more information about the SOA Web Service set of operations.

About MSS Web Service Standards and Specifications

Table 1–1 lists the standards and specifications that apply to the MSS Web Services.

Table 1–1 MSS Web Service Standards and Specifications

Standard and Specification	Version Release	Description	Compliance
JAXB	2.2	Java Architecture for XML Binding	Compliant.

Table 1–1 (Cont.) MSS Web Service Standards and Specifications

Standard and Specification	Version Release	Description	Compliance
JAX-WS	2.2	Java API for XML-based Web Services	Compliant.
SOAP	1.2	Simple Object Access Protocol (Also referred to as Service Orientated Architecture Protocol.)	Compliant. Uses XML/SOAP/HTTP and XML/SOAP/JMS.
Transport Protocols	HTTP 1.0, HTTPS 1.0 (HTTP 1.1), JMS 1.1	NA	NA
WSDL	1.1	Web Service Definition Language	Compliant.
WS-Security	1.1	Web Service Security	Compliant.
WS-Policy	1.2	Web Service Policy Framework	Compliant.
XML	1.1	NA	Compliant. Uses XML/SOAP/HTTP and XML/SOAP/JMS.

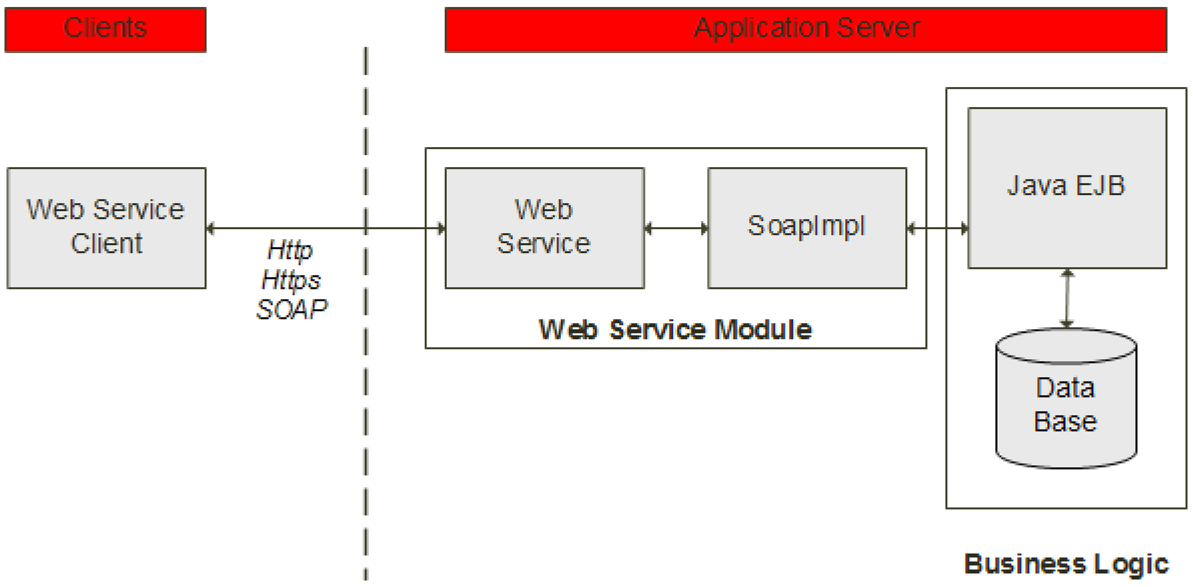
For more information, refer to the information in the *Features and Standards Support by WebLogic Web Services* chapter in the Oracle Fusion Middleware documentation *Understanding WebLogic Web Services for Oracle WebLogic Server* at this website:

`http://docs.oracle.com/middleware/1221/wls/WSOVR/toc.htm`

About the MSS Web Services Framework

Figure 1–1 shows the path traveled by a call originating from the web service client.

Figure 1–1 MSS Web Services Framework



The path of the web service includes:

- Web service client

  This client represents the web service user. Web service operations are called by sending SOAP messages over HTTP or HTTPS.

- Web service module

  This module represents all the sub-modules required for implementing a web service, for instance, the web service framework, the WSDL interfaces, and the WSDL implementations. The web service module is deployed as an EAR file.

  See "About the MSS Web Service Module" for more information.

- Business logic

  The business logic includes all the MSS sub-modules required for implementing business functionality.

About the MSS Web Service Module

The web service module is installed as part of the MSS installation. All web services are included in the **MSS_WebService.ear** file. When the MSS installer deploys the **EAR** file, the following modules of the MSS Web Services are automatically deployed and ready to use:

- Customer Web Service
- Event Web Service
- Inventory Web Service
- Order Web Service
- Activation Web Service
- Service Order Activation (SOA) Web Service

About Transaction Handling

For transactions, the MSS Web Services use Web Services Atomic Transaction (WS-AtomicTransaction). WS-AtomicTransaction is a protocol for managing atomic transactions between distributed applications, transaction managers and resource managers. An atomic transaction is a single, irreducible component of a classic transaction, such as a create or update.

Options for Transaction Management

There are two options for transaction management. You can control the transaction with the client side code or have the web service create and handle the transaction. If the client side code controls the transaction, you pass in the user transaction details to the web service with the UserTransaction interface. If the web service receives a null UserTransaction, then MSS creates and handles its own transaction. When MSS creates its own transaction it performs the following:

- Commit on a success
- Rollback on a failure

About the UserTransaction Interface

The UserTransaction interface defines the methods that allow an application to explicitly manage transaction boundaries. The UserTransaction.begin() method starts a global transaction and associates the transaction with the calling thread. The transaction-to-thread association is managed transparently by the transaction manager.

The UserTransaction.begin() method also can throw a NotSupportedException when the calling thread is already associated with a transaction. The transaction manager implementation does not support nested transactions.

See the following website for more information on the UserTransaction interface:

`http://docs.oracle.com/middleware/1213/wls/WLAPI/weblogic/transaction/UserTransaction.html`

About Exception Stacktraces

Exception stacktraces are available in the WebLogic server logs. The error messages are thrown by the operation when an exception occurs. This information is available in the *serverName*.**mss.log** file where *serverName* is the host name of the MSS installation. This file is located in the MSS **log** directory.

MSS throws exceptions of type WSException_Exception from the web services. You must handle these exceptions from the client side code.

Understanding How MSS Defines Web Services

Web services are defined by WAR, WSDL, and schema files. For each web service, there is a WAR file. Each WAR file includes a single WSDL file and numerous schema files. The WSDL files contain the actual web service definitions. The schema files includes definitions of specific elements, complex types, and simple types.

The following sections provide details information about these files.

About WAR and WSDL Files

The WAR files are in the root directory of the **MSS_WebService.ear** file. Within each WAR file, the WSDL file is located in the **WEB-INF/wsdls** directory. The schemas are also available in the **mss_webservice_schemas.jar** file. You find this JAR file as a deliverable with the MSS installer. Table 1–2 provides the WSDL and WAR file names for each web service:

Table 1–2 WAR and WSDL Files for each Web Service

Web Service	WAR File Name	WSDL File Name
Customer Web Service	customer.war	CustomerAPI.wsdl
Event Web Service	events.war	EventsAPI.wsdl
Inventory Web Service	inventory.war	InventoryAPI.wsdl
Order Web Service	order.war	OrderAPI.wsdl
Activation Web Service	activation.war	ServiceActivationAPI.wsdl
Service Order Activation (SOA) Web Service	soa.war	SOAAPI.wsdl

About WSDL Definitions

The WSDL definitions at the beginning of the WSDL file define the web service as being deployed using the SOAP 1.1 protocol over HTTP.

See the following websites for more information about WSDL 1.1 and SOAP 1.1:

- `https://www.w3.org/TR/wsdl`
- `https://www.w3.org/TR/2000/NOTE-SOAP-20000508/`

See "About MSS Web Service Standards and Specifications" for more information about the standards that apply to the MSS Web Services.

Example 1–1 shows the definitions section of the **OrderAPI.wsdl** file.

Example 1–1 WSDL Definitions Example

```
<wsdl:definitions xmlns:http="http://schemas.xmlsoap.org/wsdl/http/"
                  xmlns:mime="http://schemas.xmlsoap.org/wsdl/mime/"
                  xmlns:s="http://www.w3.org/2001/XMLSchema"
                  xmlns:s0="http://www.openuri.org/"
                  xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
                  xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/"
                  xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
                  targetNamespace="http://www.openuri.org/">
```

About WSDL Locations

For each MSS Web Service, you can access the WSDL with a URL location.

Table 1–3 provides the URL locations where:

- *mssHost* is the name of the host on which MSS is installed
- *port* is the port number of the machine on which MSS is installed

Table 1–3 WSDL URL Location for each Web Service

Web Service	URL Location
Customer Web Service	`http://mssHost:port/MssWS/customer/CustomerAccount?WSDL`
Event Web Service	`http://mssHost:port/MssWS/events/IntegrationEvent?WSDL`
Inventory Web Service	`http://mssHost:port/MssWS/inventory/Inventory?WSDL`
Order Web Service	`http://mssHost:port/MssWS/order/Order?WSDL`
Activation Web Service	`http://mssHost:port/MssWS/activation/ServiceActivationData?WSDL`
Service Order Activation (SOA) Web Service	`http://mssHost:port/MssWS/soa/Soa?WSDL`

About WSDL Bindings

In the WSDL file, the bindings define the protocol details and message formats for the web service operations. Example 1–2 shows a portion of the WSDL file with the binding definition of the addTaskJeopardyRequest operation.

Example 1–2 OrderSoap Binding Definition Example with addTaskJeopardyRequest

```
<wsdl:binding name="OrderSoap" type="s0:OrderSoap">
    <soap:binding transport="http://schemas.xmlsoap.org/soap/http"
                  style="document"/>
    <wsdl:operation name="addTaskJeopardyRequest">
      <soap:operation soapAction="http://www.openuri.org/addTaskJeopardyRequest"
                      style="document"/>
      <wsdl:input>
        <soap:body use="literal"/>
      </wsdl:input>
      <wsdl:output>
        <soap:body use="literal"/>
      </wsdl:output>
      <wsdl:fault name="WSException">
        <soap:fault name="WSException" use="literal"/>
      </wsdl:fault>
    </wsdl:operation>
</wsdl:binding>
```

About Namespaces

Each WSDL file defines a namespace to avoid naming conflicts. The namespace definition appears after the WSDL definitions section.

You use the namespace to determine the schema file location of the schema reference. Example 1–3 shows how a namespace defined in the WSDL correlates to the supporting schema files.

In this example, the Order Web Service WSDL file named **OrderAPI.wsdl** defines and references the **ord** namespace. This excerpt shows momAddTaskJeopardyRequest has a namespace designation and how the namespace is defined:

Example 1–3 OrderAPI.wsdl Namespace Example

.

```
.
.
<s:schema
xmlns:ord="http://xmlns.oracle.com/communications/mss/OrderManagementAPI">

<s:import
namespace="http://xmlns.oracle.com/communications/mss/OrderManagementAPI"
      schemaLocation="../schemas/OrderManagementAPI.xsd"/>
.
.
.
<s:element name="addTaskJeopardyRequest">
  <s:complexType>
    <s:sequence>
      <s:element ref="ord:momAddTaskJeopardyRequest"/>
    </s:sequence>
  </s:complexType>
</s:element>
.
.
.
```

The addTaskJeopardyRequest element declaration tells you that momAddTaskJeopardyRequest is defined in the schema file that supports the specified namespace. A search for the specified namespace reveals that the **ord** namespace represents the **OrderManagementAPI** schema file.

After you determine that the **OrderManagementAPI.xsd** schema file defines the XML structure that the WSDL file references, you can navigate through the schema files to determine child XML structures.

Refer to the following website for more information on namespaces:

https://www.w3.org/TR/REC-xml-names/

Understanding MSS WSDL Operations

The WSDL file defines the web service and its operations. It also defines the input, output and fault messages of each operation. MSS WSDL operations follow the request-response or round-trip pattern. When the client sends a request message to the web service, the operation either sends a response message back or the operation sends a fault message back to the client for an error.

In the WSDL file, each service is bound to a port. For example, in the **OrderAPI.wsdl** file, the Order Web Service is bound to the OrderSoap port. Example 1–4 shows the definition of the Order service and the port name.

Example 1–4 Order Service and Port Definition

```
<wsdl:service name="Order">
  <wsdl:port name="OrderSoap" binding="s0:OrderSoap">
    <soap:address location="http://localhost:7001/MssWS/order"/>
  </wsdl:port>
</wsdl:service>
```

The Order service and port declaration reference the portType through the OrderSoap definition. The portType element combines multiple message elements to define the list of web service operations and their parameters.

Example 1–5 demonstrates the following aspects of the portType definition in the Order Web Service:

- The portType element definition of the name as OrderSoap
- The definition of addTaskJeopardyRequest as a round-trip operation
- The declaration of the input, output and fault message types for addTaskJeopardyRequest

Example 1–5 The WSDL portType and addTaskJeopardyRequest Definition

```
<wsdl:portType name="OrderSoap">
  <wsdl:operation name="addTaskJeopardyRequest">
    <wsdl:input message="s0:addTaskJeopardyRequestSoapIn"/>
    <wsdl:output message="s0:addTaskJeopardyRequestSoapOut"/>
    <wsdl:fault message="s0:WSException" name="WSException"/>
  </wsdl:operation>
.
.
.
</wsdl:portType>
```

Message definitions determine the input and output parameters for the operation. Example 1–6 shows a WSDL excerpt with the message definitions and their respective elements.

Example 1–6 Web Service Message Definitions for addTaskJeopardyRequest Operation

```
  <wsdl:message name="addTaskJeopardyRequestSoapIn">
    <wsdl:part name="parameters" element="s0:addTaskJeopardyRequest"/>
  </wsdl:message>
  <wsdl:message name="addTaskJeopardyRequestSoapOut">
    <wsdl:part name="parameters" element="s0:addTaskJeopardyRequestResponse"/>
  </wsdl:message>
  <wsdl:message name="WSException">
   <wsdl:part name="fault" element="s0:WSException"/>
  </wsdl:message>
```

Table 1–4 summarizes the definitions in Example 1–5 and Example 1–6 to list the message types, messages and elements for the addTaskJeopardyRequest operation.

Table 1–4 addTaskJeopardyRequest Messages and Elements

Message Type	Message	Element
input	addTaskJeopardyRequestSoapIn	addTaskJeopardyRequest
output	addTaskJeopardyRequestSoapOut	addTaskJeopardyRequestResponse
fault	WSException	WSException

The details of the element for each message are defined in declarations in the WSDL file. For example, the input request, output response, and exception/fault elements listed in Table 1–4 are defined by the declarations shown in Example 1–7.

Example 1–7 Element Definitions for Input, Output and Fault for addTaskJeopardyRequest

```
      <s:element name="addTaskJeopardyRequest">
        <s:complexType>
          <s:sequence>
            <s:element ref="ord:momAddTaskJeopardyRequest"/>
          </s:sequence>
        </s:complexType>
```

```
</s:element>
<s:element name="addTaskJeopardyRequestResponse">
  <s:complexType>
    <s:sequence>
      <s:element ref="ord:addTaskJeopardyResponse"/>
    </s:sequence>
  </s:complexType>
</s:element>
<s:element name="WSException">
  <s:complexType>
    <s:sequence>
      <s:element name="faultCode" type="s:string" minOccurs="0"/>
      <s:element name="faultString" type="s:string" minOccurs="0"/>
    </s:sequence>
  </s:complexType>
</s:element>
```

The elements include references to the locations in schema files where they are fully defined. The references include namespaces that you use to locate the schema file for each element. For example, the namespace **ord** was defined earlier to point to the **OrderManagementAPI.xsd** file.

The schema files can themselves include references to other schema files.

Refer to "About Namespaces" for more information on namespaces, and "About Schema Files" for more information on schema files.

About Schema Files

Numerous schema files support the MSS Web Services. Within the WAR files, you find the schema files located in the **WEB-INF/wsdls** directory. The schemas are also available in the **mss_webservice_schemas.jar** file. You find this JAR file as a deliverable with the MSS installer.

These schemas are categorized as common schemas, entity schemas, data schemas, and API schemas. The entity, data, and API schema files are different for each web service.

Identical common schema files are included in the WAR files for all web services. The following are the common schema files:

- Common.xsd
- Core.xsd
- Customer.xsd
- DataTypes.xsd
- Location.xsd
- Resource.xsd
- Service.xsd

Refer to the individual web service chapters for more information about schema files.

Getting Information about MSS WSDL Operations

This guide contains high-level documentation about each web service and its operations.

There is a reference chapter dedicated to each web service. Each web service chapter includes an overview of the web service and a section for each operation. Each operation section contains the following:

- An overview of the operation's purpose
- High-level schema information for the input or request message payload
- High-level schema information for the output or response message payload
- When applicable, a description of multiple different types of requests
- When the operation is complex, an XML sample of a request

You use the XSD schema files as reference for field-level information such as descriptions and field formatting guidance. The field-level documentation is provided by annotations in the XSD files.

Using the Message Payload Documentation

The reference chapters for the web services include sections for the request and response documentation. For each operation's request and response, one or more tables describe the primary XSD elements and types. An operation can have multiple tables describe complex payloads.

These tables enable you to navigate the request or response in the following ways:

- They provide an overall view of the payload.
- They aid in locating the request or response definitions and their XSD file location.
- They show you XSD definitions, for instance, whether they are elements or a complex types.

The table describing a request contains the following information:

- The input request element defined in the first row.
- A row for each element, describing how the element or complexType is defined in the XSD, its type, and the file name where it is defined.
- A row for each type listed in the table, placed following the row in which the type is first mentioned.

Table 1–5 is an example of the request elements and complex types for the addTaskJeopardyRequest operation.

Table 1–5 Elements and Types for the Request Example

Name	Defined As	Type Description	File Name
addTaskJeopardyRequest	element	momAddTaskJeopardyRequest	OrderAPI.wsdl
momAddTaskJeopardyRequest	element	addTaskJeopardyRequestValue	OrderManagementAPI.xsd
addTaskJeopardyRequestValue	element	AddTaskJeopardyRequestValueType	OrderManagementAPI.xsd
AddTaskJeopardyRequestValueType	complexType	complexType with a list of required and optional fields	OrderManagementData.xsd

In this example, addTaskJeopardyRequest is the name value in first row. This addTaskJeopardyRequest element is defined as the input message element for the operation. It is defined as an element and is located in the **OrderAPI.wsdl** file. (See Example 1–6, "Web Service Message Definitions for addTaskJeopardyRequest Operation" for an excerpt of this input message definition.)

The type for the addTaskJeopardyRequest element is momAddTaskJeopardyRequest. This type is listed in the second row and similarly described.

This pattern continues until one of the following is true:

- The item referenced is a core or common schema item.
- The item referenced is a series of fields where the series contains simple types or complex types that are located in the same schema file.

To get specific information about fields, you consult the XSD schema file itself. For the AddTaskJeopardyRequestValueType, you find the field information in the **OrderManagementData.xsd** file. Annotations in the file describe the field details of AddTaskJeopardyRequestValueType.

Similarly, response messages are defined in tables that have the same format and layout as the request information. Optionally, the response section includes a list of error conditions that can be thrown by the operation.

Table 1–6 shows an example of the possible error messages for the addTaskJeopardyRequest operation.

Table 1–6 Error Messages for addTaskJeopardyRequest

Error Message	Cause	Resolution
The document number is not found.	The provided document number does not exist in the database.	Verify and pass in a valid document number.
The jeopardy reason code is not valid.	An invalid jeopardy reason code was provided in the request.	Verify and provide a valid jeopardy reason code value in the request.

2 Migrating XML APIs to Web Services

This chapter provides information about migrating existing Oracle Communications MetaSolv Solution (MSS) XML APIs to web services.

About MSS XML APIs

Previous releases of MSS including 6.2.1 supported XML APIs, which ran on WebLogic Server 10.3.1. The requirements for providing XML APIs are the following from WebLogic:

- WebLogic Integration - WLI
- WebLogic Workshop

The current release of MSS runs on a WebLogic Server release that does not support these requirements for providing XML APIs. Therefore, MSS provides a new set of web services to replace the XML API functionality. This chapter includes instructions on migrating to the web services, and a mapping of the XML API operations to the web service operations.

Migrating XML APIs to Web Services

Migrating XML APIs to web services can be done in several ways. The following sections describe a sampling of the possible migration paths:

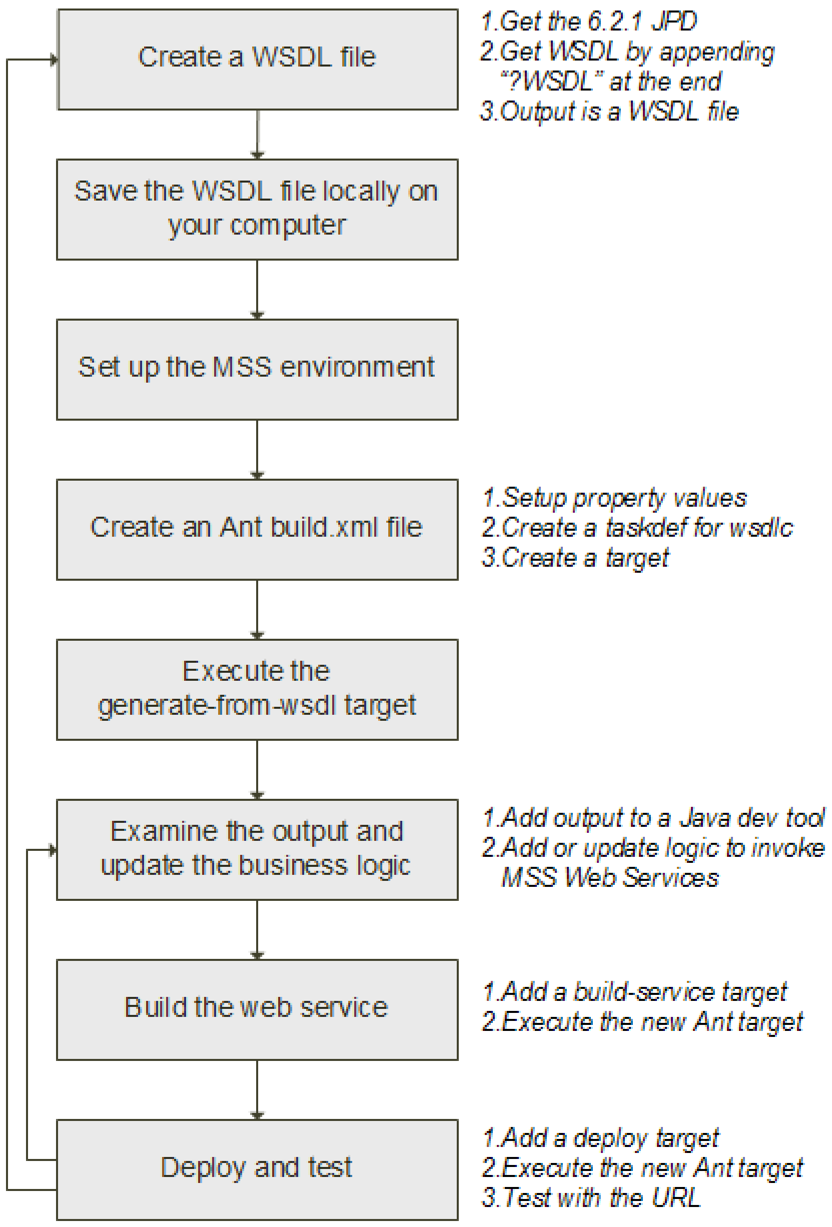
- Migrating XML APIs Developed Using Workshop Using Java
- Migrating XML APIs Using Oracle SOA Suite
- Migrating XML APIs Using a Java Class

Migrating XML APIs Developed Using Workshop Using Java

Since the XML APIs were developed using Workshop, it is not possible to migrate them using a tool. Therefore, the best way to migrate these XML APIs to web services is to write them in Java.

Figure 2–1 describes an overview of the migration steps.

Figure 2–1 Overview of Migration Steps Using Java


Create a WSDL file
1.Get the 6.2.1 JPD
2.Get WSDL by appending "?WSDL" at the end
3.Output is a WSDL file
Save the WSDL file locally on your computer
Set up the MSS environment
Create an Ant build.xml file
1.Setup property values
2.Create a taskdef for wsdlc
3.Create a target
Execute the generate-from-wsdl target
Examine the output and update the business logic
1.Add output to a Java dev tool
2.Add or update logic to invoke MSS Web Services
Build the web service
1.Add a build-service target
2.Execute the new Ant target
Deploy and test
1.Add a deploy target
2.Execute the new Ant target
3.Test with the URL

The following steps show an example of how to write Java functionality for a sample XML API:

1. Create a WSDL File. This step assumes you are using Java Process Definition (JPD) for your integration.

   a. For your XML API, get the WSDL file for the API which must be migrated.

   b. Find your JPD file (these have a .jpd or .java extension) for the XML API you are converting. You can get the WSDL file by adding "?WSDL" after the JPD file information.

   ```
   http://hostname:port/custom/testAPI.jpd?WSDL
   ```

   This will give you the complete WSDL for your JPD file.

Note: In WebLogic Integration, Process Definition for Java (JPD) defines a business process.

2. Save the file to a new name, for example: **testAPI.wsdl**.
3. Set up your WebLogic Server and MSS environment.

   Open a command window and execute the **setDomainEnv.cmd** (Windows) or **setDomainEnv.sh** (UNIX) script, located in the **bin** subdirectory of your domain directory.
4. Create a working directory, for example:

   ```
   c:\opt\customWebService
   ```

   where *customWebService* is the name of the XML API that you want to migrate.
5. Put your WSDL file into an accessible directory on your computer.

   In this example, it is assumed that your WSDL file is called **testAPI.wsdl** and place the WSDL file in the '**\***customWebService* **\wsdl_file**' directory.
6. Create a standard Ant **build.xml** file in the project directory. Set up the values of the property names for items such as the URL, server name, and password.

   ```
   <project name="GenerateWS" default="all">
   <property name="build" value="output" />
   <property name="adminurl" value="t3://192.0.2.251:7001"/> //Server IP & Port
   <property name="serverName" value="m63server" /> // Server Name
   <property name="userName" value="weblogic" /> // User Name
   <property name="password" value="password" /> // Password
   <property name="ear.name" value="Custom_WebService" /> // Ear Name
   ```

   In this example, the **build.xml** file will be located under the **c:\opt\***customWebService* directory.

   See "Sample Build XML File" for an example of a complete **build.xml** file.
7. Add a **taskdef** item to specify the full Java classname of the **wsdlc** task.

   ```
   <taskdef name="jwsc" classname="weblogic.wsee.tools.anttasks.JwscTask" />
   <taskdef name="wsdlc" classname="weblogic.wsee.tools.anttasks.WsdlcTask"/>
   ```
8. Create a **generate-from-wsdl** target and add the following to the **wsdlc** Ant task in the **build.xml** file:

   ```
   <target name="generate-from-wsdl">
        <wsdlc srcWsdl="wsdl_file/testAPI.wsdl"
               destJwsDir="jarJws"
               destImplDir="src" type="JAXWS"
               packageName="custom.mss.test"/>
   </target>
   ```

   Before generating the web service from WSDL file, the following changes need to be made on the MSS WSDL file for a successful compilation:

   - Remove one <wsdl:port> which contains the name with JpdSoap under <wsdl:services>.
   - Remove the corresponding <wsdl:binding> for JpdSoap.
   - Remove the corresponding <wsdl:port type> for JpdSoap.

- Remove the corresponding <wsdl:message> for both Input and Output messages.

9. Execute the **generate-from-wsdl** target to run the **wsdlc** Ant task by running this command line:

   ```
   ant generate-from-wsdl
   ```

   See the output directory to examine the artifacts and files generated by the **wsdlc** Ant task.

   The **wsdlc** task in the examples generates the JAR file that contains the JWS Service Endpoint Interface (SEI) and JAXB data binding artifacts into the **jarJws** directory under the current directory.

   It also generates a partial implementation file (**testAPISoapImpl.java**) of the JWS SEI into the **src\custom\mss\test\** directory (which is a combination of the output directory specified by destImplDir and the directory hierarchy specified by the package name). All generated JWS files will be packaged in the **custom.mss.test** package.

10. Update the implementation file.

    a. Run the wsimport Java command to generate the web service client files. The wsimport command creates JAX-WS portable artifacts that can be packaged in a web application archive or WAR file.

       Refer to the following website for more information on the wsimport command:

       https://docs.oracle.com/javase/8/docs/technotes/tools/unix/wsimport.html

    b. Package the generated files with the web service project or package it as a JAR file and include it as a library.

    c. Create a transaction to use for the API calls and do a lookup for the UserTransaction.

       ```
       ctx = new InitialContext(env);
       utx = (UserTransaction)
             ctx.lookup("javax.transaction.UserTransaction");
       ```

       where the env variable contains the server information.

    d. Add code to convert the API input object to an MSS input object. (This is the same concept as xquery converter logic in Workshop.) This is an example passing an input document number of type String to the Start Order input object.

       ```
       StartOrderByKeyRequest startOrderByKeyReq =
           new StartOrderByKeyRequest();
       MetaSolvOrderKeyChoice metaSolvOrderKeyChoi =
           new MetaSolvOrderKeyChoice();
       OrderKey ordKey = new OrderKey();
       ordKey.setPrimaryKey("11111");
       ordKey.setType("String");
       ```

    e. Initialize the web service port.

       ```
       OrderSoap ord = getWebService();
       ```

       The getWebService() method is a separate method and contains the Atomic Transaction along with the web service reference which points to the server

WSDL file. The @Transactional and @WebServiceRef are annotations used by the getWebService() method.

```
@Transactional(value = Transactional.TransactionFlowType.SUPPORTS,
                version = Transactional.Version.DEFAULT);
@WebServiceRef(wsdlLocation =
                "http://ipaddress:port/MssWS/order/Order?WSDL",
                value =  client.Order.class)Order service;

private OrderSoap getWebService() {
   return service.getOrderSoap();
}
```

The client.Order.class is the Order class generated by wsimport command.

f. Start the transaction.

```
utx.begin();
```

g. Secure the web service port using the credentials.

```
List<CredentialProvider> credProviders =
      new ArrayList<CredentialProvider>();
String username = "username";
String password = "password";
CredentialProvider cp =
    new ClientUNTCredentialProvider(username.getBytes(),
                                    password.getBytes());
credProviders.add(cp);
Map<String, Object> rc = ((BindingProvider)ord).getRequestContext();
rc.put(WSSecurityContext.CREDENTIAL_PROVIDER_LIST, credProviders);
```

h. Invoke the web service operation. For instance, call the start order by key web service.

```
StartOrderByKeyResponse StartOrderByKeyRes =
     ord.startOrderRequest(startOrderByKeyReq);
```

i. Map the MSS response object to the custom response object using converter logic.

j. Close the transaction.

```
utx.commit();
```

Note: In addition to these steps, ensure that all exceptions are handled properly using a catch block, and throw the error encountered.

For multiple operations, you must repeat these steps and the transaction has to be handled.

11. Build the web service.

a. Add a build-service target to the **build.xml** file that executes the **jwsc** Ant task against the updated JWS implementation class. Use the compiledWsdl attribute of **jwsc** task to specify the name of the JAR file generated by the **wsdlc** Ant task:

```
<target name="build-service">
```

```
     <mkdir dir="${build}/${ear.name}" />
     <jwsc srcdir="src" destdir="${build}/${ear.name}">
          <jws file="src/custom/mss/test/testAPISoapImpl.java"
com-piledWsdl="jarJws/testAPI_wsdl.jar" type="JAXWS"/>
     </jwsc>
</target>
```

b. Ensure you have the wsimport client class files or the JAR file in the classpath. This is required to successfully compile the implementation files.

c. After adding the build-service target, go to command prompt and run the following command

```
ant build-service
```

This will compile the web service class that contains the business logic.

12. Generate the custom web service EAR file using build.deliverable target.

```
<target name="build.deliverable" depends="build-service"
        description="Generates the WebService EAR">
  <delete file="Custom_WebService.ear"/>
  <ear destfile="Custom_WebService.ear"
   appxml="descriptors/META-INF/application.xml">
  <fileset dir="output/${ear.name}">
    <include name="**"/>
  </fileset>
  </ear>
</target>
```

This target creates an EAR file using the contents in **output/***earFileName* directory.

13. Deploy the EAR file on the server:

a. Deploy the web service, packaged in an Enterprise Application, to WebLogic Server, using the **wldeploy** Ant task.

b. To use the **wldeploy** Ant task, add the following target to the **build.xml** file:

```
<taskdef name="wldeploy"
          classname="weblogic.ant.taskdefs.management.WLDeploy"/>
  <target name="deploy">
    <wldeploy action="deploy" name="wsdlcEar"
      source="output/wsdlcEar" user="${username}"
      password="${password}" verbose="true"
      adminurl="t3://${hostname}:${port}"
      targets="${server.name}" />
  </target>
```

Substitute the values for *username*, *password*, *hostname*, *port*, and *server.name* that correspond to your WebLogic Server instance.

c. Deploy the file by executing the deploy target:

```
ant deploy
```

14. Test that the web service is deployed correctly by invoking its WSDL in your browser:

```
http://hostname:port/custom/mss/test/testAPI?WSDL
```

The context path and service URI section of the preceding URL are specified by the original WSDL. Use the *hostname* and *port* relevant to your WebLogic Server instance. The deployed and original WSDL files are the same, except for the host and port of the endpoint address.

For more information on “Developing JAX-WS Web Services for Oracle WebLogic Server”, refer to the following Oracle WebLogic Server website:

`https://docs.oracle.com/middleware/1212/wls/WSGET/toc.htm`

Migrating XML APIs Using Oracle SOA Suite

Oracle SOA Suite enables system developers to set up and manage services and to orchestrate them into composite applications and business processes. Oracle SOA Suite runs on the same platform as WebLogic Integration (WebLogic Server).

To migrate a Workshop project to an Oracle SOA Suite project, you must design and build it. In Oracle SOA Suite, workflows are defined as Business Process Execution Language (BPEL) files. You create these files in the same way as you create JPD files in Workshop. Oracle SOA Suite has features which are similar to Workshop features that assist in the design process of workflows.

See the following website for information on the community of joint WLI/SOA users and the “SOA Suite Essentials for WLI users” series:

`http://www.oracle.com/technetwork/topics/soa/index-085039.html`

Migrating XML APIs Using a Java Class

Often you can have a call to an XML API as part of integration code that may call into several different systems. This section addresses how a simple Java class can invoke MSS Web Services.

The following steps show an example of how to write a Java class to invoke MSS Web Services:

1. Create a Java class with a main method that invokes another method, for example getOrderDetails(). The getOrderDetails() method calls the web service to retrieve order details and also contains the invocation logic.

```
public static void main(String[] args)  {
   getOrderDetails();
}
```

2. Create the method getOrderDetails() that invokes the MSS Web Service.
3. The first step of this method is to determine the WSDL URL and add a definition for that string variable. For instance, the following contains a line of code with a sample URL format for the Order Web Service:

```
String url = "http://username:password@ipAddress:port/MssWS/order/Order?WSDL";
```

   where:

   - *username* is the username to log in to the web service
   - *password* is the password to log in to the web service
   - *ipAddress* is the IP Address for the WSDL location, such as 192.0.2.12
   - *port* is the port for the WSDL location, such as 1521

4. Define the soap message which is passed in as input XML.

Note: The input XML can be read in from a file. These details are included to illustrate how the input message is built.

```
MessageFactory mf = MessageFactory.newInstance();
SOAPMessage message = mf.createMessage();
SOAPPart soapPart = message.getSOAPPart();
```

5. Define the message header with the XML namespaces and add these to the envelope.

```
/* XML NameSpaces */
String soapenv = "http://schemas.xmlsoap.org/soap/envelope/";
String open="http://www.openuri.org/";
String ord="http://xmlns.oracle.com/communications/mss/OrderManagementAPI";
String
ord1="http://xmlns.oracle.com/communications/mss/OrderManagementEntities";
String com="http://java.sun.com/products/oss/xml/Common";
String wsse =
"http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-secext-1.0.
xsd";
String wsu =
"http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-1.0
.xsd";

/* Adding Name Spaces to the Envelope */
SOAPEnvelope envelope = soapPart.getEnvelope();
envelope.addNamespaceDeclaration("soapenv", soapenv);
envelope.addNamespaceDeclaration("open", open);
envelope.addNamespaceDeclaration("ord", ord);
envelope.addNamespaceDeclaration("ord1", ord1);
envelope.addNamespaceDeclaration("com", com);
envelope.addNamespaceDeclaration("wsse", wsse);
envelope.addNamespaceDeclaration("wsu", wsu);
SOAPHeader soapheader = envelope.getHeader();
SOAPBody soapbody = envelope.getBody();
```

6. Because MSS Web Services are secured, you must pass proper credentials which update the security header.

```
SOAPElement soapsecurity = soapheader.addChildElement("Security","wsse");
QName soapsecurityattribute = new QName("soapenv:mustUnderstand");
soapsecurity.addAttribute(soapsecurityattribute, "1");

SOAPElement soapusernametoken =
      soapsecurity.addChildElement("UsernameToken","wsse");
QName usernameattribute = new QName("wsu:Id");
soapusernametoken.addAttribute(usernameattribute,
      "UsernameToken-983A273E3EDA90F960148681422777119");

SOAPElement soapusername =
      soapusernametoken.addChildElement("Username","wsse");
soapusername.addTextNode("yourUsername");
SOAPElement soappassword =
      soapusernametoken.addChildElement("Password","wsse");
QName passwordattribute = new QName("Type");
soappassword.addAttribute(passwordattribute,

"http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-username-token-profile
-1.0#PasswordText");
```

```
/* password is your password value set in the addTextNode method */
soappassword.addTextNode("password");
```

7. Add the contents to the message body.

```
/* Add the SOAP body */
SOAPElement soapgetOrder = soapbody.addChildElement("getOrderByKey", "open");
SOAPElement soapmomGetOrderByKeyRequest =
      soapgetOrder.addChildElement("momGetOrderByKeyRequest", "ord");
SOAPElement soapmommekey =
      soapmomGetOrderByKeyRequest.addChildElement("mommekey","ord");
SOAPElement soapmetaSolvOrderKey =
      soapmommekey.addChildElement("metaSolvOrderKey","ord1");
SOAPElement soapapplicationDN =
      soapmetaSolvOrderKey.addChildElement("applicationDN","com");
SOAPElement soaptype = soapmetaSolvOrderKey.addChildElement("type","com");
SOAPElement soapprimaryKey =
      soapmetaSolvOrderKey.addChildElement("primaryKey","ord1");
/* This DOCUMENT_NUMBER is for the order number */
soapprimaryKey.addTextNode(DOCUMENT_NUMBER);
```

8. Add the code to invoke the web service. The soapResponse variable contains the output details.

```
/* Call with the SOAP message being returned */
SOAPMessage soapResponse = con.call(message, url);
```

Mapping Existing XML APIs to Web Services

The goal of the web services release is that the web service functionality matches the functionality of the XML APIs. However, there are some changes to the method names and to some schema file names. Table 2–1, " Mapping XML APIs to Web Services" shows the changes from the XML APIs to the web service operation names.

Note: Table 2–1 through Table 2–10 refer to changes in moving from the XML APIs to the web services for MSS Release 6.3. Table 2–11 refers to changes made in MSS Release 6.3.0.1.

Table 2–1 Mapping XML APIs to Web Services

XML API Method Name	Web Service Operation Name
GetServiceActivationData	getActivationData
DeleteCustomerAccount	deleteCustomerRequest
GetCustomerAccountSync	getCustomerAccountByKey
ImportCustomerAccountSync	importCustomerAccount
GetIntegrationEvent	getIntegrationEventData
InboundEventStatusUpdate	updateInboundEventStatus
UpdateIntegrationEvent	updateIntegrationEventStatus
OutboundEventStatusUpdate	updateOutboundEventStatus
AuditTrail	auditTrailRecording
CreateEntitySync	createEntityByValueRequest

Table 2–1 (Cont.) Mapping XML APIs to Web Services

XML API Method Name	Web Service Operation Name
CreateLocationSync	createLocationRequest
CreateNewInventoryItem	createNewInventoryItemRequest
DeleteLocationSync	deleteLocationRequest
GetAvailablePhysicalPorts	getAvailablePhysicalPortsRequest
GetDlrByKeySync	getDlrByKeyRequest
GetDlrByOrderKeySync	getDlrByOrderKey
GetEntityByKeySync	getEntityByKeyRequest
GetIPAddresses	getIpAddressesRequest
GetLocationSync	getLocationRequest
GetNetworkAreasByGeoArea	getNetworkAreasByGeoAreaRequest
GetNetworkComponents	getNetworkComponentsRequest
GetServiceLocationSync	getServiceLocationRequest
CreateInventoryAssociation	inventoryAssociationRequest
TNRecall	processTNRecallRequestDocument
QueryEndUserLocation	queryEndUserLocation
QueryInventoryManagementSync	queryInventoryManagementRequest
QueryNetworkLocation	queryNetworkLocation
TNValidation	tnValidationRequest
UpdateEntitySync	updateEntityByValueRequest
UpdateLocationSync	updateLocationRequest
UpdateTelephoneNumber	updateTNRequest
AddTaskJeopardy	addTaskJeopardyRequest
AssignProvPlanSync	assignProvPlanRequest
CompletOrderTask	completeTaskRequest
CreateAttachment	createAttachment
CreateISROrderSync	createISROrderRequest
CreateOrderMainSync	createOrderRequest
CreateOrderRelationship	createOrderRelationship
CreatePSROrderSync	createPSROrderRequest
GetCnamData	getCnamData
GetE911Data	getE911Data
GetLidbData	getLidbData
GetPSROrderByTN	getPSROrderByTN
GetPSROrderSync	getOrderByKey
GetTaskJeopardy	TaskJeopardyRequest
GetTaskViewDetail	TaskRequest
ProcessBillingTelephoneNumber	billingTelephoneNumberRequest

Table 2–1 (Cont.) Mapping XML APIs to Web Services

XML API Method Name	Web Service Operation Name
ProcessSuppOrder	processSuppOrderRequest
QueryOrderManagementSync	queryOrderManagementRequest
ReopenTask	reopenTaskRequest
StartOrderSync	startOrderRequest
TransferTaskSync	transferTaskRequest
UpdateCnamData	updateCnamData
UpdateE911Data	updateE911DataRequest
UpdateEstimatedCompletionDate	updateEstimationCompletedDateRequest
UpdateGatewayEventSync	updateGWEventRequest
UpdateLidbData	updateLidbData
UpdateOrderMainSync	updateOrderRequest
UpdatePSROrderSync	updatePSROrderRequest
CreateSoaMessage	createSoaMessageRequest
GetSoaDefaults	getSoaDefaultsRequest
GetSoaInformation	getSoaInformationRequest
GetSoaMessagesToSend	getSoaMessageToSendRequest
GetSoaTnsForOrder	getSoaTnsForOrderRequest
SetTnSoaComplete	setTnSoaCompleteRequest

Table 2–2 through Table 2–10 show the details of the following:

- An element or complex type name change.
- An XSD definition change.

Table 2–2 OrderManagementData.xsd

Before	After
PsrOrderItemPriceType	MomPsrOrderItemPriceType
LocationTypeEnumType	MomLocationTypeEnumType

Table 2–3 OrderManagementEntities.xsd

Before	After
name="primaryKey" type="string" nillable="true" minOccurs="0"	name="primaryKey" type="string" minOccurs="1"
name="taskPrimaryKey" type="string"	name="taskPrimaryKey" type="string" minOccurs="1"

Table 2–4 ServiceData.xsd

Before	After
HuntGroupType	MsdHuntGroupType
TrunkGroupType	MsdTrunkGroupType

Table 2–4 (Cont.) ServiceData.xsd

Before	After
ISDNTrunkGroupInformationType	MsdISDNTrunkGroupInformationType

Table 2–5 ServiceEntities.xsd

Before	After
name="serviceSpecificationPrimaryKey" type="string" nillable="true" minOccurs="0"	name="serviceSpecificationPrimaryKey" type="string" minOccurs="1"
name="servicePrimaryKey" type="string" default="0" nillable="true" minOccurs="0"	name="servicePrimaryKey" type="string" default="0" minOccurs="1"

Table 2–6 InventoryManagementEntities.xsd

Before	After
LocationValue	MimLocationValue
name="networkLocationPrimaryKey" type="string" nillable="true" minOccurs="0"	name="networkLocationPrimaryKey" type="string" minOccurs="1"
name="endUserLocationPrimaryKey" type="string" nillable="true" minOccurs="0"	name="endUserLocationPrimaryKey" type="string" minOccurs="1"

Table 2–7 CustomerManagementAPI.xsd

Before	After
key	mcmmekey
value	mcmmevalue

Table 2–8 OrderManagementAPI.xsd

Before	After
PsrOrderItemPriceType	PsrOrderItemPriceType
LocationTypeEnumType	MomLocationTypeEnumType
QueryValue	MomQueryValue
key	mommekey
value	mommevalue
addTaskJeopardyRequest	momAddTaskJeopardyRequest
createAttachmentResponse	momCreateAttachmentResponse
createOrderRelationshipResponse	momCreateOrderRelationshipResponse
getE911DataResponse	momGetE911DataResponse
getCNAMDataResponse	momGetCnamDataResponse
getPSROrderByTNResponse	momGetPSROrderByTNResponse
getLIDBDataResponse	momGetLIDBDataResponse
getOrderByKeyRequest	momGetOrderByKeyRequest
getOrderByKeyResponse	momGetOrderByKeyResponse
queryOrderManagementRequest	momQueryOrderManagementRequest

Table 2–8 (Cont.) OrderManagementAPI.xsd

Before	After
processSuppOrderRequest	momProcessSuppOrderRequest
reopenTaskRequest	momReopenTaskRequest
transferTaskRequest	momTransferTaskRequest
updateE911DataRequest	momUpdateE911DataRequest

Table 2–9 ServiceOrderActivationAPI.xsd

Before	After
createSOAMessageRequest	msoaCreateSOAMessageRequest
getSOADefaultsRequest	msoaGetSOADefaultsRequest
getSOAInformationRequest	msoaGetSOAInformationRequest
getSOATNsForOrderRequest	msoaGetSOATNsForOrderRequest
setTNSOACompleteRequest	msoaSetTNSOACompleteRequest

Table 2–10 InventoryManagementAPI.xsd

Before	After
queryValue	MimQueryValue
queryResponse	MimQueryResponse
getAvailablePhysicalPortsRequest	mimGetAvailablePhysicalPortsRequest
CreateLocationRequest	mimCreateLocationRequest
GetLocationRequest	mimGetLocationRequest
UpdateLocationRequest	mimUpdateLocationRequest
createEntityByValueRequest	mimCreateEntityByValueRequest
updateEntityByValueRequest	mimUpdateEntityByValueRequest
getEntityByKeyRequest	mimGetEntityByKeyRequest
queryInventoryManagementRequest	mimQueryInventoryManagementRequest
updateTNRequest	mimUpdateTNRequest
getNetworkComponentsRequest	mimGetNetworkComponentsRequest
getNetworkAreasByGeoAreaRequest	mimGetNetworkAreasByGeoAreaRequest
getIpAddressesRequest	mimGetIpAddressesRequest
createNewInventoryItemRequest	mimCreateNewInventoryItemRequest
queryNetworkLocationResponse	mimQueryNetworkLocationResponse
queryEndUserLocationResponse	mimQueryEndUserLocationResponse

Note: Table 2–11 describes the schema changes in MSS Release 6.3.0.1.

Table 2–11 ServiceOrderActivationData.xsd

Before	After
NpaNxxNumericStringType length value="3" NpaNxxNumericStringType pattern value="[0123456789]"	NpaNxxNumericStringType pattern value="[0-9]{3}"
LineRangeNumericStringType length value="4" LineRangeNumericStringType pattern value="[0123456789]"	LineRangeNumericStringType pattern value="[0-9]{4}"

Functional Differences from the XML APIs

The following sections highlight areas that have changed from the XML APIs to the new MSS Web Services.

- Empty Structures on Responses
- Get DLR Response Field Changes
- Schema File Changes

Empty Structures on Responses

In using the XML APIs a subordinate or child structure may be returned with empty data. In using the MSS Web Services, subordinate structures are only returned with data for a response. For instance, if you retrieve a PSR order with a web service operation and the user data does not exist, the user data structure is not returned as part of the response.

Get DLR Response Field Changes

In retrieving the Design Layout Report (DLR) details using the getServiceRequestDLRs operation, the field values for the circuit type are modified. Table 2–12 shows the before and after values for the 'type' field in the DLRResult complexType.

Table 2–12 Changes to the 'type' Field in DLRResult

Field Name	Description	Before	After
type	Circuit type in DLRResult	C	Connection
		F	Facility
		S	Special
		T	Trunk
		P	Product
		V	Virtual
		B	Bandwidth

Schema File Changes

The **ServiceActivation.xsd** file is removed from the schemas and is not referenced in WSDL files. The references from the **ServiceActivation.xsd** schema file are now located in the **Service.xsd** schema file.

Sample Build XML File

In the section "Migrating XML APIs Developed Using Workshop Using Java" sections of an Ant **build.xml** file are described. For reference, Example 2–1 shows a complete sample build.xml file. This XML file is used as an input to Ant to run the targets defined in the file.

Example 2–1 Full build.xml File

```
<project name="GenerateWS" default="all">

<property name="build" value="output"/>
<property name="adminurl" value="t3://192.0.2.251:7001"/> // Server IP and Port
<property name="serverName" value="m63server"/> // Server Name
<property name="userName" value="weblogic"/> // User Name
<property name="password" value="password"/>
<property name="ear.name" value="Custom_WebService"/> // Ear Name
<property name="main_dist.dir" value="../../../dist/lib"/>

<taskdef name="jwsc" classname="weblogic.wsee.tools.anttasks.JwscTask"/>
<taskdef name="clientgen" classname="weblogic.wsee.tools.anttasks.ClientGenTask"/>
<taskdef name="wsdlc" classname="weblogic.wsee.tools.anttasks.WsdlcTask"/>
<taskdef name="wldeploy" classname="weblogic.ant.taskdefs.management.WLDeploy"/>

<target name="all" depends="generate-from-wsdl,wait"/>

<target name="generate-from-wsdl">
  <wsdlc srcWsdl="wsdl_file/testAPI.wsdl" destJwsDir="jarJws" destImplDir="src"
   type="JAXWS" packageName="custom.mss.test"/>
</target>

<target name="wait">
  <echo message=" Now you need to provide your own Implementation for sayHello()
   of [[[ ws/testAPISoapImpl.java ]]]"/>
  <echo message=" then you need to run 'ant deploy' to rebuild your edited Service
   and to Deploy it Mon the Server..."/>
</target>

<target name="build-service">
  <mkdir dir="${build}/${ear.name}"/>
  <jwsc srcdir="src" destdir="${build}/${ear.name}">
    <jws file="src/custom/mss/test/testAPISoapImpl.java"
     com-piledWsdl="jarJws/testAPI_wsdl.jar" type="JAXWS"/>
  </jwsc>
</target>

<target name="build.deliverable" depends="build-service"
        description="Generates the WebService EAR">
  <delete file="Custom_WebService.ear"/>
  <ear destfile="Custom_WebService.ear"
   appxml="descriptors/META-INF/application.xml">
  <fileset dir="output">
    <include name="**"/>
  </fileset>
  </ear>
</target>

<target name="deploy">
  <wldeploy action="deploy" name="wsdlcEar" source="output/wsdlcEar"
   user="${username}" password="${password}" verbose="true"
   adminurl="t3://${hostname}:${port}" targets="${server.name}"/>
</target>
</project>
```

3

Deploying, Testing, and Invoking Secure Web Services

This chapter provides information about deploying, testing, and invoking secure Oracle Communications MetaSolv Solution (MSS) Web Services.

About Deploying MSS Web Services

You can deploy an MSS Web Service in a single server or in a clustered server environment. If you already have MSS Web Services deployed then you will need to first undeploy them before you can deploy again.

Undeploying MSS Web Services

To undeploy MSS Web Services:

1. Start the WebLogic Server Administration Console with the following URL:

   ```
   http://serverName:port/console
   ```

   where:

   - *serverName* is the host name for MSS Web Services
   - *port* is the port number of the system on which MSS Web Services are installed

2. Enter the administration user name and password.
3. Click **Login**.
4. Under Change Center, click **Lock & Edit**.
5. Expand the Domain Structure tree and click **Deployments**.

   The Summary of Deployments window appears.
6. On the **Control** tab, select **MSS_WebService**.
7. From the Stop list, select **Force Stop Now**.

   Ensure that the state of the MSS Web Service application has changed from Active to Prepared.
8. Click the **Configuration** tab.
9. Select **MSS_WebService** and click **Delete**.

   The Delete Application Assistant window appears.
10. Click **Yes**.

MSS Web Services are now undeployed.

Deploying MSS Web Services

To deploy MSS Web Services:

> **Note:** Ensure that you have undeployed any existing MSS Web Services before deploying.
>
> See "Undeploying MSS Web Services" for more information.

1. Ensure that the administration server is running.

   If it is not running, start it with the following startup script:

   - UNIX

     `domainDirectory/startserverName.sh`

   - Windows

     `domainDirectory/startserverName.cmd`

   where:

   – *domainDirectory* is the WebLogic server domain directory

   – *serverName* is the name of the administration server

   For a clustered server environment, ensure that any managed servers are also running with the following startup script:

   - UNIX

     `domainDirectory/startserverName.sh`

   - Windows

     `domainDirectory/startserverName.cmd`

   where:

   – *domainDirectory* is the WebLogic server domain directory

   – *serverName* is the name of the managed server

   > **Note:** Ensure for the clustered server environment that the administration and proxy servers are also running.

2. Start the WebLogic Server Administration Console using the following URL:

   `http://serverName:port/console`

   - *serverName* is the host name for MSS Web Services
   - *port* is the port number of the system on which MSS Web Services are installed

3. Enter the WebLogic administration user name and password.
4. Click **Login**.
5. Under the Change Center area, click **Lock & Edit**.

6. Expand the **Domain Structure** tree and click **Deployments.**

   The Summary of Deployments window appears.

7. On the **Configuration** tab, click **Install.**

   The Install Application Assistant window appears.

8. Under **Current Location**, navigate to the deploy directory where the desired MSS Web Service EAR file is located.

9. Select **MSS_WebService**, and click **Next**.

   The Choose Targeting Style window appears.

10. Select **Install this deployment as an application** and click **Next**.

11. Under Source Accessibility, select **I will make the deployment accessible from the following location**.

12. Click **Finish**.

13. Under the Change Center area, click **Activate Changes**.

14. Expand the **Domain Structure** tree and click **Deployments**.

    The Summary of Deployments window appears.

15. On the **Control** tab, select **MSS_WebService**.

16. From the **Start** list, select **Servicing all requests**.

    The Start Application Assistant window appears.

17. Click **Yes**.

18. Ensure that the state of the **MSS_WebService** application has changed from Prepared to Active.

    MSS Web Services should now be deployed.

Note: You can ignore the following warning message when it appears on the **appserver.mss.log** file during deployment:

WARNING: Non unique body parts! In a port, according to BP 1.1 R2710 operations must have unique operation signature on the wire for successful dispatch.

About Testing MSS Web Services

You can test MSS Web Services in several ways, for instance by testing using SOAPUI or by writing a Java test client.

Testing MSS Web Services using SOAPUI

SOAPUI is a testing tool that can be used for testing requests like web services. Before testing with SOAPUI, complete the following prerequisites:

1. Install SOAPUI.

2. Modify the following settings in SOAPUI:

   - In Global SoapUI Preferences, select the **Authenticate Preemptively** check box under the HTTP Settings tab. This will add authentication information to the outgoing request.

- Still in the HTTP Settings tab, set the Socket Time Exception to 10 min (600000 ms).

3. Ensure that the MSS server is running.

After completing the prerequisites, complete the following steps for testing with SOAPUI:

1. Get the WSDL location of the module that you need to test from the WebLogic console:
   a. Log in to the WebLogic console.
   b. Under Deployment, expand **MSS WebService**.
   c. Select the module to test (for example, Order).
   d. Click the **Testing** tab.

      You should now see the "?WSDL" URL under Test Point.
   e. Click the URL to display the WSDL location.
2. Copy the WSDL location.
3. Open SOAPUI.
4. Click **File** and then click **New Soap Project**.
5. Enter a project name (for example, "OrderTest").
6. Enter the initial WSDL by pasting the WSDL location.

   Click **OK**.

   The project is now ready with the list of operations for the respective module.
7. Choose the operation that you want to test (for example, "assignProvPlanRequest").
8. Populate the following fields in the **Request Properties** tab on the left side of the screen.
   - Set Username to a valid WebLogic user name.
   - Set Password to the corresponding WebLogic password.
   - Set WSS-Password Type to PasswordText.
   - Set WSS-TimeToLive to 200 (and for a long running API, consider increasing the value).

   Valid values for these fields are required by the secured MSS Web Services.
9. After providing login credentials and the required fields if there are any, click **Run**.

   SOAPUI calls the operation and displays the results.

Note: If you are testing a web service within an SSL environment, you must set the following vmoptions before creating the SOAPUI project. You update the SoapUI vmoptions file by adding the following new line at the end of the file:

```
-Dsoapui.https.protocols=TLSv1.2
```

You find the file with the vmoptions extension under the following directory:

*SoapInstallationFolder***bin**\

where *SoapInstallationFolder* is the folder where SOAPUI was installed on your computer.

Testing MSS Web Services with a Java Test Client

MSS Web Services can also be tested by writing a Java test client. MSS Web Services use JAX-WS, the Java API for web services.You can write a simple Java class which invokes the web service, and it should return the expected data.

You must ensure that you handle transaction management properly. MSS Web Services use Web Services Atomic Transaction (WS-AT). You can refer to this website for more information:

`http://docs.oracle.com/middleware/12212/wls/WLACH/pagehelp/J2EEwebservicewebserviceconfigatomictransactiontitle.html`

For any operation that needs a commit of the transaction, you need to use the Servlet Test Client which should do a commit/rollback based on the response.

The following URL from WebLogic explains how to create a JAX-WS client:

`http://docs.oracle.com/middleware/12212/wls/WSGET/jax-ws-client.htm`

About Invoking Secure MSS Web Services

The MSS Web Service operations are secured using WS-Security. You must be a registered WebLogic user to access the web service operations externally. The following sections provide information on securing the MSS Web Services:

- Securing Web Services
- About Policy Files
- Modifying Web Service Security

Refer to the following WebLogic Server website for more detailed information securing web services using WS-security:

`https://docs.oracle.com/middleware/1221/wls/WSSOV/ws-security-message.htm`

Securing Web Services

The MSS Web Services has security enabled upon installation. Specifically, the web service ports are associated to the default WebLogic security policy file, **usernametoken.xml**. As a result, a user name and password must be sent in clear text over a secure tunnel.

About Policy Files

A policy file can be associated to a port, or to a specific operation defined for the port. When a policy file is associated to a port, it automatically secures all operations defined for the web service. When a policy file is not associated to a port, a policy file can be associated to one or more operations. If necessary, each operation can specify a different policy file. If no policy file is associated to the port, or to any operations, the web service is unsecured and no security validations are performed.

Upon installation of MSS, the WebLogic default policy file, **usernametoken.xml**, is associated to IntegrationEventSoap, InventorySoap, OrderSoap, ServiceActivationDataSoap and SoaSoap. So, all operations are automatically secured, and all operations under each port require a user name and password in the SOAP message header. Example 3–1 shows a SOAP message header with a user name and password specified, where *password* would be the specific password value.

Example 3–1 SOAP Message Header

```
<soapenv:Envelope xmlns:open="http://www.openuri.org/"
xmlns:ord="http://xmlns.oracle.com/communications/mss/OrderManagementAPI"
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
   <soapenv:Header>
   <wsse:Security soapenv:mustUnderstand="1"
   xmlns:wsse="http://docs.oasis-open.org/wss/2004/01/oasis-200401-
   wss-wssecurity-secext-1.0.xsd"
   xmlns:wsu="http://docs.oasis-open.org/wss/2004/01/oasis-200401-
   wss-wssecurity-utility-1.0.xsd">
   <wsse:UsernameToken wsu:Id="UsernameToken-D4DD3881F54839961414839602556291">
   <wsse:Username>admin</wsse:Username>
   <wsse:Password Type="http://docs.oasis-open.org/wss/2004/01/oasis-200401-
   wss-username-token-profile-1.0#PasswordText">password</wsse:Password>
   <wsse:Nonce EncodingType="http://docs.oasis-open.org/wss/2004/01/oasis-
   200401-wss-soap-message-security-1.0#Base64Binary">/qtCBQf4FsS3Y7GdEk0xcw==
   </wsse:Nonce>
   <wsu:Created>2017-01-09T11:10:55.625Z</wsu:Created>
   </wsse:UsernameToken>
   </wsse:Security>
   </soapenv:Header>
   <soapenv:Body>
      <open:getE911Data>
        .
        .
        .
      </open:getE911Data>
   </soapenv:Body>
</soapenv:Envelope>
```

Modifying Web Service Security

You can modify the default security settings through the WebLogic Server Administration Console.

To modify the default web service security settings, see the following:

- Accessing Security
- Associating a Policy File
- Disassociating a Policy File

Accessing Security

To access security:

1. Log in to the WebLogic Server Administration Console.
2. In the left panel, under Domain Structure, click the **Deployments** link.

   The Summary of Deployments page appears.
3. Expand **MSS_WebService**.
4. Under **MSS_WebService**, expand **Web Services**.
5. Under **Web Services**, click the link that represents the name of the web service.

   For example, click the **Order** link.
6. Click the **Configuration** tab, then click the **WS-Policy** tab.

   The WS-Policy tab lists the policy files associated with the web service. Upon installation, this page shows OrderSoap with the **usernametoken.xml** policy file associated.
7. Expand the port.

   All operations are listed under the port.

Associating a Policy File

You can associate a policy file to a port, or to a specific operation defined for the port.

To associate a policy file:

1. Access security for the web service.

   See "Accessing Security" for more information.
2. Click the port or a specific operation.

   The available policy files are listed on the left, and the policy files associated with the port or operation are listed on the right.
3. In the left side, select an available policy file to associate to the port or operation.
4. Click the right arrow, which moves the available policy file to the list of associated policy files.
5. Click **OK**.

Disassociating a Policy File

You can disassociate a policy file from a port or from a specific operation defined for the port.

To disassociate a policy file:

1. Access security for the web service.

   See "Accessing Security" for more information.
2. Click the port or a specific operation.

   The available policy files are listed on the left, and the policy files associated with the port or operation are listed on the right.
3. In the right side, select the policy file to disassociate from the port or operation.
4. Click the left arrow, which moves the associated policy file to the list of available policy files.

5. Click **OK**.

Enabling Transactions Transport Security Mode

You use the SSL (Secure Sockets Layer) protocol to encrypt and secure information transported in a web service operation. In an SSL environment where the HTTPS port is enabled and the HTTP port is disabled, the Transactions Transport Security Mode must have an "SSL Required" setting. For example, you can invoke the following in this scenario:

```
https://hostname:httpsPort/MssWS/customer/CustomerAccount?WSDL
```

where the *hostname* and *httpsPort* are relevant to your WebLogic Server instance.

To enable this setting, perform the following steps:

1. Start the WebLogic Server Administration Console with the following URL:

   ```
   http://serverName:port/console
   ```

   where:

   - *serverName* is the host name for MSS Web Services
   - *port* is the port number of the system on which MSS Web Services are installed

2. Enter the administration user name and password.
3. Click **Login**.
4. Under the Change Center, click **Lock & Edit**.
5. Expand the Domain Structure tree and select your domain.
6. On the **JTA** tab, select **Advanced**.
7. For the **Web Service Transactions Transport Security Mode**, set it to a **SSL Required** value.
8. In the Change Center area of the Administration Console, click **Activate Changes**, which activates these changes.
9. Restart the Administration server.

Troubleshooting

You use the following section to help resolve items that you can encounter during your testing.

Connecting to the WSDL Location

When trying to connect to the MSS Web Service WSDL location using the IP Address, you can get an error. For example, the following location:

```
http://IPaddress:port/MssWS/order/Order?WSDL
```

can give the error "The page cannot be displayed" but the server is running. The issue involves the internal and external IP Address defined for the computer. When the computer has both an internal and an external IP Address defined, then the MSS Web Service WSDL location might not work using the IP Address.

Solution

The workaround for this issue is to use the host name instead of an IP Address. For example:

```
http://hostname:port/MssWS/order/Order?WSDL
```

This location with the host name provides the proper WSDL details.

Redirecting Log Messages

When an MSS Web Service operation fails, sometimes the error is not logged in the **mss.log** or the **AppServerLog.xml** file. Instead, it is logged to the **AppServerLog_misc.xml** file which is also available on the **logs** directory.

Solution

You can redirect the logs to the **AppServerLog.xml** file. You add the section in Example 3–2 to the **loggingConfig.xml** file which is under the following directory:

```
MSLV_HOME\m63Server\AppServer\Config
```

Example 3–2 Snippet Addition to loggingConfig.xml File

```
<category name="cmm.Integration.API"
      class="com.metasolv.common.framework.logging.api.log4jext.MSLVLogger"
      additivity="false">
   <level value ="error"
      class="com.metasolv.common.framework.logging.api.log4jext.MSLVLevel"/>
      <appender-ref ref="XMLFileApp"/>
</category>
```

Invoking MSS Web Services in an SSL Environment

When you invoke an MSS Web Service operation in an SSL environment where the HTTP port is disabled sometimes an error can occur. This error occurs when you invoke the web service through the HTTPS port and the API throws the following error:

```
weblogic.wsee.server.ServerURLNotFoundException: Cannot resolve uri for
protocol http/https
```

Solution

You must enable the Transactions Transport Security Mode setting through the WebLogic Console to "SSL Required" value. See "Enabling Transactions Transport Security Mode" for the steps to set this mode.

4 Customer Web Service Reference

This chapter provides information about the Oracle Communications MetaSolv Solution (MSS) Customer Web Service.

About the Customer Web Service

The Customer Web Service enables an external system to import and maintain customer accounts in MSS. Customer Web Service operations enable you to:

- Import or update a customer account into MSS.
- Get a customer account details using customer account id from MSS.
- Delete a customer account using customer account id from MSS.

About the Customer Web Service Packaging

The Customer Web Service is packaged in the **MSS_WebService.ear** file which contains the **customer.war** file. When the installer deploys the **EAR** file, the Customer Web Service is automatically deployed and ready to use.

Note: The **MSS_WebService.ear** file also includes the other MSS web service operations. See Chapter 1, "Web Services Overview" for information about these operations.

About the Customer WSDL, WAR, and Schema Files

The Customer Web Service is defined by the **CustomerAPI.wsdl** file and is supported by numerous schema files. The WSDL file and supporting schema files are located in the **customer.war** file.

See "Understanding How MSS Defines Web Services" for more information about WSDL and WAR files, and about their directory locations in the EAR file.

About Customer Schema Files

Several schema files support the Customer Web Service. Within **customer.war** file, the schema files are located in the **WEB-INF/wsdls** directory. These schemas are categorized as common schemas, entity schemas, data schemas, and API schemas.

Common Schemas

For information about the common schema files, see "About Schema Files".

Entity Schemas

The entity schemas define elements, such as keys and types, specific to the web service.

The Customer entity schemas are defined in the following files:

- CustomerManagementEntities.xsd
- InventoryManagementEntities.xsd
- MIPCommonEntities.xsd
- OrderManagementEntities.xsd
- OrderManagementEvents.xsd
- ServiceEntities.xsd

Data Schemas

The data schemas contain numerous complex type structures, enumerations, and simple types.

The Customer data schemas are defined in the following files:

- CustomerManagementData.xsd
- InventoryManagementData.xsd
- OrderAncillaryManagementData.xsd
- OrderManagementData.xsd
- ServiceData.xsd

API Schemas

The API schemas contain the high-level response and request type definitions and exception definitions.

The Customer API schemas are defined in the following files:

- CustomerManagementAPI.xsd
- OrderManagementAPI.xsd

deleteCustomerRequest Operation

This operation enables you to delete a customer account. It deletes all of the child accounts if there is no telephone number reservation against the customer account and if they are not dependent on the following:

- Service request
- Service items

The following are the request and response structures:

Request Structure: deleteCustomerRequest

Response Structure: deleteCustomerRequestResponse

deleteCustomerRequest

The deleteCustomerRequest element contains the input information for the operation. Each row in Table 4–1 describes the element or type name, the XSD declaration, the type description, and the file name that contains the item's definition.

Table 4–1 Payload Information for the Request

Name	Defined As	Type Description	File Name
deleteCustomerRequest	element	deleteCustomerAccountByKeyRequest	CustomerAPI.wsdl
deleteCustomerAccountByKeyRequest	element	mcmmekey	CustomerManagementAPI.xsd
mcmmekey	element	MetaSolvCustomerAccountKeyChoice	CustomerManagementEntities.xsd
MetaSolvCustomerAccountKeyChoice	complexType	metaSolvCustomerAccountKey	CustomerManagementEntities.xsd
metaSolvCustomerAccountKey	element	MetaSolvCustomerAccountKey	CustomerManagementEntities.xsd
MetaSolvCustomerAccountKey	complexType	Extension of CustomerAccountKey Contains a list of fields	CustomerManagementEntities.xsd

Table 4–2 describes the required fields for deleteCustomerRequest.

Table 4–2 Required Fields for deleteCustomerRequest

Field Name	Data Type	Field Description
customerAccountPrimaryKey	string	Customer Account ID of the customer account.

deleteCustomerRequestResponse

The deleteCustomerRequestResponse element contains the output information for the operation. The information returned in the response indicates if the operation is successful. Table 4–3 describes the returned information in the response.

Table 4–3 Payload Information for the Response

Name	Defined As	Type Description	File Name
deleteCustomerRequestResponse	element	deleteCustomerAccountByKeyResponse	CustomerAPI.wsdl
deleteCustomerAccountByKeyResponse	element	Contains the status string	CustomerManagementAPI.xsd

Table 4–4 describes the error messages for the operation.

Table 4–4 Error Messages for the Operation

Error Message	Cause	Resolution
Record not found: PSRCustomerAccount.custAcctID=	Customer account ID provided is not valid.	Populate a valid customer account ID.
PSRServReqServItemProcessorMSG001:The customer account cannot be deleted as it has relationships with other customer accounts or service requests.	The customer account ID provided is associated with other customer account or service requests.	Check the dependent customer accounts or service requests before deleting the customer account.
PSRServReqServItemProcessorMSG002:This Customer account or child customer/billing account have service items. It cannot be deleted.	This customer account or child customer or billing account have service items.	Populate the customer account without active service items associated.

getCustomerAccountByKey Operation

You can use this operation to retrieve customer account details by using the Customer Account ID which will be passed as an input value.

The following are the request and response structures:

Request Structure: getCustomerAccountByKey

Response Structure: getCustomerAccountByKeyResponse

getCustomerAccountByKey

The getCustomerAccountByKey element contains the input information for the operation. Each row in Table 4–5 describes the element or type name, the XSD declaration, the type description, and the file name that contains the item's definition.

Table 4–5 Payload Information for the Request

Name	Defined As	Type Description	File Name
getCustomerAccountByKey	element	getCustomerAccountByKeyRequest	CustomerAPI.wsdl
getCustomerAccountByKeyRequest	element	mcmmekey	CustomerManagementAPI.xsd
mcmmekey	element	MetaSolvCustomerAccountKeyChoice	CustomerManagementEntities.xsd
MetaSolvCustomerAccountKeyChoice	complexType	Complex type with metaSolvCustomerAccountKey element	CustomerManagementEntities.xsd
metaSolvCustomerAccountKey	element	MetaSolvCustomerAccountKey	CustomerManagementEntities.xsd
MetaSolvCustomerAccountKey	complexType	Extension of CustomerAccountKey Contains a list of fields	CustomerManagementEntities.xsd

Table 4–6 describes the required fields for getCustomerAccountByKey.

Table 4–6 Required Fields for getCustomerAccountByKey

Field Name	Data Type	Field Description
customerAccountPrimaryKey	string	Customer Account ID of the customer account.

getCustomerAccountByKeyResponse

The getCustomerAccountByKeyResponse element contains the output information for the operation. The information returned in the response indicates if the operation is successful. Table 4–7 describes the returned information in the response.

Table 4–7 Payload Response for the Response

Name	Defined As	Type Description	File Name
getCustomerAccountByKeyResponse	element	CusgetCustomerAccountByKeyResponse	CustomerAPI.wsdl
CusgetCustomerAccountByKeyResponse	element	Contains the mcmmevalue element	CustomerManagementAPI.xsd
mcmmevalue	element	MetaSolvCustomerAccountValueChoice	CustomerManagementEntities.xsd
MetaSolvCustomerAccountValueChoice	complexType	complextype with metaSolvCustomerAccountValue element	CustomerManagementEntities.xsd
metaSolvCustomerAccountValue	element	MetaSolvCustomerAccountValue	CustomerManagementEntities.xsd
MetaSolvCustomerAccountValue	complexType	Extension of CustomerAccountValue Contains the metaSolvCustomerAccountKey element and a list of fields	CustomerManagementEntities.xsd
metaSolvCustomerAccountKey	element	MetaSolvCustomerAccountKey	CustomerManagementEntities.xsd
MetaSolvCustomerAccountKey	complexType	Extension of CustomerAccountKey Contains a list of fields	CustomerManagementEntities.xsd

Table 4–8 describes the error messages for the operation.

Table 4–8 Error Messages for the Operation

Error Message	Cause	Resolution
No customer for account id	Customer account ID provided is not valid.	Populate the ID with a valid customer account ID.

importCustomerAccount Operation

This operation creates a new customer account or modifies an existing customer account.

The following are the request and response structures:

Request Structure: importCustomerAccount

Response Structure: importCustomerAccountResponse

importCustomerAccount

The importCustomerAccount element contains the input information for the operation. Each row in Table 4–9 describes the element or type name, the XSD declaration, the type description, and the file name that contains the item's definition.

Table 4–9 Payload Information for the Request

Name	Defined As	Type Description	File Name
importCustomerAccount	element	updateCustomerAccountByValueRequest	CustomerAPI.wsdl
updateCustomerAccountByValueRequest	element	Contains the mcmmevalue element	CustomerManagementAPI.xsd
mcmmevalue	element	MetaSolvCustomerAccountValueChoice	CustomerManagementEntities.xsd
MetaSolvCustomerAccountValueChoice	complexType	Contains the metaSolvCustomerAccountValue element	CustomerManagementEntities.xsd
metaSolvCustomerAccountValue	element	MetaSolvCustomerAccountValue	CustomerManagementEntities.xsd
MetaSolvCustomerAccountValue	complexType	See Table 4–10	CustomerManagementEntities.xsd

Table 4–10 defines MetaSolvCustomerAccountValue.

Table 4–10 MetaSolvCustomerAccountValue Definition

Name	Defined As	Type Description	File Name
MetaSolvCustomerAccountValue	complexType	Extension of CustomerAccountValue Contains the metaSolvCustomerAccountKey element	CustomerManagementEntities.xsd
metaSolvCustomerAccountKey	element	MetaSolvCustomerAccountKey	CustomerManagementEntities.xsd
MetaSolvCustomerAccountKey	complexType	Extension of CustomerAccountKey Contains a list of fields	CustomerManagementEntities.xsd
customerBillingAccount	element	CustomerBillingAccountType	CustomerManagementEntities.xsd
CustomerBillingAccountType	complexType	billingAccount	CustomerManagementData.xsd
billingAccount	element	BillingAccountType	CustomerManagementData.xsd
BillingAccountType	complexType	Contains a list of fields	CustomerManagementData.xsd
customerAccountBillCycle	element	CustomerAccountBillCycleType	CustomerManagementData.xsd
CustomerAccountBillCycleType	complexType	Contains a list of fields	CustomerManagementData.xsd
autoPayments	element	AutoPaymentType	CustomerManagementData.xsd
AutoPaymentType	complexType	Contains a list of fields	CustomerManagementData.xsd
prepayments	element	PrePaymentType	CustomerManagementData.xsd
PrePaymentType	complexType	Contains a list of fields	CustomerManagementData.xsd

Table 4–10 (Cont.) MetaSolvCustomerAccountValue Definition

Name	Defined As	Type Description	File Name
creditApps	element	CreditAppsType	CustomerManagementEntities.xsd
CreditAppsType	complexType	Contains the following: ■ psrResCreditApplications ■ psrOrgCreditApplications	CustomerManagementData.xsd
psrResCreditApplications	element	PsrResCreditApplicationType	CustomerManagementData.xsd
PsrResCreditApplicationType	complexType	Contains a list of fields	CustomerManagementData.xsd
psrOrgCreditApplications	element	PsrOrgCreditApplicationType	CustomerManagementData.xsd
PsrOrgCreditApplicationType	complexType	Contains a list of fields	CustomerManagementData.xsd
customerSicCode	element	CustomerSicCodeType	CustomerManagementEntities.xsd
CustomerSicCodeType	complexType	sicCode	CustomerManagementData.xsd
sicCode	element	SicCodeType	CustomerManagementData.xsd
SicCodeType	complexType	Contains a list of fields	CustomerManagementData.xsd
customerServiceCategory	element	CustomerServiceCategoryType	CustomerManagementEntities.xsd
CustomerServiceCategoryType	complexType	Contains a list of fields	CustomerManagementData.xsd
customerNotes	element	CustomerNoteType	CustomerManagementEntities.xsd
CustomerNoteType	complexType	Contains a list of fields	CustomerManagementData.xsd
customerSalesModules	element	CustomerSalesModuleType	CustomerManagementEntities.xsd
CustomerSalesModuleType	complexType	Contains a list of fields	CustomerManagementData.xsd
salesPersons	element	SalesPersonType	CustomerManagementData.xsd
SalesPersonType	complexType	Contains a list of fields	CustomerManagementData.xsd
customerContacts	element	CustomerContactType	CustomerManagementEntities.xsd
CustomerContactType	complexType	Contains a list of fields	CustomerManagementData.xsd
customerSpecialHandlings	element	CustomerSpecialHandlingType	CustomerManagementEntities.xsd
CustomerSpecialHandlingType	complexType	Contains a list of fields	CustomerManagementData.xsd

Table 4–10 (Cont.) MetaSolvCustomerAccountValue Definition

Name	Defined As	Type Description	File Name
customerAddresses	element	CustomerAddressType	CustomerManagementEntities.xsd
CustomerAddressType	complexType	Contains a list of fields	CustomerManagementData.xsd
structureFormat	element	StructureFormatType	CustomerManagementData.xsd
StructureFormatType	complexType	Contains a list of fields	CustomerManagementData.xsd
structureFormatComponents	element	StructureFormatComponentType	CustomerManagementData.xsd
StructureFormatComponentType	complexType	Contains a list of fields	CustomerManagementData.xsd
Discounts	element	DiscountType	CustomerManagementEntities.xsd
DiscountType	complexType	Contains a list of fields	CustomerManagementData.xsd
Agreements	element	AgreementType	CustomerManagementEntities.xsd
AgreementType	complexType	Contains a list of fields	CustomerManagementData.xsd
taxExemptions	element	TaxExemptionType	CustomerManagementEntities.xsd
TaxExemptionType	complexType	Contains a list of fields	CustomerManagementData.xsd
internalAccounts	element	InternalAccountType	CustomerManagementEntities.xsd
InternalAccountType	complexType	Contains a list of fields	CustomerManagementData.xsd
metaSolvUserDataValue	element	metaSolvUserDataValue	CustomerManagementEntities.xsd
metaSolvUserDataValue	complexType	MetaSolvUserDataValueType	MIPCommonEntities.xsd
MetaSolvUserDataValueType	complexType	UserDataValueType	MIPCommonEntities.xsd
UserDataValueType	complexType	Contains a list of fields	MIPCommonEntities.xsd
BaseNameValueType	complexType	Contains a list of fields	MIPCommonEntities.xsd
metaSolvPartyRoleValue	element	metaSolvPartyRoleValue	CustomerManagementEntities.xsd
metaSolvPartyRoleValue	element	MetaSolvPartyRoleUpdateType	MIPCommonEntities.xsd
MetaSolvPartyRoleUpdateType	complexType	Contains a list of fields	MIPCommonEntities.xsd
PartyRoleType	complexType	Contains a list of fields	MIPCommonEntities.xsd

Required Fields for the importCustomerAccount Request Operation

The following tables describe the required fields for the importCustomerAccount request operation. The following are required for the operation:

- metaSolvCustomerAccountValue
- BillingAccountType
- CustomerAccountBillCycleType

The remaining tables and fields are only required if that structure is included. For instance, the fields in AutoPaymentType are only required if the auto-payment option is provided.

Table 4–11 describes the required fields for metaSolvCustomerAccountValue.

Table 4–11 Required Fields for metaSolvCustomerAccountValue

Field Name	Data Type	Field Description
accessCustomer Nr	integer	Industry standard code that identifies a PSR customer account as another carrier. Input a zero value if this is not applicable.
custAcctParentId	string	Return customer account parent ID.
accountStatus	Enum	Valid values of AccountStatusEnumType. Identifies the current condition of the customer or billing account.
billingInterfaceC d	Enum	Valid values of BillingInterfaceEnumType. Indicates the status of this customer information in the billing system.
tradingName	string	Only required for a business service category. If a business customer, the full name of the customer or billing account.
accountType	Enum	Valid values of AccountTypeEnumType. Type of account you are establishing.
extractCreationD ate	CbeDateTim e	Extract creation date.
customerNr	string	User-defined number or the system default number. The system assigns a sequential number to each record. Users can override this default by populating this field.
givenName	string	Only required for a residential service category. First name of a customer.
startDate	CbeDateTim e	Start Date.
familyName	string	Only required for a residential service category. Last name of a customer.
middleInitial	CharType	Middle initial of a customer.
familyGeneratio n	string	Suffix of a customers name.
formOfAddress	string	Form of address to be used with a customer.
priorityCd	string	Classification of the customer or billing account that indicates the way a particular customer will be handled for collection issues and problems.
suspendNonPay	string	Indicates the customer account has a non-sufficient funds situation.
endDate	CbeDateTim e	Indicates the date of the non-sufficient funds event. This field must be non-null. It may have a zero date value.

Table 4–12 describes the required fields for BillingAccountType.

Table 4–12 Required Fields for BillingAccountType

Field Name	Data Type	Field Description
autoPayInd	string	Indicates whether this account will pay its bills automatically (through credit card or bank draft).
billFormatCd	string	Billing format, or template used for invoices sent to this billing account.
billInArrearsInd	string	Indicates that you will bill this customer in arrears.
billingCapAmt	float	Billing cap amount.
creditClass	string	Description of the credit grouping for a customer.
currencyId	integer	Type of currency used in conducting business with this customer.
estUsage	float	Currency amount of the total estimated usage for this billing account.
languageCd	string	Language used in conducting business with this customer.
nsfEffDate	CbeDateTime	If the NSF check box is checked, the date that the customer payment was returned.
nsfInd	string	Indicates whether the billing account is in a non-sufficient funds situation.
supersedureFormInd	string	Indicates whether an Agreement of Change of Responsibility form is on file for the customer.

Table 4–13 describes the required fields for CustomerAccountBillCycleType.

Table 4–13 Required Fields for CustomerAccountBillCycleType

Field Name	Data Type	Field Description
billCycle	string	Indicator of how often (bill frequency) and when (billing period) the customer will receive a bill.
billCycleSeq	string	Sequence for the bill cycle.
startDate	CbeDateTime	Start date of bill cycle.
endDate	CbeDateTime	End date of bill cycle.
billPeriod	string	Two-digit code used for determining the bill cycle. The billing period is used to automatically create the last two characters of the bill cycle number.
billFreqCd	Enum	Valid values of BillingFrequencyEnumType. Cyclical billing frequency for the selected bill cycle.

Table 4–14 describes the required fields for AutoPaymentType.

Table 4–14 Required Fields for AutoPaymentType

Field Name	Data Type	Field Description
autoPaySeq	string	Auto payment sequence.
bankAcctNr	string	Account number from which funds will be drafted.
bankETN	string	Electronic transfer number of the bank holding the customer's account.
bankName	string	Name of the bank that issued the credit card or that holds the bank account.
ccExpDate	CbeDateTime	Month, day, and year that the credit card expires.
ccNr	string	Credit card number for the billing account.

Table 4–14 (Cont.) Required Fields for AutoPaymentType

Field Name	Data Type	Field Description
ccNrSuf	string	Credit card number suffix for the billing account.
startDate	CbeDateTime	Start date for auto payment.
nameOnAcct	string	Name of the customer, as it appears on its credit card or bank account.
paymentTypeCd	string	Method that the customer will use to pay its invoices automatically.
primaryInd	string	Indicates whether this payment method is the customer's first choice. This is only required when the AutoPayment parent tag is provided. This field is only applicable when the customer supplies more than one Payment Method.

Table 4–15 describes the required fields for PrePaymentType.

Table 4–15 Required Fields for PrePaymentType

Field Name	Data Type	Field Description
currencyId	integer	Type of currency used in conducting business with this customer.
documentNumber	integer	Order number.
prepayID	integer	Pre payment ID.
subAcctInd	string	Sub Account Indicator.
Amount	float	Amount of the required prepayment, deposit, or interest.
amtRequested	float	Requested amount.
interestAmt	float	Corresponding interest amount.
amtValueType	string	Measurement that describes the entry in the Payment Amount field.
paymentMethod	string	Method that the customer will use to make the payment.
paymentType	string	Type of required payment that the customer is making.
prepaySeq	string	Sequence for pre payment.
dtRequested	CbeDateTime	Date that the customer was requested to make the required payment.
dtReturned	CbeDateTime	Only applicable for deposits. Date that the deposit was returned to the customer. This field must be non-null. It may have a zero date value.
paymentDt	CbeDateTime	Date that the customer made the required payment.

Table 4–16 describes the required fields for PsrOrgCreditApplicationType

Table 4–16 Required Fields for PsrOrgCreditApplicationType

Field Name	Data Type	Field Description
bankruptInd	string	Indicates whether this company has ever been through bankruptcy proceedings.
tradingName	string	Full company name of the business customer.
companyType	Enum	Valid values of CompanyTypeEnumType. The legal entity under which the company was formed.
dtOrgFormed	string	Date that the business was formed.

Table 4–16 (Cont.) Required Fields for PsrOrgCreditApplicationType

Field Name	Data Type	Field Description
fedIdNr	string	Federal identification number assigned to the company.
nbrYrsInBusiness	string	Number of years that the company has been in business.
registrationDate	CbeDateTime	Date that the organization was legally registered or incorporated.
registrationStCd	string	State in which the organization was registered.
baseApp	Enum	Valid values of BaseAppType.

Table 4–17 describes the required fields for PsrResCreditApplicationType

Table 4–17 Required Fields for PsrResCreditApplicationType

Field Name	Data Type	Field Description
coApplicantNm	string	Full name of the customer's spouse or other co-applicant.
coApplicantSSN	string	Social security number of the customer's co-applicant.
employerAddress	string	Address of the customer's employer.
employerNm	string	Name of the customer's employer.
givenName	string	First name of the customer.
houseHoldIncome	float	Total income, in dollars, of the customer and his/her spouse, if applicable.
familyName	string	Last name of the customer.
middleInitial	CharType	Middle initial of the customer.
familyGeneration	string	Suffix of the customer name, such as Jr.
formOfAddress	string	Form of address to be used with the customer.
nrOfDependents	string	Number of people, other than the customer, that are currently dependent on the household income.
ownerOfHomeInd	string	Indicates whether the customer owns his or her own home.
Ssn	string	Social security number assigned to the customer.
baseApp	Enum	Valid values of baseApp.

Table 4–18 describes the required fields for SicCodeType.

Table 4–18 Required Fields for SicCodeType

Field Name	Data Type	Field Description
sicCd	string	Standard Industry Classification (SIC) code that is used to categorize customer listings for telephone directory publishing purposes.
stdCdInd	string	Indicates whether it is an standard industry code.
subAccountInd	string	Sub Account indicator.

Table 4–19 describes the required fields for CustomerServiceCategoryType.

Table 4–19 Required Fields for CustomerServiceCategoryType

Field Name	Data Type	Field Description
catgKey	integer	Category Key.
catgName	string	Category of service that you offer to customers. Service categories are tied to both customer records and products in the product catalog.
catgType	Enum	Valid values of CategoryTypeEnumType. A broad classification of the types of customers to whom you offer service.

Table 4–20 describes the required fields for CustomerNoteType.

Table 4–20 Required Fields for CustomerNoteType

Field Name	Data Type	Field Description
noteKey	integer	Note key value.
noteText	string	Additional information about the customer.
userId	string	User ID of the person entering the note.

Table 4–21 describes the required fields for CustomerSalesModuleType.

Table 4–21 Required Fields for CustomerSalesModuleType

Field Name	Data Type	Field Description
houseAccountInd	string	Indicates this sales module does not have commissions associated with it.
roleType	string	Identifies the role a sales module can play, such as salesperson or customer care representative.
salesModuleKey	string	Sales module key.
endDate	CbeDateTime	End date.

Table 4–22 describes the required fields for SalesPersonType.

Table 4–22 Required Fields for SalesPersonType

Field Name	Data Type	Field Description
tradingName	string	Name of the company for whom the salesperson works.
emailAddress	string	Email address of the salesperson.
givenName	string	Name of the salesperson.
familyName	string	Last name of the salesperson.
mgrName	string	Name of the salesperson's manager.
salesPersonKey	integer	Sales person key value.
Ssn	string	Social security number of the salesperson.
telephoneNr	string	Telephone number of the salesperson.

Table 4–23 describes the required fields for CustomerContactType.

Table 4–23 Required Fields for CustomerContactType

Field Name	Data Type	Field Description
contactSeq	string	Contact sequence.
contactType	string	Description of this contact's role. This is used to determine who to call for different situations and needs.
givenName	string	First name of the contact.
familyName	string	Last name of the contact.
startDate	CbeDateTime	Start date.
sendToBillingInd	string	Indicates whether this contact information will be fed to the billing system.
telephoneNr	string	Telephone number of the contact.
endDate	CbeDateTime	End date.
gaInstanceKeyCity	integer	City in which the contact is located.
gaInstanceKeyState	integer	State or province in which the contact is located.
gaInstanceKeyCountry	integer	Country in which the contact is located.

Table 4–24 describes the required fields for CustomerSpecialHandlingType.

Table 4–24 Required Fields for CustomerSpecialHandlingType

Field Name	Data Type	Field Description
startDate	CbeDateTime	Creation Date of record.
label	string	Type of customer for whom a special handling code is appropriate.
requiredInd	string	Indicates the special handling code is required for the selected customer account type.
specialHandlingCd	string	Code used to identify customer or billing accounts as requiring special treatment or special handling of invoices and disconnect notices.
specialHandlingSeq	string	Special handling sequence value.
systemDefaultCd	string	Indicates whether a value is the default value for a handling code.
endDate	CbeDateTime	Deletion date of record.
Value	string	Different options associated with the handling code.
valueCd	string	Code associated with each value type.

Table 4–25 describes the required fields for CustomerAddressType.

Table 4–25 Required Fields for CustomerAddressType

Field Name	Data Type	Field Description
tradingName	string	If a business customer, the full name of the customer or billing account.
customerAddressKey	integer	Customer address key value.
dispatchMethodCd	string	Media that is used to bill the customer, such as paper or electronic mail.
givenName	string	First name of a residential customer or the contact for a business customer.
familyName	string	Last name of a residential customer or the contact for a business customer.
Function	enum	Valid values of AddressFunctionEnumType. The function of an address for a customer, such as primary or secondary billing address.
secondaryBillName	string	Name of a department or group that further identifies the customer being billed.
sendToBillingInd	string	Send to Billing indicator value.
generalDelInd	string	Indicates if the customer wants mail sent general delivery by the post office.
incorporatedCd	Enum	Valid values of CityIncorporatedCdEnumType. Identifies whether an area code is inside the incorporated area of that city.
disconnectInd	boolean	Disconnect indicator.

Table 4–26 describes the required fields for StructureFormatType.

Table 4–26 Required Fields for StructureFormatType

Field Name	Data Type	Field Description
sfType	string	Category that describes a structure.
name	string	Name of a specific customization of a structured format type.

Table 4–27 describes the required fields for StructureFormatComponentType.

Table 4–27 Required Fields for StructureFormatComponentType

Field Name	Data Type	Field Description
Id	int	ID of a component of a structured format.
name	string	Name of a component of a structured format.
componentType	string	Type of component, such as table-driven drop-down or valid value drop-down.
Value	string	Name of a value for a component.

Table 4–28 describes the required fields for DiscountType.

Table 4–28 Required Fields for DiscountType

Field Name	Data Type	Field Description
agrmntID	integer	Agreement ID.
discountType	string	Type of discount offered to the customer, either percent or fixed amount.
agrmntNbr	string	Agreement number.
agrmntType	string	Agreement Type.

Table 4–28 (Cont.) Required Fields for DiscountType

Field Name	Data Type	Field Description
discountMinUom	string	Discount minimum value.
discountRole	string	Role of discount.
discountSeq	string	Sequence for discount.
periodValueUom	string	Period value UOM.
agrmntDt	CbeDateTime	Date of agreement.
toEffDt	CbeDateTime	Date the discount ceases to be in effect. This field must be non-null. It may have a zero date value.
fromEffDt	CbeDateTime	Date the discount is effective.

Table 4–29 describes the required fields for AgreementType.

Table 4–29 Required Fields for AgreementType

Field Name	Data Type	Field Description
agrmntNbr	string	Number printed on the signed agreement.
agrmntType	string	Type of agreement associated with the customer.
nmOnAgrmnt	string	Name of the customer contact providing the agreement.
fromEffDt	CbeDateTime	First date on which the agreement is effective.
toEffDt	CbeDateTime	Last date on which the agreement is effective. This field must be non-null. It may have a zero date value.

Table 4–30 describes the required fields for TaxExemptionType.

Table 4–30 Required Fields for TaxExemptionType

Field Name	Data Type	Field Description
certificateNbr	string	Number of the customer's tax exemption certificate.
Description	string	Description of the tax exemption.
taxExemptSeq	string	Sequence value for tax exemption.
taxExemptType	string	Short name or identifier for a type of tax exemption.
fromEffDt	CbeDateTime	Date that the tax exemption is effective.
toEffDt	CbeDateTime	Date on which the tax exemption ceases to be in effect. This field must be non-null. It may have a zero date value.

Table 4–31 describes the required fields for InternalAccountType.

Table 4–31 Required Fields for InternalAccountType

Field Name	Data Type	Field Description
internalAcctID	integer	Internal account ID value.
internalAcctNbr	string	Internal Account Number.
companyNm	string	Name of a group or department within the customer account or billing account.
fromEffDt	CbeDateTime	Start date.
toEffDt	CbeDateTime	End date.
accountStatus	Enum	Valid values of AccountStatusEnumType.
billingInterfaceCd	Enum	Valid values of BillingInterfaceEnumType.

Table 4–32 describes the required fields for UserDataValueType.

Table 4–32 Required Fields for UserDataValueType

Field Name	Data Type	Field Description
tableNm	string	ASAP table where the user data will be stored. This table corresponds to where the user data was originally defined.
keyColumnNm	string	Actual primary key column name of the product area. This is required so the user data row can be associated with one item.
keyValue	integer	Actual value of the product area.

Table 4–33 describes the required fields for BaseNameValueType.

Table 4–33 Required Fields for BaseNameValueType

Field Name	Data Type	Field Description
column	string	Field is the user data column that will be updated.
value	string	Value entered into the defined column.
dataType	string	Function of a user data field. Determines the kind of information a user can enter in a field on the User Data window. Valid Values: Number, Decimal, VARCHAR2, Date, Dropdown.

Table 4–34 describes the required fields for MetaSolvPartyRoleUpdateType.

Table 4–34 Required Fields for MetaSolvPartyRoleUpdateType

Field Name	Data Type	Field Description
actionCd	string	Code for the action to be taken for the party role.

Table 4–35 describes the required fields for PartyRoleType.

Table 4–35 Required Fields for PartyRoleType

Field Name	Data Type	Field Description
partyId	integer	Party id value of the party role of the level 1 service item. This field along with the partyRoleSeq defines the service provider.
partyRoleSeq	integer	Party role sequence value of the level 1 service item. This field along with the partyId defines the service provider.

importCustomerAccountResponse

The importCustomerAccountResponse element contains the output information for the operation. The information returned in the response indicates if the operation is successful. Table 4–36 describes the returned information in the response.

Table 4–36 Payload Information for the Response

Name	Defined As	Type Description	File Name
importCustomerAccountResponse	element	updateCustomerAccountByValueResponse	CustomerAPI.wsdl
updateCustomerAccountByValueResponse	complexType	Complex Type with element mcmmekey	CustomerManagementAPI.xsd
mcmmekey	element	MetaSolvCustomerAccountKeyChoice	CustomerManagementEntities.xsd
MetaSolvCustomerAccountKeyChoice	complexType	Complex Type with element metasolv-cmme:metaSolvCustomerAccountKey	CustomerManagementEntities.xsd
metaSolvCustomerAccountKey	complexType	Complex Type with a list of elements	CustomerManagementEntities.xsd

Table 4–37 describes the error messages for the operation.

Table 4–37 Error Messages for the Operation

Error Message	Cause	Resolution
Record Exists: PSRCustomerAccount.custAcctID=.	A customer account ID exists with the same account number already exists.	Populate the account number which is not used.
Missing Data: PSRCustomerAccount.companyNm.	The company name is missing in the input.	Populate the company name in the input structure.

5 Event Web Service Reference

This chapter provides information about Oracle Communications MetaSolv Solution (MSS) Event Web Service.

About the Event Web Service

The Event Web Service lets an external system retrieve and update gateway event information in MSS. Event Web Service operations enable you to:

- Get integration event data.
- Update the event status for inbound, integration, and outbound event status values.

About the Event Web Service Packaging

The Event Web Service is packaged in the **MSS_WebService.ear** file, which contains the **events.war** file. When the installer deploys the **EAR** file, the Event Web Service is automatically deployed and ready to use.

Note: The **MSS_WebService.ear** file also includes the other MSS web service operations. See Chapter 1, "Web Services Overview" for information about these operations.

About the Event WSDL, WAR, and Schema Files

The Event Web Service is defined by the **EventsAPI.wsdl** file and is supported by numerous schema files. The WSDL file and supporting schema files are located in the **events.war** file.

See "Understanding How MSS Defines Web Services" for more information about WSDL and WAR files, and about their directory locations in the EAR file.

About Event Schema Files

Several schema files support the Event Web Service. Within **events.war** file, the schema files are located in the **WEB-INF/wsdls** directory. These schemas are categorized as common schemas, entity schemas, and data schemas.

Common Schemas

For information about the common schema files, see "About Schema Files".

Entity Schemas

The entity schemas define elements, such as keys and types, specific to the web service.

The Event entity schemas are defined in the following files:

- CustomerManagementEntities.xsd
- InventoryManagementEntities.xsd
- MIPCommonEntities.xsd
- OrderManagementEntities.xsd
- OrderManagementEvents.xsd
- ServiceEntities.xsd

Data Schemas

The data schemas contain numerous complex type structures, enumerations, and simple types.

The Event data schemas are defined in the following files:

- CustomerManagementData.xsd
- InventoryManagementData.xsd
- OrderAncillaryManagementData.xsd
- OrderManagementData.xsd
- ServiceData.xsd

API Schemas

The API schemas contain the high-level response and request type definitions and exception definitions.

The Event API schemas are defined in the following file:

- OrderManagementAPI.xsd

getIntegrationEventData Operation

The getIntegrationEventData operation enables you to retrieve the details of the gateway event by passing the external system name and its corresponding key.

The following are the request and response structures:

Request Structure: getIntegrationEventData

Response Structure: getIntegrationEventDataResponse

getIntegrationEventData

The getIntegrationEventData element contains the input information for the operation. Each row in Table 5–1 describes the element or type name, the XSD declaration, the type description, and the file name that contains the item's definition.

Table 5–1 Payload Information for the Request

Name	Defined As	Type Description	File Name
getIntegrationEventData	complexType	Contains the following fields: ■ pExternalSystemName ■ pExternalSystemKey	EventsAPI.wsdl

Table 5–2 describes the required fields for getIntegrationEventData.

Table 5–2 Required Fields for getIntegrationEventData

Field Name	Data Type	Field Description
pExternalSystem Name	string	Field that defines the name for an external system order.
pExternalSystem Key	string	Field that defines the key for an external system order.

getIntegrationEventDataResponse

The getIntegrationEventDataResponse element contains the output information for the operation. The information returned in the response indicates if the operation is successful. Table 5–3 describes the information returned in the response.

Table 5–3 Payload Information for the Response

Name	Defined As	Type Description	File Name
getIntegrationEventDataRespons e	element	metasolvOrderTaskEventStatusCha nge	EventsAPI.wsdl
metasolvOrderTaskEventStatusC hange	element	MetaSolvOrderTaskEventStatusCh angeType	OrderManagementEv ents.xsd
MetaSolvOrderTaskEventStatusC hangeType	complexType	Extension of BaseEventType Contains a list of fields	OrderManagementEv ents.xsd

Table 5–4 describes the error messages for the operation.

Table 5–4 Error Messages for the Operation

Error Message	Cause	Resolution
No data found for externalSystemKey: *key* and externalSystemName: *name*	The external system key and external system name passed are invalid values.	Verify the external system details that are passed as Input.

updateInboundEventStatus Operation

The updateInboundEventStatus operation enables you to update the inbound gateway events by passing the valid input data.

The following are the request and response structures:

Request Structure: updateInboundEventStatus

Response Structure: updateInboundEventStatusResponse

updateInboundEventStatus

The updateInboundEventStatus element contains the input information for the operation. Each row in Table 5–5 describes the element or type name, the XSD declaration, the type description, and the file name that contains the item's definition.

Table 5–5 Payload Information for the Request

Name	Defined As	Type Description	File Name
updateInboundEventStatus	element	updateOrderTaskEventProcedureValue	EventsAPI.wsdl
updateOrderTaskEventProcedureValue	element	UpdateOrderTaskEventProcedureValue	OrderManagementAPI.xsd
UpdateOrderTaskEventProcedureValue	complexType	Extension of UpdateProcedureValue Contains a list of fields	OrderManagementAPI.xsd

Table 5–6 describes the required fields for updateInboundEventStatus.

Table 5–6 Required Fields for updateInboundEventStatus

Field Name	Data Type	Field Description
gatewayName	string	Name of the third party vendor defined gateway in MSS.
eventName	string	Name of the gateway event that is assigned to a provisioning plan task or to a gateway event behavior.
documentNumber	long	Number of the order that relates to the gateway event.
servItemId	long	Oracle generated sequence that uniquely identifies a service item in the MetaSolv Solution database. Note: This value is only supplied for an item-level gateway event.
taskStatus	TaskEventStatusEnumType	Status of the task associated to the gateway event.
errorMessage	string	Error message that needs to be updated if the status is set as ERROR.

updateInboundEventStatusResponse

The updateInboundEventStatusResponse element contains the output information for the operation. The information returned in the response indicates if the operation is successful. Table 5–7 describes the information returned in the response.

Table 5–7 Payload Information for the Response

Name	Defined As	Type Description	File Name
updateInboundEventStatusResponse	element	Contains the updateInboundEventStatusResult string	EventsAPI.wsdl

Table 5–8 shows the error messages for the operation.

Table 5–8 Error Messages for the Operation

Error Message	Cause	Resolution
No Gateway for Gateway Name	The gateway name provided in input is not a valid gateway name.	Populate a valid Gateway Name in the input structure.
No Gateway Event found for event name in Gateway	There is no gateway event found for the provided event name in gateway name.	Populate the event name with gateway id which has a gateway event associated to it.
No usage (ServReqGateway) for event name in gateway for document	There is no ServReqGatewayEvents that match this gatewayeventid.	Populate a valid gateway event details in the input.
Multiple Gateway Events exist for document number event name	Multiple Gateway Events exist for document number, event name, and update to many is set to false.	Populate the input by executing one by one instead of executing multiple or set the updateMany as true.
SRSI_GATEWAY_EVENT row not found for document number, task id, event name, event version, serv item	There is no data exist for the provided input.	Populate the data which has the data for the given input.

updateIntegrationEventStatus Operation

The UpdateIntegrationEventStatus operation enables an update of integration gateway events and integration details like external key and external name.

The following are the request and response structures:

Request Structure: updateIntegrationEventStatus

Response Structure: updateIntegrationEventStatusResponse

updateIntegrationEventStatus

The updateIntegrationEventStatus element contains the input information for the operation. Each row in Table 5–9 describes the element or type name, the XSD declaration, the type description, and the file name that contains the item's definition.

Table 5–9 Payload Information for the Request

Name	Defined As	Type Description	File Name
updateIntegrationEventStatus	element	metasolvOrderTaskEventStatusChange	EventsAPI.wsdl
metasolvOrderTaskEventStatusChange	element	MetaSolvOrderTaskEventStatusChangeType	OrderManagementEvents.xsd
MetaSolvOrderTaskEventStatusChangeType	complexType	Extension of BaseEventType Contains a list of fields	OrderManagementEvents.xsd
eventNotes	element	GatewayEventNoteType	OrderManagementEvents.xsd
GatewayEventNoteType	complexType	Extension of BaseEventType Contains a list of fields	OrderManagementEvents.xsd
metasolvIntegrationStatus	element	MetaSolvIntegrationEventStatusChangeType	OrderManagementEvents.xsd
MetaSolvIntegrationEventStatusChangeType	complexType	Extension of BaseEventType Contains a list of fields	OrderManagementEvents.xsd

Table 5–10 describes the required fields for updateIntegrationEventStatus

Table 5–10 Required Fields for updateIntegrationEventStatus

Field Name	Data Type	Field Description
eventName	string	Name of the gateway event that is assigned to a provisioning plan task or to a gateway event behavior.
eventId	long	Number that identifies the ID of the Gateway Event.
documentNumber	long	Number of the order that relates to the gateway event.
servItemId	long	Oracle generated sequence that uniquely identifies a service item in the MetaSolv Solution database. Note: This value is only supplied for an item-level gateway event.
taskNumber	long	Oracle generated sequence that uniquely identifies a task in the MetaSolv Solution database.
taskStatus	TaskEventStatusEnumType	Status of the task associated to the gateway event.
eventVersion	string	Error message that needs to be updated if the status is set as ERROR.
externalSystemName	string	Field that defines the name for an external system order.
externalSystemKey	string	Field that defines the key for an external system order.
Status	Enum	Integration event valid status.

updateIntegrationEventStatusResponse

The updateIntegrationEventStatusResponse element contains the output information for the operation. The information returned in the response indicates if the operation is successful. Table 5–11 describes the information returned in the response.

Table 5–11 Payload Information for the Response

Name	Defined As	Type Description	File Name
updateIntegrationEventStatusResponse	element	Contains the updateIntegrationEventStatusResult string	EventsAPI.wsdl

Table 5–12 describes the error messages for the operation.

Table 5–12 Error Messages for the Operation

Error Message	Cause	Resolution
IntegrationEventManager: Error message is required when the event status is ERROR.	The error message is not populated, but the event status is set as ERROR.	Populate the Error message if the error status is set as ERROR.
IntegrationEventManager: Preference for Gateway Event Notes is set as 'N'.	The preference to update the gateway event notes is set as N.	Enable the "GATEWAYEVENTNOTES" preference in work management else do not populate the gateway event notes section.
IntegrationEventManager: Given External System Name and Key already exists for another record.	The provided integration data already exists in the database.	Populate the right integration details in the input.

updateOutboundEventStatus Operation

The updateOutboundEventStatus operation enables you to update the outbound gateway events by passing the input values.

The following are the request and response structures:

Request Structure: updateOutboundEventStatus

Response Structure: updateOutboundEventStatusResponse

updateOutboundEventStatus

The updateOutboundEventStatus element contains the input information for the operation. Each row in Table 5–13 describes the element or type name, the XSD declaration, the type description, and the file name that contains the item's definition.

Table 5–13 Payload Information for the Request

Name	Defined As	Type Description	File Name
updateOutboundEventStatus	element	metasolvOrderTaskEventStatusChange	EventsAPI.wsdl
metasolvOrderTaskEventStatusChange	element	MetaSolvOrderTaskEventStatusChangeType	OrderManagementEvents.xsd
MetaSolvOrderTaskEventStatusChangeType	complexType	Extension of BaseEventType Contains a list of fields	OrderManagementEvents.xsd
eventNotes	element	GatewayEventNoteType	OrderManagementEvents.xsd
GatewayEventNoteType	complexType	Extension of BaseEventType Contains a list of fields	OrderManagementEvents.xsd
metasolvIntegrationStatus	element	MetaSolvIntegrationEventStatusChangeType	OrderManagementEvents.xsd
MetaSolvIntegrationEventStatusChangeType	complexType	Extension of BaseEventType Contains a list of fields	OrderManagementEvents.xsd

Table 5–14 describes the required fields for updateOutboundEventStatus

Table 5–14 Required Fields for updateOutboundEventStatus

Field Name	Data Type	Field Description
eventName	string	Name of the gateway event that is assigned to a provisioning plan task or to a gateway event behavior.
eventId	long	Number that identifies the ID of the Gateway Event.
documentNumber	long	Number of the order that relates to the gateway event.
servItemId	long	Oracle generated sequence that uniquely identifies a service item in the MSS database. **Note:** This value is only supplied for an item-level gateway event.
taskNumber	long	Oracle generated sequence that uniquely identifies a task in the MetaSolv Solution database.
taskStatus	TaskEventStatusEnumType	Status of the task associated to the gateway event.
eventVersion	string	Error message that needs to be updated if the status is set as ERROR.
externalSystemName	string	Field defines the name for an external system order.
externalSystemKey	string	Field defines the key for an external system order.
Status	Enum	Integration event valid status.

updateOutboundEventStatusResponse

The updateOutboundEventStatusResponse element contains the output information for the operation. The information returned in the response indicates if the operation is successful. Table 5–15 describes the returned information in the response.

Table 5–15 Payload Information for the Response

Name	Defined As	Type Description	File Name
updateOutboundEventStatusResponse	element	Contains the updateIntegrationEventStatusResult string	EventsAPI.wsdl

Table 5–16 describes the error messages for the operation.

Table 5–16 Error Messages for the Operation

Error Message	Cause	Resolution
No Gateway for Gateway Name	The gateway name provided in input is not a valid gateway name.	Populate a valid Gateway Name in the input structure.
No Gateway Event found for event name in Gateway	There is no gateway event found for the provided event name in gateway name.	Populate the event name with gateway ID which has a gateway event associated to it.
No usage (ServReqGateway) for event name in gateway for document	There is no ServReqGatewayEvents that match this gateway event ID.	Populate a valid gateway event ID in the input.

6 Inventory Web Service Reference

This chapter provides information about Oracle Communications MetaSolv Solution (MSS) Inventory Web Service.

About the Inventory Web Service

The Inventory Web Service enables an external system to and maintain inventory in MetaSolv Solution. Inventory Web Service operations enable you to:

- Create, get, and update locations.
- Create new inventory items.
- Get available physical ports.
- Create, get, and update entities.
- Get IP addresses.
- Get network areas, locations, and components.
- Get, validate, and update telephone numbers.

About the Inventory Web Service Packaging

The Inventory Web Service is packaged in the **MSS_WebService.ear** file, which contains the **inventory.war** file. When the installer deploys the **EAR** file, the Inventory Web Service is automatically deployed and ready to use.

Note: The **MSS_WebService.ear** file also includes the other MSS web service operations. See Chapter 1, "Web Services Overview" for information about these operations.

About the Inventory WSDL, WAR, and Schema Files

The Inventory Web Service is defined by the **InventoryAPI.wsdl** file and is supported by numerous schema files. The WSDL file and supporting schema files are located in the **inventory.war** file.

See "Understanding How MSS Defines Web Services" for more information about WSDL and WAR files, and about their directory locations in the EAR file.

About Inventory Schema Files

Several schema files support the Inventory Web Service. Within **inventory.war** file, the schema files are located in the **WEB-INF/wsdls** directory. These schemas are categorized as common schemas, entity schemas, and data schemas.

Common Schemas

For information about the common schema files, see "About Schema Files".

Entity Schemas

The entity schemas define elements, such as keys and types, specific to the web service.

The Inventory entity schemas are defined in the following files:

- CustomerManagementEntities.xsd
- InventoryManagementEntities.xsd
- MIPCommonEntities.xsd
- OrderManagementEntities.xsd
- OrderManagementEvents.xsd
- ServiceEntities.xsd

Data Schemas

The data schemas contain numerous complex type structures, enumerations, and simple types.

The Inventory data schemas are defined in the following files:

- CustomerManagementData.xsd
- InventoryManagementData.xsd
- OrderAncillaryManagementData.xsd
- OrderManagementData.xsd
- ServiceData.xsd

API Schemas

The API schemas contain the high-level response and request type definitions and exception definitions.

The Inventory API schemas are defined in the following files:

- InventoryManagementAPI.xsd
- OrderManagementAPI.xsd

auditTrailRecording Operation

This operation returns the success or failure of AuditTrail recording information.

The following are the request and response structures:

Request Structure: auditTrailRecording

Response Structure: auditTrailRecordingResponse

auditTrailRecording

The auditTrailRecording element contains the input information for the operation. Each row in Table 6–1 describes the element or type name, the XSD declaration, the type description, and the file name that contains the item's definition.

Table 6–1 Payload Information for the Request

Name	Defined As	Type Description	File Name
auditTrailRecording	element	queryInventoryManagementResponse	InventoryAPI.wsdl
queryInventoryManagementResponse	element	MimQueryResponse	InventoryManagementAPI.xsd
MimQueryResponse	element	MetaSolvInventoryQueryResponseChoice	InventoryManagementAPI.xsd
MetaSolvInventoryQueryResponseChoice	complexType	queryMSAGResponse This operation uses only queryMSAGResponse and not queryTelephoneNumberInventoryResponse.	InventoryManagementAPI.xsd
queryMSAGResponse	element	msagSearch	InventoryManagementAPI.xsd
msagSearch	element	MetaSolvMSAGSearch	InventoryManagementEntities.xsd
MetaSolvMSAGSearch	complexType	StructureFormatType	InventoryManagementEntities.xsd
StructureFormatType	complexType	StructureFormatComponentType	CustomerManagementData.xsd
StructureFormatComponentType	complexType	Contains a list of fields	CustomerManagementData.xsd

This operation only uses queryMSAGResponse and not queryTelephoneNumberInventoryResponse. Therefore, only queryMSAGResponse is defined in this operation.

Table 6–2 describes the required fields for auditTrailRecording.

Table 6–2 Required Fields for auditTrailRecording

Field Name	Data Type	Field Description
partnerId	string	Partner ID.
successOrFailure	string	Success or failure response for the operation.

Table 6–3 describes the required fields for MetaSolvMSAGSearch.

Table 6–3 Required Fields for MetaSolvMSAGSearch

Field Name	Data Type	Field Description
keyValue	string	Specifies whether the Master Street Address Guide (MSAG) information uses the NENA 2.0 standards.
partnerId	string	Partner ID.

Table 6–4 describes the required fields for StructureFormatType.

Table 6–4 Required Fields for StructureFormatType

Field Name	Data Type	Field Description
sfType	string	Category that describes a structure.
name	string	Name of a specific customizing of a structured format type.

Table 6–5 describes the required fields for StructureFormatComponentType.

Table 6–5 Required Fields for StructureFormatComponentType

Field Name	Data Type	Field Description
Id	int	ID of a component of a structured format.
name	string	Name of a component of a structured format.
componentType	string	Type of component, such as table-driven drop-down or valid value drop-down.
Value	string	Name of a value for a component.

auditTrailRecordingResponse

The auditTrailRecordingResponse element contains the output information for the operation. The information returned in the response indicates if the operation is successful. Table 6–6 describes the returned information in the response.

Table 6–6 Payload Information for the Response

Name	Defined As	Type Description	File Name
auditTrailRecordingResponse	element	Contains the auditTrailRecordingResult string	InventoryAPI.wsdl

Table 6–7 describes the error messages for the operation.

Table 6–7 Error Messages for the Operation

Error Messages	Cause	Resolution
Not executing the stored procedure SP_PSR_MSAG_CUSTOM_VALIDATION because the partyId is either null or is an empty string	The party ID passed on the input is either null or an empty string.	Populate a valid party ID in the input structure to get a successful output.

createEntityByValueRequest Operation

This operation creates an end user location or a network location and returns the key for the new object. This operation also creates a new inventory item using createNumberInventoryValue.

The following are the request and response structures:

Request Structure: createEntityByValueRequest

Response Structure: createEntityByValueResponse

createEntityByValueRequest

The createEntityByValueRequest element contains the input information for the operation. Each row in Table 6–8 describes the element or type name, the XSD declaration, the type description, and the file name that contains the item's definition.

Table 6–8 Payload Information for the Request

Name	Defined As	Type Description	File Name
createEntityByValueRequest	element	mimCreateEntityByValueRequest	InventoryAPI.wsdl
mimCreateEntityByValueRequest	element	createEntityValue	InventoryManagementAPI.xsd
createEntityValue	element	CreateEntityValueChoice	InventoryManagementAPI.xsd
CreateEntityValueChoice	complexType	Contains the following: ▪ createPSRServiceLocationValue ▪ createNumberInventoryValue	InventoryManagementAPI.xsd
createPSRServiceLocationValue	element	CreatePSRServiceLocationValue	InventoryManagementAPI.xsd
CreatePSRServiceLocationValue	complexType	serviceLocationValue	InventoryManagementAPI.xsd
serviceLocationValue	complexType	Extension of LocationValue serviceLocationKey	InventoryManagementEntities.xsd
serviceLocationKey	element	EndUserLocationKey	InventoryManagementEntities.xsd
networkLocationKey	element	networkLocationKey	InventoryManagementEntities.xsd
networkLocationKey	element	networkLocationKey	InventoryManagementEntities.xsd
structureFormats	element	StructureFormatType	InventoryManagementEntities.xsd
StructureFormatType	complexType	Contains a list of fields	CustomerManagementData.xsd

Table 6–9 describes the required fields for serviceLocationValue.

Table 6–9 Required Fields for serviceLocationValue

Field Name	Data Type	Field Description
serviceLocationRef	integer	Reference value of Service Location.
endUserLocatoinName	string	End user location name.
servLocClliCodeTn	string	Network location with switch detail. If you select a value from the drop-down, you must enter a value in the **Switch Network Area** field on the **End User Location Maintenance Window - Network Areas** tab.
servLocClliCodeData	string	Network location that is not in one of the following categories: Customer or Non-building.
e911PayableEntity	string	Company that receives the E911 surcharge collected for a customer with E911 service.
switchedNetworkArea	integer	Switch network area associated with an end user location.
associatedNA	integer	Network areas associated with the end user location.
updateAddressMode	Enum	Valid values of UpdateModeEnumType.

Table 6–10 describes the required fields for EndUserLocationKey.

Table 6–10 Required Fields for EndUserLocationKey

Field Name	Data Type	Field Description
endUserLocationPrimaryKey	string	The end user location ID value of the end user location.

Table 6–11 describes the required fields for NetworkLocationKey.

Table 6–11 Required Fields for NetworkLocationKey

Field Name	Data Type	Field Description
networkLocationPrimaryKey	string	The network location ID value of the network location.

Table 6–12 describes the required fields for NumberInventoryValue.

Table 6–12 Required Fields for NumberInventoryValue

Field Name	Data Type	Field Description
identText	string	Logical identifier of the inventory item stored in the repository, for instance, IP Addresses and Domain Names.
identTextSuffix	string	Value that uniquely defines the Ident Text within an item type category.
userPassword	string	User password for access to the internet.
emailPassword	string	User password for access to the email box.
nIStatus	string	Specifies the status of the inventory item.
apiActivity	string	Code set by the API to pre-assign or flag Number Inventories (for instance, telephone numbers and email addresses) that are used in a subsequent API order import process. The API pre-assign process exists because of the time-lag between the assignment of the number inventory to a customer and the actual assignment of that number inventory through the API order import.
toEffectDate	CbeDateTime	Date the entity type instance became inactive.
fromEffectDate	CbeDateTime	Date the entity type instance became effective.
numberInventory Type	string	Defines the valid Number Inventory Types that can be stored in number inventory.

Table 6–13 describes the required fields for metaSolvNumberInventoryKey.

Table 6–13 Required Fields for metaSolvNumberInventoryKey

Field Name	Data Type	Field Description
numberInventoryKey	int	Foreign key to the entity NI_NBR_INV. The TEL_NUM_INV table stores all the US telephone numbers. This table is a subtype of number inventory table. This new attribute establishes a relationship between TEL_NUM_INV and the Number Inventory tables.

createEntityByValueResponse

The createEntityByValueResponse element contains the output information for the operation. The information returned in the response indicates if the operation is successful. Table 6–14 describes the returned information in the response.

Table 6–14 Payload Information for the Response

Name	Defined As	Type Definition	File Name
createEntityByValueRequestResponse	element	createEntityByValueResponse	InventoryAPI.wsdl
createEntityByValueResponse	element	createEntityValueResponse	InventoryManagementAPI.xsd
createEntityValueResponse	element	CreateEntityResponseChoice	InventoryManagementAPI.xsd
CreateEntityResponseChoice	complexType	Contains the following: ▪ createPSRServiceLocationResponse ▪ createNumberInventoryResponse	InventoryManagementAPI.xsd
createPSRServiceLocationResponse	element	CreatePSRServiceLocationRespone	InventoryManagementAPI.xsd
CreatePSRServiceLocationResponse	complexType	Contains the following: ▪ endUserLocationKey ▪ addressKey	InventoryManagementAPI.xsd
endUserLocationKey	complexType	Contains a list of fields	InventoryManagementEntities.xsd
addressKey	element	String	InventoryManagementEntities.xsd
createNumberInventoryResponse	element	CreateNumberInventoryRespone	InventoryManagementAPI.xsd
CreateNumberInventoryRespone	complexType	metaSolvNumberInventoryKey	InventoryManagementAPI.xsd
metaSolvNumberInventoryKey	complexType	Contains a list of fields	InventoryManagementEntities.xsd

Table 6–15 describes the error messages for the operation.

Table 6–15 Error Messages for the Operation

Error Message	Cause	Resolution
Invalid Relationship: Network Area is Invalid for switch.	The input switch details do not map with network area information.	Populate the switch details which has the provided network area.
Missing Data: End User Location does not exist.	The input end user location ID does not exist in the database.	Verify the end user location ID and provide a valid value.
Missing Data: Network Location does not exist for this LocationIdSr.	The input network location ID (location_id_sr) does not exist in the database.	Please verify the network location id (location_id_sr) and provide a valid value.
Error. E-mail name is required for importing E-mail Number Inventory Item	The input email name is empty.	For creating inventory of EMAIL, you must pass the email ID.
Error. URL Domain is required for importing URL Number Inventory Item.	The input URL domain value is passed as an empty value.	For creating URL inventory, URL domain name is required in the input structure.

createLocationRequest Operation

This operation creates an end user, or network location object in the system and returns the key for the new object.

The following are the request and response structures:

Request Structure: createLocationRequest

Response Structure: createLocationRequestResponse

createLocationRequest

The createLocationRequest element contains the input information for the operation. Each row in Table 6–16 describes the element or type name, the XSD declaration, the type description, and the file name that contains the item's definition.

Table 6–16 Payload Information for the Request

Name	Defined As	Type Description	File Name
createLocationRequest	element	mimCreateLocationRequest	InventoryAPI.wsdl
mimCreateLocationRequest	element	locationValue	InventoryManagementAPI.xsd
locationValue	element	MimLocationValue	InventoryManagementEntities.xsd
MimLocationValue	complexType	Extension of LocationValue Contains a list of fields	InventoryManagementEntities.xsd
networkLocationValue	element	NetworkLocationValue	InventoryManagementEntities.xsd
NetworkLocationValue	complexType	NetworkLocationRelationship	InventoryManagementEntities.xsd
NetworkLocationRelationship	complexType	Contains a list of fields	InventoryManagementEntities.xsd
endUserLocationValue	element	endUserLocationValue	InventoryManagementEntities.xsd
endUserLocationValue	complexType	Contains the following: ■ endUserLocationKey ■ EndUserLocationAddressRange ■ SecondaryLSO	InventoryManagementEntities.xsd
endUserLocationKey	complexType	endUserLocationPrimaryKey	InventoryManagementEntities.xsd
EndUserLocationAddressRange	complexType	Contains a list of fields	InventoryManagementEntities.xsd
SecondaryLSO	complexType	Contains a list of fields	InventoryManagementEntities.xsd
netLocAddress	element	netLocAddress	InventoryManagementEntities.xsd
netLocAddress	complexType	sfAddress	InventoryManagementEntities.xsd
sfAddress	element	StructureFormatType	InventoryManagementEntities.xsd
StructureFormatType	complexType	StructureFormatComponentType	CustomerManagementData.xsd
StructureFormatComponentType	complexType	Contains a list of fields	CustomerManagementData.xsd

Table 6–16 (Cont.) Payload Information for the Request

Name	Defined As	Type Description	File Name
caUsageInstance	element	CaUsageInstanceType	InventoryManagementEntities.xsd
CaUsageInstanceType	complexType	caUsageValue	OrderManagementData.xsd
caUsageValue	element	CaUsageValueType	OrderManagementData.xsd
CaUsageValueType	complexType	Contains a list of fields	OrderManagementData.xsd
metaSolvUserDataValue	element	metaSolvUserDataValue	InventoryManagementEntities.xsd
metaSolvUserDataValue	complexType	MetaSolvUserDataValueType	MIPCommonEntities.xsd
MetaSolvUserDataValueType	complexType	UserDataValueType	MIPCommonEntities.xsd
UserDataValueType	complexType	BaseNameValueType	MIPCommonEntities.xsd
tandemType	element	TandemType	InventoryManagementEntities.xsd
TandemType	complexType	Contains a list of fields	InventoryManagementEntities.xsd
associatedNetworkArea	element	AssociatedNetworkArea	InventoryManagementEntities.xsd
AssociatedNetworkArea	complexType	Contains a list of fields	InventoryManagementEntities.xsd

Table 6–17 describes the required fields for MimLocationValue.

Table 6–17 Required Fields for MimLocationValue

Field Name	Data Type	Field Description
locationType	Enum	Valid values of LocationTypeEnumType. Indicates if a location is a network location, end-user location, or both (an end-user location with a network location alias).
locationId	string	Numeric value. New location when locationId is equal to 0. If locationId ID greater then 0, then it is an update of the location.

Table 6–18 describes the required fields for NetworkLocationValue.

Table 6–18 Required Fields for NetworkLocationValue

Field Name	Data Type	Field Description
locationCodeFormat	string	Select CLLI if you use 8- to 11-character Telcordia standard code.
locationCode	string	Location identification code that identifies specific locations or terminations. This code may be free-form and user-defined, or the Common Language Location. Identification (CLLI) code administered by Telcordia.
locationName	string	Full name of the network location.
switchingOfficeClass	string	Class of a switch in a telephone network.

Table 6–18 (Cont.) Required Fields for NetworkLocationValue

Field Name	Data Type	Field Description
exchangeArea	string	The CO exchange area, or central office, related to the network location. This item is created on the CO Exchange Area window.
locationCategory	string	Class of network locations.
locationTypeId	string	Function a network location plays in supporting a network and a business.
intraNodalSwitching	boolean	Indicates a network location can provide switching functions on either a routine or an emergency basis.
emergencySwitching	boolean	Indicates the network location that can provide switching functions on an emergency basis only.
owned	boolean	Indicates you own the network location and the network location does not serve as an access customer terminal location (ACTL)/Gateway.
ACTL	boolean	Indicates the network location serves as an access customer terminal location or gateway.

Table 6–19 describes the required fields for NetworkLocationRelationship.

Table 6–19 Required Fields for NetworkLocationRelationship

Field Name	Data Type	Field Description
locationIdRel	string	Location ID rel value.
locationRelTypeCode	string	Relationship to the network location entered in the Exchange Area field on the Network Location Assistant - Physical Location Information: Classes and Functions window.
actionCode	string	Action code.

Table 6–20 describes the required fields for EndUserLocationValue.

Table 6–20 Required Fields for EndUserLocationValue

Field Name	Data Type	Field Description
endUserLocationName	string	Full name of the end user location.
endUserLocationType	string	Identifies the type of location.
e911PayableEntity	string	Company that receives the E911 surcharge collected for a customer with E911 service.
telephoneNumberSwitchLocationId	string	Sequence number that uniquely identifies a location. In this case it is the location associated with this address.
dataSwitchLocationId	string	Sequence number that uniquely identifies a location. In this case it is the location associated with this address.

Table 6–21 describes the required fields for EndUserLocationKey.

Table 6–21 Required Fields for EndUserLocationKey

Field Name	Data Type	Field Description
endUserLocationPrimaryKey	string	End user location ID value.

Table 6–22 describes the required fields for EndUserLocationAddressRange.

Table 6–22 Required Fields for EndUserLocationAddressRange

Field Name	Data Type	Field Description
fromRange	long	Starting point for a range.
toRange	long	Ending number in a range of address numbers.
numberPattern	string	Enables selection of odd or even numbering for an address number range.

Table 6–23 describes the required fields for SecondaryLSO.

Table 6–23 Required Fields for SecondaryLSO

Field Name	Data Type	Field Description
networkLocationId	string	Code for identifying a physical point in a network.
servingOfficeTypeCode	string	Type of local serving office (LSO) used for a circuit.
actionCode	string	Action code.

Table 6–24 describes the required fields forNetLocAddress.

Table 6–24 Required Fields for NetLocAddress

Field Name	Data Type	Field Description
primaryIndicator	boolean	Indicates if this record is the primary network location address and whether to display the primary address of a network location in infrastructure.
addressImpactMode	string	Address impact mode.
addressId	string	Address ID.

Table 6–25 describes the required fields for CaUsageInstanceType.

Table 6–25 Required Fields for CaUsageInstanceType

Field Name	Data Type	Field Description
caUsageKey	integer	Unique ID of a custom attribute.
caValueLabel	string	Label of a custom attribute.

Table 6–26 describes the required fields for CaUsageValueType.

Table 6–26 Required Fields for CaUsageValueType

Field Name	Data Type	Field Description
caValueKey	integer	Key of valid value CA.
caValue	string	Value of a custom attribute.
caUom	string	Unit of measure, for example: meters, feet.
caUsageVvKey	integer	Key of valid value CA.

Table 6–27 describes the required fields for UserDataValueType.

Table 6–27 Required Fields for UserDataValueType

Field Name	Data Type	Field Description
tableNm	string	Table where the user data is stored. This table corresponds to where the user data was originally defined.
keyColumnNm	string	Actual primary key column name of the product area. This is required so the user data row can be associated with one item.
keyValue	integer	Actual value of the product area.

Table 6–28 describes the required fields for BaseNameValueType.

Table 6–28 Required Fields for BaseNameValueType

Field Name	Data Type	Field Description
column	string	User data column that is updated.
value	string	Value entered into the defined column.
dataType	string	Function of a user data field. Determines the kind of information a user can enter in a field on the User Data window. Valid Values: Number, Decimal, VARCHAR2, Date, Dropdown.

Table 6–29 describes the required fields for TandemType.

Table 6–29 Required Fields for TandemType

Field Name	Data Type	Field Description
tandemLocationId	string	Unique identifier for a specific location. This ID, visible only to the system, is used to store and retrieve information about the location.
tandemServiceType	string	Code that identifies the type of function performed by the tandem and whether this function is for originating (ORIG) or terminating (TERM) traffic from or to a subtending End Office.
tandemTrafficType	string	Code indicating whether the end office uses this tandem for ORIG or TERM traffic of a specific type.
actionCode	string	Action code.

Table 6–30 describes the required fields for AssociatedNetworkArea.

Table 6–30 Required Fields for AssociatedNetworkArea

Field Name	Data Type	Field Description
networkAreaId	string	Sequence number that uniquely identifies entities of this type.
Name	string	Name of the network area.
actionCode	string	Action code value.

createLocationRequestResponse

The createLocationRequestResponse element contains the output information for the operation. The information returned in the response indicates if the operation is successful. Table 6–31 describes the returned information in the response.

Table 6–31 Payload Information for the Response

Name	Defined As	Type Description	File Name
createLocationRequestResponse	element	createLocationResponse	InventoryAPI.wsdl
createLocationResponse	element	CreateLocationResponse	InventoryManagementAPI.xsd
CreateLocationResponse	complexType	networkLocationKey	InventoryManagementAPI.xsd
networkLocationKey	complexType	Contains a list of fields	InventoryManagementEntities.xsd

Table 6–32 describes the error messages for the operation.

Table 6–32 Error Messages for the Operation

Error Message	Cause	Resolution
Location not found.	The input network location ID does not exist.	Populate a valid network location ID in the input
The End User Location Type must be populated with a valid value (NEWARCH, PREMISE, PRILOC, or SECLOC).	The input end user location type is not valid.	Populate the end user location type with one of the valid values NEWARCH, PREMISE, PRILOC, or SECLOC.
The network area {0} cannot be a switch network area for this Location because it is not served by the specified TN Switch.	The populated network area is not served by the TN switch populated.	Populate the network area which is associated to the specified TN switch.

createNewInventoryItemRequest Operation

This operation creates a new inventory item by populating the inventory item structure.

The following are the request and response structures:

Request Structure: createNewInventoryItemRequest

Response Structure: createNewInventoryItemResponse

createNewInventoryItemRequest

The createNewInventoryItemRequest element contains the input information for the operation. Each row in Table 6–33 describes the element or type name, the XSD declaration, the type description, and the file name that contains the item's definition.

Table 6–33 Payload Information for the Request

Name	Defined As	Type Description	File Name
createNewInventoryItemRequest	element	mimCreateNewInventoryItemRequest	InventoryAPI.wsdl
mimCreateNewInventoryItemRequest	element	inventoryItem	InventoryManagementAPI.xsd
inventoryItem	complexType	Contains the following: ■ metaSolvNumberInventoryKey ■ inventoryType ■ inventorySubType ■ inventoryStatus	InventoryManagementEntities.xsd
metaSolvNumberInventoryKey	complexType	Contains a list of fields	InventoryManagementEntities.xsd

Table 6–33 (Cont.) Payload Information for the Request

Name	Defined As	Type Description	File Name
inventoryType	complexType	Contains a list of fields	InventoryManagementData.xsd
inventorySubType	complexType	Contains a list of fields	InventoryManagementData.xsd
inventoryStatus	complexType	Contains a list of fields	InventoryManagementData.xsd

Table 6–34 describes the required fields for MetaSolvNumberInventoryKey.

Table 6–34 Required Fields for MetaSolvNumberInventoryKey

Field Name	Data Type	Field Description
numberInventoryKey	int	Number Inventory Identifier is the foreign key to the entity NI_NBR_INV. The Tel_Num_Inv table stores all the US telephone numbers. This table is a subtype of number inventory table. This new attribute establishes a relationship between Tel_Num_Inv and the Number Inventory tables.

Table 6–35 describes the required fields for inventoryType.

Table 6–35 Required Fields for inventoryType

Field Name	Data Type	Field Description
Code	Enum	Valid values of InventoryTypeCodeTypeEnum. The code that identifies the type of inventory item stored.
Alpha	string	Alpha equivalent of the Number Inventory type code.

Table 6–36 describes the required fields for InventorySubType.

Table 6–36 Required Fields for InventorySubType

Field Name	Data Type	Field Description
Code	Enum	Valid values of InventoryTypeCodeTypeEnum. The code that identifies the inventory item type subtype.
Alpha	string	Alpha equivalent of valid subtype code of inventory item type.

Table 6–37 describes the required fields for InventoryStatus.

Table 6–37 Required Fields for InventoryStatus

Field Name	Data Type	Field Description
Code	Enum	Valid values of InventoryTypeCodeTypeEnum. The code that identifies the status of the inventory item.
Alpha	string	Alpha equivalent of the Number Inventory status code Alpha Description.

createNewInventoryItemResponse

The createNewInventoryItemResponse element contains the output information for the operation. The information returned in the response indicates if the operation is successful. Table 6–38 describes the returned information in the response.

Table 6–38 Payload Information for the Response

Name	Defined As	Type Description	File name
createLocationRequestResponse	element	createNewInventoryItemResponse	InventoryAPI.wsdl
createLocationResponse	element	CreateLocationResponse	InventoryManagementAPI.xsd
CreateLocationResponse	complexType	networkLocationKey	InventoryManagementAPI.xsd
networkLocationKey	complexType	Contains a list of fields	InventoryManagementEntities.xsd

getAvailablePhysicalPortsRequest Operation

This operation retrieves the available physical port details for the provided network node key.

The following are the request and response structures:

Request Structure: getAvailablePhysicalPortsRequest

Response Structure: getAvailablePhysicalPortsRequestResponse

getAvailablePhysicalPortsRequest

The getAvailablePhysicalPortsRequest element contains the input information for the operation. Each row in Table 6–39 describes the element or type name, the XSD declaration, the type description, and the file name that contains the item's definition.

Table 6–39 Payload Information for the Request

Name	Defined As	Type Description	File Name
getAvailablePhysiclaPortsRequest	element	mimGetAvailablePhysicalPortsRequest	InventoryAPI.wsdl
mimGetAvailablePhysicalPortsRequest	element	GetAvailablePhysicalPortsRequest Value	InventoryManagementAPI.xsd
GetAvailablePhysicalPortsRequestValue	element	getAvailablePhysicalPortsRequestValueType	InventoryManagementAPI.xsd
getAvailablePhysicalPortsRequest ValueType	element	networkNodeKey	InventoryManagementData.xsd
networkNodeKey	complexType	networkNodePrimaryKey	InventoryManagementEntities.xsd

Table 6–40 describes the required fields for networkNodeKey.

Table 6–40 Required Fields for networkNodeKey

Field Name	Data Type	Field Description
networkNodePrimaryKey	string	The network node ID where the physical port details are configured.

getAvailablePhysicalPortsRequestResponse

The getAvailablePhysicalPortsRequestResponse element contains the output information for the operation. The information returned in the response indicates if the operation is successful. Table 6–41 describes the returned information in the response.

Table 6–41 Payload Information for the Response

Name	Defined As	Type Description	File Name
getAvailablePhysicalPortsRequest Response	element	getAvailablePhysicalPortsResponse	InventoryAPI.wsdl
getAvailablePhysicalPortsRespon se	element	physicalPort	InventoryManageme ntAPI.xsd
physicalPort	complexType	Contains the following: ■ Port ■ hasSubPorts	InventoryManageme ntAPI.xsd
Port	complexType	Contains a list of fields	InventoryManageme ntData.xsd
equipmentKey	element	EquipmentKey	InventoryManageme ntData.xsd

Table 6–42 describes the error messages for the operation.

Table 6–42 Error Messages for the Operation

Error Message	Cause	Resolution
Network Node Key is required criteria.	The network node ID is not populated in the input structure.	Populate the network node ID value on the network node key.

getDlrByKeyRequest Operation

This operation retrieves the location information by using the supplied location key.

The following are the request and response structures:

Request Structure: getDlrByKeyRequest

Response Structure: getDlrByKeyRequestResponse

getDlrByKeyRequest

The getDlrByKeyRequest element contains the input information for the operation. Each row in Table 6–43 describes the element or type name, the XSD declaration, the type description, and the file name that contains the item's definition.

Table 6–43 Payload Information for the Request

Name	Defined As	Type Description	File Name
getDlrByKeyRequest	element	mimGetEntityByKeyRequest	InventoryAPI.wsdl
mimGetEntityByKeyRequest	element	getEntityByKeyValueChoice	InventoryManageme ntAPI.xsd
getEntityByKeyValueChoice	complextype	Contains the following: ■ getPSRServiceLocationByKey ■ getDlrByKey	InventoryManageme ntAPI.xsd
getPSRServiceLocationByKey	element	serviceLocationKey	InventoryManageme ntAPI.xsd
serviceLocationKey	complexType	endUserLocationKey	InventoryManageme ntEntities.xsd

Table 6–43 (Cont.) Payload Information for the Request

Name	Defined As	Type Description	File Name
endUserLocationKey	complexType	endUserLocationPrimaryKey	InventoryManagementEntities.xsd
getDlrByKey	element	metaSolvDLRKey	InventoryManagementAPI.xsd
metaSolvDLRKey	element	MetaSolvDLRKey	InventoryManagementEntities.xsd
MetaSolvDLRKey	complexType	Extension of EntityKey Contains: dlrPrimaryKey	InventoryManagementEntities.xsd
dlrPrimaryKey	complexType	DLRPrimaryKey	InventoryManagementEntities.xsd

Table 6–44 describes the required fields for endUserLocationKey.

Table 6–44 Required Fields for endUserLocationKey

Field Name	Data Type	Field Description
endUserLocationPrimaryKey	string	End user location ID of the location.

Table 6–45 describes the required fields for DLRPrimaryKey.

Table 6–45 Required Fields for DLRPrimaryKey

Field Name	Data Type	Field Description
circuitDesignIDNumber	long	Circuit design ID value for which the DLR information is required.
Issue	long	Issue number of the circuit for which the DLR is required. Zero (0) should be passed if all the issue details are required.

getDlrByKeyRequestResponse

The getDlrByKeyRequestResponse element contains the output information for the operation. The information returned in the response indicates if the operation is successful. Table 6–46 describes the returned information in the response.

Table 6–46 Payload Information for the Response

Name	Defined As	Type Description	File Name
getDlrByKeyRequestResponse	element	getEntityByKeyResponse	InventoryAPI.wsdl
getEntityByKeyResponse	element	getEntityByKeyResponseChoice	InventoryManagementAPI.xsd
getEntityByKeyResponseChoice	complexType	Contains the following: ■ getPSRServiceLocationResponse ■ getDlrByKeyResponse	InventoryManagementAPI.xsd
getPSRServiceLocationResponse	complexType	Contains the following: ■ serviceLocationValue ■ addressKey	InventoryManagementAPI.xsd
addressKey	complexType	addressPrimaryKey	InventoryManagementEntities.xsd

Table 6–46 (Cont.) Payload Information for the Response

Name	Defined As	Type Description	File Name
serviceLocationValue	complexType	Contains a list of fields	InventoryManagementEntities.xsd
serviceLocationKey	element	EndUserLocationKey	InventoryManagementEntities.xsd
networkLocationKey	element	networkLocationKey	InventoryManagementEntities.xsd
structureFormats	element	StructureFormatType	InventoryManagementEntities.xsd
getDlrByKeyResponse	element	dlrValue	InventoryManagementAPI.xsd
dlrValue	complexType	Contains a list of fields	InventoryManagementEntities.xsd
dlrKey	element	MetaSolvDLRKey	InventoryManagementEntities.xsd
dlrCircuitInfo	element	DLRCircuitInfo	InventoryManagementEntities.xsd
dlrAdminInfo	element	DLRAdminInfo	InventoryManagementEntities.xsd
dlrDesignInfo	element	DLRDesignInfo	InventoryManagementEntities.xsd
dlrEndUserInfo	element	DLREndUserInfo	InventoryManagementEntities.xsd
designLines	element	DLRDesignLine	InventoryManagementEntities.xsd
Notes	element	DLRNote	InventoryManagementEntities.xsd

Table 6–47 describes the error messages for the operation.

Table 6–47 Error Messages for the Operation

Error Message	Cause	Resolution
DLR Data Not Found for CircuitDesignId=*circuitDesignIdInput* Issue Number=*issueNumberInput*	The DLR data does not exist for the provided input values of circuit design ID and issue number.	Populate the circuit design ID and issue number with values that has DLR data existing in the database.
Circuit design id is not valid	The circuit design ID passed in the input is not valid.	Populate the circuit design ID with a value that is valid and exists in the database.
DLR for circuit issue not found	For the provided circuit design ID, the issue number does not exist.	Populate a valid issue number of the circuit design ID to get the DLR output.

getDlrByOrderKey Operation

This operation retrieves the Circuit Design (DLR) information for an input order key (document number).

The following are the request and response structures:

Request Structure: getDlrByOrderKey

Response Structure: getDlrByOrderKeyResponse

getDlrByOrderKey

The getDlrByOrderKey element contains the input information for the operation. Each row in Table 6–48 describes the element or type name, the XSD declaration, the type description, and the file name that contains the item's definition.

Table 6–48 Payload Information for the Request

Name	Defined As	Type Description	File Name
getDlrByOrderKey	element	getServiceRequestDLRsValue	InventoryAPI.wsdl
getServiceRequestDLRsValue	element	GetServiceRequestDLRsValue	OrderManagementAPI.xsd
GetServiceRequestDLRsValue	complexType	Extension of MomQueryValue Contains the following: ▪ orderKey	OrderManagementAPI.xsd
orderKey	element	OrderKey	OrderManagementAPI.xsd
OrderKey	complexType	primaryKey	OrderManagementEntities.xsd

Table 6–49 describes the required fields for OrderKey.

Table 6–49 Required Fields for OrderKey

Field Name	Data Type	Field Description
primaryKey	string	Document number of the circuit for which the DLR information is required.

getDlrByOrderKeyResponse

The getDlrByOrderKeyResponse element contains the output information for the operation. The information returned in the response indicates if the operation is successful. Table 6–50 describes the returned information in the response.

Table 6–50 Payload Information for the Response

Name	Defined As	Type Description	File Name
getDlrByOrderKeyResponse	element	getServiceRequestDLRsResponse	InventoryAPI.wsdl
getServiceRequestDLRsResponse	complexType	Contains the following: ▪ orderKey ▪ dlrResults	OrderManagementAPI.xsd
orderKey	element	primaryKey	OrderManagementAPI.xsd
dlrResults	element	DLRResult	OrderManagementAPI.xsd
DLRResult	complexType	Contains a list of fields	OrderManagementAPI.xsd
dlrIssues	element	DLRIssue	OrderManagementAPI.xsd
DLRIssue	complexType	Contains a list of fields	InventoryManagementEntities.xsd

Table 6–51 describes the error messages for the operation.

Table 6–51 Error Messages for the Operation

Error Message	Cause	Resolution
Document Number is not populated.	The input value of document number is not populated.	Populate the document number with a valid value.

getEntityByKeyRequest Operation

The getEntityByKeyRequest operation provides the following:

- Retrieving the PSR Service Location details.
- Retrieving the DLR information.

The following are the request and response structures:

Request Structure: getEntityByKeyRequest

Response Structure: getEntityByKeyRequestResponse

getEntityByKeyRequest

The getEntityByKeyRequest element contains the input information for the operation. Each row in Table 6–52 describes the element or type name, the XSD declaration, the type description, and the file name that contains the item's definition.

Table 6–52 Payload Information for the Request

Name	Defined As	Type Description	File Name
getEntityByKeyRequest	element	mimGetEntityByKeyRequest	InventoryAPI.wsdl
mimGetEntityByKeyRequest	element	getEntityByKeyValueChoice	InventoryManagementAPI.xsd
getEntityByKeyValueChoice	complexType	Contains the following: ■ getPSRServiceLocationByKey ■ getDlrByKey	InventoryManagementAPI.xsd
getPSRServiceLocationByKey	complexType	serviceLocationKey	InventoryManagementAPI.xsd
serviceLocationKey	complexType	endUserLocationKey	InventoryManagementEntities.xsd
getDlrByKey	element	metaSolvDLRKey	InventoryManagementAPI.xsd
metaSolvDLRKey	element	MetaSolvDLRKey	InventoryManagementEntities.xsd
MetaSolvDLRKey	complexType	Extension of EntityKey Contains: dlrPrimaryKey	InventoryManagementEntities.xsd
dlrPrimaryKey	complexType	DLRPrimaryKey	InventoryManagementEntities.xsd

Table 6–53 describes the required fields for EndUserLocationKey.

Table 6–53 Required Fields for EndUserLocationKey

Field Name	Data Type	Field Description
endUserLocationPrimaryKey	string	The end user location ID value of the end user location.

Table 6–54 describes the required fields for DLRPrimaryKey.

Table 6–54 Required Fields for DLRPrimaryKey

Field Name	Data Type	Field Description
circuitDesignIDNumber	long	Circuit design ID value for which the DLR information is required.
Issue	long	Issue number of the circuit for which the DLR is required. Zero (0) should be passed if all the issue details are required.

getEntityByKeyRequestResponse

The getEntityByKeyRequestResponse element contains the output information for the operation. The information returned in the response indicates if the operation is successful. Table 6–55 describes the returned information in the response.

Table 6–55 Payload Information for the Response

Name	Defined As	Type Description	File Name
getEntityByKeyRequestResponse	element	getEntityByKeyResponse	InventoryAPI.wsdl
getEntityByKeyResponse	element	getEntityByKeyResponseChoice	InventoryManagementAPI.xsd
getEntityByKeyResponseChoice	complexType	Contains the following: ■ getPSRServiceLocationResponse ■ getDlrByKeyResponse	InventoryManagementAPI.xsd
getPSRServiceLocationResponse	complexType	Contains the following: ■ serviceLocationValue ■ addressKey	InventoryManagementAPI.xsd
addressKey	complexType	addressPrimaryKey	InventoryManagementEntities.xsd
serviceLocationValue	complexType	Contains a list of fields	InventoryManagementEntities.xsd
serviceLocationKey	element	EndUserLocationKey	InventoryManagementEntities.xsd
networkLocationKey	element	networkLocationKey	InventoryManagementEntities.xsd
structureFormats	element	StructureFormatType	InventoryManagementEntities.xsd
getDlrByKeyResponse	element	dlrValue	InventoryManagementAPI.xsd
dlrValue	complexType	Contains a list of fields	InventoryManagementEntities.xsd
dlrKey	element	MetaSolvDLRKey	InventoryManagementEntities.xsd
dlrCircuitInfo	element	DLRCircuitInfo	InventoryManagementEntities.xsd
dlrAdminInfo	element	DLRAdminInfo	InventoryManagementEntities.xsd
dlrDesignInfo	element	DLRDesignInfo	InventoryManagementEntities.xsd

Table 6–55 (Cont.) Payload Information for the Response

Name	Defined As	Type Description	File Name
dlrEndUserInfo	element	DLREndUserInfo	InventoryManagementEntities.xsd
designLines	element	DLRDesignLine	InventoryManagementEntities.xsd
Notes	element	DLRNote	InventoryManagementEntities.xsd

Table 6–56 describes the error messages for the operation.

Table 6–56 Error Messages for the Operation

Error Message	Cause	Resolution
Location data not found for Location Key	The location details does not exist for the provided location ID.	Populate the location ID which has the service location details.
Document Number is not populated.	The input value of document number is not populated.	Populate the document number with a valid value.

getIpAddressesRequest Operation

This operation retrieves IP Addresses based on the criteria specified in the request. You must specify either the network area name or network area key.

The following are the request and response structures:

Request Structure: getIpAddressesRequest

Response Structure: getIpAddressesRequestResponse

getIpAddressesRequest

The getIpAddressesRequest element contains the input information for the operation. Each row in Table 6–57 describes the element or type name, the XSD declaration, the type description, and the file name that contains the item's definition.

Table 6–57 Payload Information for the Request

Name	Defined As	Type Description	File Name
getIpAddressesRequest	element	mimGetIpAddressesRequest	InventoryAPI.wsdl
mimGetIpAddressesRequest	element	GetIpAddressesRequestValue	InventoryManagementAPI.xsd
GetIpAddressesRequestValue	element	IpAddressCriteria	InventoryManagementAPI.xsd
IpAddressCriteria	complexType	Contains the following: ▪ networkArea ▪ ipAddressesValue	InventoryManagementData.xsd
networkArea	complexType	networkAreaKey	InventoryManagementEntities.xsd
networkAreaKey	complexType	networkAreaPrimaryKey	InventoryManagementEntities.xsd
ipAddressesValue	complexType	Contains a list of fields	InventoryManagementData.xsd

Table 6–58 describes the required fields for networkAreaKey.

Table 6–58 Required Fields for networkAreaKey

Field Name	Data Type	Field Description
networkAreaPrimaryKey	string	Identifier or key for the network area.

Table 6–59 describes the required fields for ipAddressesValue.

Table 6–59 Required Fields for ipAddressesValue

Field Name	Data Type	Field Description
IPaddress	string	IP address for which the details are required.
Subnet	string	Defines the portion of the IP address that is the network prefix. The remaining portion of the IP address is the host number. If not populated, then the value is defaulted to 0.
Status	Enum	Valid values of IpAddressStatusEnumType. Assignment status of the IP address. If not defined or value equal to 0, then the search ignores the status.
useGroupCode	string	Used to describe the intended use for IP addresses. If not populated, then the value is defaulted to 0. If value is 0, then the use group is ignored.
useCodeCode	string	Used with use groups to identify intended IP address usage more specifically. If not populated, then the value is defaulted to 0 and the use code is ignored.
retrivalLimit	string	Number of IP Addresses that should be returned. If not defined or value equal to 0, then all IP Addresses are returned.

getIpAddressesRequestResponse

The getIpAddressesRequestResponse element contains the output information for the operation. The information returned in the response indicates if the operation is successful. Table 6–60 describes the returned information in the response.

Table 6–60 Payload Information for the Response

Name	Defined As	Type Description	File Name
getIpAddressesRequestResponse	element	getIpAddressesResponse	InventoryAPI.wsdl
getIpAddressesResponse	element	GetIpAddressesResponseValue	InventoryManagementAPI.xsd
GetIpAddressesResponseValue	element	IpAddressesValue	InventoryManagementAPI.xsd
IpAddressesValue	complexType	Contains a list of fields	InventoryManagementData.xsd

getLocationRequest Operation

This operation retrieves the location information for the input location key.

The following are the request and response structures:

Request Structure: getLocationRequest

Response Structure: getLocationRequestResponse

getLocationRequest

The getLocationRequest element contains the input information for the operation. Each row in Table 6–61 describes the element or type name, the XSD declaration, the type description, and the file name that contains the item's definition.

Table 6–61 Payload Information for the Request

Name	Defined As	Type Description	File Name
getLocationRequest	element	getLocationRequest	InventoryAPI.wsdl
getLocationRequest	element	mimGetLocationRequest	InventoryManagementAPI.xsd
mimGetLocationRequest	element	networkLocationKey	InventoryManagementAPI.xsd
networkLocationKey	complexType	networkLocationPrimaryKey	InventoryManagementEntities.xsd

getLocationRequestResponse

The getLocationRequestResponse element contains the output information for the operation. The information returned in the response indicates if the operation is successful. Table 6–62 describes the returned information in the response.

Table 6–62 Payload Information for the Response

Name	Defined As	Type Description	File Name
getLocationRequestResponse	element	getLocationResponse	InventoryAPI.wsdl
getLocationResponse	element	locationValue	InventoryManagementAPI.xsd
locationValue	element	MimLocationValue	InventoryManagementEntities.xsd
MimLocationValue	complexType	Contains a list of fields	InventoryManagementEntities.xsd
locationType	element	LocationTypeEnumType	InventoryManagementEntities.xsd
networkLocationValue	element	networkLocationValue	InventoryManagementEntities.xsd
endUserLocationValue	element	endUserLocationValue	InventoryManagementEntities.xsd
netLocAddress	element	netLocAddress	InventoryManagementEntities.xsd
caUsageInstance	element	CaUsageInstanceType	InventoryManagementEntities.xsd
metaSolvUserDataValue	element	metaSolvUserDataValue	InventoryManagementEntities.xsd
tandemType	element	TandemType	InventoryManagementEntities.xsd
associatedNetworkArea	element	AssociatedNetworkArea	InventoryManagementEntities.xsd

Table 6–63 describes the error messages for the operation.

Table 6–63 Error Messages for the Operation

Error Message	Cause	Resolution
Location data not found for Location Key	The location details do not exist for the provided location ID.	Populate the location ID with a valid value that has service location details.

getNetworkAreasByGeoAreaRequest Operation

This operation retrieves the network areas given the input geographical area criteria. You must pass the geographic area values in the order of the Geo Area Type (GAT) hierarchy defined for the country in the Utilities application. A GAT hierarchy can consist of up to seven GA types.

The following are the request and response structures:

Request Structure: getNetworkAreasByGeoAreaRequest

Response Structure: getNetworkAreasByGeoAreaRequestResponse

getNetworkAreasByGeoAreaRequest

The getNetworkAreasByGeoAreaRequest element contains the input information for the operation. Each row in Table 6–64 describes the element or type name, the XSD declaration, the type description, and the file name that contains the item's definition.

Table 6–64 Payload Information for the Request

Name	Defined As	Type Description	File Name
getNetworkAreasByGeoAreaRequest	element	mimGetNetworkAreasByGeoAreaRequest	InventoryAPI.wsdl
mimGetNetworkAreasByGeoAreaRequest	element	geoAreaCriteria	InventoryManagementAPI.xsd
geoAreaCriteria	complexType	Contains a list of fields	InventoryManagementEntities.xsd

Table 6–65 describes the required fields for GeoAreaCriteria.

Table 6–65 Required Fields for GeoAreaCriteria

Field Name	Data Type	Field Description
geoAreaValue	string	Value represents the name or abbreviation of a geographic area.
isAbbreviation	boolean	Element indicates if the value defined in geoAreaValue element is abbreviated.

getNetworkAreasByGeoAreaRequestResponse

The getNetworkAreasByGeoAreaRequestResponse element contains the output information for the operation. The information returned in the response indicates if the operation is successful. Table 6–66 describes the returned information in the response.

Table 6–66 Payload Information for the Response

Name	Defined As	Type Description	File Name
getNetworkAreasByGeoAreaRequestResponse	element	getNetworkAreasByGeoAreaResponse	InventoryAPI.wsdl
getNetworkAreasByGeoAreaResponse	element	networkArea	InventoryManagementAPI.xsd
networkArea	complexType	Contains the following: ▪ networkAreaName ▪ networkAreaKey	InventoryManagementEntities.xsd
networkAreaKey	element	networkAreaPrimaryKey	InventoryManagementEntities.xsd

Table 6–67 describes the error messages for the operation.

Table 6–67 Error Messages for the Operation

Error Message	Cause	Resolution
At least one Geographic area is required.	No geographic area details are populated in the input.	Populate any one of the geographic area details in the input.

getNetworkComponentsRequest Operation

This operation retrieves network components based on the network area, component type and component status. The operation only retrieves network elements if the schema element networkElementOnly is set to true. You must either specify the network area name or network area key.

The following are the request and response structures:

Request Structure: getNetworkComponentsRequest

Response Structure: getNetworkComponentsRequestResponse

getNetworkComponentsRequest

The getNetworkComponentsRequest element contains the input information for the operation. Each row in Table 6–68 describes the element or type name, the XSD declaration, the type description, and the file name that contains the item's definition.

Table 6–68 Payload Information for the Request

Name	Defined As	Type Description	File Name
getNetworkComponentsRequest	element	mimGetNetworkComponentsRequest	InventoryAPI.wsdl
mimGetNetworkComponentsRequest	element	GetNetworkComponentsRequestValue	InventoryManagementAPI.xsd
GetNetworkComponentsRequestValue	element	getNetworkComponentsRequestValueType	InventoryManagementEntities.xsd
getNetworkComponentsRequestValueType	complexType	networkArea	InventoryManagementData.xsd
networkArea	element	networkArea	InventoryManagementData.xsd

Table 6–69 describes the required fields for getNetworkComponentsRequestValueType.

Table 6–69 Required Fields for getNetworkComponentsRequestValueType

Field Name	Data Type	Field Description
componentType	string	User-defined category for a network component, such as DSLAM. If populated, must be a valid component type.
componentStatus	Enum	Valid values of ComponentStatusEnumType. Status of the network component.
networkElementsOnly	boolean	Indicates that the query should return only components that are defined as network elements and are contained in a network system. If no value is passed, it defaults to false.

getNetworkComponentsRequestResponse

The getNetworkComponentsRequestResponse element contains the output information for the operation. The information returned in the response indicates if the operation is successful. Table 6–70 describes the returned information in the response.

Table 6–70 Payload Information for the Response

Name	Defined As	Type Description	File Name
getNetworkComponents RequestResponse	element	getNetworkComponentsResponse	InventoryAPI.wsdl
getNetworkComponents Response	element	networkComponent	InventoryManagementAPI.xsd
networkComponent	complexType	Contains a list of fields	InventoryManagementEntities.xsd
componentStatus	element	ComponentStatusEnumType	InventoryManagementEntities.xsd
componentKey	element	componentKey	N/A
componentLocationKey	element	componentLocationKey	N/A
networkNodeKey	element	networkNodeKey	N/A

Table 6–71 describes the error messages for the operation.

Table 6–71 Error Messages for the Operation

Error Message	Cause	Resolution
At least one network area is required.	No network area details are populated in the input.	Populate network area details in the input.
Network area name and network area key are not populated. Either can be populated, but both cannot be blank.	Network area name and key is not populated.	Populate with either a network area name or key or both. At least one value is required.

inventoryAssociationRequest Operation

This operation associates inventory details such as IP Address to IP Address, IP Address to Domain, and Domain to URL. You define the parent and child inventory IDs along with the type of association as input.

The following are the request and response structures:

Request Structure: inventoryAssociationRequest

Response Structure: inventoryAssociationRequestResponse

inventoryAssociationRequest

The inventoryAssociationRequest element contains the input information for the operation. Each row in Table 6–72 describes the element or type name, the XSD declaration, the type description, and the file name that contains the item's definition.

Table 6–72 Payload Information for the Request

Name	Defined As	Type Description	File Name
inventoryAssociationRequest	element	createInventoryAssociationRequest	InventoryAPI.wsdl
createInventoryAssociationRequest	element	importInventoryAssociationType	InventoryManagement API.xsd
importInventoryAssociationType	element	ImportInventoryAssociation	InventoryManagement Data.xsd
ImportInventoryAssociation	complexType	Contains a list of fields	InventoryManagement Data.xsd
parentInventoryID	element	MetaSolvNumberInventoryKey	InventoryManagement Data.xsd
childInventoryID	element	MetaSolvNumberInventoryKey	InventoryManagement Data.xsd

Table 6–73 describes the required fields for ImportInventoryAssociation.

Table 6–73 Required Fields for ImportInventoryAssociation

Field Name	Data Type	Field Description
inventoryRelationTyp eCode	Enum	Valid values of InventoryRelationTypeCodeTypeEnum. Indicates the relation of the inventory items passed.

inventoryAssociationRequestResponse

The inventoryAssociationRequestResponse element contains the output information for the operation. The information returned in the response indicates if the operation is successful. Table 6–74 describes the returned information in the response.

Table 6–74 Payload Information for the Response

Name	Defined As	Type Description	File Name
inventoryAssociationRequestResponse	element	createInventoryAssociationResponse	InventoryAPI.wsdl
createInventoryAssociationResponse	complexType	Status	InventoryManage mentAPI.xsd

processTNRecallRequestDocument Operation

This operation takes the input telephone number and recalls the specified telephone number.

The following are the request and response structures:

Request Structure: processTNRecallRequest

Response Structure: processTNRecallResponse

processTNRecallRequest

The processTNRecallRequest element contains the input information for the operation. Each row in Table 6–75 describes the element or type name, the XSD declaration, the type description, and the file name that contains the item's definition.

Table 6–75 Payload Information for the Request

Name	Defined As	Type Description	File Name
processTNRecallRequestDocument	element	processTNRecallRequest	InventoryAPI.wsdl
processTNRecallRequest	complexType	tn	InventoryManagementAPI.xsd

Table 6–76 describes the required fields for processTNRecallRequest.

Table 6–76 Required Fields for processTNRecallRequest

Field Name	Data Type	Field Description
tn	string	Telephone number for which the TN Recall should be invoked.

processTNRecallResponse

The processTNRecallResponse element contains the output information for the operation. The information returned in the response indicates if the operation is successful. Table 6–77 describes the returned information in the response.

Table 6–77 Payload Information for the Response

Name	Defined As	Type Description	File Name
processTNRecallRequestDocumentResponse	element	processTNRecallResponse	InventoryManagementAPI.xsd
processTNRecallResponse	element	ProcessTNRecallResponseValue	InventoryManagementAPI.xsd
ProcessTNRecallResponseValue	complexType	ProcessTNRecallResponseValueType	InventoryManagementAPI.xsd
ProcessTNRecallResponseValueType	complexType	Contains a list of fields	InventoryManagementData.xsd

Table 6–78 describes the error messages for the operation.

Table 6–78 Error Messages for the Operation

Error Message	Cause	Resolution
Telephone Number {0} is not in the inventory	The telephone number does not exist in the inventory.	Populate the TN which exist in the inventory for RECALL.

queryEndUserLocation Operation

This operation requests end user location for specific criteria and limiting the number of records returned.

The following are the request and response structures:

Request Structure: queryEndUserLocation

Response Structure: queryEndUserLocationResponse

queryEndUserLocation

The queryEndUserLocation element contains the input information for the operation. Each row in Table 6–79 describes the element or type name, the XSD declaration, the type description, and the file name that contains the item's definition.

Table 6–79 Payload Information for the Request

Name	Defined As	Type Description	File Name
queryEndUserLocation	element	queryEndUserLocationRequest	InventoryAPI.wsdl
queryEndUserLocationRequest	complexType	queryEndUserLocationRequestValue	InventoryManagementAPI.xsd
queryEndUserLocationRequestValue	element	EndUserLocationQueryValue	InventoryManagementData.xsd
EndUserLocationQueryValue	complexType	sfAddress	InventoryManagementEntities.xsd
sfAddress	element	StructureFormatType	InventoryManagementEntities.xsd
StructureFormatType	complexType	StructureFormatComponentType	CustomerManagementData.xsd
StructureFormatComponentType	complexType	Contains a list of fields	CustomerManagementData.xsd

Table 6–80 describes the required fields for EndUserLocationQueryValue.

Table 6–80 Required Fields for EndUserLocationQueryValue

Field Name	Data Type	Field Description
networkLocationName	string	Name for a location.
enduserLocationName	string	Name of the end user location where the circuit is being provided.
countryId	long	Unique ID implemented with an Oracle sequence that identifies a GA instance for a structure format.
custAcctNbr	string	Customer account number.

queryEndUserLocationResponse

The queryEndUserLocationResponse element contains the output information for the operation. The information returned in the response indicates if the operation is successful. Table 6–81 describes the returned information in the response.

Table 6–81 Payload Information for the Response

Name	Defined As	Type Description	File Name
queryEndUserLocationResponse	element	mimQueryEndUserLocationResponse	InventoryAPI.wsdl
mimQueryEndUserLocationResponse	element	queryEndUserLocationResponseValue	InventoryManagementAPI.xsd
queryEndUserLocationResponseValue	element	EndUserLocationResultValue	InventoryManagementData.xsd
EndUserLocationResultValue	complexType	Contains a list of fields	InventoryManagementEntities.xsd

Table 6–82 describes the error messages for the operation.

Table 6–82 Error Messages for the Operation

Error Message	Cause	Resolution
Country not found in the database.	The country passed as input does not exist in the database.	Populate the country which exists in the database.
Cannot have a structure format address and the address lines or country id populated on the same query.	Both structure format and country ID has been populated in the input.	Either one of the structured format or country ID has to be populated.

queryInventoryManagementRequest Operation

This operation retrieves the inventory for queryTelephoneNumberInventoryValue and queryMSAGInventoryValue.

The following are the request and response structures:

Request Structure: queryInventoryManagementRequest

Response Structure: queryInventoryManagementRequestResponse

queryInventoryManagementRequest

The queryInventoryManagementRequest element contains the input information for the operation. Each row in Table 6–83 describes the element or type name, the XSD declaration, the type description, and the file name that contains the item's definition.

Table 6–83 Payload Information for the Request

Name	Defined As	Type Description	File Name
queryInventoryManagementRequest	element	mimQueryInventoryManagementRequest	InventoryAPI.wsdl
mimQueryInventoryManagementRequest	complexType	MimQueryValue	InventoryManagementAPI.xsd
MimQueryValue	element	MetaSolvInventoryQueryValueChoice	InventoryManagementAPI.xsd
MetaSolvInventoryQueryValueChoice	complexType	Contains the following: ▪ queryTelephoneNumberInventoryValue ▪ queryMSAGInventoryValue	InventoryManagementAPI.xsd
queryTelephoneNumberInventoryValue	complexType	telephoneNumberSearchCritera	InventoryManagementAPI.xsd
telephoneNumberSearchCritera	element	MetaSolvTelephoneNumberSearchCriteria	InventoryManagementEntities.xsd
MetaSolvTelephoneNumberSearchCriteria	complexType	Contains the following: ▪ strucFmt ▪ customerAccountKey ▪ statusSeq ▪ telnbrTypeCd	InventoryManagementEntities.xsd
strucFmt	element	StructureFormatType	InventoryManagementEntities.xsd
queryMSAGInventoryValue	complexType	msagSearch	InventoryManagementAPI.xsd

Table 6–83 (Cont.) Payload Information for the Request

Name	Defined As	Type Description	File Name
msagSearch	element	MetaSolvMSAGSearch	InventoryManagementEntities.xsd
MetaSolvMSAGSearch	complexType	StructureFormatType	InventoryManagementEntities.xsd
StructureFormatType	complexType	StructureFormatComponentType	CustomerManagementData.xsd
StructureFormatComponentType	complexType	Contains a list of fields	CustomerManagementData.xsd
customerAccountKey	complexType	customerAccountPrimaryKey	CustomerManagementEntities.xsd
statusSeq	element	TelephoneNumberStatusEnumArray	InventoryManagementEntities.xsd
TelephoneNumberStatusEnumArray	complexType	telephoneNumberStatus	InventoryManagementData.xsd
telephoneNumberStatus	element	TelephoneNumberStatusEnumType	InventoryManagementData.xsd
telnbrTypeCd	element	TelephoneNumberTypeEnumArray	InventoryManagementEntities.xsd
TelephoneNumberTypeEnumArray	complexType	telephoneNumberType	InventoryManagementData.xsd
telephoneNumberType	element	TelephoneNumberTypeEnumType	InventoryManagementData.xsd

Table 6–84 describes the required fields for MetaSolvTelephoneNumberSearchCriteria.

Table 6–84 Required Fields for MetaSolvTelephoneNumberSearchCriteria

Field Name	Data Type	Field Description
inventorySubTypeCd	int	Inventory sub type code.
locationCode	string	Location identification code that identifies specific locations or terminations.
networkAreaCity	int	Oracle sequence that uniquely identifies a GA instance for a city. It enables the ability to tie a network area to an actual city.
networkAreaStateProv	int	Oracle sequence that uniquely identifies a GA instance for a state.
networkAreaCountry	int	Oracle sequence that uniquely identifies a GA instance for a country.
tniCategoryId	int	Telephone number category ID.
tniSubcategoryId	int	Telephone number sub category ID.

Table 6–85 describes the required fields for MetaSolvMSAGSearch.

Table 6–85 Required Fields for MetaSolvMSAGSearch

Field Name	Data Type	Field Description
keyValue	string	Specifies whether the Master Street Address Guide (MSAG) information uses the NENA 2.0 standards.
partnerId	string	Partner ID.

Table 6–86 describes the required fields for StructureFormatType.

Table 6–86 Required Fields for StructureFormatType

Field Name	Data Type	Field Description
sfType	string	Category that describes a structure.
name	string	Name of a specific customizing of a structured format type.

Table 6–87 describes the required fields for StructureFormatComponentType.

Table 6–87 Required Fields for StructureFormatComponentType

Field Name	Data Type	Field Description
Id	int	ID of a component of a structured format.
name	string	Name of a component of a structured format.
componentType	string	Type of component, such as table-driven drop-down or valid value drop-down.
Value	string	Name of a value for a component.

Table 6–88 describes the required fields for customerAccountKey.

Table 6–88 Required Fields for customerAccountKey

Field Name	Data Type	Field Description
MetaSolvCustom customerAccountPrimaryKey	string	Customer account primary key value which is the customer account number.

Table 6–89 describes the required fields for TelephoneNumberStatusEnumArray.

Table 6–89 Required Fields for TelephoneNumberStatusEnumArray

Field Name	Data Type	Field Description
telephoneNumberStatus	Enum	Valid values of TelephoneNumberStatusEnumType.

Table 6–90 describes the required fields for TelephoneNumberTypeEnumArray.

Table 6–90 Required Fields for TelephoneNumberTypeEnumArray

Field Name	Data Type	Field Description
telephoneNumberType	Enum	Valid values of TelephoneNumberTypeEnumType.

queryInventoryManagementRequestResponse

The queryInventoryManagementRequestResponse element contains the output information for the operation. The information returned in the response indicates if the operation is successful. Table 6–91 describes the returned information in the response.

Table 6–91 Payload Information for the Response

Name	Defined As	Type Description	File Name
queryInventoryManagementRequestResponse	element	queryInventoryManagementResponse	InventoryAPI.wsdl
queryInventoryManagementResponse	element	MimQueryResponse	InventoryManagementAPI.xsd
MimQueryResponse	element	MetaSolvInventoryQueryResponseChoice	InventoryManagementAPI.xsd
MetaSolvInventoryQueryResponseChoice	complexType	Contains the following: ■ queryTelephoneNumberInventoryResponse ■ queryMSAGResponse	InventoryManagementAPI.xsd
QueryTelephoneNumberInventoryResponse	complexType	Contains the following: ■ telephoneNumberArray ■ checkOutId	InventoryManagementAPI.xsd
telephoneNumberArray	element	TelephoneNumberValueArray	InventoryManagementEntities.xsd
TelephoneNumberValueArray	element	item	InventoryManagementEntities.xsd
Item	element	MetaSolvTelephoneNumberValue	InventoryManagementEntities.xsd
MetaSolvTelephoneNumberValue	complexType	Contains a list of fields	InventoryManagementEntities.xsd
metaSolvNumberInventoryKey	element	metaSolvNumberInventoryKey	InventoryManagementEntities.xsd
customerAccountKey	element	MetaSolvCustomerAccountKey	InventoryManagementEntities.xsd
structureFormat	element	StructureFormatType	InventoryManagementEntities.xsd
Reservation	element	TelephoneNumberReservationType	InventoryManagementEntities.xsd
QueryMSAGResponse	complexType	Contains the following: ■ msagAddress ■ errorMessages	InventoryManagementAPI.xsd
msagAddress	element	MetaSolvMSAGAddress	InventoryManagementEntities.xsd
MetaSolvMSAGAddress	element	MSAGAddressType	InventoryManagementEntities.xsd
MSAGAddressType	complexType	Contains a list of fields	InventoryManagementData.xsd

Table 6–92 describes the error messages for the operation.

Table 6–92 Error Messages for the Operation

Error Message	Cause	Resolution
Status, Customer Id or Reservation is required.	Status, customer ID or reservation is not populated.	Populate either status, customer ID or reservation in the input structure.
Inventory subtype code is required.	Inventory subtype code is not populated.	Populate Inventory subtype code in the input structure.
Miscellaneous Error: No Network Preference set up.	EnableNetworkAreas preference is not set.	Enable the EnableNetworkAreas preference.

queryNetworkLocation Operation

This operation requests network locations for specific criteria and limits the number of records returned.

The following are the request and response structures:

Request Structure: queryNetworkLocation

Response Structure: queryNetworkLocationResponse

queryNetworkLocation

The queryNetworkLocation element contains the input information for the operation. Each row in Table 6–93 describes the element or type name, the XSD declaration, the type description, and the file name that contains the item's definition.

Table 6–93 Payload Information for the Request

Name	Defined As	Type Description	File Name
queryNetworkLocation	element	queryNetworkLocationRequest	InventoryAPI.wsdl
queryNetworkLocationRequest	complexType	queryNetworkLocationRequestValue	InventoryManagementAPI.xsd
queryNetworkLocationRequestValue	element	NetworkLocationQueryValue	InventoryManagementData.xsd
NetworkLocationQueryValue	complexType	sfAddress	InventoryManagementEntities.xsd
StructureFormatType	complexType	StructureFormatComponentType	CustomerManagementData.xsd
StructureFormatComponentType	complexType	Contains a list of fields	CustomerManagementData.xsd

Table 6–94 describes the required fields for NetworkLocationQueryValue.

Table 6–94 Required Fields for NetworkLocationQueryValue

Field Name	Data Type	Field Description
locationCodeFormat	long	Indicates the industry format used to create the location code.
networkLocationCode	string	Location identification code that identifies specific locations or terminations. This code may be free-form and user-defined, or the Common Language Location Identification (CLLI) code administered by Telcordia.
networkLocationName	string	Name for a location. This name often appears in the Description field of the Telcordia CLLI practice.

Table 6–94 (Cont.) Required Fields for NetworkLocationQueryValue

Field Name	Data Type	Field Description
networkLocationTypeId	long	Unique identifier sequence number that identifies the Network Location Type.
countryId	long	Unique ID implemented with an Oracle sequence that identifies a GA instance for a structure format. This is required if the query is done using Country ID. This is not required if the search uses structure format.
version	string	Used to convert the container to the appropriate generated structure class. Currently not used and defaulted to 1.
maxRow	string	Number of End User Location that should be returned. If not defined or the value is equal to 0, then all End User Locations are returned.

queryNetworkLocationResponse

The queryNetworkLocationResponse element contains the output information for the operation. The information returned in the response indicates if the operation is successful. Table 6–95 describes the returned information in the response.

Table 6–95 Payload Information for the Response

Name	Defined As	Type Description	File Name
queryNetworkLocationResponse	element	mimQueryNetworkLocationResponse	InventoryAPI.wsdl
mimQueryNetworkLocationResponse	element	queryNetworkLocationResponseValue	InventoryManagementAPI.xsd
queryNetworkLocationResponseValue	element	NetworkLocationResultValue	InventoryManagementData.xsd
NetworkLocationResultValue	complexType	Contains a list of fields	InventoryManagementEntities.xsd
sfAddress	element	StructureFormatType	InventoryManagementEntities.xsd

processtnValidationRequest Operation

This operation validates an input telephone number including the additional input as criteria.

The following are the request and response structures:

Request Structure: processTNValidationRequest

Response Structure: processTNValidationResponse

processTNValidationRequest

The processTNValidationRequest element contains the input information for the operation. Each row in Table 6–96 describes the element or type name, the XSD declaration, the type description, and the file name that contains the item's definition.

Table 6–96 Payload Information for the Request

Name	Defined As	Type Description	File Name
tnValidationRequest	element	processTNValidationRequest	InventoryAPI.wsdl
processTNValidationRequest	complexType	Contains a list of fields	InventoryManagementAPI.xsd

Table 6–97 describes the required fields for processTNValidationRequest.

Table 6–97 Required Fields for processTNValidationRequest

Field Name	Data Type	Field Description
tnNumber	string	Telephone number for which the validation is required.
validationType	Enum	Valid values of TNValidationTypeEnumType.
activityType	Enum	Valid values of TNActivityTypeEnumType.
allowICP	boolean	Indicates whether to allow the ICP or not.

processTNValidationResponse

The processTNValidationResponse element contains the output information for the operation. The information returned in the response indicates if the operation is successful. Table 6–98 describes the returned information in the response.

Table 6–98 Payload Information for the Response

Name	Defined As	Type Description	File Name
tnValidationRequestResponse	element	processTNValidationResponse	InventoryAPI.wsdl
processTNValidationResponse	complexType	ProcessTNValidationResponseValue	InventoryManagementAPI.xsd

Table 6–99 describes the error messages for the operation.

Table 6–99 Error Messages for the Operation

Error Message	Cause	Resolution
The Telephone Number is incomplete.	The populated telephone number is not complete number.	Populate a valid telephone number in the input structure.
The Telephone Number activity is invalid.	The activity populated is not valid.	Populate a valid TN activity in the input structure.

updateEntityByValueRequest Operation

This operation updates the inventory details for the following:

- The attributes of the location that are populated (updatePSRServiceLocationValue)
- Releases the previously reserved telephone numbers (updatePreAssignTelephoneNumberValue)

This operation only updates the attributes that are populated.

The following are the request and response structures:

Request Structure: updateEntityByValueRequest

Response Structure: updateEntityByValueRequestResponse

updateEntityByValueRequest

The updateEntityByValueRequest element contains the input information for the operation. Each row in Table 6–100 describes the element or type name, the XSD declaration, the type description, and the file name that contains the item's definition.

Table 6–100 Payload Information for the Request

Name	Defined As	Type Description	File Name
updateEntityByValueRequest	element	mimUpdateEntityByValueRequest	InventoryAPI.wsdl
mimUpdateEntityByValueRequest	element	updateEntityValue	InventoryManagementAPI.xsd
updateEntityValue	element	UpdateEntityValueChoice	InventoryManagementAPI.xsd
UpdateEntityValueChoice	complexType	Contains the following: ■ updatePSRServiceLocationValue ■ updatePreAssignTelephoneNumberValue	InventoryManagementAPI.xsd
updatePSRServiceLocationValue	element	serviceLocationValue	InventoryManagementAPI.xsd
serviceLocationValue	complexType	Contains the following: ■ serviceLocationKey ■ networkLocationKey ■ updateAddressMode ■ structureFormats	InventoryManagementEntities.xsd
serviceLocationKey	element	EndUserLocationKey	InventoryManagementEntities.xsd
networkLocationKey	element	networkLocationKey	InventoryManagementEntities.xsd
updateAddressMode	element	UpdateModeEnumType	InventoryManagementEntities.xsd
structureFormats	element	StructureFormatType	InventoryManagementEntities.xsd
updatePreAssignTelephoneNumberValue	element	UpdatePreAssignTelephoneNumberValue	InventoryManagementAPI.xsd
UpdatePreAssignTelephoneNumberValue	complexType	preAssignTelephoneNumberValue	InventoryManagementAPI.xsd
preAssignTelephoneNumberValue	element	PreAssignTelephoneNumberValue	InventoryManagementAPI.xsd
PreAssignTelephoneNumberValue	complexType	Extension of ResourceValue Contains a list of fields and the following: metaSolvNumberInventoryKey	InventoryManagementEntities.xsd
metaSolvNumberInventoryKey	complexType	See Table 6–13, " Required Fields for metaSolvNumberInventoryKey".	InventoryManagementEntities.xsd

Table 6–101 describes the required fields for EndUserLocationKey.

Table 6–101 Required Fields for EndUserLocationKey

Field Name	Data Type	Field Description
endUserLocationPrimaryKey	string	The end user location ID value of the end user location.

Table 6–102 describes the required fields for NetworkLocationKey.

Table 6–102 Required Fields for NetworkLocationKey

Field Name	Data Type	Field Description
networkLocationPrimaryKey	string	The network location ID value of the network location.

Table 6–103 describes the required fields for StructureFormatType.

Table 6–103 Required Fields for StructureFormatType

Field Name	Data Type	Field Description
sfType	string	Category that describes a structure.
name	string	Name of a specific customizing of a structured format type.

Table 6–104 describes the required fields for PreAssignTelephoneNumberValue.

Table 6–104 Required Fields for PreAssignTelephoneNumberValue

Field Name	Data Type	Field Description
inventorySubTypeCd	int	The code that identifies the inventory item type subtype.
checkOutId	int	System generated number that is set by the operation when a number is exported.
quantity	int	Number to range from the starting point.
releaseIndicator	boolean	Specifies if these telephone numbers should be released.
activityIndicator	Enum	Valid values of TelephoneNumberActivityEnumType. Activity code set by the operation to pre-assign or flag Number Inventories (for instance, telephone numbers and email addresses) that are used in a subsequent operation order import process.

updateEntityByValueRequestResponse

The updateEntityByValueRequestResponse element contains the output information for the operation. The information returned in the response indicates if the operation is successful. Table 6–105 describes the returned information in the response.

Table 6–105 Payload Information for the Response

Name	Defined As	Type Description	File Name
updateEntityByValueRequestResponse	element	updateEntityByValueResponse	InventoryAPI.wsdl
updateEntityByValueResponse	element	updateEntityResponse	InventoryManagementAPI.xsd
updateEntityResponse	complextype	Contains the following: ▪ updatePSRServiceLocationResponse ▪ updatePreAssignTelephoneNumberResponse	InventoryManagementAPI.xsd
updatePSRServiceLocationResponse	complexType	Contains the following: ▪ locationKey ▪ addressKey	InventoryManagementAPI.xsd

Table 6–105 (Cont.) Payload Information for the Response

Name	Defined As	Type Description	File Name
locationKey	element	MetaSolvLocationKeyChoice	InventoryManagementEntities.xsd
MetaSolvLocationKeyChoice	complexType	Contains the following: ▪ serviceLocationKey ▪ endUserLocationKey ▪ networkLocationKey	InventoryManagementEntities.xsd
ServiceLocationKey	element	ServiceLocationKey	InventoryManagementEntities.xsd
ServiceLocationKey	complexType	EndUserLocationKey	InventoryManagementEntities.xsd
EndUserLocationKey	complexType	Contains a list of fields	InventoryManagementEntities.xsd
endUserLocationKey	element	EndUserLocationKey	InventoryManagementEntities.xsd
EndUserLocationKey	complexType	Contains a list of fields	InventoryManagementEntities.xsd
networkLocationKey	element	NetworkLocationKey	InventoryManagementEntities.xsd
NetworkLocationKey	complexType	Contains a list of fields	InventoryManagementEntities.xsd
addressKey	element	AddressKey	InventoryManagementEntities.xsd
AddressKey	complexType	Contains a list of fields	InventoryManagementEntities.xsd
updatePreAssignTelephoneNumberResponse	complexType	telephoneNumberArray	InventoryManagementAPI.xsd
telephoneNumberArray	element	TelephoneNumberValueArray	InventoryManagementEntities.xsd
TelephoneNumberValueArray	complexType	MetaSolvTelephoneNumberValue	InventoryManagementEntities.xsd
MetaSolvTelephoneNumberValue	complexType	Contains a list of fields	InventoryManagementEntities.xsd
metaSolvNumberInventoryKey	element	metaSolvNumberInventoryKey	InventoryManagementEntities.xsd
customerAccountKey	element	MetaSolvCustomerAccountKey	InventoryManagementEntities.xsd
structureFormat	element	StructureFormatType	InventoryManagementEntities.xsd
Reservation	element	TelephoneNumberReservationType	InventoryManagementEntities.xsd

updateLocationRequest Operation

This operation changes the attributes of a location. The operation only updates the attributes that are populated.

The following are the request and response structures:

Request Structure: updateLocationRequest

Response Structure: updateLocationRequestResponse

updateLocationRequest

The updateLocationRequest element contains the input information for the operation. Each row in Table 6–106 describes the element or type name, the XSD declaration, the type description, and the file name that contains the item's definition.

Table 6–106 Payload Information for the Request

Name	Defined As	Type Description	File Name
updateLocationRequest	element	updateLocationRequest	InventoryAPI.wsdl
updateLocationRequest	element	mimUpdateLocationRequest	InventoryManagementAPI.xsd
mimUpdateLocationRequest	element	locationValue	InventoryManagementAPI.xsd
locationValue	element	MimLocationValue	InventoryManagementEntities.xsd
MimLocationValue	complexType	Contains the following: ■ NetworkLocationValue ■ EndUserLocationValue ■ NetLocAddress ■ CaUsageInstanceType ■ metaSolvUserDataValue ■ TandemType ■ AssociatedNetworkArea	InventoryManagementEntities.xsd
networkLocationValue	element	NetworkLocationValue	InventoryManagementEntities.xsd
endUserLocationValue	element	endUserLocationValue	InventoryManagementEntities.xsd
netLocAddress	element	netLocAddress	InventoryManagementEntities.xsd
caUsageInstance	element	CaUsageInstanceType	InventoryManagementEntities.xsd
metaSolvUserDataValue	element	metaSolvUserDataValue	InventoryManagementEntities.xsd
tandemType	element	TandemType	InventoryManagementEntities.xsd
associatedNetworkArea	element	AssociatedNetworkArea	InventoryManagementEntities.xsd

updateLocationRequestResponse

The updateLocationRequestResponse element contains the output information for the operation. The information returned in the response indicates if the operation is successful. Table 6–107 describes the returned information in the response.

Table 6–107 Payload Information for the Response

Name	Defined As	Type Description	File Name
updateLocationRequestResponse	element	updateLocationResponse	InventoryAPI.wsdl
updateLocationResponse	element	UpdateLocationResponse	InventoryManagementAPI.xsd
UpdateLocationResponse	complexType	networkLocationKey	InventoryManagementAPI.xsd
networkLocationKey	complexType	Contains a list of fields	InventoryManagementEntities.xsd

updateTNRequest Operation

This operation updates the telephone number.

The following are the request and response structures:

Request Structure: updateTNRequest

Response Structure: updateTNRequestResponse

updateTNRequest

The updateTNRequest element contains the input information for the operation. Each row in Table 6–108 describes the element or type name, the XSD declaration, the type description, and the file name that contains the item's definition.

Table 6–108 Payload Information for the Request

Name	Defined As	Type Description	File Name
updateTNRequest	element	mimUpdateTNRequest	InventoryAPI.wsdl
mimUpdateTNRequest	element	UpdateTNRequestValue	InventoryManagementAPI.xsd
UpdateTNRequestValue	element	MetaSolvTelephoneNumberValue	InventoryManagementData.xsd
MetaSolvTelephoneNumberValue	complexType	Contains the following: ▪ metaSolvNumberInventoryKey ▪ customerAccountKey ▪ structureFormat ▪ reservation	InventoryManagementEntities.xsd
metaSolvNumberInventoryKey	element	metaSolvNumberInventoryKey	InventoryManagementEntities.xsd
customerAccountKey	element	MetaSolvCustomerAccountKey	InventoryManagementEntities.xsd
structureFormat	element	StructureFormatType	InventoryManagementEntities.xsd
reservation	element	TelephoneNumberReservationType	InventoryManagementEntities.xsd
telephoneNumberStatus	element	TelephoneNumberStatusEnumType	InventoryManagementEntities.xsd
telephoneNumberTypeCode	element	TelephoneNumberTypeEnumType	InventoryManagementEntities.xsd

Table 6–109 describes the required fields for MetaSolvTelephoneNumberValue.

Table 6–109 Required Fields for MetaSolvTelephoneNumberValue

Field Name	Data Type	Field Description
countryCode	string	Identifies the country code for a telephone number. This is different from the postal country code on the country table. It defaults from a preference.
telephoneNumberSuf	string	Used when numbers are in high demand and limited supply. This allows for faster turn around time on reuse of a number as toll can be tracked down to this PIN.

Table 6–109 (Cont.) Required Fields for MetaSolvTelephoneNumberValue

Field Name	Data Type	Field Description
telephoneNumberStatus	Enum	Valid values of TelephoneNumberStatusEnumType. This gives information about the state of a telephone number.
telephoneNumberTypeCode	Enum	Valid values of TelephoneNumberTypeEnumType. Determines the type of telephone number.
locationId	long	Required code for identifying a network location.

Table 6–110 defines the reservation element.

Table 6–110 reservation Definition

Name	Defined As	Type Description	File Name
reservation	element	TelephoneNumberReservationType	InventoryManagementData.xsd
TelephoneNumberReservationType	complexType	Contains the following: ▪ telephoneNumberReservationKey ▪ customerAccountKey	InventoryManagementData.xsd
TelephoneNumberReservationKey	complexType	reservationKey	InventoryManagementData.xsd
customerAccountKey	element	MetaSolvCustomerAccountKey	InventoryManagementData.xsd
MetaSolvCustomerAccountKey	complexType	customerAccountPrimaryKey	CustomerManagementEntities.xsd

Table 6–111 describes the required fields for reservation.

Table 6–111 Required Fields for reservation

Field Name	Data Type	Field Description
qtyOfWtn	int	Number of telephone numbers that are reserved on this telephone number reservation.
reservationNbr	int	Identifier of the reservation that is meaningful for the customer/user reserving the phone numbers.
reservationExpiredInd	YesNoIndicatorType	If a telephone number reservation has expired, then set this indicator to Y. The other valid value is the default N.
endUserLoc	string	Free form text area where the customer reserving the telephone numbers can indicate an end user location for which the numbers are being reserved.
reservedReason	string	Indicates why this telephone number is reserved. For example, it can be reserved because it is a vanity number, or because a customer has requested it.
requestedByName	string	Person who initiated the request for the customer, such as the salesperson.

Table 6–112 describes the required fields for TelephoneNumberReservationKey.

Table 6–112 Required Fields for TelephoneNumberReservationKey

Field Name	Data Type	Field Description
reservationKey	integer	Reservation key.

updateTNRequestResponse

The updateTNRequestResponse element contains the output information for the operation. The information returned in the response indicates if the operation is successful. Table 6–113 describes the returned information in the response.

Table 6–113 Payload Information for the Response

Name	Defined As	Type Description	File Name
updateTNRequestResponse	element	updateTNResponse	InventoryAPI.wsdl
updateTNResponse	complexType	UpdateTNReponseValue	InventoryManagementAPI.xsd

7 Order Web Service Reference

This chapter provides information about Oracle Communications MetaSolv Solution (MSS) Order Web Service.

About the Order Web Service

The Order Web Service enables an external system to import and maintain order information in MSS. Order Web Service operations enable you to:

- Create, get, update, and finish an order.
- Complete, reopen, and a transfer task for an order.
- Assign a provisioning plan to an order.
- Process a supplement order.
- Get and update CNAM/LIDB and E911 data.

  The following summarizes the CNAM/LIDB and E911 data terms:

  - Calling Name (CNAM) is a telephone service used to display the name and telephone number of the caller.
  - Line Database (LIDB) is a telephone service used to verify a telephone number for toll service and third-party billing, such as for validation of calling card numbers.
  - E911 is a telephone service used to provide emergency (911) operators with the caller's telephone number and location.

About the Order Web Service Packaging

The Order Web Service is packaged in the **MSS_WebService.ear** file, which contains the **order.war** file. When the installer deploys the **EAR** file, the Order Web Service is automatically deployed and ready to use.

Note: The **MSS_WebService.ear** file also includes the other MSS web service operations. See Chapter 1, "Web Services Overview" for information about these operations.

About the Order WSDL, WAR, and Schema Files

The Order Web Service is defined by the **OrderAPI.wsdl** file and is supported by numerous schema files. The WSDL file and supporting schema files are located in the **order.war** file.

See "Understanding How MSS Defines Web Services" for more information about WSDL and WAR files, and about their directory locations in the EAR file.

About Order Schema Files

Numerous schema files support the Order Web Service. Within the **order.war** file, the schema files are located in the **WEB-INF/wsdls** directory. These schemas are categorized as common schemas, entity schemas, and data schemas, and API schemas.

Common Schemas

For information about the common schema files, see "About Schema Files".

Entity Schemas

The entity schemas define elements, such as keys and types, specific to the web service.

The Order entity schemas are defined in the following files:

- CustomerManagementEntities.xsd
- InventoryManagementEntities.xsd
- MIPCommonEntities.xsd
- OrderManagementEntities.xsd
- ServiceEntities.xsd

Data Schemas

The data schemas contain numerous complex type structures, enumerations, and simple types.

The Order data schemas are defined in the following files:

- CustomerManagementData.xsd
- InventoryManagementData.xsd
- OrderAncillaryManagementData.xsd
- OrderManagementData.xsd
- ServiceData.xsd
- ServiceOrderData.xsd

API Schemas

The API schemas contain the high-level response and request type definitions and exception definitions.

The Order API schemas are defined in the following files:

- InventoryManagementAPI.xsd
- OrderAPI.wsdl

- OrderManagementAPI.xsd

addTaskRequest Operation

The addTaskRequest operation enables external systems to add a task to an existing provisioning plan assigned order which is not completed.

The following are the request and response structures:

Request Structure: addTaskRequest

Response Structure: addTaskRequestResponse

addTaskRequest

The addTaskRequest element contains the input information for the operation. Each row in Table 7–1 describes the element or type name, the XSD declaration, the type description, and the file name that contains the item's definition.

Table 7–1 Payload Information for the Request

Name	Defined As	Type Description	File Name
addTaskRequest	element	momAddTaskRequest	OrderAPI.wsdl
momAddTaskRequest	element	addTaskRequestValue	OrderManagementAPI.xsd
addTaskRequestValue	element	AddTaskRequestValueType	OrderManagementAPI.xsd
AddTaskRequestValueType	ComplexType	ComplexType with task, taskKey, addTaskPolicy	OrderManagementData.xsd
task	element	Task	OrderManagementData.xsd
Task	ComplexType	ComplexType with a list of fields	OrderManagementEntities.xsd
taskKey	element	TaskOrderKey	OrderManagementData.xsd
TaskOrderKey	ComplexType	ComplexType with a list of fields	OrderManagementEntities.xsd
addTaskPolicy	element	AddTaskPolicy	OrderManagementData.xsd
AddTaskPolicy	ComplexType	ComplexType with a list of fields	OrderManagementEntities.xsd

Table 7–2 describes the required fields for addTaskRequest.

Table 7–2 Required Fields

File Name	Data Type	Field Description
primaryKey	string	A system-assigned identifier for the document number. Value should be zero for creating new order.

addTaskRequestResponse

The addTaskRequestResponse element contains the output information for the operation. The information returned in the response indicates if the operation is successful. Table 7–3 describes the returned information in the response.

Table 7–3 Payload Information for the Request

Name	Defined As	Type Description	File Name
addTaskRequestResponse	element	addTaskResponse	OrderAPI.wsdl
addTaskResponse	element	addTaskResponseValue	OrderManagementAPI.xsd
addTaskResponseValue	element	AddTaskResponseValueType	OrderManagementAPI.xsd
AddTaskResponseValueType	ComplexType	ComplexType with task, planKey, planDefinition	OrderManagementData.xsd
task	element	Task	OrderManagementData.xsd
Task	ComplexType	ComplexType with a list of fields	OrderManagementEntities.xsd
planKey	element	PlanKey	OrderManagementData.xsd
PlanKey	ComplexType	ComplexType with a list of fields	OrderManagementEntities.xsd
planDefinition	element	PlanDefinition	OrderManagementData.xsd
PlanDefinition	ComplexType	ComplexType with a list of fields	OrderManagementEntities.xsd

addTaskJeopardyRequest Operation

This operation enables external systems to add jeopardy information for a task. Adding jeopardy information with reason codes identifies a task as being at risk of completing late and specifies the reason for the risk.

The following are the request and response structures:

Request Structure: addTaskJeopardyRequest

Response Structure: addTaskJeopardyResponse

addTaskJeopardyRequest

The addTaskJeopardyRequest element contains the input information for the operation. Each row in Table 7–4 describes the element or type name, the XSD declaration, the type description, and the file name that contains the item’s definition.

Table 7–4 Payload Information for the Request

Name	Defined As	Type Description	File Name
addTaskJeopardyRequest	element	momAddTaskJeopardyRequest	OrderAPI.wsdl
momAddTaskJeopardyRequest	element	addTaskJeopardyRequestValue	OrderManagementAPI.xsd
addTaskJeopardyRequestValue	element	AddTaskJeopardyRequestValueType	OrderManagementAPI.xsd
AddTaskJeopardyRequestValueType	complexType	Contains a list of fields	OrderManagementData.xsd

Table 7–5 describes the required fields for addTaskJeopardyRequest.

Table 7–5 Required Fields

Field Name	Data Type	Field Description
jeopId	int	Oracle-generated sequence that uniquely identifies the object in the MSS database.
docNum	OrderKey	System-assigned identifier for the document number.
taskNum	TaskKey	System-assigned identifier for the task.
jeopardyReason Code	string	Four-digit numeric code for identifying why a key date was missed and why a task was not performed by the scheduled completion date.

addTaskJeopardyResponse

The addTaskJeopardyResponse element contains the output information for the operation. The information returned in the response indicates if the operation is successful. Table 7–6 describes the returned information in the response.

Table 7–6 Payload Information for the Response

Name	Defined As	Type Description	File Name
addTaskJeopardyRequestResponse	element	Contains the addTaskJeopardyResponse string	OrderAPI.wsdl

Table 7–7 describes the error messages for the operation.

Table 7–7 Error Messages for the Operation

Error Message	Cause	Resolution
The document number is not found.	The provided document number does not exist in the database.	Verify and pass in a valid document number.
The jeopardy reason code is not valid.	An invalid jeopardy reason code was provided in the request.	Verify and provide a valid jeopardy reason code value in the request.

assignProvPlanRequest Operation

This operation assigns a provisioning plan to the order. This operation takes the provisioning plan ID and order key as the input.

The following are the request and response structures:

Request Structure: assignProvPlanRequest

Response Structure: assignProvPlanRequestResponse

assignProvPlanRequest

The assignProvPlanRequest element contains the input information for the operation. Each row in Table 7–8 describes the element or type name, the XSD declaration, the type description, and the file name that contains the item's definition.

Table 7–8 Payload Information for the Request

Name	Defined As	Type Description	File Name
assignProvPlanRequest	element	assignProvisionPlanProcedureRequest	OrderAPI.wsdl
assignProvisionPlanProcedureRequest	element	AssignProvisionPlanProcedureValue	OrderManagementAPI.xsd
AssignProvisionPlanProcedureValue	complexType	Extension of MetaSolvAssignProvPlanValue Contains a list of fields	OrderManagementAPI.xsd

Table 7–9 describes the required fields for ProvPlanRequest.

Table 7–9 Required Fields for ProvPlanRequest

Field Name	Data Type	Field Description
OrderKey	OrderKey	Document number of the order to assign the provisioning plan.
ProvisioningPlanKey	string	Provisioning plan key value to be assigned to this order.

assignProvPlanRequestResponse

The assignProvPlanRequestResponse element contains the output information for the operation. The information returned in the response indicates if the operation is successful. Table 7–10 describes the returned information in the response.

Table 7–10 Payload Information for the Response

Name	Defined As	Type Description	File Name
assignProvPlanRequestResponse	element	assignProvisionPlanProcedureResponse	OrderAPI.wsdl
assignProvisionPlanProcedureResponse	element	mommekey	OrderManagementAPI.xsd
Mommekey	element	MetaSolvOrderKeyChoice	OrderManagementAPI.xsd
MetaSolvOrderKeyChoice	complexType	metaSolvOrderKey	OrderManagementEntities.xsd
metaSolvOrderKey	element	OrderKey	OrderManagementEntities.xsd
OrderKey	complexType	Extension of ManagedEntityKey Contains a list of fields	OrderManagementEntities.xsd

Table 7–11 describes the error messages for the operation.

Table 7–11 Error Messages for the Operation

Error Message	Cause	Resolution
Miscellaneous Error: Tasks cannot be assigned. Order status is invalid.	Tasks are requested assignment when the order status is not valid for task assignment.	Change the status of the input order number or change the input order number.

billingTelephoneNumberRequest Operation

This operation assigns the billing telephone number (BTN) on an order that has multiple telephone numbers. This operation assigns and unassigns the billing telephone number. It takes the number inventory ID as input for the telephone number.

The following are the request and response structures:

Request Structure: billingTelephoneNumberRequest

Response Structure: assignProvPlanRequestResponse

billingTelephoneNumberRequest

The billingTelephoneNumberRequest element contains the input information for the operation. Each row inTable 7–12 describes the element or type name, the XSD declaration, the type description, and the file name that contains the item's definition.

Table 7–12 Payload Information for the Request

Name	Defined As	Type Definition	File Name
billingTelephoneNumberRequest	element	billingTelephoneNumber	OrderAPI.wsdl
billingTelephoneNumber	complexType	Contains a list of fields	OrderManagementAPI.xsd

Table 7–13 describes the required fields for billingTelephoneNumberRequest.

Table 7–13 Required Fields for billingTelephoneNumberRequest

Field Name	Data Type	Field Description
documentNumber	long	Document number of the order where the billing telephone number is being set.
servItemId	long	Corresponding Service Item ID for the billing telephone number values.
BtnFunctionEnum	Enum	Corresponding BTN function ' 0 - Assign and 1 - Unassign'.
nbrInvId	long	Corresponding Number Inventory ID of the telephone number.

billingTelephoneNumberRequestResponse

The billingTelephoneNumberRequestResponse element contains the output information for the operation. The information returned in the response indicates if the operation is successful. Table 7–14 describes the returned information in the response.

Table 7–14 Payload Information for the Response

Name	Defined As	Type Description	File Name
billingTelephoneNumberRequestResponse	element	billingTelephoneNumberResponse	OrderAPI.wsdl
billingTelephoneNumberResponse	complexType	Contains the documentNumber integer for the order	OrderManagementAPI.xsd

Table 7–15 describes the error messages for the operation.

Table 7–15 Error Messages for the Operation

Error Message	Cause	Resolution
BTN Assign/UnAssign is allowed only for "pending" or "electronically received" orders.	You are requesting an assign or unassign of billing telephone number for an order that is not in correct status.	You can only perform this action when the order is pending or electronically received.

completeTaskRequest Operation

This operation completes a task for an order. It takes order number and task number as input and marks the task as complete.

Note: This completeTaskRequest operation is used for only complete task requests. The following are in the schema files, however this operation does not support:

- updateServicesInOrderProcedureValue
- updateOrderTaskGWEventValue

To execute these operations use the updateOrderRequest operation.

The following are the request and response structures:

Request Structure: completeTaskRequest

Response Structure: completeTaskRequestResponse

completeTaskRequest

The completeTaskRequest element contains the input information for the operation. Each row in Table 7–16 describes the element or type name, the XSD declaration, the type description, and the file name that contains the item's definition.

Table 7–16 Payload Information for the Request

Name	Defined As	Type Description	File Name
completeTaskRequest	element	updateOrderManagementRequest	OrderAPI.wsdl
updateOrderManagementRequest	element	updateValue	OrderManagementAPI.xsd
updateValue	element	MetaSolvUpdateProcedureValueChoice	OrderManagementAPI.xsd
MetaSolvUpdateProcedureValueChoice	complexType	completeTaskProcedureValue	OrderManagementAPI.xsd
completeTaskProcedureValue	element	CompleteTaskProcedureValue	OrderManagementAPI.xsd
CompleteTaskProcedureValue	complexType	See Table 7–17	OrderManagementAPI.xsd

Table 7–17 defines CompleteTaskProcedureValue.

Table 7–17 CompleteTaskProcedureValue Definition

Name	Defined As	Type Description	File Name
CompleteTaskProcedureValue	complexType	Contains a list of fields	OrderManagementAPI.xsd
whyMissInputs	element	WhyMissInputType	OrderManagementAPI.xsd
WhyMissInputType	complexType	Contains a list of fields	OrderManagementData.xsd

Table 7–18 describes the required fields for completeTaskRequest.

Table 7–18 Required Fields for completeTaskRequest

Field Name	Data Type	Field Description
taskNumber	long	Task Number of the task to be completed.
OrderKey	OrderKey	Document Number of the order where the task is assigned.
whyMissReason Code	string	Reason code for why the task was not completed on time. This field is only required when the task exceeds the actual completion date.

completeTaskRequestResponse

The completeTaskRequestResponse element contains the output information for the operation. The information returned in the response indicates if the operation is successful.Table 7–19 describes the returned information in the response.

Table 7–19 Payload Information for the Response

Name	Defined As	Type Description	File Name
completeTaskRequestResponse	element	completeTaskProcedureResponse	OrderAPI.wsdl
completeTaskProcedureResponse	element	CompleteTaskProcedureResponse	OrderManagementAPI.xsd
CompleteTaskProcedureResponse	complexType	mommekey	OrderManagementAPI.xsd
mommekey	element	MetaSolvOrderKeyChoice	OrderManagementAPI.xsd
MetaSolvOrderKeyChoice	complexType	metaSolvOrderKey	OrderManagementEntities.xsd
metaSolvOrderKey	element	OrderKey	OrderManagementEntities.xsd
OrderKey	complexType	Extension of ManagedEntityKey Contains a list of fields	OrderManagementEntities.xsd

Table 7–20 describes the error messages for the operation.

Table 7–20 Error Messages for the Operation

Error Message	Cause	Resolution
Task row not found for document number, task number.	For the input document number and task number, the task could not be found in inventory.	Check that the document number and task number exist for the document number/order.
The task requires a why missed code before it can be completed.	For the input task number, the why-missed code is not yet populated to complete the task.	Populate the why-missed code for the task and then request the task completion again.

createAttachment Operation

The createAttachment operation creates an attachment for the input order.

The following are the request and response structures:

Request Structure: createAttachment

Response Structure: createAttachmentResponse

createAttachment

The createAttachment element contains the input information for the operation. Each row in Table 7–21 describes the element or type name, the XSD declaration, the type description, and the file name that contains the item's definition.

Table 7–21 Payload Information for the Request

Name	Defined As	Type Description	File Name
CreateAttachment	element	createAttachmentRequest	OrderAPI.wsdl
createAttachmentRequest	element	CreateAttachmentRequestValue	OrderManagementAPI.xsd
CreateAttachmentRequestValue	element	CreateAttachmentType	OrderManagementAPI.xsd
CreateAttachmentType	complexType	Contains a list of fields	OrderManagementData.xsd

Table 7–22 describes the required fields for createAttachment.

Table 7–22 Required Fields for createAttachment

Field Name	Data Type	Field Description
attachmentKey	string	Valid PSR document number.
attachmentName	string	Valid attachment name which appears in GUI.
transformationName	string	Name of the XSL file stored in ms_attachment_transform table.
attachmentType	Enum	ORDER is the valid enum value for attachment.

createAttachmentResponse

The createAttachmentResponse element contains the output information for the operation. The information returned in the response indicates if the operation is successful. Table 7–23 describes the returned information in the response.

Table 7–23 Payload Information for the Response

Name	Defined As	Type Description	File Name
createAttachmentResponse	element	momCreateAttachmentResponse	OrderAPI.wsdl
momCreateAttachmentResponse	element	Contains the CreateAttachmentResponseValue long	OrderManagementAPI.xsd

createISROrderRequest Operation

This operation creates a new ISR order in the system and returns the new order key. A single OrderValue is the only value passed into the request. The state values in OrderValue are ignored; the state is initialized to STARTED by the system.

The following are the request and response structures:

Request Structure: createISROrderRequest

Response Structure: createISROrderResponse

createISROrderRequest

The createISROrderRequest element contains the input information for the operation. Each row in Table 7–24 describes the element or type name, the XSD declaration, the type description, and the file name that contains the item's definition.

Note: This operation only supports the metaSolvISROrderValue schema in the request. For creating a PSR order using the metaSolvPSROrderValue schema, use the createOrderRequest and createPSROrderRequest operations.

Table 7–24 Payload Information for the Request

Name	Defined As	Type Description	File Name
createISROrderRequest	element	createOrderByValueRequest	OrderAPI.wsdl
createOrderByValueRequest	element	Mommevalue	OrderManagementAPI.xsd
Mommevalue	element	MetaSolvOrderValueChoice	OrderManagementAPI.xsd
MetaSolvOrderValueChoice	complexType	complexType with a choice of type: ▪ metaSolvPSROrderValue ▪ metaSolvISROrderValue	OrderManagementEntities.xsd
metaSolvISROrderValue	complexType	See Table 7–41, " metaSolvISROrderValue Definition"	OrderManagementEntities.xsd

The complexType metaSolvISROrderValue for the create IRS order operation. metaSolvISROrderValue is a choice in the MetaSolvOrderValueChoice definition. The metaSolvISROrderValue element is mutually exclusive with the PSR order information metaSolvPSROrderValue.

Table 7–25 describes the required fields for ISROrderHeaderType.

Table 7–25 Required Fields for ISROrderHeaderType

Field Name	Data Type	Field Description
externalOrderKey	string	External order key from the external system.
requestType	string	Request type for the ISR order.
activityCd	string	Activity code for the order.
broadbandServiceCateg ory	string	Broad band service category if required.
uniValue	string	UNI value for the order.
organizationId	string	Organization ID for the order.
userId	string	User ID for the order.
desiredDueDate	string	Desired due date for the order that is planned.
serviceOrderDate	string	Service order date for the order that is planned.
servReqStatus	int	Service request status of the order.
quantityUnit	string	Quantity on the order.
quantityCircuitUnit	int	Quantity of circuit on the order.
quantityVirtConnUnit	int	Quantity of virtual circuit on the order.
activeInd	string	Active Indicator for the order.

Table 7–26 describes the required fields for ISRContactType.

Table 7–26 Required Fields for ISRContactType

Field Name	Data Type	Field Description
actionCd	string	Action code for the contact.
orderKey	int	Document number of the order.
companyContact	string	Company contact details of the order.
assocContactName	string	Associate contact details.
customerContactName	string	Customer contact details.

Table 7–27 describes the required fields for ISRNoteValueType.

Table 7–27 Required Fields for ISRNoteValueType

Field Name	Data Type	Field Description
noteKey	int	Note key value.
actionCd	string	Action code for the note.
noteText	string	Data to be entered on the note.
circuitNoteInd	string	Note for the circuit.

Table 7–28 describes the required fields for ISRLocationValueType.

Table 7–28 Required Fields for ISRLocationValueType

Field Name	Data Type	Field Description
isTerminationPoint	boolean	Indicates if this is a termination point.
netLocUse	string	Network location use code.

Table 7–29 describes the required fields for ISRServiceValueType.

Table 7–29 Required Fields for ISRServiceValueType

Field Name	Data Type	Field Description
actionCd	string	Action code for the service.
activityCd	string	Activity code for the service.
serviceItemKey	string	Service item id of the ISR order.
orderKey	int	Document number of the order.

Table 7–30 describes the required fields for ISRRemarkType.

Table 7–30 Required Fields for ISRRemarkType

Field Name	Data Type	Field Description
orderKey	int	Document number of the order.
formId	string	Form ID value.
seqNumber	int	Sequence number of the order.
remarkText	string	Remark that can be updated.

createISROrderResponse

The createISROrderResponse element contains the output information for the operation. The information returned in the response indicates if the operation is successful. Table 7–31 describes the returned information in the response.

Table 7–31 Payload Information for the Response

Name	Defined As	Type Description	File Name
createISROrderResponse	element	createOrderByValueResponse	OrderAPI.wsdl
createOrderByValueResponse	element	mommekey	OrderManagementAPI.xsd
mommekey	element	MetaSolvOrderKeyChoice	OrderManagementAPI.xsd
MetaSolvOrderKeyChoice	complexType	metaSolvOrderKey	OrderManagementEntities.xsd
metaSolvOrderKey	element	OrderKey	OrderManagementEntities.xsd
OrderKey	complexType	Contains a list of fields	OrderManagementEntities.xsd

createOrderRelationship Operation

This operation creates a relationship between two input order numbers. The input values are parent order ID and child order ID.

The following are the request and response structures:

Request Structure: createOrderRelationship

Response Structure: createOrderRelationshipResponse

createOrderRelationship

The createOrderRelationship element contains the input information for the operation. Each row in Table 7–32 describes the element or type name, the XSD declaration, the type description, and the file name that contains the item's definition.

Table 7–32 Payload Information for the Request

Name	Defined As	Type Description	File Name
createOrderRelationship	element	createOrderRelationshipRequest	OrderAPI.wsdl
createOrderRelationshipRequest	complexType	Contains the following: ■ parentOrderKey ■ childOrderKey	OrderManagementAPI.xsd

Table 7–33 describes the required fields for createOrderRelationship.

Table 7–33 Required Fields for createOrderRelationship

Field Name	Data Type	Field Description
parentOrderKey	OrderKey	Document number of the parent order of the relationship.
childOrderKey	OrderKey	Document number of the child order of the relationship.

createOrderRelationshipResponse

The createOrderRelationshipResponse element contains the output information for the operation. The information returned in the response indicates if the operation is successful. Table 7–34 describes the returned information in the response.

Table 7–34 Payload Information for the Response

Name	Defined As	Type Description	File Name
createOrderRelationshipResponse	element	Contains the momCreateOrderRelationshipResponse string	OrderAPI.wsdl

createOrderRequest Operation

The createOrderRequest operation enables external systems to create orders in MSS. You can request two different types of orders with this operation: PSR orders and ISR orders.

The following are the request and response structures:

Request Structure: createOrderRequest

Response Structure: createOrderRequestResponse

createOrderRequest

The createOrderRequest element contains the input information for the operation. Each row in Table 7–35 describes the element or type name, the XSD declaration, the type description, and the file name that contains the item's definition.

Table 7–35 Payload Information for the Request

Name	Defined As	Type Description	File Name
createOrderRequest	element	createOrderByValueRequest	OrderAPI.wsdl
createOrderByValueRequest	element	mommevalue	OrderManagementAPI.xsd
mommevalue	element	MetaSolvOrderValueChoice	OrderManagementAPI.xsd
MetaSolvOrderValueChoice	complexType	complexType with a choice of type: ▪ metaSolvPSROrderValue ▪ metaSolvISROrderValue	OrderManagementEntities.xsd
metaSolvPSROrderValue	complexType	Described in Table 7–36	OrderManagementEntities.xsd
metaSolvISROrderValue	complexType	Described in Table 7–41	OrderManagementEntities.xsd

metaSolvPSROrderValue

Table 7–36 describes the metaSolvPSROrderValue definition information for the create PSR order operation. metaSolvPSROrderValue was referenced in Table 7–35 as a choice in the MetaSolvOrderValueChoice definition. The metaSolvPSROrderValue element is mutually exclusive with the ISR order information.

Table 7–36 metaSolvPSROrderValue Definition

Name	Defined As	Type Description	File Name
metaSolvPSROrderValue	element	MetaSolvPSROrderValue	OrderManagementEntities.xsd
MetaSolvPSROrderValue	complexType	Extension of OrderValue OrderHeaderType metaSolvUserDataValue MetaSolvServiceValueChoice LineDirType	OrderManagementEntities.xsd
OrderValue	complexType	Extension of ManagedEntityValue	OrderManagementEntities.xsd
OrderHeaderType	complexType	complexType with a list of fields	OrderManagementData.xsd
metaSolvUserDataValue	complexType	MetaSolvUserDataValueType	MIPCommonEntities.xsd
MetaSolvUserDataValueType	complexType	UserDataValueType	MIPCommonEntities.xsd
UserDataValueType	complexType	complexType with a list of fields BaseNameValueType	MIPCommonEntities.xsd
MetaSolvServiceValueChoice	complexType	choice of metaSolvServiceValue	ServiceEntities.xsd
metaSolvServiceValue	element	MetaSolvServiceValue	ServiceEntities.xsd

Table 7–36 (Cont.) metaSolvPSROrderValue Definition

Name	Defined As	Type Description	File Name
MetaSolvServiceValue	complexType	Extension of ServiceValue and a list of fields	ServiceEntities.xsd
ServiceValue	complexType	Extension of EntityValue complexType with a list of fields	Service.xsd
EntityValue	complexType	CBEManagedEntityValue	Core.xsd
LineDirType	complexType	lineDirectoryType	OrderManagementData.xsd
lineDirectoryType	complexType	complexType with a list of fields	OrderManagementData.xsd

Table 7–37 describes the required fields for OrderValue.

Table 7–37 Required Fields for OrderValue

Field Name	Data Type	Field Description
PrimaryKey	OrderKey	Value should be 0 if it is new or change order and the document number for update order.

Table 7–38 describes the required fields for OrderHeaderType.

Table 7–38 Required Fields for OrderHeaderType

Field Name	Data Type	Field Description
organizationId	string	Organization responsible for processing the order and providing service.
servReqStatus	Enum	Valid values are defined in OrderStatusEnumType. Any new or change order created through an API should have osRECEIVED which refers as Electronically Received order. For update order, the corresponding order status should be passed.
customerAccountKey	integer	Customer account key to associate to the order.
orderedByFirstName	string	First name of the individual placing the order.
orderedByLastName	string	Last name of the individual placing the order.
orderedByTelephoneNr	string	Telephone number of the individual placing the order.
desiredDueDate	CbeDateTime	Date that the customer desires the products to be installed and in-service.
serviceOrderDate	CbeDateTime	Date the order is placed (entered).
responsibleParty	string	Person responsible for the order, such as the customer care rep or the sales person.
orderActivityCd	Enum	Valid values are oaNEW, oaCHANGE and oaDISCONNECT
billActivationDate	CbeDateTime	Date for the customer invoice.
suppType	Enum	Valid values are stcorrect, stcancell, stduedate and stnone. For a new order, this is always the value stnone and for update orders, this can be updated according to the requirement.

Table 7–39 describes the required fields for UserDataValueType.

Table 7–39 Required Fields for UserDataValueType

Field Name	Data Type	Field Description
tableNm	string	Table where the user data is stored. This table corresponds to where the user data was originally defined.
keyColumnNm	string	Actual primary key column name of the product area. This is required so the user data row can be associated with one item.
keyValue	integer	Actual value of the product area.

Table 7–40 describes the required fields for BaseNameValueType.

Table 7–40 Required Fields for BaseNameValueType

Field Name	Data Type	Field Description
column	string	Field is the user data column that is updated.
value	string	Value entered into the defined column.
dataType	string	Function of a user data field. Determines the kind of information a user can enter in a field on the User Data window. Valid values are Number, Decimal, VARCHAR2, Date, Dropdown.

metaSolvISROrderValue

Table 7–41 describes the metaSolvISROrderValue definition information for the create IRS order operation. metaSolvISROrderValue was referenced in Table 7–35 as a choice in the MetaSolvOrderValueChoice definition. The metaSolvISROrderValue element is mutually exclusive with the PSR order information metaSolvPSROrderValue.

Table 7–41 metaSolvISROrderValue Definition

Name	Defined As	Type Description	File Name
metaSolvISROrderValue	element	MetaSolvISROrderValue	OrderManagementEntities.xsd
MetaSolvISROrderValue	complexType	Extension of OrderValue ISROrderHeaderType	OrderManagementEntities.xsd
ISROrderHeaderType	complexType	complexType with a list of fields, contact, notes, locations, serviceValue, remarks, userData	OrderManagementData.xsd
Contact	element	ISRContactType	OrderManagementData.xsd
ISRContactType	complexType	Contains a list of fields	OrderManagementData.xsd
notes	element	ISRNoteValueType	OrderManagementData.xsd
ISRNoteValueType	complexType	Contains a list of fields	OrderManagementData.xsd
locations	element	ISRLocation	OrderManagementData.xsd
ISRLocation	complexType	locationValueA, locationValueB	OrderManagementData.xsd

Table 7–41 (Cont.) metaSolvISROrderValue Definition

Name	Defined As	Type Description	File Name
locationValueA	element	ISRLocationValueType	OrderManagementData.xsd
locationValueB	element	ISRLocationValueType	OrderManagementData.xsd
ISRLocationValueType	complexType	MetaSolvLocationKey	OrderManagementData.xsd
serviceValue	element	ISRServiceValueType	OrderManagementData.xsd
ISRServiceValueType	complexType	Contains a list of fields	OrderManagementData.xsd
remarks	element	ISRRemarkType	OrderManagementData.xsd
ISRRemarkType	complexType	Contains a list of fields	OrderManagementData.xsd
userData	element	UserDataValueType	OrderManagementData.xsd
UserDataValueType	complexType	Contains a list of fields	OrderManagementData.xsd

Table 7–42 describes the required fields for MetaSolvServiceValue.

Table 7–42 Required Fields for MetaSolvServiceValue

Field Name	Data Type	Field Description
externalServiceKey	string	External service key value from the external system.
activityCd	Enum	Valid values of ActivityCodeEnumType. Activity to be performed on this item. Pass the value based on the scenario.
productGroup	integer	The name of the product grouping. This field should be zero if a value is not applicable.

Table 7–43 describes the required fields for serviceKey and parentServiceKey.

Table 7–43 Required Fields for serviceKey and parentServiceKey

Field Name	Data Type	Field Description
servicePrimaryKey	string	Field is required only for change orders and is populated using service item ID. For new orders, this value is 0.

Table 7–44 describes the required fields for parentSpecKey and describingSpecificationKey.

Table 7–44 Required Fields for parentSpecKey and describingSpecificationKey

Field Name	Data Type	Field Description
serviceSpecificationPrimaryKey	string	Field is populated using the product catalog ID which is added under the service structure.

Table 7–45 describes the required fields for serviceLocationKey.

Table 7–45 Required Fields for serviceLocationKey

Field Name	Data Type	Field Description
locationPrimaryKey	string	Value of the location ID where the product is assigned. This is populated for products which are under Service Location and is 0 for Global products.

Table 7–46 describes the required fields for ipNumberInventoryKey and addressKey.

Table 7–46 Required Fields for ipNumberInventoryKey and addressKey

Field Name	Data Type	Field Description
resourcePrimaryKey	string	Value of the address ID where the product is assigned. This is populated for products which are under Service Location and is 0 for Global products.

Table 7–47 describes the required values for serviceItemValues.

Table 7–47 Required Values for serviceItemValues

Field Name	Data Type	Field Description
label	string	Name or purpose of additional information that you want to capture for a service item.
Value	string	Specific value for a given value label. Values can be lists of predefined choices, or text-entry fields.
valueCd	string	Code used to identify a value, such as a shortened version of the value's "name" or a number.
activityCd	Enum	Valid values of ActivityCodeEnumType. Activity to be performed on this item. Pass the value based on the scenario.

assignedTelephoneNr

Table 7–48 defines assignedTelephoneNr information.

This tag applies for PSR orders which has products that require Telephone Numbers to be assigned.

Table 7–48 assignedTelephoneNr Definition

Name	Defined As	Type Description	File Name
assignedTelNr	element	AssignedTelNrType	OrderManagementData.xsd
AssignedTelNrType	complexType	Set of simple data types: ▪ CnamType ▪ LidbType ▪ StructureFormatType	OrderManagementData.xsd
CnamType	element	psrCnam	OrderManagementData.xsd
psrCnam	element	PsrCnamType	OrderManagementData.xsd
PsrCnamType	complexType	Contains a list of fields	OrderManagementData.xsd
LidbType	element	psrLidbType	OrderManagementData.xsd

Table 7–48 (Cont.) assignedTelephoneNr Definition

Name	Defined As	Type Description	File Name
psrLidbType	element	PsrLidbType	OrderManagementData.xsd
PsrLidbType	complexType	Contains a list of fields	OrderManagementData.xsd
StructureFormatType	complexType	structureFormatComponents	CustomerManagementData.xsd
structureFormatComponents	element	StructureFormatComponentType	CustomerManagementData.xsd
StructureFormatComponentType	complexType	Contains a list of fields	CustomerManagementData.xsd
assignedTelNrRel	element	AssignedTelNrRelType	OrderManagementData.xsd
AssignedTelNrRelType	complexType	Set of simple data types: ■ AssignedTelNrType ■ MetaSolvServiceKey	OrderManagementData.xsd
AssignedTelNrType	complexType	Contains a list of fields	OrderManagementData.xsd
MetaSolvServiceKey	element	MetaSolvServiceKey	ServiceEntities.xsd
MetaSolvServiceKey	complexType	servicePrimaryKey	ServiceEntities.xsd

Table 7–49 describes the required fields for MetaSolvServiceKey.

Table 7–49 Required Fields for MetaSolvServiceKey

Field Name	Data Type	Field Description
servicePrimaryKey	string	Field is required only for change orders and is populated using service item ID and for new orders, this is 0.

Table 7–50 describes the required fields for assignedTelephoneNr.

Table 7–50 Required Fields for assignedTelephoneNr

Field Name	Data Type	Field Description
relationshipType	Enum	Relationship between two telephone numbers. Valid values are rtRCF, rtPorted, rtPortedTo, rtTollfree and rtNone.

Table 7–51 describes the required fields for AssignedTelNrType.

Table 7–51 Required Fields for AssignedTelNrType

Field Name	Data Type	Field Description
telNrSuf	string	System tracks a number's history by assigning a suffix counter to each number. Each time the number is recalled into an Unassigned status, the counter increases by one.
respOrg	string	Organization that owns the NPA NXX.
telNrTypeCd	Enum	Telephone number type code and valid values are Valid values are tntINPOUT, tntRESALE, tntNPIN, tntTFWTN, tntTF, tntFORWTN, tntNPOUT, tntWTN, tntNone.

Table 7–51 (Cont.) Required Fields for AssignedTelNrType

Field Name	Data Type	Field Description
activityCd	Enum	Valid values of ActivityCodeEnumType. Activity to be performed on this item. Pass the value based on the scenario.
identityCd	Enum	Identify whether the telephone number as main telephone number or alternate. Valid values are: MPTN, ALT and NONE.
locationId	Long	The location ID of the telephone number and if not mapped, it should be 0.
btnInd	Boolean	Default is false and should be changed to true if the TN needs to set as billing telephone number.

Table 7–52 describes the required fields for PsrCnamType.

Table 7–52 Required Fields for PsrCnamType

Field Name	Data Type	Field Description
callingName	string	The name of the business or residential customer, as desired for the caller ID display.
presentationInd	Enum	Valid values of PresentationIndEnumType. Indicates that caller ID name information is sent for calls made from this number.
activityCode	Enum	Valid values of ActivityCodeEnumType. Activity to be performed on this item.
manualIndicator	Enum	Valid values of IndicatorEnumType. Indicates that you do not want to send the CNAM information electronically to the gateway vendor. If you do not send the information electronically, you must send it manually.
effectiveDateChg	CbeDateTime	Indicates a future date for any change to be effective, if it is not effective immediately.

Table 7–53 describes the required fields for PsrLidbType.

Table 7–53 Required Fields for PsrLidbType

Field Name	Data Type	Field Description
bnsCode	Enum	Valid values of BNSCodeEnumType. The Billed Number Screening Code is a code to indicate the type of screening, if any, applied to this number.
marketInd	Enum	Valid values of MarketIndicatorEnumType. Indicates the type of market this number is assigned to.
serviceClassCode	Enum	Valid values of ClassCodeEnumType. A code that describes how a service is used by a business, residential, or pay phone customer. Examples include business, residential, public coin, public non-coin, and semi-public coin.
manualIndicator	Enum	Valid values of IndicatorEnumType. Indicates that you do not want to send the LIDB information electronically to the gateway vendor. If you do not send the information electronically, you must send it manually.
activityCd	Enum	Valid values of ActivityCodeEnumType. Activity to be performed on this item.
effectiveDateChg	CbeDateTime	Indicates a future date for any change to be effective, if it is not effective immediately.

Table 7–54 describes the required fields for StructureFormatType.

Table 7–54 Required Fields for StructureFormatType

Field Name	Data Type	Field Description
sfType	string	Category that describes a structure.
name	string	Name of a specific customizing of a structured format type.

Table 7–55 describes the required fields for StructureFormatComponentType.

Table 7–55 Required Fields for StructureFormatComponentType

Field Name	Data Type	Field Description
Id	int	ID of a component of a structured format.
name	string	Name of a component of a structured format.
componentType	string	Type of component, such as table-driven drop-down or valid value drop-down.
Value	string	Name of a value for a component.

Table 7–56 describes Access.

Table 7–56 Access Definition

Name	Defined As	Type Description	File Name
Access	element	AccessType	ServiceEntities.xsd
AccessType	complexType	AccessInformationType	OrderManagementData.xsd
AccessInformationType	complexType	Contains a list of fields	OrderManagementData.xsd

Table 7–57 describes the required fields for AccessInformationType.

Table 7–57 Required Fields for AccessInformationType

Field Name	Data Type	Field Description
Floor	string	The floor where the product is installed.
Room	string	The room where the product is installed.
jackCd	string	A standard code for the registered or non-registered jack used to terminate service at the service location.
jackNr	string	If the jack is existing, the number associated with the jack.
jackPosition	integer	The position of the circuit within the jack.
jackStatus	string	Indicates whether the jack used to terminate service is new or existing.
accessInfo	string	Special instructions regarding access to the service location and times that access is available.
additionalLocationDetails	string	Specific details not included in the Additional Information field.
locationIdSpot	integer	The CLLI code for the physical point of termination at the end user location.
activityCd	Enum	Valid values of ActivityCodeEnumType. Activity to be performed on this item.

Table 7–58 describes isdnTrunkGroup.

Table 7–58 isdnTrunkGroup Definition

Name	Defined As	Type Description	File Name
isdnTrunkGroup	element	IsdnTrunkGroupType	ServiceEntities.xsd
IsdnTrunkGroupType	complexType	isdnTrunkGroupInformation	OrderManagementData.xsd
isdnTrunkGroupInformation	complexType	IsdnTrunkGroupInformationType	OrderManagementData.xsd
IsdnTrunkGroupInformationType	complexType	Contains a list of fields	OrderManagementData.xsd

Table 7–59 describes the required fields for IsdnTrunkGroupInformationType.

Table 7–59 Required Fields for IsdnTrunkGroupInformationType

Field Name	Data Type	Field Description
bchanFunctionCd	Enum	Valid values of BchanFuncEnumType. Identifies the purpose of the bearer channels.
nfasBkupDchanInd	Enum	Valid values of IndicatorWNAEnumType. Indicates if a backup D-Channel is used on a subsequent DS1 in a multiple DS1 trunk group.
premierCallInd	Enum	Valid values of IndicatorEnumType. Identifies whether the premier calling features are available.
nfasInd	Enum	Valid values of IndicatorEnumType. Indicates whether the DS1 circuits in the ISDN trunk group shares the same D channel for signaling. When NFAS is not used, each DS1 uses its own D channel.
dialableGroup	Enum	Valid values of DialableGroupEnumType. The grouping method for the DS0 time slot (channel).
dialableSearch	Enum	Valid values of DialableSrchEnumType.

Table 7–60 defines trunkGroup.

Table 7–60 trunkGroup Definition

Name	Defined As	Type Description	File Name
trunkGroup	element	TrunkGroupType	ServiceEntities.xsd
TrunkGroupType	complexType	TrunkGroupInformation	OrderManagementData.xsd
TrunkGroupInformation	complexType	TrunkGroupInformationType	OrderManagementData.xsd
TrunkGroupInformationType	complexType	Contains the following: ▪ intoCO ▪ outFromCO	OrderManagementData.xsd
intoCO	element	IntoCOType	OrderManagementData.xsd
outFromCO	element	OutFromCOType	OrderManagementData.xsd

Table 7–61 describes the required fields TrunkGroupInformationType.

Table 7–61 Required Fields for TrunkGroupInformationType

Field Name	Data Type	Field Description
coDirection	string	Describes the direction of the signals traveling along this trunk.
interfaceType	string	Indicates whether the interface for the trunk group is digital or analog.
glareAction	string	Indicates who has control of the trunks when both ends are seized at the same time for different uses or by different users.
ncCd	string	Network Channel code - A code that identifies the type of service provided by the trunk.
ncCdOption	string	Two optional characters further identify the performance specifications and transmission options of the trunk.
nciCd	string	Network Channel Interface code - A code that identifies the electrical conditions of the trunk at the customer location.
twoSixCd	string	A code that identifies the trunk group. The code consists of a two-character company code, followed by the six-digit serial number of the trunk.
framing	string	The type of frame pattern used for DS1 service.
lineCoding	string	The type of pulse code modulation (PCM) pattern used for DS1 service.
trunkSegment	string	Suffix added to a trunk to make it unique. This suffix is created by a local telephone trunk service.
selectionSequenc e	string	The method of selecting idle trunks in a trunk group during call processing. This field is required for all types of special trunk groups.

Table 7–62 describes the required fields for IntoCOType.

Table 7–62 Required Fields for IntoCOType

Field Name	Data Type	Field Description
digitsQty	integer	The number of digits that are to be received by the central office switch from the customer for a Direct Outward-Dial (DOD) trunk.
maxDigit	string	The maximum number of digits that can be sent from a PBX, terminal equipment, or similar device to the central office switch.
minDigit	string	The minimum number of digits that can be sent from a PBX, terminal equipment, or similar device to the central office switch.
pulseType	string	The pulsing method used to send digits from a customer to a central office switch.
startSignal	string	The timing or method used to start signaling from the customer to the central office switch.
supervisionSigna l	string	The type of signaling sent from a PBX, terminal equipment, or similar device to a central office switch.

Table 7–63 describes the required fields for OutFromCOType.

Table 7–63 Required Fields for OutFromCOType

Field Name	Data Type	Field Description
digitsQty	integer	The number of digits sent from the central office switch to the customer for a Direct In-Dial (DID) trunk.
maxDigit	string	The actual digits that are inserted before the Digits Outpulsed and sent from the central office switch to the PBX, terminal equipment, or similar device.
minDigit	string	The quantity of digits that are extracted from the Digits Outpulsed before they are sent from the central office switch to the PBX, terminal equipment, or similar device.
pulseType	string	The pulsing method used to send digits from a central office switch to a customer.
startSignal	string	The timing or method used to start signaling from the customer to the central office switch.
supervisionSigna l	string	The type of signaling sent from a PBX, terminal equipment, or similar device to a central office switch.

Table 7–64 defines Circuit.

Table 7–64 Circuit Definition

Name	Defined As	Type Description	File Name
circuit	element	CircuitType	ServiceEntities.xsd
CircuitType	complexType	circuitInformation	OrderManagementDa ta.xsd
circuitInformation	complexType	CircuitInformationType	OrderManagementDa ta.xsd
CircuitInformationType	complexType	Contains a list of fields	OrderManagementDa ta.xsd

Table 7–65 describes the required fields for CircuitInformationType.

Table 7–65 Required Fields for CircuitInformationType

Field Name	Data Type	Field Description
rateCd	string	Identifies the transmission rate of the requested circuit (measured in bits per second).
aPot	string	The CLLI associated with the additional termination point of the circuit, if applicable.
framing	string	The type of frame pattern used for DS1 service.
framingAnsiInd	Enum	Valid values of IndicatorWNAEnumType. Only applicable if framing is used.
lineCoding	string	The type of pulse code modulation (PCM) pattern used for DS1 service.
ownedLeasedCd	Enum	Valid values of CktServiceTypeEnumType. Description of the service provided, broken down into Special Services (for IntraLATA/Boundary and LATA/Boundary Access), Switched Services, and Facility Services.
serviceTypeExt	string	Service a circuit provides. The extension further defines a service type code.
serviceTypeCd	string	Service type code.

Table 7–65 (Cont.) Required Fields for CircuitInformationType

Field Name	Data Type	Field Description
ncCd	string	Identifies performance and other technical specifications for the circuit.
ncCdOption	string	Identifies performance and other technical specifications for the circuit.
Jurisdiction	string	Jurisdiction classification, which categorizes the circuit (for its originating/terminating end) for toll separation purposes.

Table 7–66 defines circuitLocation.

Table 7–66 circuitLocation Definition

Name	Defined As	Field Description	File Name
circuitLocation	element	CircuitLocationType	ServiceEntities.xsd
CircuitLocationType	complexType	circuitLocationInformation	OrderManagementData.xsd
circuitLocationInformation	complexType	CircuitLocationInformationType	OrderManagementData.xsd
CircuitLocationInformationType	complexType	Contains the following: ▪ primaryLocation ▪ secondaryLocation	OrderManagementData.xsd
primaryLocation	element	LocationType	OrderManagementData.xsd
secondaryLocation	element	LocationType	OrderManagementData.xsd
LocationType	complexType	CodedLocationType, EndUserLocationType	OrderManagementData.xsd
CodedLocationType	complexType	locationKey	OrderManagementData.xsd
locationKey	element	MetaSolvLocationKey	OrderManagementData.xsd
EndUserLocationType	complexType	Contains the following: ▪ locationKey ▪ jackInformation	OrderManagementData.xsd
jackInformation	element	JackInformationType	OrderManagementData.xsd
JackInformationType	complexType	Contains a list of fields	OrderManagementData.xsd

Table 7–67 describes the required fields for CodedLocationType.

Table 7–67 Required Fields for CodedLocationType

Field Name	Data Type	Field Description
serviceLocationRef	integer	Relates to the Service Location field that appears for End User locations.
clliCd	string	Only applicable for network locations. The network location for the circuit connection. This field is only applicable to network locations and is only required when the CodedLocationType is populated.
nciCd	string	Network Channel Interface code that identifies the interface specifications associated with a circuit.
cfa	string	Connecting Facility Assignment field. If applicable, the circuit on another provider's network where the ordered circuit is connected and the channel that it is assigned.
activityCd	Enum	Valid values of ActivityCodeEnumType. Activity to be performed on this item.
supervisionSignal	string	Type of signaling sent from a PBX, terminal equipment, or similar device to a central office switch.

Table 7–68 describes the required fields for EndUserLocationType.

Table 7–68 Required Fields for EndUserLocationType

Field Name	Data Type	Field Description
serviceLocationRef	integer	Relates to the Service Location field that appears for End User locations.
endUserClli	Enum	Valid values of MomLocationTypeEnumType. The network location for the circuit connection.
nciCd	string	Network Channel Interface code that identifies the interface specifications associated with a circuit.
cfa	string	Connecting Facility Assignment field. If applicable, the circuit on another provider's network where the ordered circuit is connected and the channel that it is assigned.
backboardSpace	Enum	Valid values of IndicatorWNAEnumType. This field is only required if the parent tag is populated. This field is applicable for private line (including LAN) connections. Indicates whether backboard space for equipment installation is available, not available, or not applicable at this location.
offNetType	Enum	Valid values of NniUniCDEnumType. This field is only required if the parent tag is populated. This field is applicable for frame relay orders when the On Net Location box is not checked.
equipSpace	Enum	Valid values of IndicatorWNAEnumType. Indicates whether equipment space is available, not available, or not applicable at this location.
seperatlyFused	Enum	Valid values of IndicatorEnumType. Indicates whether the power available to this location is separately fused.
buildingGround	Enum	Valid values of IndicatorEnumType. Indicates whether a building ground is available at this location.
powerAvail	Enum	Valid values of IndicatorEnumType. Indicates whether power is available at this location.
localLoop	Enum	Valid values of IndicatorEnumType. Indicates whether you need to order a local loop connection to this location.
Muxing	Enum	Valid values of IndicatorEnumType. Indicates whether the customer requests multiplexing (muxing).

Table 7–68 (Cont.) Required Fields for EndUserLocationType

Field Name	Data Type	Field Description
onNetLoc	Enum	Valid values of IndicatorEnumType. Indicates whether your company currently serves this location.
powerType	Enum	Valid values of PowerTypeEnumType. A description of the type of power available at this location.
activityCd	Enum	Valid values of ActivityCodeEnumType. Activity to be performed on this item.

Table 7–69 describes the required fields for JackInformationType.

Table 7–69 Required Fields for JackInformationType

Field Name	Data Type	Field Description
jackCd	string	Standard code for the registered or non-registered jack used to terminate service at the service location.
jackPosition	integer	Position of the circuit within the jack.
jackNr	string	If the jack is existing, the number associated with the jack.
jackStatus	Enum	Valid values of JackStatusEnumType. Indicates whether the jack used to terminate service is new or existing.
activityCd	Enum	Valid values of ActivityCodeEnumType. Activity to be performed on this item.

Table 7–70 defines Connector.

Table 7–70 Connector Definition

Name	Defined As	Type Description	File Name
connector	element	ConnectorType	ServiceEntities.xsd
ConnectorType	complexType	connectorInformation	OrderManagementData.xsd
connectorInformation	complexType	ConnectorInformationType	OrderManagementData.xsd
ConnectorInformationType	complexType	Contains the following: ■ serviceLocationKeyA ■ serviceLocationKeyB	OrderManagementData.xsd
serviceLocationKeyA	element	MetaSolvLocationKey	OrderManagementData.xsd
serviceLocationKeyB	element	MetaSolvLocationKey	OrderManagementData.xsd

Table 7–71 describes the required fields for ConnectorInformationType.

Table 7–71 Required Fields for ConnectorInformationType

Field Name	Data Type	Field Description
compTypeConId	integer	Type of connection on an order.
rateCd	string	Identifies the transmission rate of the requested circuit (measured in bits per second).
Framing	string	Type of frame pattern used for DS1 service.
lineCoding	string	Type of pulse code modulation (PCM) pattern used for DS1 service.

Table 7–71 (Cont.) Required Fields for ConnectorInformationType

Field Name	Data Type	Field Description
Jurisdiction	string	Jurisdiction classification, which categorizes the circuit (for its originating/terminating end) for toll separation purposes.
serviceLocationRefA	integer	Primary location for the circuit termination for A.
addressKeyA	integer	Address key A value.
serviceLocationRefB	integer	Primary location for the circuit termination for B.
addressKeyB	integer	Address key B value.

Table 7–72 defines caUsageInstance.

Table 7–72 caUsageInstance Definition

Name	Defined As	Type Description	File Name
caUsageInstance	element	CaUsageInstanceType	ServiceEntities.xsd
CaUsageInstanceType	complexType	caUsageValue	OrderManagementData.xsd
caUsageValue	element	CaUsageValueType	OrderManagementData.xsd
CaUsageValueType	complexType	Contains a list of fields	OrderManagementData.xsd

Table 7–73 describes the required fields for CaUsageInstanceType.

Table 7–73 Required Fields for CaUsageInstanceType

Field Name	Data Type	Field Description
caUsageKey	integer	Unique ID of custom attribute.
caValueLabel	string	Label of custom attribute.

Table 7–74 describes the required fields for CaUsageValueType.

Table 7–74 Required Fields for CaUsageValueType

Field Name	Data Type	Field Description
caValueKey	integer	Key of valid value custom attribute.
caValue	string	Value of custom attribute.
caUom	string	Unit of measure. For example, meters, yards, or feet.
caUsageVvKey	integer	Key of valid value custom attribute.

Table 7–75 defines picInfo.

Table 7–75 picInfo Definition

Name	Defined As	Type Description	File Name
picInfo	element	PicInfoType	ServiceEntities.xsd
PicInfoType	complexType	Contains a list of fields	OrderManagementData.xsd

Table 7–76 describes the required fields for PicInfoType.

Table 7–76 Required Fields for PicInfoType

Field Name	Data Type	Field Description
Pic	string	A Primary Interexchange (or IntraLata) Carrier code or Carrier identification code (CIC) that represents an end user's toll provider.
picTypeCd	string	Toll calling area of a PIC, including intraLATA/Boundary, interLATA/Boundary, and international.
freezePic	Enum	Valid values of IndicatorType. Indicates the customer requested the freeze option on the PIC. The end user is specifying that the PIC can only be changed when authorized by the end user.
partyName	string	Name for the given primary interexchange code. This field is information only not used in business logic for create or update transactions.
activityCd	Enum	Valid values of ActivityCodeEnumType. Activity to be performed on this item.

Table 7–77 defines orderItemPrice.

Table 7–77 orderItemPrice Definition

Name	Defined As	Field Description	File Name
orderItemPrice	element	PSROrderItemPriceType	ServiceEntities.xsd
PSROrderItemPriceType	complexType	psrOrderItemPrice	OrderManagementData.xsd
psrOrderItemPrice	complexType	MomPsrOrderItemPriceType	OrderManagementData.xsd
MomPsrOrderItemPriceType	complexType	Contains a list of fields	OrderManagementData.xsd

Table 7–78 describes the required fields for MomPsrOrderItemPriceType.

Table 7–78 Required Fields for MomPsrOrderItemPriceType

Field Name	Data Type	Field Description
priceID	integer	ID for the price.
basePriceSeq	string	Base price sequence.
priceVarSeq	string	Price variation sequence.
priceOverride	Float	Indicator on whether to override the price.
activityCD	Enum	Valid values of ActivityCodeEnumType. Activity to be performed on this item.

Table 7–79 defines EmailInfo.

Table 7–79 EmailInfo Definition

Name	Defined As	Type Description	File Name
EmailInfo	element	EmailInfoType	ServiceEntities.xsd
EmailInfoType	complexType	internetEmailInformation	OrderManagementDa ta.xsd
internetEmailInformation	complexType	InternetEmailInformationType	OrderManagementDa ta.xsd
InternetEmailInformationType	complexType	complexType with a list of fields	OrderManagementDa ta.xsd

Table 7–80 describes the required fields for InternetEmailInformationType.

Table 7–80 Required Fields for InternetEmailInformationType

Field Name	Data Type	Field Description
emailName	string	Email that is used by the customer.
Domain	string	Domain name.
domainSuffix	string	Suffix for the domain name.

Table 7–81 defines DialupInfo.

Table 7–81 DialupInfo Definition

Name	Defined As	Field Description	File Name
DialupInfo	element	DialupInfoType	ServiceEntities.xsd
DialupInfoType	complexType	internetDialupInformation	OrderManagementDa ta.xsd
internetDialupInformation	complexType	InternetDialupInformationType	OrderManagementDa ta.xsd
InternetDialupInformationType	complexType	Contains a list of fields	OrderManagementDa ta.xsd

Table 7–82 describes the required fields for InternetDialupInformationType.

Table 7–82 Required Fields for InternetDialupInformationType

Field Name	Data Type	Field Description
url	string	URL for the dial up.
Domain	string	Domain name.
domainSuffix	string	Suffix for the domain name.
userId	string	User ID for the dial up.
accessTnNbrInvI d	integer	Associated Number inventory ID.

Table 7–83 defines TemplateInfo.

Table 7–83 TemplateInfo Definition

Name	Defined As	Type Definition	File Name
TemplateInfo	element	TemplateInfoType	ServiceEntities.xsd
TemplateInfoType	complexType	templateInformation	OrderManagementData.xsd
templateInformation	complexType	TemplateInformationType	OrderManagementData.xsd
TemplateInformationType	complexType	Contains a list of fields	OrderManagementData.xsd

Table 7–84 describes the required fields for TemplateInformationType.

Table 7–84 Required Fields for TemplateInformationType

Field Name	Data Type	Field Decription
configTypeId	integer	Config type ID.
shortName	string	Short name that is going to be defined for the template product.
longName	string	Long name that is going to be defined for the template product.
providerId	string	Provider ID of the template.
customerId	string	Customer ID that is assigned to this template.

Table 7–85 defines ElementInfo.

Table 7–85 ElementInfo Definition

Name	Defined As	Type Description	File Name
ElementInfo	element	ElementInfoType	ServiceEntities.xsd
ElementInfoType	complexType	elementInformation	OrderManagementData.xsd
elementInformation	complexType	ElementInformationType	OrderManagementData.xsd
ElementInformationType	complexType	Contains a list of fields	OrderManagementData.xsd

Table 7–86 describes the required fields for ElementInformationType.

Table 7–86 Required Fields for ElementInformationType

Field Name	Data Type	Field Description
configTypeId	integer	Config type ID.
elementType	string	Type of element that is used.
elementNbr	integer	Element number.
elementNm	string	Element name that is used.
networkElementId	string	Network element ID that is assigned.
locationId	string	Location ID that is assigned.
addressId	string	Address ID that is assigned.

Table 7–87 defines EquipmentInfo.

Table 7–87 EquipmentInfo Definition

Name	Defined As	Type Description	File Name
EquipmentInfo	element	EquipmentInfoType	ServiceEntities.xsd
EquipmentInfoType	complexType	equipmentInformation	OrderManagementData.xsd
equipmentInformation	complexType	EquipmentInformationType	OrderManagementData.xsd
EquipmentInformationType	complexType	Contains a list of fields	OrderManagementData.xsd

Table 7–88 describes the required fields for EquipmentInformationType.

Table 7–88 Required Fields for EquipmentInformationType

Field Name	Data Type	Field Description
equipmentSpecId	integer	Equipment specification ID that is assigned to this order.
equipmentId	integer	Equipment id that is assigned to this order.

Table 7–89 defines PSRDirServ.

Table 7–89 PSRDirServ Definition

Name	Defined As	Type Description	File Name
PSRDirServ	element	PSRDirServType	ServiceEntities.xsd
PSRDirServType	complexType	directoryServiceRequest	OrderManagementData.xsd
directoryServiceRequest	complexType	directoryServiceType	OrderManagementData.xsd
directoryServiceType	complexType	Contains the following: ■ BANUsageOne ■ BANUsageTwo ■ ACNAInfo	OrderManagementData.xsd
BANUsageOne	element	PSRBillingAccountNumberUsageType	OrderManagementData.xsd
BANUsageTwo	element	PSRBillingAccountNumberUsageType	OrderManagementData.xsd
PSRBillingAccountNumberUsageType	complexType	Contains a list of fields	OrderManagementData.xsd
ACNAInfo	element	PSRDirServReqACNAType	OrderManagementData.xsd
PSRDirServReqACNAType	complexType	Contains a list of fields	OrderManagementData.xsd

Table 7–90 describes the required fields for directoryServiceType.

Table 7–90 Required Fields for directoryServiceType

Field Name	Data Type	Field Description
serviceCenter1	string	Service center 1 details.
serviceCenter2	string	Service center 2 details.
accountNumber	string	Identifies the main account number assigned by the NSP.
accountTelNbr	string	Identifies the account telephone number assigned by the NSP.
TOSTYPE	Enum	Valid values of TypeOfServiceEnumType. Type of Service identifies the type of service or the line ordered. Byte 1 of 4-byte field.
TOSPROD	Enum	Valid values of TypeOfServiceProdEnumType. Type of Service identifies the type of service for the line ordered. Byte 2 of 4-byte field.
TOSCLASS	Enum	Valid values of TypeOfServiceClassEnumType. Type of Service identifies the type of service or the line ordered. Byte 3 of 4-byte field.
TOSCHAR	Enum	Valid values of TypeOfServiceClassEnumType. Type of Service identifies the type of service or the line ordered. Byte 4 of 4-byte field.

Table 7–91 describes the required fields for PSRBillingAccountNumberUsageType.

Table 7–91 Required Fields for PSRBillingAccountNumberUsageType

Field Name	Data Type	Field Description
billingAcctNbr	string	Identifies the billing account number for the recurring and non-recurring charges for this request.
banSvcCat	Enum	Valid values of BanServiceTypeEnumType. Identifies the type of billing account number.

Table 7–92 describes the required fields for PSRDirServReqACNAType.

Table 7–92 Required Fields for PSRDirServReqACNAType

Field Name	Data Type	Field Description
Acna	string	ACNA value.
partyPrimaryName	string	Primary name of the party.
partySecondary Name	string	Secondary name of the party.

Table 7–93 defines PSRDirListing.

Table 7–93 PSRDirListing Definition

Name	Defined As	Type Description	File Name
PSRDirListing	element	PSRDirListingType	ServiceEntities.xsd
PSRDirListingType	complexType	directoryListing	OrderManagementData.xsd
directoryListing	complexType	directoryListingType	OrderManagementData.xsd

Table 7–93 (Cont.) PSRDirListing Definition

Name	Defined As	Type Description	File Name
directoryListingType	complexType	Contains the following: ▪ address ▪ telNbr ▪ delivery ▪ addlInfo ▪ text	OrderManagementData.xsd
Address	element	PSRDirectoryListingAddress	OrderManagementData.xsd
PSRDirectoryListingAddress	complexType	Contains a list of fields	OrderManagementData.xsd
telNbr	element	PSRDirectoryListingTelephoneNumber	OrderManagementData.xsd
PSRDirectoryListingTelephoneNumber	complexType	Contains a list of fields	OrderManagementData.xsd
delivery	element	PSRDirectoryListingDeliveryLocation	OrderManagementData.xsd
PSRDirectoryListingDeliveryLocation	complexType	Contains a list of fields PSRDirectoryListingDirectoryTypeSegments	OrderManagementData.xsd
addlInfo	element	PSRDirectoryListingAdditionalInformation	OrderManagementData.xsd
PSRDirectoryListingAdditionalInformation	complexType	Contains a list of fields	OrderManagementData.xsd
Text	element	PSRDirectoryListingLinesOfText	OrderManagementData.xsd
PSRDirectoryListingLinesOfText	complexType	Contains a list of fields	OrderManagementData.xsd
PSRDirectoryListingDirectoryTypeSegments	complexType	Contains a list of fields	OrderManagementData.xsd

Table 7–94 describes the required fields for directoryListingType

Table 7–94 Required Fields for directoryListingType

Field Name	Data Type	Field Description
ALICd	string	Uniquely identifies each listing for an account number.
listingType	Enum	Valid values of listingType. For DSR, identifies the type of listing being submitted for the publication and directory assistance (DA) appearance rules.
listAcctType	Enum	Valid values of TypeOfAccountEnum. Identifies the type of account for this listing and determines the placement of the listing in split directories and directory assistance.
styleCd	Enum	Valid values of StyleCodeEnum. Identifies if the listing is a straight line, caption header.
Yphdgcd	string	Identifies the heading under which a business listing should appear in the Yellow Pages Directory.
Yphdgverbiage	string	Identifies the heading under which a business listing should appear in the Yellow Pages.

Table 7–94 (Cont.) Required Fields for directoryListingType

Field Name	Data Type	Field Description
placeListingAs	string	When a customer wishes to override the normal sequencing, the special filing words that are used rather than using the listed name.
dirName	string	Identifies the name of the directory in which the listing is to be placed.
listedNmAddrAddl	string	This is a twelve-character alphaNumeric field.
listedNmLineageDual	Enum	Valid values of LineageNameEnum.
listedNmAddrDual	Enum	Valid values of AddressNameEnum.

Table 7–95 describes the required fields for PSRDirectoryListingAddress.

Table 7–95 Required Fields for PSRDirectoryListingAddress

Field Name	Data Type	Field Description
listedNmLineage	Enum	Valid values of LineageNameEnum.
listedNmLast	string	Last name for the listing.
listedNmFirst	string	First name for the listing.
houseNbrPref	string	Identifies the prefix for the house number.
houseNbr	string	The number of the street address, not including extensions such as 1/2, A, D.
streetNm	string	Name of the road of the address.
City	string	Community of the street name/house number as designated by the Master Street Address Guide, MSAG.
stateCd	string	A two-character abbreviation for the State associated with the address.
zipCode	string	Zip code of the address.
omitAddr	Enum	Valid values of CheckBoxOptionEnum.
omitTelNbr	Enum	Valid values of CheckBoxOptionEnum.

Table 7–96 describes the required fields for PSRDirectoryListingTelephoneNumber.

Table 7–96 Required Fields for PSRDirectoryListingTelephoneNumber

Field Name	Data Type	Field Description
Listingtelnbr	string	Indicates the telephone number to be placed in the directory and quoted in Directory Assistance (DA) as appropriate based on LTY, RTY, and STYC field entries.
mainTelNbr	string	Field identifies the main telephone number used to link the main listing with any other associated listings.
Telnbrnpa	string	Area code (NPA) portion of the telephone number. The first three digits of a ten-digit phone number.

Table 7–96 (Cont.) Required Fields for PSRDirectoryListingTelephoneNumber

Field Name	Data Type	Field Description
Telnbrnxx	string	Central Office, or Exchange, portion of the phone number. The first three digits of a seven-digit telephone number, and the fourth through sixth digits of a ten-digit phone number.
Telnbrlinerange	string	Line number of the telephone. This is the last four digits a ten-digit telephone number, and the last four digits of a seven-digit number.
Telnbrsuf	string	Suffix or PIN used to differentiate between two otherwise identical telephone numbers.

Table 7–97 describes the required fields for PSRDirectoryListingDeliveryLocation.

Table 7–97 Required Fields for PSRDirectoryListingDeliveryLocation

Field Name	Data Type	Field Description
dirDirectoryDeliveryId	integer	Oracle sequence uniquely identifies the delivery location information.
delRefNbr	string	Identifies the delivery address/information segment and each additional delivery address/information segment with a unique number.
delAddressType	Enum	Valid values of deliveryAddTypeEnum. Identifies a delivery address segment as being valid for hand delivery, postal delivery or both. (Represents the DATY field on the DL Form.).
endUserName	string	Recipient name for the delivery address.
houseNbrPref	string	Identifies the prefix for the house number.
houseNbr	string	Number of the street address, not including extensions such as 1/2, A, D.
streetnm	string	Name of the road of the address.
City	string	Community of the street name/house number as designated by the Master Street Address Guide, MSAG.
stateCd	string	Two-character abbreviation for the State associated with the address.
zipCd	string	Zip code of the address.

Table 7–98 describes the required fields for PSRDirectoryListingAdditionalInformation.

Table 7–98 Required Fields for PSRDirectoryListingAdditionalInformation

Field Name	Data Type	Field Description
ttyttdCd	Enum	Valid values of TTYTDDEnum. Identifies that this listing should include a special TTY or TDD phrase and which phrase should be included.
localNpaNxx	string	The telephone number being listed is not local to the service address and this column indicates the provider's local NPA NXX. Examples of when this might be used include foreign exchange numbers and ported numbers.
noSoliciationCd	Enum	Valid values of CheckBoxOptionEnum.
listedNmPlacement	Enum	Valid values of CheckBoxOptionEnum.
omitStAddr	Enum	Valid values of CheckBoxOptionEnum. Identifies if this listing is to be omitted from the street address (reverse) directory.
omitDirectMail	Enum	Valid values of CheckBoxOptionEnum. Indicates whether this listing is to be omitted from any direct mail lists.

Table 7–98 (Cont.) Required Fields for PSRDirectoryListingAdditionalInformation

Field Name	Data Type	Field Description
omitTeleMktg	Enum	Valid values of CheckBoxOptionEnum. Indicates that this listing is to be omitted from any telemarketing lists.
placementOverrideCd	Enum	Valid values of OverrideCodeEnum. Identifies an override of the normal placement of business or residence listings.
profList	Enum	Valid values of CheckBoxOptionEnum. Indicates that this is a professional listing.
doNotAbbrev	Enum	Valid values of doNotAbbrevEnum. Identifies that data in specific fields must not be abbreviated.
existingAdvert	Enum	Valid values of ExistingAdvertisingEnum. Identifies the end user's directory advertising status.

Table 7–99 describes the required fields for PSRDirectoryListingDirectoryTypeSegments.

Table 7–99 Required Fields for PSRDirectoryListingDirectoryTypeSegments

Field Name	Data Type	Field Description
dirTypeCd	Enum	Valid values of DirectoryTypeEnum. Identifies the type of directory to be delivered.
dirAnnualQty	integer	Identifies the number of directories to be delivered on an annual basis.
dirNCQty	integer	Identifies the number of directories to be delivered at the time of new connection.
dirDelCd	integer	Identifies the directory code of the book to be delivered.
dirName	string	Identifies the name of the directory in which the listing is to be placed.

Table 7–100 defines AuthCDUsage.

Table 7–100 AuthCDUsage Definition

Name	Defined As	Type Description	File Name
AuthCDUsage	element	AuthCDUsageType	ServiceEntities.xsd
AuthCDUsageType	complexType	authCodeUsage	OrderManagementData.xsd
authCodeUsage	complexType	AuthCodeUsageType	OrderManagementData.xsd
AuthCodeUsageType	complexType	Contains a list of fields	OrderManagementData.xsd

Table 7–101 describes the required fields for AuthCodeUsageType.

Table 7–101 Required Fields for AuthCodeUsageType

Field Name	Data Type	Field Description
authTypeCD	string	User recognizable code for the auth type being defined.
authCD	string	Actual code assigned to the service item.
Translation	string	For a verified auth code, you must enter a verbal translation which is the response to the telephone user after the user enters the verified auth code correctly.

Table 7–101 (Cont.) Required Fields for AuthCodeUsageType

Field Name	Data Type	Field Description
authCDName	string	Describes who is using an auth code. This name can be used on the bill format.
fromEffectiveDateTime	CbeDateTime	The date the entity became effective.
toEffectiveDateTime	CbeDateTime	Date product became inactive. This field must be non-null. It may have a zero date value.
activityCD	Enum	Valid values of ActivityCodeEnumType. An enumerated type describing the activity for this item. Set this type to indicate how the operation processes this structure.

Table 7–102 defines huntGroup.

Table 7–102 huntGroup Definition

Name	Defined As	Type Description	File Name
huntGroup	element	HuntGroupType	ServiceEntities.xsd
HuntGroupType	complexType	huntGroupToFrom	OrderManagementData.xsd
huntGroupToFrom	element	HuntToFromType	OrderManagementData.xsd
HuntToFromType	complexType	huntToServItemKey, huntFromServItemKey	OrderManagementData.xsd
huntToServItemKey	element	MetaSolvServiceKey	OrderManagementData.xsd
huntFromServItemKey	element	MetaSolvServiceKey	OrderManagementData.xsd

Table 7–103 describes the required fields for HuntGroupType.

Table 7–103 Required Fields for HuntGroupType

Field Name	Data Type	Field Description
huntKey	integer	Unique Key that identifies a hunt group in the database.
huntType	string	Hunt Type of hunting requested.
huntName	string	Hunt Group Name
huntNumber	string	Number of members in the hunt group.
activityCd	Enum	Valid values of ActivityCodeEnumType. Activity to be performed on this item.

Table 7–104 describes the required fields for HuntToFromType.

Table 7–104 Required Fields for HuntToFromType

Field Name	Data Type	Field Description
activityCd	Enum	Valid values of ActivityCodeEnumType. Activity to be performed on this item.

Table 7–105 defines lineDirectoryType.

Table 7–105 lineDirectoryType Definition

Name	Defined As	Type Description	File Name
lineDirectoryType	complexType	Contains the following: ■ address ■ telNbr ■ addlInfo ■ text	OrderManagementData.xsd
Address	element	PSRDirectoryListingAddress	OrderManagementData.xsd
PSRDirectoryListingAddress	complexType	Contains a list of fields	OrderManagementData.xsd
telNbr	element	PSRDirectoryListingTelephoneNumber	OrderManagementData.xsd
PSRDirectoryListingTelephoneNumber	complexType	Contains a list of fields	OrderManagementData.xsd
delivery	element	PSRDirectoryListingDeliveryLocation	OrderManagementData.xsd
PSRDirectoryListingDeliveryLocation	complexType	Contains a list of fields PSRDirectoryListingDirectoryTypeSegments	OrderManagementData.xsd
addlInfo	element	PSRDirectoryListingAdditionalInformation	OrderManagementData.xsd
PSRDirectoryListingAdditionalInformation	complexType	Contains a list of fields	OrderManagementData.xsd
Text	element	PSRDirectoryListingLinesOfText	OrderManagementData.xsd
PSRDirectoryListingLinesOfText	complexType	Contains a list of fields	OrderManagementData.xsd
PSRDirectoryListingDirectoryTypeSegments	complexType	Contains a list of fields	OrderManagementData.xsd

Table 7–106 describes the required fields for lineDirectoryType.

Table 7–106 Required Fields for lineDirectoryType

Field Name	Data Type	Field Description
rcdType	Enum	Valid values of RecordTypeEnum
rcdTypeArea	Enum	Valid values of RecordTypeAreaEnum
referenceNumber	string	A number that uniquely identifies a delivery location.
listingType	Enum	Valid values of ListingTypeEnum. For DSR, identifies the type of listing being submitted for the publication and directory assistance (DA) appearance rules.
styleCd	Enum	Valid values of StyleCodeEnum. Identifies if the listing is a straight line, caption header.
listAcctType	Enum	Valid values of TypeOfAccountEnum. Identifies the type of account for this listing and determines the placement of the listing in split directories and directory assistance.
degreeofIdent	Long	Identifies the degree of indention for this listing.

Table 7–106 (Cont.) Required Fields for lineDirectoryType

Field Name	Data Type	Field Description
listedNmLineage Dual	Enum	Valid values of LineageNameEnum
listedNmAddrD ual	Enum	Valid values of AddressNameEnum
ALICd	string	Uniquely identifies each listing for an account number
Yphdgcd	string	Identifies the heading under which a business listing should appear in the Yellow Pages Directory.
Yphdgverbiage	string	Identifies the heading under which a business listing should appear in the Yellow Pages.
sicCode	string	Industry code to identify yellow page heading.

createOrderRequestResponse

The createOrderRequestResponse element contains the output information for the operation. The information returned in the response indicates if the operation is successful. Table 7–107 describes the returned information in the response.

Table 7–107 Payload Information for the Response

Name	Defined As	Type Description	File Name
createOrderRequestResponse	element	createOrderByValueResponse	OrderAPI.wsdl
createOrderByValueResponse	element	mommekey	OrderManagementAPI.xsd
mommekey	element	MetaSolvOrderKeyChoice	OrderManagementAPI.xsd
MetaSolvOrderKeyChoice	complexType	metaSolvOrderKey	OrderManagementEntities.xsd

createPSROrderRequest Operation

This operation creates a new PSR order in the system and returns the key for the new order created. A single OrderValue is the only value passed into the request. The state values in OrderValue are ignored; the state is initialized to STARTED by the system.

The following are the request and response structures:

Request Structure: createPSROrderRequest

Response Structure: createPSROrderRequestResponse

createPSROrderRequest

The createPSROrderRequest element contains the input information for the operation. Each row in Table 7–108 describes the element or type name, the XSD declaration, the type description, and the file name that contains the item's definition.

Table 7–108 Payload Information for the Request

Name	Defined As	Type Description	File Name
createPSROrderRequest	element	createOrderByValueRequest	OrderAPI.wsdl
createOrderByValueRequest	element	Mommevalue	OrderManagementA PI.xsd

Table 7–108 (Cont.) Payload Information for the Request

Name	Defined As	Type Description	File Name
Mommevalue	element	MetaSolvOrderValueChoice	OrderManagementAPI.xsd
MetaSolvOrderValueChoice	complexType	Contains a choice of types: ▪ metaSolvPSROrderValue ▪ metaSolvISROrderValue	OrderManagementEntities.xsd
metaSolvPSROrderValue	complexType	Described in Table 7–36, " metaSolvPSROrderValue Definition"	OrderManagementEntities.xsd

createPSROrderRequestResponse

The createPSROrderRequestResponse element contains the output information for the operation. The information returned in the response indicates if the operation is successful. Table 7–109 describes the returned information in the response.

Table 7–109 Payload Information for the Response

Name	Defined As	Type Description	File Name
createPSROrderRequestResponse	element	createOrderByValueResponse	OrderAPI.wsdl
createOrderByValueResponse	element	mommekey	OrderManagementAPI.xsd
mommekey	element	MetaSolvOrderKeyChoice	OrderManagementAPI.xsd
MetaSolvOrderKeyChoice	complexType	metaSolvOrderKey	OrderManagementEntities.xsd
metaSolvOrderKey	element	OrderKey	OrderManagementEntities.xsd
OrderKey	complexType	Contains a list of fields	OrderManagementEntities.xsd

Table 7–110 describes the error messages for the operation.

Table 7–110 Error Messages for the Operation

Error Message	Cause	Resolution
Miscellaneous Error: Telephone Number already exists in inventory and has status of statusValue.	The telephone number you specified already exists in the database.	Provide a new telephone number.
Telephone Number already exists in inventory with a different telephone number type.	The telephone number type you specified does not match the type in the database or the telephone number already exists.	Retrieve the type for the telephone number and provide the matching type or provide a new telephone number.
Unable to copy item valueForItem to the order as the Service Item is on another open PSR Order. The Preference to Copy Pending PSR Items is set to No.	You requested a copy of a service item that is on an open PSR.	Remove the request to copy the item or change the preference for the Copy Pending PSR Items to Yes.

getCnamData Operation

This operation retrieves the Calling Name (CNAM) records. This operation provides the extract sequence for updating the Calling Name information.

The following are the request and response structures:

Request Structure: getCnamData

Response Structure: getCnamDataResponse

getCnamData

The getCnamData element contains the input information for the operation. Each row in Table 7–111 describes the element or type name, the XSD declaration, the type description, and the file name that contains the item's definition.

Table 7–111 Payload Information for the Request

Name	Defined As	Type Description	File Name
getCnamData	element	getCNAMDataRequest	OrderAPI.wsdl

getCnamDataResponse

The getCnamDataResponse element contains the output information for the operation. The information returned in the response indicates if the operation is successful. Table 7–112 describes the returned information in the response.

Table 7–112 Payload Information for the Response

Name	Defined As	Type Descriptions	File Name
getCnamDataResponse	element	momGetCNAMDataResponse	OrderAPI.wsdl
momGetCNAMDataResponse	element	CNAMDataValue	OrderManagementAPI.xsd
CNAMDataValue	element	CNAMDataType	OrderManagementAPI.xsd
CNAMDataType	complexType	CNAMRecords	OrderManagementEntities.xsd
CNAMRecords	element	CNAMRecordType	OrderManagementEntities.xsd
CNAMRecordType	complexType	Contains a list of fields	OrderAncillaryManagementData.xsd

getE911Data Operation

This operation retrieves the E911 records. This operation provides the extract sequence for updating the E911 information.

The following are the request and response structures:

Request Structure: getE911Data

Response Structure: getE911DataResponse

getE911Data

The getE911Data element contains the input information for the operation. Each row in Table 7–113 describes the element or type name, the XSD declaration, the type description, and the file name that contains the item's definition.

Table 7–113 Payload Information for the Request

Name	Defined As	Type Description	File Name
getE911Data	element	getE911DataRequest	OrderAPI.wsdl

getE911DataResponse

The getE911DataResponse element contains the output information for the operation. The information returned in the response indicates if the operation is successful. Table 7–114 describes the returned information in the response.

Table 7–114 Payload Information for the Response

Name	Defined As	Type Description	File Name
getE911DataResponse	element	momGetE911DataResponse	OrderAPI.wsdl
momGetE911DataResponse	element	e911DataValue	OrderManagementAPI.xsd
e911DataValue	element	E911DataType	OrderManagementAPI.xsd
E911DataType	complexType	e911Records	OrderManagementEntities.xsd
e911Records	element	E911RecordType	OrderManagementEntities.xsd
E911RecordType	complexType	Contains a list of fields	OrderAncillaryManagementData.xsd

getLidbData Operation

This operation retrieves the Line Database (LIDB) records. This operation provides the extract sequence used for updating the Line Database details.

The following are the request and response structures:

Request Structure: getLidbData

Response Structure: getLidbDataResponse

getLidbData

The getLidbData element contains the input information for the operation. Each row in Table 7–115 describes the element or type name, the XSD declaration, the type description, and the file name that contains the item's definition.

Table 7–115 Payload Information for the Request

Name	Defined As	Type Description	File Name
getLidbData	element	getLIDBDataRequest	OrderAPI.wsdl

getLidbDataResponse

The getLidbDataResponse element contains the output information for the operation. The information returned in the response indicates if the operation is successful. Table 7–116 describes the returned information in the response.

Table 7–116 Payload Information for the Response

Name	Defined As	Type Description	File Name
getLidbDataResponse	element	momGetLIDBDataResponse	OrderAPI.wsdl
momGetLIDBDataResponse	element	LIDBDataValue	OrderManagementAPI.xsd
LIDBDataValue	element	LIDBDataType	OrderManagementAPI.xsd
LIDBDataType	complexType	LIDBRecords	OrderManagementEntities.xsd
LIDBRecords	element	LIDBRecordType	OrderManagementEntities.xsd
LIDBRecordType	complexType	Contains a list of fields	OrderAncillaryManagementData.xsd

getOrderByKey Operation

This operation retrieves the order details given an input order number (document number of the order). The output consists of order header, user data, and services information.

The following are the request and response structures:

Request Structure: getOrderByKey

Response Structure: getOrderByKeyResponse

getOrderByKey

The getOrderByKey element contains the input information for the operation. Each row in Table 7–117 describes the element or type name, the XSD declaration, the type description, and the file name that contains the item's definition.

Table 7–117 Payload Information for the Request

Name	Defined As	Type Description	File Name
getOrderByKey	element	momGetOrderByKeyRequest	OrderAPI.wsdl
momGetOrderByKeyRequest	element	mommekey	OrderManagementAPI.xsd
Mommekey	element	MetaSolvOrderKeyChoice	OrderManagementAPI.xsd
MetaSolvOrderKeyChoice	complexType	metaSolvOrderKey	OrderManagementEntities.xsd
metaSolvOrderKey	element	OrderKey	OrderManagementEntities.xsd
OrderKey	complexType	primaryKey	OrderManagementEntities.xsd

Table 7–118 describes the required fields for getOrderByKey.

Table 7–118 Required Fields for getOrderByKey

Field Name	Data Type	Field Description
PrimaryKey	string	Document number of the order where the details are required.

getOrderByKeyResponse

The getOrderByKeyResponse element contains the output information for the operation. The information returned in the response indicates if the operation is successful. Table 7–119 describes the returned information in the response.

Table 7–119 Payload Information for the Response

Name	Defined As	Type Description	File Name
getOrderByKeyResponse	element	momGetOrderByKeyResponse	OrderAPI.wsdl
momGetOrderByKeyResponse	element	Mommevalue	OrderManagementAPI.xsd
mommevalue	element	MetaSolvOrderValueChoice	OrderManagementAPI.xsd
MetaSolvOrderValueChoice	complexType	Contains the following: ▪ metaSolvPSROrderValue ▪ metaSolvISROrderValue	OrderManagementEntities.xsd
metaSolvPSROrderValue	complexType	Described in Table 7–36, " metaSolvPSROrderValue Definition"	OrderManagementEntities.xsd
metaSolvISROrderValue	complexType	Described in Table 7–41, " metaSolvISROrderValue Definition"	OrderManagementEntities.xsd

getPSROrderByTN Operation

This operation retrieves the PSR order details given the input telephone number and TN format. The output includes the order header, user data and service details.

The following are the request and response structures:

Request Structure: getPSROrderByTN

Response Structure: createPSROrderRequestResponse

getPSROrderByTN

The getPSROrderByTN element contains the input information for the operation. Each row in Table 7–120 describes the element or type name, the XSD declaration, the type description, and the file name that contains the item's definition.

Table 7–120 Payload Information for the Request

Name	Defined As	Type Description	File Name
getPSROrderByTN	element	getPSROrderByTNRequest	OrderAPI.wsdl
getPSROrderByTNRequest	element	GetPSROrderByTNRequestValue	OrderManagementAPI.xsd
GetPSROrderByTNRequestValue	element	GetPSROrderByTNRequestValueType	OrderManagementAPI.xsd
GetPSROrderByTNRequestValueType	complexType	Contains a list of fields	OrderManagementData.xsd

Table 7–121 describes the required fields for getPSROrderByTN.

Table 7–121 Required Fields for getPSROrderByTN

Field Name	Data Type	Field Description
TelephoneNbr	string	Element is used to pass the telephone number and the details are required.
TelephoneNbrFormat	string	Element is used to specify the format of the telephone number. For example in case of US format i.e NPA-NXX-LINE the format is 'xxx-xxx-xxxx'.

getPSROrderByTNResponse

The getPSROrderByTNResponse element contains the output information for the operation. The information returned in the response indicates if the operation is successful. Table 7–122 describes the returned information in the response.

Table 7–122 Payload Information for the Response

Name	Defined As	Type Description	File Name
getPSROrderByTNResponse	element	momGetPSROrderByTNResponse	OrderAPI.wsdl
momGetPSROrderByTNResponse	element	mommevalue	OrderManagementAPI.xsd
mommevalue	element	MetaSolvOrderValueChoice	OrderManagementAPI.xsd
MetaSolvOrderValueChoice	complexType	Contains the following: ■ metaSolvPSROrderValue ■ metaSolvISROrderValue	OrderManagementEntities.xsd
metaSolvPSROrderValue	complexType	Described in Table 7–36, " metaSolvPSROrderValue Definition"	OrderManagementEntities.xsd
metaSolvISROrderValue	complexType	Described in Table 7–41, " metaSolvISROrderValue Definition"	OrderManagementEntities.xsd

processSuppOrderRequest Operation

This operation requests a supplement to the order. It takes in the supplement type, and order number. You have the option to reopen tasks and gateway events attached to the order.

The following are the request and response structures:

Request Structure: processSuppOrderRequest

Response Structure: assignProvPlanRequestResponse

processSuppOrderRequest

The processSuppOrderRequest element contains the input information for the operation. Each row in Table 7–123 describes the element or type name, the XSD declaration, the type description, and the file name that contains the item's definition.

Table 7–123 Payload Information for the Request

Name	Defined As	Type Description	File Name
processSuppOrderRequest	element	momProcessSuppOrderRequest	OrderAPI.wsdl
momProcessSuppOrderRequest	element	ProcessSuppOrderRequestValueType	OrderManagementAPI.xsd
ProcessSuppOrderRequestValueType	complexType	orderKey	OrderManagementData.xsd
orderKey	element	OrderKey	OrderManagementData.xsd
OrderKey	complexType	Contains a list of fields	OrderManagementEntities.xsd

Table 7–124 describes the required fields for processSuppOrderRequest.

Table 7–124 Required Fields for processSuppOrderRequest

Field Name	Data Type	Field Description
suppType	Enum	Valid values of SuppTypeEnumType. This field indicates the reason that the order is being supplemented, or changed.
reopenTasksIndicator	boolean	Indicates whether the found tasks need to be re-opened for later processing.
reopenGatewayEventsIndicator	boolean	Indicates whether the found Gateway Events need to re-opened for later processing.

Table 7–125 describes the required fields for OrderKey.

Table 7–125 Required Fields for OrderKey

Field Name	Data Type	Field Description
PrimaryKey	string	Document number of the order where the details are required.

processSuppOrderRequestResponse

The processSuppOrderRequestResponse element contains the output information for the operation. The information returned in the response indicates if the operation is successful. Table 7–126 describes the returned information in the response.

Table 7–126 Payload Information for the Response

Name	Defined As	Type Description	File Name
processSuppOrderRequestResponse	element	processSuppOrderResponse	OrderAPI.wsdl

queryOrderManagementRequest Operation

This operation retrieves the order management information for:

- validateOrderQueryValue
- getTaskGWEventQueryValue
- getServReqTasksQueryValue
- getServiceRequestDLRsValue
- getDLRInfosByOrderAndServiceItemIdValue
- getDLRInfosByServiceItemIdInServiceValue
- getServItemReferenceValue
- getServItemsValue

The following are the request and response structures:

Request Structure: queryOrderManagementRequest

Response Structure: queryOrderManagementRequestResponse

queryOrderManagementRequest

The queryOrderManagementRequest element contains the input information for the operation. Each row in Table 7–127 describes the element or type name, the XSD declaration, the type description, and the file name that contains the item's definition.

Table 7–127 Payload Information for the Request

Name	Defined As	Type Description	File Name
queryOrderManagementRequest	element	momQueryOrderManagementRequest	OrderAPI.wsdl
momQueryOrderManagementRequest	element	queryValue	OrderManagementAPI.xsd
queryValue	element	MetaSolvQueryValueChoice	OrderManagementAPI.xsd
MetaSolvQueryValueChoice	complexType	Contains a choice of: ■ getTaskGWEventQueryValue ■ getServReqTasksQueryValue ■ getServiceRequestDLRsValue ■ getDLRInfosByOrderAndServiceItemIdValue ■ getDLRInfosByServiceItemIdInServiceValue ■ getServItemReferenceValue ■ getServItemsValue ■ getProductCatalog ■ getOrderStatus	OrderManagementAPI.xsd

Table 7–128 defines getTaskGWEventQueryValue.

Table 7–128 getTaskGWEventQueryValue Definition

Name	Defined As	Type Description	File Name
getTaskGWEventQueryValue	complexType	Extension of MomQueryValue Contains the following: ▪ orderKey ▪ taskKey ▪ requestId	OrderManagementAPI.xsd
orderKey	element	OrderKey	OrderManagementAPI.xsd
taskKey	element	TaskKey	OrderManagementAPI.xsd

Table 7–129 describes the required fields for getTaskGWEventQueryValue.

Table 7–129 Required Fields for getTaskGWEventQueryValue

Field Name	Data Type	Field Description
requestId	short	Request ID for get task gateway event query.

Table 7–130 describes the required fields for orderKey.

Table 7–130 Required Fields for OrderKey

Field Name	Data Type	Field Description
primaryKey	string	Document number of the order in the OrderKey type.

Table 7–131 describes the required fields for TaskKey.

Table 7–131 Required Fields for TaskKey

Field Name	Data Type	Field Description
taskPrimaryKey	string	Task number of the order.

Table 7–132 defines getServReqTasksQueryValue.

Table 7–132 getServReqTasksQueryValue Definition

Name	Defined As	Type Description	File Name
getServReqTasksQueryValue	complexType	Extension of MomQueryValue mommekey	OrderManagementAPI.xsd
mommekey	element	mommekey	OrderManagementAPI.xsd
Mommekey	element	MetaSolvOrderKeyChoice	OrderManagementAPI.xsd
MetaSolvOrderKeyChoice	element	metaSolvOrderKey	OrderManagementEntities.xsd
metaSolvOrderKey	element	OrderKey	OrderManagementEntities.xsd

Table 7–133 describes the required fields for getServTasksQueryValue.

Table 7–133 Required Fields for getServReqTasksQueryValue

Field Name	Data Type	Field Description
timeZone	string	Time zone.

Table 7–134 defines getServiceRequestDLRsValue.

Table 7–134 getServiceRequestDLRsValue Definition

Name	Defined As	Type Description	File Name
getServiceRequestDLRsValue	element	GetServiceRequestDLRsValue	OrderManagementAPI.xsd
GetServiceRequestDLRsValue	complexType	Extension of MomQueryValue orderKey	OrderManagementAPI.xsd
orderKey	element	OrderKey	OrderManagementAPI.xsd

Table 7–135 defines getDLRInfosByOrderAndServiceItemIdValue.

Table 7–135 getDLRInfosByOrderAndServiceItemIdValue Definition

Name	Defined As	Type Description	File Name
getDLRInfosByOrderAndServiceItemIdValue	complexType	Extension of MomQueryValue orderKey	OrderManagementAPI.xsd
orderKey	element	OrderKey	OrderManagementAPI.xsd

Table 7–136 describes the required fields for getDLRInfosByOrderAndServiceItemIdValue.

Table 7–136 Required Fields for getDLRInfosByOrderAndServiceItemIdValue

Field Name	Data Type	Field Description
serviceItemId	long	Service item ID of the product where the DLR information is retrieved.

Table 7–137 defines getServItemReferenceValue.

Table 7–137 getServItemReferenceValue Definition

Name	Defined As	Type Definition	File Name
getServItemReferenceValue	complexType	Contains a list of fields	OrderManagementAPI.xsd
orderKey	element	OrderKey	OrderManagementAPI.xsd

Table 7–138 describes the required fields for getServItemReferenceValue.

Table 7–138 Required Fields for getServItemReferenceValue.

Field Name	Data Type	Field Description
serviceItemId	long	The service item ID of the product where the DLR information is retrieved.

Table 7–139 defines getProductCatalog.

Table 7–139 getProductCatalog Definition

Name	Defined As	Type Description	File Name
getProductCatalog	complexType	Contains the following: ▪ productCatalogPolicy ▪ criteria	OrderManagementAPI.xsd
productCatalogPolicy	element	productCatalogPolicy	OrderManagementAPI.xsd
criteria	element	CatalogExportCriteria	OrderManagementAPI.xsd
CatalogExportCriteria	complexType	Contains the following: ▪ specificationKey ▪ partyRoleCriteria	MIPCommonEntities.xsd
specificationKey	element	MetaSolvServiceSpecificationKey	MIPCommonEntities.xsd
MetaSolvServiceSpecificationKey	complexType	serviceSpecificationPrimaryKey	ServiceEntities.xsd

Table 7–140 defines CatalogExportCriteria.

Table 7–140 CatalogExportCriteria Definition

Name	Defined As	Type Description	File Name
CatalogExportCriteria	complexType	Contains the following: ▪ specificationKey ▪ partyRoleCriteria	MIPCommonEntities.xsd
specificationKey	element	MetaSolvServiceSpecificationKey	MIPCommonEntities.xsd
MetaSolvServiceSpecificationKey	complexType	serviceSpecificationPrimaryKey	ServiceEntities.xsd

Table 7–141 describes the required fields for productCatalogPolicy.

Table 7–141 Required Fields for productCatalogPolicy

Field Name	Data Type	Field Description
customAttribute	boolean	Specifies whether the custom attribute should be included with the product catalog.

Table 7–142 describes the required fields for CatalogExportCriteria.

Table 7–142 Required Fields for CatalogExportCriteria

Field Name	Data Type	Field Description
exportInd	boolean	Indicates if you only want level one products.
applicationUseCode	string	Application use code indicates which module uses the product catalog item created. If the application use code is Global the product catalog item is used by a PSR order.

Table 7–143 describes the required fields for MetaSolvServiceSpecificationKey.

Table 7–143 Required Fields for MetaSolvServiceSpecificationKey

Field Name	Data Type	Field Description
serviceSpecificationPrimaryKey	string	Product specification ID of the product

Table 7–144 defines getOrderStatus.

Table 7–144 getOrderStatus Definition

Name	Defined As	Type Description	File Name
getOrderStatus	complexType	Contains a list of fields	OrderManagementAPI.xsd
orderKey	element	OrderKey	OrderManagementAPI.xsd

queryOrderManagementRequestResponse

The queryOrderManagementRequestResponse element contains the output information for the operation. The information returned in the response indicates if the operation is successful. Table 7–145 describes the returned information in the response.

Table 7–145 Payload Information for the Response

Name	Defined As	Type Description	File Name
queryOrderManagementRequestResponse	element	queryOrderManagementResponse	OrderAPI.wsdl
queryOrderManagementResponse	element	queryResponse	OrderManagementAPI.xsd
queryResponse	element	MetaSolvQueryResponseChoice	OrderManagementAPI.xsd
MetaSolvQueryResponseChoice	complexType	Contains a choice of the following: ■ getTaskGWEventQueryResponse ■ getServReqTasksQueryResponse ■ getDLRInfosByOrderAndServiceItemIdResponse ■ getDLRInfosByServiceItemIdInServiceResponse ■ getServItemReferenceResponse ■ getServItemsResponse ■ getProductCatalogResponse ■ getOrderStatusResponse	OrderManagementAPI.xsd

Table 7–146 defines getTaskGWEventQueryResponse.

Table 7–146 getTaskGWEventQueryResponse Definition

Name	Defined As	Type Description	File Name
getTaskGWEventQueryResponse	complexType	Contains a list of fields	OrderManagementAPI.xsd
gWEventTypes	element	MetaSolvGatewayEventValue	OrderManagementAPI.xsd
MetaSolvGatewayEventValue	element	GatewayEventType	OrderManagementEntities.xsd
GatewayEventType	complexType	Contains a list of fields	OrderManagementEvents.xsd
gatewayEventKey	element	GatewayEventKey	OrderManagementEvents.xsd
taskKeyPredecessor	element	TaskKey	OrderManagementEvents.xsd

Table 7–147 defines GetServReqTasksQueryResponse.

Table 7–147 GetServReqTasksQueryResponse Definition

Name	Defined As	Type Description	File Name
GetServReqTasksQueryResponse	complexType	Contains a list of fields	OrderManagementAPI.xsd
predFollows	element	PredFollowType	OrderManagementAPI.xsd
PredFollowType	complexType	Contains a list of fields	OrderManagementData.xsd
taskKey	element	TaskKey	OrderManagementEvents.xsd
orderKey	element	OrderKey	OrderManagementEvents.xsd

Table 7–148 defines getDLRInfosByOrderAndServiceItemIdResponse.

Table 7–148 getDLRInfosByOrderAndServiceItemIdResponse Definition

Name	Defined As	Type Description	File Name
getDLRInfosByOrderAndServiceItemIdResponse	complexType	Contains a list of fields	OrderManagementAPI.xsd
orderKey	element	OrderKey	OrderManagementAPI.xsd
dlrInfos	element	DLRInfo	OrderManagementAPI.xsd
DLRInfo	complexType	Contains a list of fields	OrderManagementAPI.xsd

Table 7–149 defines getDLRInfosByServiceItemIdInServiceResponse.

Table 7–149 getDLRInfosByServiceItemIdInServiceResponse Definition

Name	Defined As	Type Description	File Name
getDLRInfosByServiceItemIdInServiceResponse	complexType	Contains a list of fields	OrderManagementAPI.xsd
orderKey	element	OrderKey	OrderManagementAPI.xsd
dlrInfos	element	DLRInfo	OrderManagementAPI.xsd
DLRInfo	complexType	Contains a list of fields	OrderManagementAPI.xsd

Table 7–150 defines getServItemReferenceResponse.

Table 7–150 getServItemReferenceResponse Definition

Name	Defined As	Type Description	File Name
getServItemReferenceResponse	complexType	Contains a list of fields	OrderManagementAPI.xsd
orderKey	element	OrderKey	OrderManagementAPI.xsd
serviceItemAlias	string	Contains a list of fields	OrderManagementAPI.xsd
numberKey	element	MetaSolvResourceKey	OrderManagementAPI.xsd

Table 7–151 defines getServItemsResponse.

Table 7–151 getServItemsResponse Definition

Name	Defined As	Type Description	File Name
getServItemsResponse	complexType	metaSolvServiceValue	OrderManagementAPI.xsd
metaSolvServiceValue	element	metaSolvServiceValue	OrderManagementAPI.xsd
metaSolvServiceValue	complexType	Contains a list of fields	ServiceEntities.xsd
serviceKey	element	MetaSolvServiceKey	ServiceEntities.xsd
parentSpecKey	element	MetaSolvServiceSpecificationKey	ServiceEntities.xsd
describingSpecificationKey	element	MetaSolvServiceSpecificationKey	ServiceEntities.xsd
parentServiceKey	element	MetaSolvServiceKey	ServiceEntities.xsd
serviceLocationKey	element	MetaSolvLocationKey	ServiceEntities.xsd
addressKey	element	MetaSolvResourceKey	ServiceEntities.xsd
ipNumberInventoryKey	element	MetaSolvResourceKey	ServiceEntities.xsd
serviceItemValues	element	ServiceItemValueType	ServiceEntities.xsd
assignedTelephoneNr	element	AssignedTelephoneNrType	ServiceEntities.xsd
Access	element	AccessType	ServiceEntities.xsd
isdnTrunkGroup	element	IsdnTrunkGroupType	ServiceEntities.xsd
trunkGroup	element	TrunkGroupType	ServiceEntities.xsd

Table 7–151 (Cont.) getServItemsResponse Definition

Name	Defined As	Type Description	File Name
Circuit	element	CircuitType	ServiceEntities.xsd
circuitLocation	element	CircuitLocationType	ServiceEntities.xsd
Connector	element	ConnectorType	ServiceEntities.xsd
caUsageInstance	element	CaUsageInstanceType	ServiceEntities.xsd
picInfo	element	PicInfoType	ServiceEntities.xsd
orderItemPrice	element	PSROrderItemPriceType	ServiceEntities.xsd
EmailInfo	element	EmailInfoType	ServiceEntities.xsd
DialupInfo	element	DialupInfoType	ServiceEntities.xsd
TemplateInfo	element	TemplateInfoType	ServiceEntities.xsd
ElementInfo	element	ElementInfoType	ServiceEntities.xsd
EquipmentInfo	element	EquipmentInfoType	ServiceEntities.xsd
PSRDirServ	element	PSRDirServType	ServiceEntities.xsd
PSRDirListing	element	PSRDirListingType	ServiceEntities.xsd
AuthCDUsage	element	AuthCDUsageType	ServiceEntities.xsd
huntGroup	element	HuntGroupType	ServiceEntities.xsd

Table 7–152 defines getProductCatalogResponse.

Table 7–152 getProductCatalogResponse Definition

Name	Defined As	Type Description	File Name
getProductCatalogResponse	complexType	Contains a list of fields	OrderManagementAPI.xsd
productSpecification	element	productSpecification	OrderManagementAPI.xsd
pSRSpecItemType	element	pSRSpecItemType	OrderManagementEntities.xsd
pSRSpecItemType	complexType	Contains a list of fields	OrderManagementData.xsd
specificationKey	element	MetaSolvServiceSpecificationKey	OrderManagementData.xsd
levelOneSpecKey	element	LevelOneSpecKey	OrderManagementData.xsd
parentSpecId	element	parentSpecId	OrderManagementData.xsd
Netarea	element	NetworkAreaType	OrderManagementData.xsd
Prices	element	PSRSpecItemPriceType	OrderManagementData.xsd
Labels	element	PSRDefaultValueType	OrderManagementData.xsd
levelOne	element	LevelOneUnionType	OrderManagementData.xsd
Standard	element	IndicatorEnumType	OrderManagementData.xsd
Required	element	IndicatorEnumType	OrderManagementData.xsd
global	element	IndicatorEnumType	OrderManagementData.xsd
guideInd	element	IndicatorEnumType	OrderManagementData.xsd

Table 7–152 (Cont.) getProductCatalogResponse Definition

Name	Defined As	Type Description	File Name
softProvInd	element	IndicatorEnumType	OrderManagementData.xsd
CaUsages	element	CaUsageType	OrderManagementData.xsd
partyRoles	element	PSRSpecItemPartyRoleType	OrderManagementData.xsd

Table 7–153 defines getOrderStatusResponse.

Table 7–153 getOrderStatusResponse Definition

Name	Defined As	Type Description	File Name
getOrderStatusResponse	complexType	orderStatus	OrderManagementAPI.xsd
orderStatus	element	OrderStatusEnumType	OrderManagementAPI.xsd

reopenTaskRequest Operation

This operation requests a task to be reopened. This is for a task that is already completed. This operation takes an input order number and task number.

The following are the request and response structures:

Request Structure: reopenTaskRequest

Response Structure: reopenTaskRequestResponse

reopenTaskRequest

The reopenTaskRequest element contains the input information for the operation. Each row in Table 7–154 describes the element or type name, the XSD declaration, the type description, and the file name that contains the item's definition.

Table 7–154 Payload Information for the Request

Name	Defined As	Type Description	File Name
reopenTaskRequest	element	momReopenTaskRequest	OrderAPI.wsdl
momReopenTaskRequest	element	reopenTaskValue	OrderManagementAPI.xsd
reopenTaskValue	element	ReopenTaskValueType	OrderManagementAPI.xsd
ReopenTaskValueType	complexType	Contains a list of fields	OrderManagementData.xsd
orderNumber	element	OrderKey	OrderManagementData.xsd
taskNumber	element	TaskKey	OrderManagementData.xsd

reopenTaskRequestResponse

The reopenTaskRequestResponse element contains the output information for the operation. The information returned in the response indicates if the operation is successful. Table 7–155 describes the returned information in the response.

Table 7–155 Payload Information for the Response

Name	Defined As	Type Description	File Name
reopenTaskRequestResponse	element	reopenTaskResponse	OrderAPI.wsdl
reopenTaskResponse	element	orderKey	OrderManagementAPI.xsd
orderKey	element	OrderKey	OrderManagementAPI.xsd
OrderKey	complexType	Contains a list of fields	OrderManagementEntities.xsd

startOrderRequest Operation

This operation validates the order along with a finish order option request. This operation takes an input document number (order number).

The following are the request and response structures:

Request Structure: startOrderRequest

Response Structure: startOrderRequestResponse

startOrderRequest

The startOrderRequest element contains the input information for the operation. Each row in Table 7–156 describes the element or type name, the XSD declaration, the type description, and the file name that contains the item's definition.

Table 7–156 Payload Information for the Request

Name	Defined As	Type Description	File Name
startOrderRequest	element	startOrderByKeyRequest	OrderAPI.wsdl
startOrderByKeyRequest	element	mommekey	OrderManagementAPI.xsd
Mommekey	element	MetaSolvOrderKeyChoice	OrderManagementAPI.xsd
MetaSolvOrderKeyChoice	complexType	metaSolvOrderKey	OrderManagementEntities.xsd
metaSolvOrderKey	element	OrderKey	OrderManagementEntities.xsd
OrderKey	complexType	Contains a list of fields	OrderManagementEntities.xsd

startOrderRequestResponse

The startOrderRequestResponse element contains the output information for the operation. The information returned in the response indicates if the operation is successful. Table 7–157 describes the returned information in the response.

Table 7–157 Payload Information for the Response

Name	Defined As	Type Description	File Name
startOrderRequestResponse	element	startOrderByKeyResponse	OrderAPI.wsdl
startOrderByKeyResponse	element	mommekey	OrderManagementAPI.xsd
mommekey	element	MetaSolvOrderKeyChoice	OrderManagementAPI.xsd
MetaSolvOrderKeyChoice	complexType	metaSolvOrderKey	OrderManagementEntities.xsd
metaSolvOrderKey	element	OrderKey	OrderManagementEntities.xsd
OrderKey	complexType	Contains a list of fields	OrderManagementEntities.xsd

Table 7–158 describes the error messages for the operation.

Table 7–158 Error Messages for the Operation

Error Message	Cause	Resolution
The Broadband Service Category is missing for this bandwidth circuit. Please add a Broadband Service Category value to this order.	This order cannot be started until the Broadband Service Category is added.	Add the Broadband Service Category to this circuit.
Error: This service item has required labels with no values. Please add values.	You have requested a start order but values are missing from a service item.	Put in values for the required labels on the service items.

TaskJeopardyRequest Operation

This operation adds task jeopardy information for the input task number.

The following are the request and response structures:

Request Structure: TaskJeopardyRequest

Response Structure: TaskRequestResponse

TaskJeopardyRequest

The TaskJeopardyRequest element contains the input information for the operation. Each row in Table 7–159 describes the element or type name, the XSD declaration, the type description, and the file name that contains the item's definition.

Table 7–159 Payload Information for the Request

Name	Defined As	Type Description	File Name
TaskJeopardyRequest	element	getTaskJeopardyRequest	OrderAPI.wsdl
getTaskJeopardyRequest	element	GetTaskJeopardyRequestValue	OrderManagementAPI.xsd
GetTaskJeopardyRequestValue	element	GetTaskJeopardyRequestValueType	OrderManagementAPI.xsd

Table 7–159 (Cont.) Payload Information for the Request

Name	Defined As	Type Description	File Name
GetTaskJeopardyRequestValueType	element	Contains the following: ■ orderKey ■ taskKey	OrderManagementData.xsd
orderKey	element	OrderKey	OrderManagementData.xsd
taskKey	element	TaskKey	OrderManagementData.xsd

Table 7–160 describes the required fields for TaskJeopardyRequest.

Table 7–160 Required Fields for TaskJeopardyRequest

Field Name	Data Type	Field Description
circuitId	string	Circuit ID assigned to the order.
timeZone	string	Time zone.

TaskJeopardyRequestResponse

The TaskJeopardyRequestResponse element contains the output information for the operation. The information returned in the response indicates if the operation is successful. Table 7–161 describes the returned information in the response.

Table 7–161 Payload Information for the Response

Name	Defined As	Type Description	File Name
TaskJeopardyRequestResponse	element	getTaskJeopardyResponse	OrderAPI.wsdl
getTaskJeopardyResponse	element	getTaskJeopardyReponseValue	OrderManagementAPI.xsd
getTaskJeopardyReponseValue	element	GetTaskJeopardyResponseValueType	OrderManagementAPI.xsd
GetTaskJeopardyResponseValueType	complexType	Contains a list of fields	OrderManagementData.xsd
orderKey	element	OrderKey	OrderManagementData.xsd
taskKey	element	TaskKey	OrderManagementData.xsd

TaskRequest Operation

This operation retrieves task details for the input task number and document number.

The following are the request and response structures:

Request Structure: TaskRequest

Response Structure: TaskRequestResponse

TaskRequest

The TaskRequest element contains the input information for the operation. Each row in Table 7–162 describes the element or type name, the XSD declaration, the type description, and the file name that contains the item's definition.

Table 7–162 Payload Information for the Request

Name	Defined As	Type Description	File Name
TaskRequest	element	getTaskDetailRequest	OrderAPI.wsdl
getTaskDetailRequest	element	getTaskDetailValue	OrderManagementAPI.xsd
getTaskDetailValue	element	getTaskDetailRequestValueType	OrderManagementAPI.xsd
getTaskDetailRequestValueType	complexType	Contains the following: ▪ orderKey ▪ taskKey	OrderManagementData.xsd
orderKey	element	OrderKey	OrderManagementData.xsd
taskKey	element	TaskKey	OrderManagementData.xsd

Table 7–163 describes the required fields for getTaskDetailRequestValueType.

Table 7–163 Required Fields for getTaskDetailRequestValueType

Field Name	Data Type	Field Description
timeZone	string	Time zone.

TaskRequestResponse

The TaskRequestResponse element contains the output information for the operation. The information returned in the response indicates if the operation is successful. Table 7–164 describes the returned information in the response.

Table 7–164 Payload Information for the Response

Name	Defined As	Type Description	File Name
TaskRequestResponse	element	getTaskDetailResponse	OrderAPI.wsdl
getTaskDetailResponse	element	TaskDetailResponseValue	OrderManagementAPI.xsd
TaskDetailResponseValue	element	getTaskDetailResponseValueType	OrderManagementAPI.xsd
getTaskDetailResponseValueType	complexType	Contains a list of fields	OrderManagementData.xsd
orderKey	element	OrderKey	OrderManagementData.xsd
taskKey	element	TaskKey	OrderManagementData.xsd

transferTaskRequest Operation

This operation transfers tasks from one work queue to another work queue.

The following are the request and response structures:

Request Structure: transferTaskRequest

Response Structure: transferTaskRequestResponse

transferTaskRequest

The transferTaskRequest element contains the input information for the operation. Each row in Table 7–165 describes the element or type name, the XSD declaration, the type description, and the file name that contains the item's definition.

Table 7–165 Payload Information for the Request

Name	Defined As	Type Description	File Name
transferTaskRequest	element	momTransferTaskRequest	OrderAPI.wsdl
momTransferTaskRequest	element	transferTaskValue	OrderManagementAPI.xsd
transferTaskValue	element	TransferTaskValueType	OrderManagementAPI.xsd
TransferTaskValueType	complexType	orderKey	OrderManagementAPI.xsd
orderKey	element	OrderKey	OrderManagementAPI.xsd

Table 7–166 describes the required fields for TransferTaskValueType.

Table 7–166 Required Fields for TransferTaskValueType

Field Name	Data Type	Field Description
taskNumber	long	Task that is being transferred.
currentWorkQueue	string	Work queue where a task is currently assigned.
newWorkQueue	string	Work queue where you are transferring a task.

transferTaskRequestResponse

The transferTaskRequestResponse element contains the output information for the operation. The information returned in the response indicates if the operation is successful. Table 7–167 describes the returned information in the response.

Table 7–167 Payload Information for the Response

Name	Defined As	Type Description	File Name
transferTaskRequestResponse	element	transferTaskResponse	OrderAPI.wsdl
transferTaskResponse	element	orderKey	OrderManagementAPI.xsd
orderKey	element	OrderKey	OrderManagementAPI.xsd

Table 7–168 describes the error messages for the operation.

Table 7–168 Error Messages for the Operation

Error Message	Cause	Resolution
The Transfer TO Work Queue is an invalid work queue.	For the input task, the provided work queue to transfer the task, does not exist in the database.	Populate the transfer to work queue with a valid input value that exists in the database.

updateCnamData Operation

This operation updates the details of CNAM records.

The following are the request and response structures:

Request Structure: updateCnamData

Response Structure: updateCnamDataResponse

updateCnamData

The updateCnamData element contains the input information for the operation. Each row in Table 7–169 describes the element or type name, the XSD declaration, the type description, and the file name that contains the item's definition.

Table 7–169 Payload Information for the Request

Name	Defined As	Type Description	File Name
updateCnamData	element	updateCNAMDataRequest	OrderAPI.wsdl
updateCNAMDataRequest	element	CNAMDataValue	OrderManagementAPI.xsd
CNAMDataValue	element	CNAMDataType	OrderManagementAPI.xsd
CNAMDataType	complexType	CNAMRecords	OrderManagementEntities.xsd
CNAMRecords	element	CNAMRecordType	OrderManagementEntities.xsd
CNAMRecordType	complexType	Contains a list of fields	OrderAncillaryManagementData.xsd

Table 7–170 describes the required fields for CNAMDataType.

Table 7–170 Required Fields for CNAMDataType

Field Name	Data Type	Field Description
extractSequence	long	Oracle-generated sequence that identifies a group of CNAM records returned from a call to getCNAMDataRequest. You must pass back the same extractSequence value for the corresponding set of extract records in the call to updateCNAMDataRequest.

Table 7–171 describes the required fields for CNAMRecordType.

Table 7–171 Required Fields for CNAMRecordType

Field Name	Data Type	Field Description
bsp	string	Billing service provider.
callingNm	string	Name that is shown on the called party's caller ID.
Line	string	Line portion of the telephone number. Valid values are a 4-digit number created using numeric characters (0-9).
Npa	string	Numbering plan area of the telephone number. Valid values are a 3-digit number created using numeric characters (0-9).
Nxx	string	Exchange portion of the telephone number. NXX is also known as the central office (CO) prefix or code. Valid values are a 3-digit number created using numeric characters (0-9).
ocn	string	Number assigned to a local exchange carrier or an interexchange carrier. This field is also referred to as the company code (CC).
presCd	CharType	Code that indicates if the name should be presented when the called party has caller ID. Valid values are ALLOW and RESTRICT.
rao	string	Revenue accounting office (RAO) associated with the NPA NXX. The RAO processes message usage in the form of automatic message accounting (AMA).
transCd	CharType	Indicates the action taken by the database provider. Valid values are ADD, MODIFY, and DELETE.
Date	string	The effective date of the CNAM update. This date field is only recorded to resolve discrepancies. Format: YYYY-MM-DD. Example: 2015-06-01.
entrySrc	CharType	Identifies the mechanism used to send information to the LIDB/CNAM database provider.

updateCnamDataResponse

The updateCnamDataResponse elementelementelement contains the output information for the operation. The information returned in the response indicates if the operation is successful. Table 7–172 describes the returned information in the response.

Table 7–172 Payload Information for the Response

Name	Defined As	Type Description	File Name
updateCnamDataResponse	element	updateCNAMDataReponse	OrderAPI.wsdl
updateCNAMDataReponse	element	status	OrderManagementAPI.xsd

updateE911DataRequest Operation

This operation updates the status of E911 records.

The following are the request and response structures:

Request Structure: updateE911DataRequest

Response Structure: updateE911DataRequestResponse

updateE911DataRequest

The updateE911DataRequest element contains the input information for the operation. Each row in Table 7–173 describes the element or type name, the XSD declaration, the type description, and the file name that contains the item's definition.

Table 7–173 Payload Information for the Request

Name	Defined As	Type Description	File Name
updateE911DataRequest	element	momUpdateE911DataRequest	OrderAPI.wsdl
momUpdateE911DataRequest	element	e911DataValue	OrderManagementAPI.xsd
e911DataValue	element	E911DataType	OrderManagementAPI.xsd
E911DataType	complexType	e911Records	OrderManagementEntities.xsd
e911Records	element	E911RecordType	OrderManagementEntities.xsd
E911RecordType	complexType	Contains a list of fields	OrderAncillaryManagementData.xsd
E911ErrorType	complexType	e911ErrorStruc	OrderAncillaryManagementData.xsd
e911ErrorStruc	element	E911ErrorStruc	OrderAncillaryManagementData.xsd
E911ErrorStruc	complexType	Contains a list of fields	OrderAncillaryManagementData.xsd

Table 7–174 describes the required fields for E911DataType.

Table 7–174 Required Fields for E911DataType

Field Name	Data Type	Field Description
extractSequence	long	Oracle-generated sequence that identifies a group of E911 records returned from a call to getE911DataRequest. You must pass back the same extractSequence value for the corresponding set of extract records in the call to updateE911DataRequest.

Table 7–175 describes the required fields for E911RecordType.

Table 7–175 Required Fields for E911RecordType

Field Name	Data Type	Field Description
Cls	CharType	Class of service.
cpd	string	Completion date (YYYY-MM-DD).
Ctn	string	Calling telephone number.
Exd	string	Extract Date YYYY-MM-DD
Foc	CharType	Function code (function of change).
Hno	string	House number of the service location for the calling telephone number.
Nam	string	Customer name.
Ord	string	Order number.
Pcn	string	Postal community name of the service location for the calling telephone number.
Sta	string	State/province of the service location for the calling telephone number.
Stn	string	Street name of the service location for the calling telephone number.

Table 7–176 describes the required fields for E911ErrorStruc.

Table 7–176 Required Fields for E911ErrorStruc

Field Name	Data Type	Field Description
Code	integer	Error code.
Reason	string	Reason for the error.

updateE911DataRequestResponse

The updateE911DataRequestResponse element contains the output information for the operation. The information returned in the response indicates if the operation is successful. Table 7–177 describes the returned information in the response.

Table 7–177 Payload Information for the Response

Name	Defined As	Type Description	File Name
updateE911DataRequestResponse	element	updateE911DataReponse	OrderAPI.wsdl
updateE911DataReponse	element	status	OrderManagementAPI.xsd

updateEstimationCompletedDateRequest Operation

This operation updates the estimated completion date for an input task. The request takes an input document number, task number and estimated completion date.

The following are the request and response structures:

Request Structure: updateEstimationCompletedDateRequest

Response Structure: updateEstimationCompletedDateRequestResponse

updateEstimationCompletedDateRequest

The updateEstimationCompletedDateRequest element contains the input information for the operation. Each row in Table 7–178 describes the element or type name, the XSD declaration, the type description, and the file name that contains the item's definition.

Table 7–178 Payload Information for the Request

Name	Defined As	Type Description	File Name
updateEstimationCompletedDateRequest	element	updateEstimatedCompletionDateRequest	OrderAPI.wsdl
updateEstimatedCompletionDateRequest	element	updateEstimatedCompletionDateValue	OrderManagementAPI.xsd
updateEstimatedCompletionDateValue	element	UpdateEstimatedCompletionDateValueType	OrderManagementAPI.xsd
UpdateEstimatedCompletionDateValueType	complexType	Contains the following: ■ orderNumber ■ taskNumber	OrderManagementData.xsd
orderNumber	element	OrderKey	OrderManagementData.xsd
taskNumber	element	TaskKey	OrderManagementData.xsd

Table 7–179 describes the required fields for UpdateEstimatedCompletionDateValueType.

Table 7–179 Required Fields for UpdateEstimatedCompletionDateValueType

Field Name	Data Type	Field Description
timeZone	string	Time zone associated with an NPA NXX/building location.
estCompDateTime	string	Work queue owner's estimate of the task completion date and time.

updateEstimationCompletedDateRequestResponse

The updateEstimationCompletedDateRequestResponse element contains the output information for the operation. The information returned in the response indicates if the operation is successful. Table 7–180 describes the returned information in the response.

Table 7–180 Payload Information for the Response

Name	Defined As	Type Description	File Name
updateEstimationCompletedDateRequestResponse	element	updateEstimatedCompletionDateResponse	OrderAPI.wsdl
updateEstimatedCompletionDateResponse	element	status	OrderManagementAPI.xsd

updateGWEventRequest Operation

This operation updates order management details for the following:

- updateOrderTaskGWEventValue
- completeTaskProcedureValue
- updateOrderTaskEventProcedureValue
- reopenTaskProcedureValue
- updateServicesInOrderProcedureValue

The following are the request and response structures:

Request Structure: updateGWEventRequest

Response Structure: updateGWEventRequestResponse

updateGWEventRequest

The updateGWEventRequest element contains the input information for the operation. Each row in Table 7–181 describes the element or type name, the XSD declaration, the type description, and the file name that contains the item's definition.

Table 7–181 Payload Information for the Request

Name	Defined As	Type Description	File Name
updateGWEventRequest	element	updateOrderManagementRequest	OrderAPI.wsdl
updateOrderManagementRequest	element	updateValue	OrderManagementAPI.xsd
updateValue	element	MetaSolvUpdateProcedureValueChoice	OrderManagementAPI.xsd

Table 7–181 (Cont.) Payload Information for the Request

Name	Defined As	Type Description	File Name
MetaSolvUpdateProcedureValueChoice	complexType	updateOrderTaskGWEventValue	OrderManagementAPI.xsd
updateOrderTaskGWEventValue	complexType	metaSolvGatewayEventKey	OrderManagementAPI.xsd
metaSolvGatewayEventKey	element	GatewayEventKey	OrderManagementEntities.xsd
GatewayEventKey	complexType	Contains the following: ■ orderKey ■ taskKey ■ serviceKey	OrderManagementEvents.xsd
orderKey	element	OrderKey	OrderManagementEntities.xsd
taskKey	element	TaskKey	OrderManagementEntities.xsd
serviceKey	element	MetaSolvServiceKey	ServiceEntities.xsd

Table 7–182 describes the required fields for updateOrderTaskGWEventValue.

Table 7–182 Required Fields for updateOrderTaskGWEventValue

Field Name	Data Type	Field Description
statusCode	Enum	Valid values of GatewayEventStatusEnumType. Current state of the gateway event.
requestId	short	Request ID

Table 7–183 describes the required fields for serviceKey.

Table 7–183 Required Fields for serviceKey

Field Name	Data Type	Field Description
servicePrimaryKey	string	Service item ID of the product

updateGWEventRequestResponse

The updateGWEventRequestResponse element contains the output information for the operation. The information returned in the response indicates if the operation is successful. Table 7–184 describes the returned information in the response.

Table 7–184 Payload Information for the Response

Name	Defined As	Type Description	File Name
updateGWEventRequestResponse	element	updateOrderTaskGWEventResponse	OrderAPI.wsdl
updateOrderTaskGWEventResponse	element	UpdateOrderTaskGWEventResponse	OrderManagementAPI.xsd
UpdateOrderTaskGWEventResponse	complexType	Extension of UpdateProcedureResponse metaSolvGatewayEventKey	OrderManagementAPI.xsd

Table 7–184 (Cont.) Payload Information for the Response

Name	Defined As	Type Description	File Name
metaSolvGatewayEventKey	element	GatewayEventKey	OrderManagementEntities.xsd
GatewayEventKey	complexType	Extension of EventKey Contains the following: ■ orderKey ■ taskKey ■ serviceKey	OrderManagementEvents.xsd
EventKey	complexType	Extension of ManagedEntityKey	OrderManagementEvents.xsd
orderKey	element	OrderKey	OrderManagementEvents.xsd
taskKey	element	TaskKey	OrderManagementEvents.xsd
serviceKey	element	MetaSolvServiceKey	OrderManagementEvents.xsd

updateLidbData Operation

This operation updates the details of LIDB records.

The following are the request and response structures:

Request Structure: updateLidbData

Response Structure: updateLidbDataResponse

updateLidbData

The updateLidbData element contains the input information for the operation. Each row in Table 7–185 describes the element or type name, the XSD declaration, the type description, and the file name that contains the item's definition.

Table 7–185 Payload Information for the Request

Name	Defined As	Type Description	File Name
updateLidbData	element	updateLIDBDataRequest	OrderAPI.wsdl
updateLIDBDataRequest	element	LIDBDataValue	OrderManagementAPI.xsd
LIDBDataValue	element	LIDBDataType	OrderManagementAPI.xsd
LIDBDataType	complexType	LIDBRecords	OrderManagementEntities.xsd
LIDBRecords	element	LIDBRecordType	OrderManagementEntities.xsd
LIDBRecordType	complexType	Contains a list of fields	OrderAncillaryManagementData.xsd

Table 7–186 describes the required fields for LIDBDataType.

Table 7–186 Required Fields for LIDBDataType

Field Name	Data Type	Field Description
extractSequence	long	Oracle-generated sequence that identifies a group of LIDB records returned from a call to getLIDBDataRequest. You must pass back the same extractSequence value for the corresponding set of extract records in the call to updateLIDBDataRequest.

Table 7–187 describes the required fields for LIDBRecordType.

Table 7–187 Required Fields for LIDBRecordType

Field Name	Data Type	Field Description
bnsCd	CharType	Determines what kind of calls can be made from another telephone but billed to this number.
busResCd	CharType	Indicates the type of market that this telephone number is intended. Valid values are B = BUSINESS, R = RESIDENCE.
Line	string	Line portion of the telephone number. Valid values are a 4-digit number created using numeric characters (0-9).
Npa	string	Numbering plan area of the telephone number. Valid values are a 3-digit number created using numeric characters (0-9).
Nxx	string	Exchange portion of the telephone number. NXX is also known as the central office (CO) prefix or code. Valid values are a 3-digit number created using numeric characters (0-9).
ocn	string	Number assigned to a local exchange carrier or an interexchange carrier. This field is also referred to as the company code (CC).
servClassCd	string	Class of Service code. Valid values are BUS = Business, RES = Residential, PBC = Public Coin, PBN = Public non-coin.
transCd	CharType	Indicates the action taken by the database provider. Valid values are I = ADD, C = MODIFY, D = DELETE.
entrySrc	CharType	Field identifies the mechanism used to send information to the LIDB/CNAM database provider.
thirdNbrAcc	CharType	Third number acceptance indicator. Valid values are B = DENIED (This value prevents the PSR from go into Complete status), C = VERIFIED.
collCallAcc	CharType	Collect call acceptance indicator. Valid values are B = DENIED, A = VERIFIED.

updateLidbDataResponse

The updateLidbDataResponse element contains the output information for the operation. The information returned in the response indicates if the operation is successful. Table 7–188 describes the returned information in the response.

Table 7–188 Payload Information for the Response

Name	Defined As	Type Description	File Name
updateLidbDataResponse	element	updateLIDBDataReponse	OrderAPI.wsdl
updateLIDBDataReponse	element	Success	OrderManagementAPI.xsd

updateOrderRequest Operation

This operation updates order management details for the following:

- updateServicesInOrderProcedureValue
- updateOrderTaskGWEventValue
- completeTaskProcedureValue

The following are the request and response structures:

Request Structure: updateOrderRequest

Response Structure: updateOrderRequestResponse

updateOrderRequest

The updateOrderRequest element contains the input information for the operation. Each row in Table 7–189 describes the element or type name, the XSD declaration, the type description, and the file name that contains the item's definition.

Table 7–189 Payload Information for the Request

Name	Defined As	Type Description	File Name
updateOrderRequest	element	updateOrderManagementRequest	OrderAPI.wsdl
updateOrderManagementReques t	element	updateValue	OrderManagementA PI.xsd
updateValue	element	MetaSolvUpdateProcedureValueCh oice	OrderManagementA PI.xsd
MetaSolvUpdateProcedureValue Choice	complexType	Contains the following: ■ updateServicesInOrderProcedure Value ■ updateOrderTaskGWEventValue ■ completeTaskProcedureValue	OrderManagementA PI.xsd

Table 7–190 defines UpdateServicesInOrderProcedureValue.

Table 7–190 UpDateServicesInOrderProcedureValue Definition

Name	Defined As	Type Description	File Name
UpdateServicesInOrderProcedure Value	element	metaSolvPSROrderValue	OrderManagementA PI.xsd
metaSolvPSROrderValue	complexType	Described in Table 7–36, " metaSolvPSROrderValue Definition"	OrderManagementEn tities.xsd

Table 7–191 defines updateOrderTaskGWEventValue.

Table 7–191 updateOrderTaskGWEventValue Definition

Name	Defined As	Type Description	File Name
updateOrderTaskGWEventValue	element	UpdateOrderTaskGWEventValue	OrderManagementA PI.xsd

Table 7–191 (Cont.) updateOrderTaskGWEventValue Definition

Name	Defined As	Type Description	File Name
UpdateOrderTaskGWEventValue	complexType	metaSolvGatewayEventKey, statusCode	OrderManagementAPI.xsd
metaSolvGatewayEventKey	element	GatewayEventKey	OrderManagementEntities.xsd
GatewayEventKey	complexType	Contains a list of fields	OrderManagementEvents.xsd

Table 7–192 defines CompleteTaskProcedureValue

Table 7–192 CompleteTaskProcedureValue Definition

Name	Defined As	File Description	File Name
CompleteTaskProcedureValue	complexType	Contains a list of fields	OrderManagementAPI.xsd
whyMissInputs	element	WhyMissInputType	OrderManagementAPI.xsd
WhyMissInputType	complexType	Contains a list of fields	OrderManagementData.xsd
Mommekey	element	MetaSolvOrderKeyChoice	OrderManagementAPI.xsd
MetaSolvOrderKeyChoice	element	metaSolvOrderKey	OrderManagementEntities.xsd
metaSolvOrderKey	element	OrderKey	OrderManagementEntities.xsd

updateOrderRequestResponse

The updateOrderRequestResponse element contains the output information for the operation. The information returned in the response indicates if the operation is successful. Table 7–193 describes the returned information in the response.

Table 7–193 updateOrderRequestResponse

Name	Defined As	Type Description	File Name
updateOrderRequestResponse	element	updateOrderManagementResponse	OrderAPI.wsdl
updateOrderManagementResponse	element	updateValueResponse	OrderManagementAPI.xsd
updateValueResponse	element	MetaSolvUpdateProcedureValueResponseChoice	OrderManagementAPI.xsd
MetaSolvUpdateProcedureValueResponseChoice	complexType	Contains the following: ■ updateServicesInOrderProcedureResponse ■ completeTaskProcedureResponse ■ updateOrderTaskGWEventResponse	OrderManagementAPI.xsd
updateServicesInOrderProcedureResponse	element	UpdateServicesInOrderProcedureResponse	OrderManagementAPI.xsd

Table 7–193 (Cont.) updateOrderRequestResponse

Name	Defined As	Type Description	File Name
UpdateServicesInOrderProcedure Response	complexType	Extension of UpdateProcedureResponse metaSolvOrderKey	OrderManagementA PI.xsd
metaSolvOrderKey	element	OrderKey	OrderManagementEn tities.xsd
OrderKey	complexType	Contains a list of fields	OrderManagementEn tities.xsd
completeTaskProcedureResponse	element	CompleteTaskProcedureResponse	OrderManagementA PI.xsd
CompleteTaskProcedureResponse	complexType	Extension of UpdateProcedureResponse mommekey	OrderManagementA PI.xsd
mommekey	element	MetaSolvOrderKeyChoice	OrderManagementA PI.xsd
MetaSolvOrderKeyChoice	complexType	metaSolvOrderKey	OrderManagementEn tities.xsd
metaSolvOrderKey	element	OrderKey	OrderManagementEn tities.xsd
OrderKey	complexType	Contains a list of fields	OrderManagementEn tities.xsd
updateOrderTaskGWEventRespo nse	element	UpdateOrderTaskGWEventRespon se	OrderManagementA PI.xsd
UpdateOrderTaskGWEventRespo nse	complexType	Extension of UpdateProcedureResponse metaSolvGatewayEventKey	OrderManagementA PI.xsd
metaSolvGatewayEventKey	element	GatewayEventKey	OrderManagementEn tities.xsd
GatewayEventKey	complexType	Extension of EventKey Contains the following: ▪ orderKey ▪ taskKey ▪ serviceKey	OrderManagementEv ents.xsd
EventKey	complexType	Extension of ManagedEntityKey Contains a list of fields	OrderManagementEv ents.xsd
orderKey	element	OrderKey	OrderManagementEv ents.xsd
OrderKey	complexType	Contains a list of fields	OrderManagementEn tities.xsd
taskKey	element	TaskKey	OrderManagementEv ents.xsd
TaskKey	complexType	Contains a list of fields	OrderManagementEn tities.xsd
serviceKey	element	MetaSolvServiceKey	OrderManagementEv ents.xsd
MetaSolvServiceKey	complexType	Contains a list of fields	ServiceEntities.xsd

updatePSROrderRequest Operation

This operation updates order management details for the following:

- updateServicesInOrderProcedureValue
- updateOrderTaskGWEventValue
- completeTaskProcedureValue

The following are the request and response structures:

Request Structure: updatePSROrderRequest

Response Structure: updatePSROrderRequestResponse

updatePSROrderRequest

The updatePSROrderRequest element contains the input information for the operation. Each row in Table 7–194 describes the element or type name, the XSD declaration, the type description, and the file name that contains the item's definition.

Table 7–194 Payload Information for the Request

Name	Defined As	Type Description	File Name
updatePSROrderRequest	element	updateOrderManagementRequest	OrderAPI.wsdl
updateOrderManagementRequest	element	updateValue	OrderManagementAPI.xsd
updateValue	element	MetaSolvUpdateProcedureValueChoice	OrderManagementAPI.xsd
MetaSolvUpdateProcedureValue Choice	complexType	Contains the following: ■ updateServicesInOrderProcedure Value ■ updateOrderTaskGWEventValue ■ completeTaskProcedureValue	OrderManagementAPI.xsd

Table 7–195 defines updateServicesInOrderProcedureValue.

Table 7–195 UpdateServicesInOrderProcedureValue

Name	Defined As	Type Description	File Name
updateServicesInOrderProcedure Value	element	metaSolvPSROrderValue	OrderManagementAPI.xsd
metaSolvPSROrderValue	complexType	Described in Table 7–36, " metaSolvPSROrderValue Definition"	OrderManagementEntities.xsd

Table 7–196 defines updateOrderTaskGWEventValue.

Table 7–196 updateOrderTaskGWEventValue

Name	Defined As	Type Description	File Name
updateOrderTaskGWEventValue	element	UpdateOrderTaskGWEventValue	OrderManagementAPI.xsd
UpdateOrderTaskGWEventValue	complexType	Extension of UpdateProcedureValue Contains a list of fields	OrderManagementAPI.xsd
statusCode	element	GatewayEventStatusEnumType	OrderManagementAPI.xsd
metaSolvGatewayEventKey	element	GatewayEventKey	OrderManagementEntities.xsd
GatewayEventKey	complexType	Contains a list of fields	OrderManagementEvents.xsd
EventKey	element	eventPrimaryKey	OrderManagementEvents.xsd
taskKey	element	taskKey	OrderManagementEvents.xsd
orderKey	element	taskKey	OrderManagementEvents.xsd
serviceKey	element	MetaSolvServiceKey	OrderManagementEvents.xsd

Table 7–197 defines CompleteTaskProcedureValue.

Table 7–197 CompleteTaskProcedureValue Definition

Name	Defined As	Type Description	File Name
CompleteTaskProcedureValue	complexType	Contains a list of fields	OrderManagementAPI.xsd
whyMissInputs	element	WhyMissInputType	OrderManagementAPI.xsd
WhyMissInputType	complexType	Contains a list of fields	OrderManagementData.xsd
Mommekey	element	MetaSolvOrderKeyChoice	OrderManagementAPI.xsd
MetaSolvOrderKeyChoice	element	metaSolvOrderKey	OrderManagementEntities.xsd
metaSolvOrderKey	element	OrderKey	OrderManagementEntities.xsd
OrderKey	complexType	Contains a list of fields	OrderManagementEntities.xsd

updatePSROrderRequestResponse

The updatePSROrderRequestResponse element contains the output information for the operation. The information returned in the response indicates if the operation is successful. Table 7–198 describes the returned information in the response.

Table 7–198 Payload Information for the Response

Name	Defined As	Type Description	File Name
updatePSROrderRequestRespons e	element	updateServicesInOrderProcedureR esponse	OrderAPI.wsdl
updateServicesInOrderProcedure Response	element	UpdateServicesInOrderProcedureR esponse	OrderManagementA PI.xsd
UpdateServicesInOrderProcedure Response	complexType	Extension of UpdateProcedureResponse metaSolvOrderKey	OrderManagementA PI.xsd
metaSolvOrderKey	element	OrderKey	OrderManagementEv ents.xsd
OrderKey	complexType	Contains a list of fields	OrderManagementEn tities.xsd

Table 7–199 describes the error messages for the operation.

Table 7–199 Error Messages for the Operation

Error Message	Cause	Resolution
Miscellaneous Error: Telephone Number already exists in inventory and has status of statusValue.	The telephone number you specified already exists in the database.	Provide a new telephone number.
Telephone Number already exists in inventory with a different telephone number type.	The telephone number type you specified does not match the type in the database or the telephone number is invalid.	Retrieve the type for the telephone number and provide the matching type or provide a new telephone number.
Unable to copy item valueForItem to the order as the Service Item is on another open PSR Order. The Preference to Copy Pending PSR Items is set to No.	You requested a copy of a service item that is on an open PSR.	Remove the request to copy the item or change the preference for the Copy Pending PSR Items to Yes.

8 Activation Web Service Reference

This chapter provides information about Oracle Communications MetaSolv Solution (MSS) Activation Web Service.

About the Activation Web Service

The Activation Web Service enables an external system to retrieve activation information from MSS. You use the Activation Web Service operation for the action that gets service data.

About the Activation Web Service Packaging

The Activation Web Service is packaged in the **MSS_WebService.ear** file, which contains the **activation.war** file. When the installer deploys the **EAR** file, the Activation Web Service is automatically deployed and ready to use.

Note: The **MSS_WebService.ear** file also includes the other MSS web service operations. See Chapter 1, "Web Services Overview" for information about these operations.

About the Activation WSDL, WAR, and Schema Files

The Activation Web Service is defined by the **ServiceActivationAPI.wsdl** file and is supported by numerous schema files. The WSDL file and supporting schema files are located in the **activation.war** file.

See "Understanding How MSS Defines Web Services" for more information about WSDL and WAR files, and about their directory locations in the EAR file.

About Activation Schema Files

Several schema files support the Activation Web Service. Within **activation.war** file, the schema files are located in the **WEB-INF/wsdls** directory. These schemas are categorized as common schemas, entity schemas, and data schemas.

Common Schemas

See "About Schema Files" for information about the common schema files.

Entity Schemas

The entity schemas define elements, such as keys and types, specific to the web service.

The Activation entity schemas are defined in the following files:

- CustomerManagementEntities.xsd
- InventoryManagementEntities.xsd
- MIPCommonEntities.xsd
- OrderManagementEntities.xsd
- OrderManagementEvents.xsd
- ServiceEntities.xsd

Data Schemas

The data schemas contain numerous complex type structures, enumerations, and simple types.

The Activation data schemas are defined in the following files:

- CustomerManagementData.xsd
- InventoryManagementData.xsd
- OrderAncillaryManagementData.xsd
- OrderManagementData.xsd
- ServiceData.xsd

API Schemas

The API schemas contain the high-level response and request type definitions and exception definitions.

The Activation API schemas are defined in the following files:

- InventoryManagementAPI.xsd
- OrderManagementAPI.xsd
- ServiceActivationAPI.xsd

getActivationData Operation

The getActivationData operation enables external systems to get activation data for the given order that will be used for switch activation.

The following are the request and response structures:

Request Structure: getActivationData

Response Structure: getActivationDataResponse

getActivationData

The getActivationData element contains the input information for the operation. Each row in Table 8–1 describes the element or type name, the XSD declaration, the type description, and the file name that contains the item's definition.

Table 8–1 Payload Information for the Request

Name	Defined As	Type Description	File Name
getActivationData	element	getActivationDataByKeyRequest	ServiceActivationAPI.wsdl
getActivationDataByKeyRequest	element	Contains a list of fields	OrderManagementAPI.xsd

Table 8–2 describes the required fields for getActivationDataByKeyRequest.

Table 8–2 Required Fields

Field Name	Data Type	Field Description
orderKey	OrderKey	A system-assigned identifier for the document number.
serviceKey	MetaSolvServiceKey	A system-assigned identifier for the Service Item.

getActivationDataResponse

The getActivationDataResponse element contains the output information for the operation. The information returned in the response indicates if the operation is successful. Table 8–3 describes the returned information in the response.

Table 8–3 Payload Information for the Response

Name	Defined As	Type Description	File Name
getActivationDataResponse	element	getActivationDataByKeyResponse	ServiceActivationAPI.wsdl
getActivationDataByKeyResponse	element	metaSolveServiceActivationValue	OrderManagementAPI.xsd
metaSolvServiceActivationValue	element	MetaSolveServiceActivationType	ServiceEntities.xsd
MetaSolvServiceActivationType	complexType	Extension of ManagedEntityKey Includes the following: ▪ orderDetails ▪ switchActivation ▪ networkElements ▪ networkSystems	ServiceEntities.xsd

Table 8–4 describes the error messages for the operation.

Table 8–4 Error Messages for the Operation

Error Message	Cause	Resolution
Document Number is not populated	Document number is not populated in the input.	Populate the document number in the input structure.
Serv Item Id is a required parameter	Service item id is not populated in the input.	Populate the service item id in the input structure.

9 SOA Web Service Reference

This chapter provides information about Oracle Communications MetaSolv Solution (MSS) Service Order Activation (SOA) Web Service.

About the SOA Web Service

The SOA Web Service enables an external system to activate services for previously placed orders in MetaSolv Solution. SOA Web Service operations enable you to:

- Create a service order activation message.
- Get service order activation information and defaults.
- Get service order activation telephone numbers for an order.
- Set a telephone number for service activation order completion.

About the SOA Web Service Packaging

The SOA Web Service is packaged in the **MSS_WebService.ear** file, which contains the **soa.war** file. When the installer deploys the **EAR** file, the SOA Web Service is automatically deployed and ready to use.

Note: The **MSS_WebService.ear** file includes all of the other provided web service operations as well. See "Web Services Overview" for information about the full list of these operations.

About the SOA WSDL, WAR, and Schema Files

The SOA Web Service is defined by the **SOAAPI.wsdl** file and is supported by numerous schema files. The WSDL file and supporting schema files are located in the **soa.war** file.

See "Understanding How MSS Defines Web Services" for more information about WSDL and WAR files, and about their directory locations in the EAR file.

About SOA Schema Files

Several schema files support the SOA Web Service. Within **soa.war** file, the schema files are located in the **WEB-INF/wsdls** directory. These schemas are categorized as common schemas, entity schemas, and data schemas.

Common Schemas

See, "About Schema Files" for information about the common schema files.

Entity Schemas

The entity schemas define elements, such as keys and types, specific to the web service.

The SOA entity schemas are defined in the following files:

- CustomerManagementEntities.xsd
- InventoryManagementEntities.xsd
- MIPCommonEntities.xsd
- OrderManagementEntities.xsd
- OrderManagementEvents.xsd
- ServiceEntities.xsd
- ServiceOrderActivationEntities.xsd

Data Schemas

The data schemas contain numerous complex type structures, enumerations, and simple types.

The SOA data schemas are defined in the following files:

- CustomerManagementData.xsd
- InventoryManagementData.xsd
- OrderAncillaryManagementData.xsd
- OrderManagementData.xsd
- ServiceData.xsd
- ServiceOrderActivationData.xsd

API Schemas

The API schemas contain the high-level response and request type definitions and exception definitions.

The SOA API schemas are defined in the following files:

- OrderManagementAPI.xsd
- ServiceOrderActivationAPI.xsd

createSoaMessageRequest Operation

The createSoaMessageRequest operation enables you to create and to save a SOA transaction request or response. It takes in a SOATransactionType object, validates it, and saves it to the database. The value of requestType or reponseType within the SOATransactionType indicates if a SOA transaction request or SOA transaction response is created. The value returned is the internal transaction ID or SOATransactionKey object.

The following are the request and response structures:

Request Structure: createSoaMessageRequest

Response Structure: createSoaMessageRequestResponse

createSoaMessageRequest

The createSoaMessageRequest element contains the input information for the operation. Each row in Table 9–1 describes the element or type name, the XSD declaration, the type description, and the file that contains the item's definition.

Table 9–1 Payload Information for the Request

Name	Defined As	Type Description	File Name
createSoaMessageRequest	element	msoaCreateSOAMessageRequest	SOAAPI.wsdl
msoaCreateSOAMessageRequest	element	Contains the following: ■ orderKey ■ transaction	ServiceOrderActivationAPI.xsd
orderKey	element	OrderKey	ServiceOrderActivationAPI.xsd
transaction	element	SOATransactionType	ServiceOrderActivationAPI.xsd
SOATransactionType	complexType	Extension of ManagedEntityValue Contains a list of fields	ServiceOrderActivationEntities.xsd
transactionId	element	SOATransactionKey	ServiceOrderActivationEntities.xsd
SOATransactionKey	complexType	Extension of ManagedEntityKey Contains a list of fields	ServiceOrderActivationEntities.xsd
lrn	element	LocalRoutingNumberType	ServiceOrderActivationEntities.xsd
LocalRoutingNumberType	complexType	Contains a list of fields	ServiceOrderActivationData.xsd

Required fields for createSoaMessageRequest

The following tables describe the required fields for the createSoaMessageRequest operation.

Table 9–2 describes the required fields for SOATransactionType.

Table 9–2 Required Fields for SOATransactionType

Field Name	Data Type	Field Description
requestType	Enum	Valid values of SOAMessageRequestEnumType. The request type of the SOA transaction. If this field is populated, the message type is a request.
responseType	Enum	Valid values of SOAMessageResponseEnumType. The response type of the SOA transaction.
telephoneNumber	string	Telephone number for which the SOA transaction exists.
svId	string	Subscription version identifier. This is from Number Portability Administration Center (NPAC).
sendIndicator	boolean	MetaSolv specific field that a user can mark a request as ready to send. If true it is ready to send. If this value is not provided, it will default to false.
messageDate	CbeDateTime	Date the request was sent, or the response was received.

Table 9–2 (Cont.) Required Fields for SOATransactionType

Field Name	Data Type	Field Description
tnActivityIndicator	Enum	Valid values of SOATelNumActivityIndEnumType. Indicates the activity operation on the telephone number for the order.
supplementType	Enum	Valid values of SupplementTypeEnum. Identifies the reason a supplement is being issued for a particular order.
telephoneNumberTypeCode	Enum	Valid values of TelephoneNumberTypeEnumType. Determines the type of telephone number.
portType	Enum	Valid values of SOAPortTypeEnumType. Indicates the port type of the SOA telephone number.
portToOriginal	boolean	Indicates if the telephone number is being ported to the original service provider.
lnpType	Enum	Valid values of LnpTypeEnumType. Identifies whether a subscription version is inter-service provider ported (LSPP) or intra-service provider ported (LISP) or Pooled Block Number Port (POOL).
svStatus	Enum	Valid values of SOASvStatusEnumType. Indicator set by the old Service Provider to indicate authorization or denial of Transfer of Service for the Subscription Version to the new Service Provider.
conflictDate	CbeDateTime	Date and time the Subscription Version was last placed in conflict.
activationRequestDate	CbeDateTime	Date and time the Subscription Version activation request was made by the new Service Provider.
preCancellationStatus	Enum	Valid values of SOASvStatusEnumType. Status of the Subscription Version before cancellation.
timerType	Enum	Valid values of TimerTypeEnumType. Timer type used for the subscription version.
businessHour	Enum	Valid values of BusinessHourEnumType. Business Hours used for the subscription version.
mslvRequestStatus	string	String that represents an internal status value for SOA transactions.
categoryId	string	Telephone number inventory value for the category.
subCategoryId	string	Telephone number inventory value for the sub category.

Table 9–3 describes the required fields for OrderKey.

Table 9–3 Required Fields for OrderKey

Field Name	Data Type	Field Description
primaryKey	string	Document number for the order

Table 9–4 describes the required fields for SOATransactionKey.

Table 9–4 Required Fields for SOATransactionKey

Field Name	Data Type	Field Description
primaryKey	string	This primary key is a SOATransactionKey and uniquely identifies a SOA transaction request or response.

Table 9–5 describes the required fields for LocalRoutingNumberType.

Table 9–5 Required Fields for LocalRoutingNumberType

Field Name	Data Type	Field Description
Npa	string	Simple type representing the format for the NPA portion of a telephone number.
Nxx	string	Simple type representing the format for the NPA portion of a telephone number.
line	string	Simple type representing the format for the LINE portion of a telephone number.

createSoaMessageRequestResponse

The createSoaMessageRequestResponse element contains the output information for the operation. The information returned in the response indicates if the operation is successful. Table 9–6 describes the returned information in the response.

Table 9–6 Payload Information for the Response

Name	Defined As	Type Description	File Name
createSoaMessageRequestResponse	element	createSOAMessageResponse	SOAAPI.wsdl
createSOAMessageResponse	element	SOAtransactionKey	ServiceOrderActivationAPI.xsd
SOATransactionKey	complexType	Extension of ManagedEntityKey Contains a list of fields	ServiceOrderActivationEntities.xsd

getSoaDefaultsRequest Operation

The getSoaDefaultsRequest operation enables you to retrieve the SOA defaults for the NPAC transaction from the database. The value returned is the SOADefaultsType object.

The following are the request and response structures:

Request Structure: getSoaDefaultsRequest

Response Structure: getSoaDefaultsRequestResponse

getSoaDefaultsRequest

The getSoaDefaultsRequest element contains the input information for the operation. Each row in Table 9–7 describes the element or type name, the XSD declaration, the type description, and the file that contains the item’s definition.

Table 9–7 Payload Information for the Request

Name	Defined As	Type Definition	File Name
getSoaDefaultsRequest	element	msoaGetSOADefaultsRequest	SOAAPI.wsdl
msoaGetSOADefaultsRequest	element	Contains the following: ■ orderKey ■ telephoneNumber	ServiceOrderActivationAPI.xsd

Table 9–7 (Cont.) Payload Information for the Request

Name	Defined As	Type Definition	File Name
orderKey	element	OrderKey	ServiceOrderActivationAPI.xsd
telephoneNumber	element	SOATelephoneNumberType	ServiceOrderActivationAPI.xsd
SOATelephoneNumberType	complexType	Contains a list of fields	ServiceOrderActivationData.xsd

Table 9–8 describes the required fields for SOATelephoneNumberType.

Table 9–8 Required Fields for SOATelephoneNumberType

Field Name	Data Type	Field Description
Npa	string	Simple type representing the format for the NPA portion of a telephone number.
Nxx	string	Simple type representing the format for the NPA portion of a telephone number.
line	string	Simple type representing the format for the LINE portion of a telephone number.
suffix	string	Simple type representing the format for the suffix portion of a telephone number.

getSoaDefaultsRequestResponse

The getSoaDefaultsRequestResponse element contains the output information for the operation. The information returned in the response indicates if the operation is successful. Table 9–9 describes the returned information in the response.

Table 9–9 Payload Information for the Response

Name	Defined As	Type Definition	File Name
getSoaDefaultsRequestResponse	element	getSOADefaultsResponse	SOAAPI.wsdl
getSOADefaultsResponse	element	defaults	ServiceOrderActivationAPI.xsd
defaults	element	SOADefaultsType	ServiceOrderActivationAPI.xsd
SOADefaultsType	complexType	Contains a list of fields	ServiceOrderActivationEntities.xsd

getSoaInformationRequest Operation

The getSoaInformationRequest operation enables you to retrieve SOA information for:

- Telephone numbers including telephone number inventory information
- Existing SOA requests
- Existing SOA responses

  The value returned is a SOAInformationType.

The following are the request and response structures:

Request Structure: getSoaInformationRequest

Response Structure: getSoaInformationRequestResponse

getSoaInformationRequest

The getSoaInformationRequest element contains the input information for the operation. Each row in Table 9–10 describes the element or type name, the XSD declaration, the type description, and the file that contains the item's definition.

Table 9–10 Payload Information for the Request

Name	Defined As	Type Definition	File Name
getSoaInformationRequest	element	msoaGetSOAInformationRequest	SOAAPI.wsdl
msoaGetSOAInformationRequest	complexType	Contains the following: ▪ orderKey ▪ telephoneNumber	ServiceOrderActivationAPI.xsd
orderKey	element	OrderKey	ServiceOrderActivationAPI.xsd
telephoneNumber	element	SOATelephoneNumberType	ServiceOrderActivationAPI.xsd
SOATelephoneNumberType	complexType	Contains a list of fields	ServiceOrderActivationData.xsd

getSoaInformationRequestResponse

The getSoaInformationRequestResponse element contains the output information for the operation. The information returned in the response indicates if the operation is successful. Table 9–11 describes the returned information in the response.

Table 9–11 Payload Information for the Response

Name	Defined As	Type Description	File Name
getSoaInformationRequestResponse	element	getSOAInformationResponse	SOAAPI.wsdl
getSOAInformationResponse	element	SOAInformation	ServiceOrderActivationAPI.xsd
SOAInformation	element	SOAInformationType	ServiceOrderActivationAPI.xsd
SOAInformationType	complexType	Contains the following SOATransactionTypes ▪ defaultTransactionInfo ▪ requests ▪ responses	ServiceOrderActivationEntities.xsd
defaultTransactionInfo	element	SOATransactionType	ServiceOrderActivationEntities.xsd
requests	element	SOATransactionType	ServiceOrderActivationEntities.xsd
responses	element	SOATransactionType	ServiceOrderActivationEntities.xsd
SOATransactionType	complexType	Extension of ManagedEntityValue Contains a list of fields	ServiceOrderActivationEntities.xsd

getSoaMessageToSendRequest Operation

This operation retrieves SOA transaction requests. Key inputs for this operation are the following:

- Order Number
- Telephone Number
- Gateway Event Reactivated Indicator

If the request indicator checkGatewayEventReactivated is true, the operation only returns SOA transaction requests where the gateway event has been activated. The order number is an optional input value. If the order number is not provided then all SOA transaction requests that match the telephone number input are returned. The operation returns a list of SOATransactionType objects.

The following are the request and response structures:

Request Structure:getSoaMessageToSendRequest

Response Structure:getSoaMessageToSendRequestResponse

getSoaMessageToSendRequest

The getSoaMessageToSendRequest element contains the input information for the operation. Each row in Table 9–12 describes the element or type name, the XSD declaration, the type description, and the file that contains the item's definition.

Table 9–12 Payload Information for the Request

Name	Defined As	Type Definition	File Name
getSoaMessageToSendRequest	element	msoaGetSOAInformationRequest	SOAAPI.wsdl
msoaGetSOAInformationRequest	element	Contains the following: ■ orderKey ■ telephoneNumber	ServiceOrderActivationAPI.xsd
orderKey	element	OrderKey	ServiceOrderActivationAPI.xsd
telephoneNumber	element	SOATelephoneNumberType	ServiceOrderActivationAPI.xsd
SOATelephoneNumberType	complexType	Contains a list of fields	ServiceOrderActivationData.xsd

Table 9–13 describes the required fields for getSOAMessagesToSendRequest.

Table 9–13 Required Fields for getSOAMessagesToSendRequest

Field Name	Data Type	Field Description
checkGatewayEventReactivated	boolean	This flag determines if the requests returned are only requests when the gateway event has been reactivated.

getSoaMessageToSendRequestResponse

The getSoaMessageToSendRequestResponse element contains the output information for the operation. The information returned in the response indicates if the operation is successful. Table 9–14 describes the returned information in the response.

Table 9–14 Payload Information for the Response

Name	Defined As	Type Definition	File Name
getSoaMessageToSendRequestResponse	element	getSOAMessagesToSendResponse	SOAAPI.wsdl
getSOAMessagesToSendResponse	element	transactions	ServiceOrderActivationAPI.xsd
transactions	element	SOATransactionType	ServiceOrderActivationAPI.xsd
SOATransactionType	complexType	Extension of ManagedEntityValue Contains a list of fields	ServiceOrderActivationEntities.xsd

getSoaTnsForOrderRequest Operation

This operation retrieves SOA telephone numbers for an order. The value returned is a list of existing SOA telephone numbers.

The following are the request and response structures:

Request Structure: getSoaTnsForOrderRequest

Response Structure: getSoaTnsForOrderRequestResponse

getSoaTnsForOrderRequest

The getSoaTnsForOrderRequest element contains the input information for the operation. Each row in Table 9–15 describes the element or type name, the XSD declaration, the type description, and the file that contains the item's definition.

Table 9–15 Payload Information for the Request

Name	Defined As	Type Definition	File Name
getSoaTnsForOrderRequest	element	msoaGetSOATNsForOrderRequest	SOAAPI.wsdl
msoaGetSOATNsForOrderRequest	element	Contains orderKey	ServiceOrderActivationAPI.xsd
orderKey	element	OrderKey	ServiceOrderActivationAPI.xsd

getSoaTnsForOrderRequestResponse

The getSoaTnsForOrderRequestResponse element contains the output information for the operation. The information returned in the response indicates if the operation is successful. Table 9–16 describes the returned information in the response.

Table 9–16 Payload Information for the Response

Name	Defined As	Type Definition	File Name
getSoaTnsForOrderRequestResponse	element	getSOATNsForOrderResponse	SOAAPI.wsdl

Table 9–16 (Cont.) Payload Information for the Response

Name	Defined As	Type Definition	File Name
getSOATNsForOrderResponse	element	SOAtelephoneNumbers	ServiceOrderActivationAPI.xsd
SOAtelephoneNumbers	element	SOATelephoneNumberType	ServiceOrderActivationAPI.xsd
SOATelephoneNumberType	complexType	Contains a list of fields	ServiceOrderActivationData.xsd

setTnSoaCompleteRequest Operation

This operation sets the MetaSolv request status value on the last request to indicate that processing is complete for this telephone number.

The following are the request and response structures:

Request Structure: setTnSoaCompleteRequest

Response Structure: setTnSoaCompleteRequestResponse

setTnSoaCompleteRequest

The setTnSoaCompleteRequest element contains the input information for the operation. Each row in Table 9–17 describes the element or type name, the XSD declaration, the type description, and the file that contains the item's definition.

Table 9–17 Payload Information for the Request

Name	Defined As	Type Description	File Name
setTnSoaCompleteRequest	element	msoaSetTNSOACompleteRequest	SOAAPI.wsdl
msoaSetTNSOACompleteRequest	complexType	Contains the following: ▪ orderKey ▪ telephoneNumber	ServiceOrderActivationAPI.xsd
orderKey	element	OrderKey	ServiceOrderActivationAPI.xsd
telephoneNumber	element	SOATelephoneNumberType	ServiceOrderActivationAPI.xsd
SOATelephoneNumberType	complexType	Contains a list of fields	ServiceOrderActivationData.xsd

setTnSoaCompleteRequestResponse

The setTnSoaCompleteRequestResponse element contains the output information for the operation. The information returned in the response indicates if the operation is successful. Table 9–18 describes the returned information in the response.

Table 9–18 Payload Information for the Response

Name	Defined As	Type Description	File Name
setTnSoaCompleteRequestResponse	element	setTNSOACompleteResponse	SOAAPI.wsdl
setTNSOACompleteResponse	element	successfulCompletion	ServiceOrderActivationAPI.xsd

10 Engineering Work Order Web Service Reference

This chapter provides information about Oracle Communications MetaSolv Solution (MSS) Engineering Work Order Web Service.

About the Engineering Work Order Web Service

The Engineering Work Order Web Service enables an external system to create and update engineering work orders, and retrieve engineering work order details from MSS. You use the Engineering Work Order Web Service operation for the following actions:

- Create Work Order
- Update Work Order
- Get Work Order
- Create Work Order Note
- Update Work Order Note
- Process DD Supplement
- Associate Connection to Work Order
- Associate Equipment to Work Order

About the Engineering Work Order Web Service Packaging

The Engineering Work Order Web Service is packaged in the **MSS_WebService.ear** file, which contains the **ewo.war** file. When the installer deploys the **EAR** file, the Engineering Work Order Web Service is automatically deployed and ready to use.

Note: The **MSS_WebService.ear** file also includes the other MSS web service operations. See Chapter 1, "Web Services Overview" for information about these operations.

About the Engineering Work Order WSDL, WAR, and Schema Files

The Engineering Work Order Web Service is defined by the **EWOAPI.wsdl** file and is supported by numerous schema files. The WSDL file and supporting schema files are located in the **ewo.war** file.

See "Understanding How MSS Defines Web Services" in Chapter 1, "Web Services Overview" for more information about WSDL and WAR files, and about their directory locations in the EAR file.

About Engineering Work Order Schema Files

Several schema files support the Engineering Work Order Web Service. Within **ewo.war** file, the schema files are located in the **WEB-INF/wsdls** directory. These schemas are API schemas.

API Schemas

The API schemas contain the high level, field level response and request type definitions and exception definitions.

The Engineering Work Order API schemas are defined in the following files:

- EWOInventory.xsd
- EWOWorkOrder.xsd
- DataTypes.xsd

createWorkOrder Operation

The createWorkOrder operation enables external systems to create engineering work order.

The following are the request and response structures:

Request Structure: createWorkOrder

Response Structure: createWorkOrderResponse

createWorkOrder

The createWorkOrder element contains the input information for the operation. Each row in Table 10–1 describes the element or type name, the XSD declaration, the type description, and the file name that contains the item's definition.

Table 10–1 Payload Information for the Request

Name	Defined As	Type Description	File Name
createWorkOrder	element	createWorkOrderRequest	EWOAPI.wsdl
createWorkOrderRequest	element	workOrder	EWOWorkOrder.xsd
workOrder	element	WorkOrder	EWOWorkOrder.xsd
WorkOrder	ComplexType	ComplexType with a list of fields	EWOWorkOrder.xsd

Table 10–2 describes the required fields for createWorkOrder.

Table 10–2 Required Fields

Field Name	Data Type	Field Description
documentNumber	int	A system-assigned identifier for the document number. Value should be zero for creating new order.
status	OrderStatusEnumType	Valid Enumeration values of Order Status Enumeration Type

createWorkOrderResponse

The createWorkOrderResponse element contains the output information for the operation. The information returned in the response indicates if the operation is successful. Table 10–3 describes the returned information in the response.

Table 10–3 Payload Information for the Response

Name	Defined As	Type Description	File Name
createWorkOrderResponse	element	createWorkOrderReqResponse	EWOAPI.wsdl
createWorkOrderReqResponse	element	workOrder	EWOWorkOrder.xsd
workOrder	element	WorkOrder	EWOWorkOrder.xsd
WorkOrder	ComplexType	ComplexType with a list of fields[1]	EWOWorkOrder.xsd

[1] Refers to the elements, which will have definitions of their own XSD structure. See the schemas to understand the element flow.

associateConnectionToWorkOrder Operation

The associateConnectionToWorkOrder operation enables external systems to associate connection to the passed engineering work order.

The following are the request and response structures:

Request Structure: associateConnectionToWorkOrder

Response Structure: associateConnectionToWorkOrderResponse

associateConnectionToWorkOrder

The associateConnectionToWorkOrder element contains the input information for the operation. Each row in Table 10–4 describes the element or type name, the XSD declaration, the type description, and the file name that contains the item's definition.

Table 10–4 Payload Information for the Response

Name	Defined As	Type Description	File Name
associateConnectionToWorkOrder	element	associateConnectionToWorkOrder Request	EWOAPI.wsdl
associateConnectionToWorkOrder Request	element	AssociateConnectionToWorkOrder Policy	EWOInventory.xsd
AssociateConnectionToWorkOrde rPolicy	element	AssociateConnectionToWorkOrder Policy	EWOInventory.xsd
AssociateConnectionToWorkOrde rPolicy	ComplexType	ComplexType with a list of fields	EWOInventory.xsd

Table 10–5 describes the required fields for associateConnectionToWorkOrder.

Table 10–5 Required Fields

Field Name	Data Type	Field Description
activityCode	ConnectionActivityCodeE numType	Activity code for the connection on the work order. Valid Enumeration values of Activity Code Enumeration Type
connectionId	String	The identifier for the connection

associateConnectionToWorkOrderResponse

The associateConnectionToWorkOrderResponse element contains the output information for the operation. The information returned in the response indicates if the operation is successful. Table 10–6 describes the returned information in the response.

Table 10–6 Payload Information for the Response

Name	Defined As	Type Description	File Name
associateConnectionToWorkOrder Response	element	associateConnectionToWorkOrder ReqResponse	EWOAPI.wsdl
associateConnectionToWorkOrder ReqResponse	element	associateConnectionToWorkOrder ReqResponse	EWOInventory.xsd
associateConnectionToWorkOrder ReqResponse	ComplexType	orderKey	EWOInventory.xsd

associateEquipmentToWorkOrder Operation

The associateEquipmentToWorkOrder operation enables external systems to associate equipment to the passed engineering work order.

The following are the request and response structures:

Request Structure: associateEquipmentToWorkOrder

Response Structure: associateEquipmentToWorkOrderResponse

associateEquipmentToWorkOrder

The associateEquipmentToWorkOrder element contains the input information for the operation. Each row in Table 10–7 describes the element or type name, the XSD declaration, the type description, and the file name that contains the item's definition.

Table 10–7 Payload Information for the Response

Name	Defined As	Type Description	File Name
associateEquipmentToWorkOrder	element	associateEquipmentToWorkOrderR equest	EWOAPI.wsdl
associateEquipmentToWorkOrder Request	element	AssociateEquipmentToWorkOrder Policy	EWOInventory.xsd
AssociateEquipmentToWorkOrder Policy	element	AssociateEquipmentToWorkOrder Policy	EWOInventory.xsd
AssociateEquipmentToWorkOrder Policy	ComplexType	ComplexType with a list of fields	EWOInventory.xsd

Table 10–8 describes the required fields for associateEquipmentToWorkOrder.

Table 10–8 Required Fields

Field Name	Data Type	Field Description
activityCode	EquipmentActivityCodeE numType	Activity code for the equipment on the work order. Valid Enumeration values of Activity Code Enumeration Type
childEquipmentId	String	The equipment ID of the child that needs to be associated.
equipmentId	String	The equipment ID that needs to be associated.

associateEquipmentToWorkOrderResponse

The associateEquipmentToWorkOrderResponse element contains the output information for the operation. The information returned in the response indicates if the operation is successful. Table 10–9 describes the returned information in the response.

Table 10–9 Payload Information for the Response

Name	Defined As	Type Description	File Name
associateEquipmentToWorkOrderResponse	element	associateEquipmentToWorkOrderReqResponse	EWOAPI.wsdl
associateEquipmentToWorkOrderReqResponse	element	associateEquipmentToWorkOrderReqResponse	EWOInventory.xsd
associateEquipmentToWorkOrderReqResponse	ComplexType	orderKey	EWOInventory.xsd

createWorkOrderNote Operation

The createWorkOrderNote operation enables external systems to create engineering work order note.

The following are the request and response structures:

Request Structure: createWorkOrderNote

Response Structure: createWorkOrderNoteResponse

createWorkOrderNote

The createWorkOrderNote element contains the input information for the operation. Each row in Table 10–10 describes the element or type name, the XSD declaration, the type description, and the file name that contains the item's definition.

Table 10–10 Payload Information for the Response

Name	Defined As	Type Description	File Name
createWorkOrderNote	element	createWorkOrderNoteRequest	EWOAPI.wsdl
createWorkOrderNoteRequest	element	workOrderNote	EWOWorkOrder.xsd
workOrderNote	element	workOrderNote	EWOWorkOrder.xsd
workOrderNote	ComplexType	ComplexType with a list of fields	EWOWorkOrder.xsd

createWorkOrderNoteResponse

The createWorkOrderNoteResponse element contains the output information for the operation. The information returned in the response indicates if the operation is successful. Table 10–11 describes the returned information in the response.

Table 10–11 Payload Information for the Response

Name	Defined As	Type Description	File Name
createWorkOrderNoteResponse	element	createWorkOrderNoteReqResponse	EWOAPI.wsdl
createWorkOrderNoteReqResponse	element	workOrderNote	EWOWorkOrder.xsd
workOrderNote	element	workOrderNote	EWOWorkOrder.xsd
workOrderNote	ComplexType	ComplexType with a list of fields	EWOWorkOrder.xsd

updateWorkOrderNote Operation

The updateWorkOrderNote operation enables external systems to update engineering work order note.

The following are the request and response structures:

Request Structure: updateWorkOrderNote

Response Structure: updateWorkOrderNoteResponse

updateWorkOrderNote

The updateWorkOrderNote element contains the input information for the operation. Each row in Table 10–12 describes the element or type name, the XSD declaration, the type description, and the file name that contains the item's definition.

Table 10–12 Payload Information for the Response

Name	Defined As	Type Description	File Name
updateWorkOrderNote	element	updateWorkOrderNoteRequest	EWOAPI.wsdl
updateWorkOrderNoteRequest	element	workOrderNote	EWOWorkOrder.xsd
workOrderNote	element	workOrderNote	EWOWorkOrder.xsd
workOrderNote	ComplexType	ComplexType with a list of fields	EWOWorkOrder.xsd

updateWorkOrderNoteResponse

The updateWorkOrderNoteResponse element contains the output information for the operation. The information returned in the response indicates if the operation is successful. Table 10–13 describes the returned information in the response.

Table 10–13 Payload Information for the Response

Name	Defined As	Type Description	File Name
updateWorkOrderNoteResponse	element	updateWorkOrderNoteReqRespons e	EWOAPI.wsdl
updateWorkOrderNoteReqRespon se	element	workOrderNote	EWOWorkOrder.xsd
workOrderNote	element	workOrderNote	EWOWorkOrder.xsd
workOrderNote	ComplexType	ComplexType with a list of fields	EWOWorkOrder.xsd

processDDChangeSupplement Operation

The processDDChangeSupplement operation enables external systems to process DD change supplement for engineering work order.

The following are the request and response structures:

Request Structure: processDDChangeSupplement

Response Structure: processDDChangeSupplementResponse

processDDChangeSupplement

The processDDChangeSupplement element contains the input information for the operation. Each row in Table 10–14 describes the element or type name, the XSD declaration, the type description, and the file name that contains the item's definition.

Table 10–14 Payload Information for the Response

Name	Defined As	Type Description	File Name
processDDChangeSupplement	element	processDDChangeSupplementRequest	EWOAPI.wsdl
processDDChangeSupplementRequest	ComplexType	workOrder, SuppNote	EWOWorkOrder.xsd
workOrder	element	WorkOrder	EWOWorkOrder.xsd
WorkOrder	ComplexType	ComplexType with a list of fields	EWOWorkOrder.xsd
suppNote	String	Supplement note	EWOWorkOrder.xsd

Table 10–15 describes the required fields for processDDChangeSupplement.

Table 10–15 Required Fields

File Name	Data Type	Field Description
documentNumber	int	A system-assigned identifier for the document number. Value should be zero for creating new order.
status	OrderStatusEnumType	Valid Enumeration values of Order Status Enumeration Type

processDDChangeSupplementResponse

The processDDChangeSupplementResponse element contains the output information for the operation. The information returned in the response indicates if the operation is successful. Table 10–16 describes the returned information in the response.

Table 10–16 Payload Information for the Response

Name	Defined As	Type Description	File Name
processDDChangeSupplementResponse	element	processDDChangeSupplementReqResponse	EWOAPI.wsdl
processDDChangeSupplementReqResponse	element	workOrder	EWOWorkOrder.xsd
workOrder	element	WorkOrder	EWOWorkOrder.xsd
WorkOrder	ComplexType	ComplexType with a list of fields[1]	EWOWorkOrder.xsd

[1] Refers to the elements, which will have definitions of their own XSD structure. See the schemas to understand the element flow.

updateWorkOrder Operation

The updateWorkOrder operation enables external systems to update engineering work orders.

The following are the request and response structures:

Request Structure: updateWorkOrder

Response Structure: updateWorkOrderResponse

updateWorkOrder

The updateWorkOrder element contains the input information for the operation. Each row in Table 10–17 describes the element or type name, the XSD declaration, the type description, and the file name that contains the item's definition.

Table 10–17 Payload Information for the Response

Name	Defined As	Type Description	File Name
updateWorkOrder	element	updateWorkOrderRequest	EWOAPI.wsdl
updateWorkOrderRequest	ComplexType	workOrder	EWOWorkOrder.xsd
workOrder	element	WorkOrder	EWOWorkOrder.xsd
WorkOrder	ComplexType	ComplexType with a list of fields	EWOWorkOrder.xsd

Table 10–18 describes the required fields for updateWorkOrder.

Table 10–18 Required Fields

File Name	Data Type	Field Description
documentNumber	int	A system-assigned identifier for the document number. Value should be zero for creating new order.
status	OrderStatusEnumType	Valid Enumeration values of Order Status Enumeration Type

updateWorkOrderResponse

The updateWorkOrderResponse element contains the output information for the operation. The information returned in the response indicates if the operation is successful. Table 10–19 describes the returned information in the response.

Table 10–19 Payload Information for the Response

Name	Defined As	Type Description	File Name
updateWorkOrderResponse	element	updateWorkOrderReqResponse	EWOAPI.wsdl
updateWorkOrderReqResponse	element	workOrder	EWOWorkOrder.xsd
workOrder	element	WorkOrder	EWOWorkOrder.xsd
WorkOrder	ComplexType	ComplexType with a list of fields[1]	EWOWorkOrder.xsd

[1] Refers to the elements, which will have definitions of their own XSD structure. See the schemas to understand the element flow.

getWorkOrder Operation

The getWorkOrder operation enables external systems to retrieve engineering work order details.

The following are the request and response structures:

Request Structure: getWorkOrder

Response Structure: getWorkOrderResponse

getWorkOrder

The getWorkOrder element contains the input information for the operation. Each row in Table 10–20 describes the element or type name, the XSD declaration, the type description, and the file name that contains the item's definition.

Table 10–20 Payload Information for the Response

Name	Defined As	Type Description	File Name
getWorkOrder	element	getWorkOrderRequest	EWOAPI.wsdl
getWorkOrderRequest	ComplexType	ComplexType with documentNumber	EWOWorkOrder.xsd

Table 10–21 describes the required fields for getWorkOrder.

Table 10–21 Required Fields

File Name	Data Type	Field Description
Field Name	Data Type	Field Description
documentNumber	int	The document number of the engineering work order that needs to be exported.

getWorkOrderResponse

The getWorkOrderResponse element contains the output information for the operation. The information returned in the response indicates if the operation is successful. Table 10–22 describes the returned information in the response.

Table 10–22 Payload Information for the Response

Name	Defined As	Type Description	File Name
getWorkOrderResponse	element	getWorkOrderReqResponse	EWOAPI.wsdl
getWorkOrderReqResponse	element	workOrder	EWOWorkOrder.xsd
workOrder	element	WorkOrder	EWOWorkOrder.xsd
WorkOrder	ComplexType	ComplexType with a list of fields[1]	EWOWorkOrder.xsd

[1] Refers to the elements, which will have definitions of their own XSD structure. See the schemas to understand the element flow.