Contents

Preface ....................................................................................................................................................................... v
  Audience................................................................................................................................................................. v
  Downloading Oracle Communications Documentation ......................................................................................... v
  Documentation Accessibility ...................................................................................................................................... v

1 Using Web Services
  About WSDL Files and BRM Opcodes .................................................................................................................. 1-1
  About Validating Input and Output XML Data ...................................................................................................... 1-8
  About Developing and Testing Client Applications with Web Services Manager ............................................. 1-9
    Example of Testing a Web Services Implementation Using a Client Application ........................................ 1-9
  Testing the Web Service ....................................................................................................................................... 1-12
  Using Sample Web Services Programs .................................................................................................................. 1-14
  Example of Creating an Account in BRM Using Web Services ......................................................................... 1-14
    Sample SOAP Request Input XML File ........................................................................................................... 1-15
    Sample SOAP Response Output XML File ....................................................................................................... 1-16
  About Data Masking in Web Services Responses ............................................................................................ 1-18

2 Installing Web Services Manager
  Installing Web Services Manager ............................................................................................................................ 2-1
    Supported Servers .............................................................................................................................................. 2-1
  Uninstalling Web Services Manager ..................................................................................................................... 2-1

3 Deploying Web Services Manager
  Deploying and Launching Web Services Manager on WebLogic Server ............................................................ 3-1
  Deploying and Launching Web Services Manager on Tomcat Server ............................................................... 3-4
    Deploying and Launching infranetwebsvc.war ............................................................................................... 3-4
    Deploying and Launching BrmWebServices.war ............................................................................................ 3-5

4 Configuring Web Services Manager
  About Connecting Web Services Manager to the BRM System ........................................................................ 4-1
    Connecting Web Services Manager to the BRM System ............................................................................... 4-2
    Changing the Instance of BRM to which Web Services Manager Connects ................................................... 4-4
  Configuring Security for Web Services Manager ............................................................................................... 4-6
    Configuring Security for Web Services Manager in WebLogic Server ....................................................... 4-6
Configuring Authentication for WebLogic Server ............................................................ 4-6
Configuring Role-Based Authentication for Apache Axis in WebLogic Server .......... 4-8
Configuring WebLogic Security Policy on BRM Web Services for JAX-WS in WebLogic Server 4-9
Configuring Security for Web Services Manager in Tomcat Server ......................... 4-10
Configuring Authentication for Web Services Manager for Apache Axis in Tomcat Server .. 4-10
Configuring Authentication for Web Services Manager for JAX-WS in Tomcat Server 4-11
Enabling SSL in Tomcat Server .................................................................................. 4-12
Disabling the JarScanner Feature in Tomcat Server .................................................. 4-13
Configuring Java Logging for the Application Server ............................................... 4-13
Configuring Java Logging for WebLogic Server ....................................................... 4-13
Specifying the Java Unified Logging (JUL) Mechanism ......................................... 4-13
Creating a Startup Class ............................................................................................ 4-14

5 Customizing Web Services

Setting Up Web Services Manager to Support Custom Opcodes ................................. 5-1
Creating a Custom Web Service .................................................................................. 5-4
Generating the Schema Files for Your System ......................................................... 5-6
Generating the Schema for Your Opcodes ................................................................. 5-7
Generating the Schema for an Existing Opcode ....................................................... 5-7
Creating Opcode Specification Schema Files ......................................................... 5-7
Specifying the XSL Rules to Create the Opcode Schema ........................................ 5-8
Preface

This guide provides guidelines for installing and setting up Oracle Communications Billing and Revenue Management (BRM) Web Services Manager. The sample procedures use WebLogic Server, but you can apply the concepts to any standards-compliant, vendor-supported application server that supports web services.

Before reading this guide, you should be familiar with implementing web services using WebLogic Server. See your WebLogic Server documentation for more information.

Audience

This document is intended for systems integrators, system administrators, database administrators, and other individuals who are responsible for installing, configuring, and customizing Web services for BRM.

Downloading Oracle Communications Documentation

Product documentation is located on Oracle Technology Network:

http://docs.oracle.com

Additional Oracle Communications documentation is available from the Oracle software delivery Web site:

https://edelivery.oracle.com

Documentation Accessibility

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at


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.
BRM Web Services Manager enables BRM opcodes to be exposed through Web services. Web Services Manager uses the Apache Axis framework to support SOAP Web services and is packaged as an integration pack.

This document contains information about using Oracle Communications Billing and Revenue Management (BRM) Web Services Manager.

See also:
- Installing Web Services Manager
- Deploying Web Services Manager
- Configuring Web Services Manager
- Customizing Web Services

About WSDL Files and BRM Opcodes

Web Services Manager exposes BRM opcodes as operations through different Web services.

The Web services included in Web Services Manager define the opcodes that can be called as Web service APIs and the attributes required to call a specific opcode. The Web service APIs (opcodes) are grouped by functional area into a Web service. For example, the **BRMBillServices** Web service defines the billing Web service APIs, and the **BRMPyntServices** Web service defines the payment Web service APIs. Web Services Manager includes one WSDL file for each Web service.

BRM Web Services Manager contains different WSDL files for Web services that support payload as XML string data type and for Web services that support the payload as an XML element data type.

For example:
- The **BRMBalService** Web service defines balances Web service APIs that take the payload as an XML string data type.
- The **BRMBalService_v2** Web service defines balances Web service APIs that take the payload as an XML element data type.

The file names of the WSDL files for Web services that support the payload as an XML element data type contain _v2 as a suffix.
Web services that support the payload as an XML element data type describe the input in a well-defined structure. Any standards-compliant SOAP development application can generate a client stub. The WSDL files of these Web services describe the input as a well-defined message.

Table 1–1 describes the Web services included in Web Services Manager that take the payload as an XML string.

### Table 1–1 Web Services Included in Web Services Manager that Take the Payload as an XML String

<table>
<thead>
<tr>
<th>Web Service Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BRMARServices</td>
<td>Defines the accounts receivable Web service, which includes the following opcodes:</td>
</tr>
<tr>
<td></td>
<td>- PCM_OP_AR_ACCOUNT_ADJUSTMENT</td>
</tr>
<tr>
<td></td>
<td>- PCM_OP_AR_BILL_ADJUSTMENT</td>
</tr>
<tr>
<td></td>
<td>- PCM_OP_AR_GET_ACCT_ACTION_ITEMS</td>
</tr>
<tr>
<td></td>
<td>- PCM_OP_AR_GET_ACCT_BAL_SUMMARY</td>
</tr>
<tr>
<td></td>
<td>- PCM_OP_AR_GET_ACCT_BILLS</td>
</tr>
<tr>
<td></td>
<td>- PCM_OP_AR_GET_BAL_SUMMARY</td>
</tr>
<tr>
<td></td>
<td>- PCM_OP_AR_GET_BILL_ITEMS</td>
</tr>
<tr>
<td></td>
<td>- PCM_OP_AR_ITEM_ADJUSTMENT</td>
</tr>
<tr>
<td></td>
<td>- PCM_OP_AR_EVENT_ADJUSTMENT</td>
</tr>
<tr>
<td></td>
<td>- PCM_OP_AR_GET_ACTION_ITEMS</td>
</tr>
<tr>
<td></td>
<td>- PCM_OP_AR_GET_BILLS</td>
</tr>
<tr>
<td></td>
<td>- PCM_OP_AR_RESOURCE_AGGREGATION</td>
</tr>
</tbody>
</table>

| BRMBalServices      | Defines the balances Web service, which includes the following opcodes:     |
|                    |   - PCM_OP_BAL_GET_BALANCES                                                 |
|                    |   - PCM_OP_BAL_GET_BAL_GRP_AND_SVC                                           |
|                    |   - PCM_OP_BAL_GET_ACCT_BAL_GRP_AND_SVC                                     |
|                    |   - PCM_OP_BAL_GET_ACCT_BILLINFO                                            |
|                    | See "Balance FM Standard Opcodes" in BRM Developer’s Reference for more information. |
### About WSDL Files and BRM Opcodes

#### Using Web Services

<table>
<thead>
<tr>
<th>Web Service Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>BRMBillServices</strong></td>
<td>Defines the billing Web service, which includes the following opcodes:</td>
</tr>
<tr>
<td></td>
<td>- PCM_OP_BILL_GET_ITEM_EVENT_CHARGE_DISCOUNT</td>
</tr>
<tr>
<td></td>
<td>- PCM_OP_BILL_GROUP_MOVE_MEMBER</td>
</tr>
<tr>
<td></td>
<td>- PCM_OP_BILL_MAKE_BILL_NOW</td>
</tr>
<tr>
<td></td>
<td>- PCM_OP_BILL_DEBIT</td>
</tr>
<tr>
<td></td>
<td>- PCM_OP_BILL_GROUP_GET_PARENT</td>
</tr>
<tr>
<td><strong>BRMCollectionsServices</strong></td>
<td>Defines the collections Web service, which includes the following opcode:</td>
</tr>
<tr>
<td></td>
<td>- PCM_OP_COLLECTIONS_SET_ACTION_STATUS</td>
</tr>
<tr>
<td></td>
<td>See “Collections Manager FM Standard Opcodes” in <em>BRM Developer’s Reference</em> for more information.</td>
</tr>
<tr>
<td><strong>BRMCustcareServices</strong></td>
<td>Defines the customer care Web service, which includes the following opcode:</td>
</tr>
<tr>
<td></td>
<td>- PCM_OP_CUSTCARE_MOVE_ACCT</td>
</tr>
<tr>
<td><strong>BRMCustServices</strong></td>
<td>Defines the customer Web service, which includes the following opcodes:</td>
</tr>
<tr>
<td></td>
<td>- PCM_OP_CUST_COMMIT_CUSTOMER</td>
</tr>
<tr>
<td></td>
<td>- PCM_OP_CUST_MODIFY_CUSTOMER</td>
</tr>
<tr>
<td></td>
<td>- PCM_OP_CUST_UPDATE_CUSTOMER</td>
</tr>
<tr>
<td></td>
<td>- PCM_OP_CUST_UPDATE_SERVICES</td>
</tr>
<tr>
<td></td>
<td>- PCM_OP_CUST_DELETE_ACCT</td>
</tr>
<tr>
<td></td>
<td>- PCM_OP_CUST_DELETE_PAYINFO</td>
</tr>
<tr>
<td></td>
<td>- PCM_OP_CUST_CREATE_PROFILE</td>
</tr>
<tr>
<td></td>
<td>- PCM_OP_CUST_MODIFY_PROFILE</td>
</tr>
<tr>
<td></td>
<td>- PCM_OP_CUST_DELETE_PROFILE</td>
</tr>
<tr>
<td></td>
<td>See “Customer FM Standard Opcodes” in <em>BRM Developer’s Reference</em> for more information.</td>
</tr>
<tr>
<td><strong>BRMInvServices</strong></td>
<td>Defines the invoicing Web service, which includes the following opcode:</td>
</tr>
<tr>
<td></td>
<td>- PCM_OP_INV_VIEW_INVOICE</td>
</tr>
<tr>
<td></td>
<td><strong>Important:</strong> You must configure your client application to convert the invoice data received from the PCM_OP_INV_VIEW_INVOICE opcode into the appropriate format. See “About Invoicing Output XML Data” in <em>BRM JCA Resource Adapter</em>.</td>
</tr>
<tr>
<td></td>
<td>See “Invoicing FM Standard Opcodes” in <em>BRM Developer’s Reference</em> for more information.</td>
</tr>
<tr>
<td><strong>BRMPymtServices</strong></td>
<td>Defines the payment Web service, which includes the following opcode:</td>
</tr>
<tr>
<td></td>
<td>- PCM_OP_PYMT_COLLECT</td>
</tr>
<tr>
<td></td>
<td>See “Payment FM Standard Opcodes” in <em>BRM Developer’s Reference</em> for more information.</td>
</tr>
</tbody>
</table>

---

**Table 1–1 (Cont.) Web Services Included in Web Services Manager that Take the Payload as an XML String**

<table>
<thead>
<tr>
<th>Web Service Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>BRMBillServices</strong></td>
<td>Defines the billing Web service, which includes the following opcodes:</td>
</tr>
<tr>
<td></td>
<td>- PCM_OP_BILL_GET_ITEM_EVENT_CHARGE_DISCOUNT</td>
</tr>
<tr>
<td></td>
<td>- PCM_OP_BILL_GROUP_MOVE_MEMBER</td>
</tr>
<tr>
<td></td>
<td>- PCM_OP_BILL_MAKE_BILL_NOW</td>
</tr>
<tr>
<td></td>
<td>- PCM_OP_BILL_DEBIT</td>
</tr>
<tr>
<td></td>
<td>- PCM_OP_BILL_GROUP_GET_PARENT</td>
</tr>
<tr>
<td><strong>BRMCollectionsServices</strong></td>
<td>Defines the collections Web service, which includes the following opcode:</td>
</tr>
<tr>
<td></td>
<td>- PCM_OP_COLLECTIONS_SET_ACTION_STATUS</td>
</tr>
<tr>
<td></td>
<td>See “Collections Manager FM Standard Opcodes” in <em>BRM Developer’s Reference</em> for more information.</td>
</tr>
<tr>
<td><strong>BRMCustcareServices</strong></td>
<td>Defines the customer care Web service, which includes the following opcode:</td>
</tr>
<tr>
<td></td>
<td>- PCM_OP_CUSTCARE_MOVE_ACCT</td>
</tr>
<tr>
<td><strong>BRMCustServices</strong></td>
<td>Defines the customer Web service, which includes the following opcodes:</td>
</tr>
<tr>
<td></td>
<td>- PCM_OP_CUST_COMMIT_CUSTOMER</td>
</tr>
<tr>
<td></td>
<td>- PCM_OP_CUST_MODIFY_CUSTOMER</td>
</tr>
<tr>
<td></td>
<td>- PCM_OP_CUST_UPDATE_CUSTOMER</td>
</tr>
<tr>
<td></td>
<td>- PCM_OP_CUST_UPDATE_SERVICES</td>
</tr>
<tr>
<td></td>
<td>- PCM_OP_CUST_DELETE_ACCT</td>
</tr>
<tr>
<td></td>
<td>- PCM_OP_CUST_DELETE_PAYINFO</td>
</tr>
<tr>
<td></td>
<td>- PCM_OP_CUST_CREATE_PROFILE</td>
</tr>
<tr>
<td></td>
<td>- PCM_OP_CUST_MODIFY_PROFILE</td>
</tr>
<tr>
<td></td>
<td>- PCM_OP_CUST_DELETE_PROFILE</td>
</tr>
<tr>
<td></td>
<td>See “Customer FM Standard Opcodes” in <em>BRM Developer’s Reference</em> for more information.</td>
</tr>
<tr>
<td><strong>BRMInvServices</strong></td>
<td>Defines the invoicing Web service, which includes the following opcode:</td>
</tr>
<tr>
<td></td>
<td>- PCM_OP_INV_VIEW_INVOICE</td>
</tr>
<tr>
<td></td>
<td><strong>Important:</strong> You must configure your client application to convert the invoice data received from the PCM_OP_INV_VIEW_INVOICE opcode into the appropriate format. See “About Invoicing Output XML Data” in <em>BRM JCA Resource Adapter</em>.</td>
</tr>
<tr>
<td></td>
<td>See “Invoicing FM Standard Opcodes” in <em>BRM Developer’s Reference</em> for more information.</td>
</tr>
<tr>
<td><strong>BRMPymtServices</strong></td>
<td>Defines the payment Web service, which includes the following opcode:</td>
</tr>
<tr>
<td></td>
<td>- PCM_OP_PYMT_COLLECT</td>
</tr>
<tr>
<td></td>
<td>See “Payment FM Standard Opcodes” in <em>BRM Developer’s Reference</em> for more information.</td>
</tr>
</tbody>
</table>
Table 1–2 describes the Web services included in Web Services Manager that take the payload as an XML element.

Table 1–2 Web Services Included in Web Services Manager that Take the Payload as an XML Element

<table>
<thead>
<tr>
<th>Web Service Name</th>
<th>Description</th>
</tr>
</thead>
</table>
| BRMReadServices  | Defines the read Web service, which includes the following opcodes:  
- PCM_OP_READ_FLD5  
- PCM_OP_READ_OBJ  
- PCM_OP_SEARCH  
See "LDAP Base Opcodes" in BRM Developer’s Reference for more information. |
| BRMSSubscriptionServices | Defines the subscription Web service, which includes the following opcodes:  
- PCM_OP_SUBSCRIPTION_CANCEL_PRODUCT  
- PCM_OP_SUBSCRIPTION_CANCEL_DISCOUNT  
- PCM_OP_SUBSCRIPTION_CANCEL_SUBSCRIPTION  
- PCM_OP_SUBSCRIPTION_CHANGE_DEAL  
- PCM_OP_SUBSCRIPTION_PURCHASE_DEAL  
- PCM_OP_SUBSCRIPTION_SET_BUNDLE  
- PCM_OP_SUBSCRIPTION_SET_DISCOUNT_STATUS  
- PCM_OP_SUBSCRIPTION_SET_DISCOUNTINFO  
- PCM_OP_SUBSCRIPTION_SET_PRODINFO  
- PCM_OP_SUBSCRIPTION_SET_PRODUCT_STATUS  
- PCM_OP_SUBSCRIPTION_TRANSFER_SUBSCRIPTION  
- PCM_OP_SUBSCRIPTION_GET_PURCHASED_OFFERINGS  
See "Subscription Management FM Standard Opcodes" in BRM Developer’s Reference for more information. |

Table 1–2 describes the Web services included in Web Services Manager that take the payload as an XML element.
### About WSDL Files and BRM Opcodes

#### Using Web Services

**BRMARServices_v2**
Defines the accounts receivable Web service, which includes the following opcodes:
- PCM_OP_AR_ACCOUNT_ADJUSTMENT
- PCM_OP_AR_ACCOUNT_WRITEOFF
- PCM_OP_AR_BILL_ADJUSTMENT
- PCM_OP_AR_BILL_DISPUTE
- PCM_OP_AR_BILL_SETTLEMENT
- PCM_OP_AR_BILL_WRITEOFF
- PCM_OP_AR_BILLINFO_WRITEOFF
- PCM_OP_AR_EVENT_ADJUSTMENT
- PCM_OP_AR_EVENT_DISPUTE
- PCM_OP_AR_EVENT_SETTLEMENT
- PCM_OP_AR_GET_ACCT_ACTION_ITEMS
- PCM_OP_AR_GET_ACCT_BAL_SUMMARY
- PCM_OP_AR_GET_ACCT_BILLS
- PCM_OP_AR_GET_ACTION_ITEMS
- PCM_OP_AR_GET_BAL_SUMMARY
- PCM_OP_AR_GET_BILLS
- PCM_OP_AR_GET_BILL_ITEMS
- PCM_OP_AR_GET_DISPUTES
- PCM_OP_AR_GET_DISPUTE_DETAILS
- PCM_OP_AR_GET_ITEMS
- PCM_OP_AR_GET_ITEM_DETAILS
- PCM_OP_AR_ITEM_ADJUSTMENT
- PCM_OP_AR_ITEM_DISPUTE
- PCM_OP_AR_ITEM_SETTLEMENT
- PCM_OP_AR_ITEM_WRITEOFF
- PCM_OP_AR_RESOURCE_AGGREGATION

See "Accounts Receivable FM Standard Opcodes" in **BRM Developer’s Reference** for more information.

#### BRMBALServices_v2
Defines the balances Web service, which includes the following opcodes:
- PCM_OP_BAL_CHANGE_VALIDITY
- PCM_OP_BAL_GET_BALANCES
- PCM_OP_BAL_GET_BAL_GRP_AND_SVC
- PCM_OP_BAL_GET_ACCT_BAL_GRP_AND_SVC
- PCM_OP_BAL_GET_ACCT_BILLINFO

See "Balance FM Standard Opcodes" in **BRM Developer’s Reference** for more information.

---

**Table 1–2 (Cont.) Web Services Included in Web Services Manager that Take the Payload as an XML Element**

<table>
<thead>
<tr>
<th>Web Service Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BRMARServices_v2</td>
<td>Defines the accounts receivable Web service, which includes the following opcodes:</td>
</tr>
<tr>
<td></td>
<td>- PCM_OP_AR_ACCOUNT_ADJUSTMENT</td>
</tr>
<tr>
<td></td>
<td>- PCM_OP_AR_ACCOUNT_WRITEOFF</td>
</tr>
<tr>
<td></td>
<td>- PCM_OP_AR_BILL_ADJUSTMENT</td>
</tr>
<tr>
<td></td>
<td>- PCM_OP_AR_BILL_DISPUTE</td>
</tr>
<tr>
<td></td>
<td>- PCM_OP_AR_BILL_SETTLEMENT</td>
</tr>
<tr>
<td></td>
<td>- PCM_OP_AR_BILL_WRITEOFF</td>
</tr>
<tr>
<td></td>
<td>- PCM_OP_AR_BILLINFO_WRITEOFF</td>
</tr>
<tr>
<td></td>
<td>- PCM_OP_AR_EVENT_ADJUSTMENT</td>
</tr>
<tr>
<td></td>
<td>- PCM_OP_AR_EVENT_DISPUTE</td>
</tr>
<tr>
<td></td>
<td>- PCM_OP_AR_EVENT_SETTLEMENT</td>
</tr>
<tr>
<td></td>
<td>- PCM_OP_AR_GET_ACCT_ACTION_ITEMS</td>
</tr>
<tr>
<td></td>
<td>- PCM_OP_AR_GET_ACCT_BAL_SUMMARY</td>
</tr>
<tr>
<td></td>
<td>- PCM_OP_AR_GET_ACCT_BILLS</td>
</tr>
<tr>
<td></td>
<td>- PCM_OP_AR_GET_ACTION_ITEMS</td>
</tr>
<tr>
<td></td>
<td>- PCM_OP_AR_GET_BAL_SUMMARY</td>
</tr>
<tr>
<td></td>
<td>- PCM_OP_AR_GET_BILLS</td>
</tr>
<tr>
<td></td>
<td>- PCM_OP_AR_GET_BILL_ITEMS</td>
</tr>
<tr>
<td></td>
<td>- PCM_OP_AR_GET_DISPUTES</td>
</tr>
<tr>
<td></td>
<td>- PCM_OP_AR_GET_DISPUTE_DETAILS</td>
</tr>
<tr>
<td></td>
<td>- PCM_OP_AR_GET_ITEM_DETAILS</td>
</tr>
<tr>
<td></td>
<td>- PCM_OP_AR_ITEM_ADJUSTMENT</td>
</tr>
<tr>
<td></td>
<td>- PCM_OP_AR_ITEM_DISPUTE</td>
</tr>
<tr>
<td></td>
<td>- PCM_OP_AR_ITEM_SETTLEMENT</td>
</tr>
<tr>
<td></td>
<td>- PCM_OP_AR_ITEM_WRITEOFF</td>
</tr>
<tr>
<td></td>
<td>- PCM_OP_AR_RESOURCE_AGGREGATION</td>
</tr>
<tr>
<td></td>
<td>See &quot;Accounts Receivable FM Standard Opcodes&quot; in <strong>BRM Developer’s Reference</strong> for more information.</td>
</tr>
<tr>
<td>BRMBALServices_v2</td>
<td>Defines the balances Web service, which includes the following opcodes:</td>
</tr>
<tr>
<td></td>
<td>- PCM_OP_BAL_CHANGE_VALIDITY</td>
</tr>
<tr>
<td></td>
<td>- PCM_OP_BAL_GET_BALANCES</td>
</tr>
<tr>
<td></td>
<td>- PCM_OP_BAL_GET_BAL_GRP_AND_SVC</td>
</tr>
<tr>
<td></td>
<td>- PCM_OP_BAL_GET_ACCT_BAL_GRP_AND_SVC</td>
</tr>
<tr>
<td></td>
<td>- PCM_OP_BAL_GET_ACCT_BILLINFO</td>
</tr>
<tr>
<td></td>
<td>See &quot;Balance FM Standard Opcodes&quot; in <strong>BRM Developer’s Reference</strong> for more information.</td>
</tr>
</tbody>
</table>
### Table 1–2  (Cont.) Web Services Included in Web Services Manager that Take the Payload as an XML Element

<table>
<thead>
<tr>
<th>Web Service Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BRMBILLServices_v2</td>
<td>Defines the billing Web service, which includes the following opcodes:</td>
</tr>
<tr>
<td></td>
<td>- PCM_OP_BILL_DEBIT</td>
</tr>
<tr>
<td></td>
<td>- PCM_OP_BILL_FIND</td>
</tr>
<tr>
<td></td>
<td>- PCM_OP_BILL_GET_ITEM_EVENT_CHARGE_DISCOUNT</td>
</tr>
<tr>
<td></td>
<td>- PCM_OP_BILL_GROUP_GET_PARENT</td>
</tr>
<tr>
<td></td>
<td>- PCM_OP_BILL_GROUP_MOVE_MEMBER</td>
</tr>
<tr>
<td></td>
<td>- PCM_OP_BILL_ITEM_EVENT_SEARCH</td>
</tr>
<tr>
<td></td>
<td>- PCM_OP_BILL_ITEM_REFUND</td>
</tr>
<tr>
<td></td>
<td>- PCM_OP_BILL_MAKE_BILL_NOW</td>
</tr>
<tr>
<td></td>
<td>- PCM_OP_BILL_REVERSE</td>
</tr>
<tr>
<td></td>
<td>- PCM_OP_BILL_SET_LIMIT_AND_CR</td>
</tr>
<tr>
<td></td>
<td>- PCM_OP_BILL_VIEW_INVOICE</td>
</tr>
<tr>
<td></td>
<td>See &quot;Billing FM Standard Opcodes&quot; in BRM Developer’s Reference for more information.</td>
</tr>
<tr>
<td>BRMCOLLECTIONSServices_v2</td>
<td>Defines the collections Web service, which includes the following opcode:</td>
</tr>
<tr>
<td></td>
<td>- PCM_OP_COLLECTIONS_SET_ACTION_STATUS</td>
</tr>
<tr>
<td></td>
<td>See &quot;Collections Manager FM Standard Opcodes&quot; in BRM Developer’s Reference for more information.</td>
</tr>
<tr>
<td>BRMCUSTCAREServices_v2</td>
<td>Defines the customer care Web service, which includes the following opcode:</td>
</tr>
<tr>
<td></td>
<td>- PCM_OP_CUSTCARE_MOVE_ACCT</td>
</tr>
</tbody>
</table>
Using Web Services 1-7

About WSDL Files and BRM Opcodes

Table 1–2 (Cont.) Web Services Included in Web Services Manager that Take the Payload as an XML Element

<table>
<thead>
<tr>
<th>Web Service Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BRMCUSTServices_v2</td>
<td>Defines the customer Web service, which includes the following opcodes:</td>
</tr>
<tr>
<td></td>
<td>■ PCM_OP_CUST_COMMIT_CUSTOMER</td>
</tr>
<tr>
<td></td>
<td>■ PCM_OP_CUST_CREATE_PROFILE</td>
</tr>
<tr>
<td></td>
<td>■ PCM_OP_CUST_DELETE_ACCT</td>
</tr>
<tr>
<td></td>
<td>■ PCM_OP_CUST_DELETE_PAYINFO</td>
</tr>
<tr>
<td></td>
<td>■ PCM_OP_CUST_DELETE_PROFILE</td>
</tr>
<tr>
<td></td>
<td>■ PCM_OP_CUST_FIND</td>
</tr>
<tr>
<td></td>
<td>■ PCM_OP_CUST_FIND_PAYINFO</td>
</tr>
<tr>
<td></td>
<td>■ PCM_OP_CUST_FIND_PROFILE</td>
</tr>
<tr>
<td></td>
<td>■ PCM_OP_CUST_GET_NOTE</td>
</tr>
<tr>
<td></td>
<td>■ PCM_OP_CUST_MODIFY_CUSTOMER</td>
</tr>
<tr>
<td></td>
<td>■ PCM_OP_CUST_MODIFIED_PROFILE</td>
</tr>
<tr>
<td></td>
<td>■ PCM_OP_CUST_SET_NOTE</td>
</tr>
<tr>
<td></td>
<td>■ PCM_OP_CUST_SET_STATUS</td>
</tr>
<tr>
<td></td>
<td>■ PCM_OP_CUST_SET_TAXINFO</td>
</tr>
<tr>
<td></td>
<td>■ PCM_OP_CUST_UPDATE_CUSTOMER</td>
</tr>
<tr>
<td></td>
<td>■ PCM_OP_CUST_UPDATE_SERVICES</td>
</tr>
<tr>
<td></td>
<td>See &quot;Customer FM Standard Opcodes&quot; in BRM Developer’s Reference for more information.</td>
</tr>
<tr>
<td></td>
<td>■ PCM_OP_CUST_POL_GET_PLANS</td>
</tr>
<tr>
<td></td>
<td>■ PCM_OP_CUST_POL_GET_DEALS</td>
</tr>
<tr>
<td></td>
<td>■ PCM_OP_CUST_POL_GET_PRODUCTS</td>
</tr>
<tr>
<td></td>
<td>■ PCM_OP_CUST_POL_READ_PLAN</td>
</tr>
<tr>
<td></td>
<td>See &quot;Customer FM Policy Opcodes&quot; in BRM Developer’s Reference for more information.</td>
</tr>
<tr>
<td>BRMINVServices_v2</td>
<td>Defines the invoicing Web service, which includes the following opcode:</td>
</tr>
<tr>
<td></td>
<td>■ PCM_OP_INV_VIEW_INVOICE</td>
</tr>
<tr>
<td></td>
<td>Important: You must configure your client application to convert the invoice data received from</td>
</tr>
<tr>
<td></td>
<td>the PCM_OP_INV_VIEW_INVOICE opcode into the appropriate format. See &quot;About Invoicing Output</td>
</tr>
<tr>
<td></td>
<td>XML Data&quot; in BRM JCA Resource Adapter.</td>
</tr>
<tr>
<td></td>
<td>See &quot;Invoicing FM Standard Opcodes&quot; in BRM Developer’s Reference for more information.</td>
</tr>
<tr>
<td>BRMPYMTServices_v2</td>
<td>Defines the payment Web service, which includes the following opcode:</td>
</tr>
<tr>
<td></td>
<td>■ PCM_OP_PYMT_COLLECT</td>
</tr>
<tr>
<td></td>
<td>See &quot;Payment FM Standard Opcodes&quot; in BRM Developer’s Reference for more information.</td>
</tr>
</tbody>
</table>
About Validating Input and Output XML Data

Web Services Manager validates the input and output XML by comparing the XML fields and values against the opcode XML schema.

The opcode specifications, schemas, and WSDL files are packaged along with Web Services Manager. The package includes the opspec.xsd file and the pin_opspec_to_schema utility. Use the opspec.xsd file to write opcode specifications for custom opcodes that need to be exposed as a Web service. Use the pin_opspec_to_schema utility to generate the schema files from the opcode specification files.

<table>
<thead>
<tr>
<th>Web Service Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BRMREADServices_v2</td>
<td>Defines the read Web service, which includes the following opcodes:</td>
</tr>
<tr>
<td></td>
<td>■ PCM_OP_READ_FLDS</td>
</tr>
<tr>
<td></td>
<td>■ PCM_OP_READ_OBJ</td>
</tr>
<tr>
<td></td>
<td>■ PCM_OP_SEARCH</td>
</tr>
<tr>
<td></td>
<td>■ PCM_OP_TEST_LOOPBACK</td>
</tr>
<tr>
<td></td>
<td>See &quot;LDAP Base Opcodes&quot; in BRM Developer’s Reference for more information.</td>
</tr>
<tr>
<td>BRMSUBSCRIPTIONSERVICES_v2</td>
<td>Defines the subscription Web service, which includes the following opcodes:</td>
</tr>
<tr>
<td></td>
<td>■ PCM_OP_SUBSCRIPTIONCANCEL DEAL</td>
</tr>
<tr>
<td></td>
<td>■ PCM_OP_SUBSCRIPTIONCANCEL PRODUCT</td>
</tr>
<tr>
<td></td>
<td>■ PCM_OP_SUBSCRIPTIONCANCEL DISCOUNT</td>
</tr>
<tr>
<td></td>
<td>■ PCM_OP_SUBSCRIPTIONCANCEL_SUBSCRIPTION</td>
</tr>
<tr>
<td></td>
<td>■ PCM_OP_SUBSCRIPTIONCHANGE DEAL</td>
</tr>
<tr>
<td></td>
<td>■ PCM_OP_SUBSCRIPTIONGET_HISTORY</td>
</tr>
<tr>
<td></td>
<td>■ PCM_OP_SUBSCRIPTIONPURCHASE DEAL</td>
</tr>
<tr>
<td></td>
<td>■ PCM_OP_SUBSCRIPTIONPURCHASE FEES</td>
</tr>
<tr>
<td></td>
<td>■ PCM_OP_SUBSCRIPTIONREAD ACCT PRODUCTS</td>
</tr>
<tr>
<td></td>
<td>■ PCM_OP_SUBSCRIPTIONSERVICE BALGRP TRANSFER</td>
</tr>
<tr>
<td></td>
<td>■ PCM_OP_SUBSCRIPTIONSET BUNDLE</td>
</tr>
<tr>
<td></td>
<td>■ PCM_OP_SUBSCRIPTIONSET DISCOUNT STATUS</td>
</tr>
<tr>
<td></td>
<td>■ PCM_OP_SUBSCRIPTIONSET DISCOUNTINFO</td>
</tr>
<tr>
<td></td>
<td>■ PCM_OP_SUBSCRIPTIONSET PRODINFO</td>
</tr>
<tr>
<td></td>
<td>■ PCM_OP_SUBSCRIPTIONSET PRODUCT STATUS</td>
</tr>
<tr>
<td></td>
<td>■ PCM_OP_SUBSCRIPTIONTRANSFER SUBSCRIPTION</td>
</tr>
<tr>
<td></td>
<td>■ PCM_OP_SUBSCRIPTIONTRANSITION DEAL</td>
</tr>
<tr>
<td></td>
<td>■ PCM_OP_SUBSCRIPTIONTRANSITION PLAN</td>
</tr>
<tr>
<td></td>
<td>■ PCM_OP_SUBSCRIPTIONGET PURCHASED OFFERINGS</td>
</tr>
</tbody>
</table>
|                                           | See "Subscription Management FM Standard Opcodes” in BRM Developer’s Reference for more information.

Table 1–2 (Cont.) Web Services Included in Web Services Manager that Take the Payload as an XML Element
To configure Web Services Manager to validate the input and output XML against the target opcode XML schema:

1. Open the `local_dir/WEB-INF/classes/Infranet.properties` file.
2. Add the following entries to the file:
   - `webservices.input.validation.enabled=true`
   - `webservices.output.validation.enabled=true`
3. Do one of the following:
   - If you are using WebLogic Server, copy the schema files packaged as a part of Web Services Manager installation from the `BRM_home/deploy/web_services/schemas` directory to the `local_dir/common/lib` directory.
   - If you are using any supported server, copy the schema files from the `BRM_home/deploy/web_services/schemas` directory to the `local_dir/WEB-INF/classes` directory.

### About Developing and Testing Client Applications with Web Services Manager

You can develop custom applications that interact with BRM through Web Services Manager. Use a SOAP development environment that supports importing WSDL files (for example, SoapUI) to develop and test your custom Web service applications. SOAP development applications may have minor differences in product configuration. Consult your SOAP development application documentation for configuration information.

In general, do the following to develop and test your Web services applications:

1. Download and install a SOAP development application.
2. Configure a new project in your SOAP development application.
3. Write a client application that communicates with Web services using the SOAP protocol.
4. Import the Web service definitions using the WSDL files. See "About WSDL Files and BRM Opcodes" for more information on accessing WSDL files.
5. Run the required commands to set up your application server environment.
6. Configure the properties of the Web services operations in your SOAP development environment with valid credentials.
7. Send a Web service request to BRM from the SOAP development environment client.
8. View the Web service response in the SOAP development environment.

### Example of a Testing a Web Services Implementation Using a Client Application

To test your Web services implementation, you must write a client application that communicates with the Web service using the SOAP protocol.

The sample procedures use WebLogic Server, but you can apply the concepts to any other supported application server.
This sample procedure demonstrates how to use the TestClient.java sample code with the PCM_OP_TEST_LOOPBACK opcode to verify communication between BRM and the Web service.

Using WebLogic Server, for example:

1. Run one of the following commands, which set up the WebLogic Server environment:
   - If WebLogic is installed on a UNIX host: `WebLogic_home/wlserver/server/bin/setWLSEnv.sh`
   - If WebLogic is installed on a Windows host: `WebLogic_home/server/bin/setenv.exe`
     where `WebLogic_home` is the directory in which you installed the WebLogic Server.

2. Create an XML file (`some_name_1.xml`) using the following text:

   ```xml
   <project name="buildWebservice" default="all">
   <property name="buildDir" value="./myapps" />
   <property name="jarFiles" value="jars" />
   <target name="all" depends="jar" description="builds everything">
     <clientgen
       wssdl="http://198.51.100.1:7001/infranetwebsvc/services/Infranet?wsdl"
       packageName="test_client"
       destDir="./myapps"/>
     <target name="compile" depends="generate-client" description="compile source files">
       <echo> Compiling adapter files</echo>
       <javac destdir="${buildDir}">
         <src path="${buildDir}"/>
       </javac>
     </target>
     <target name="jar" depends="compile" description="generate jar file(s)">
       <jar jarfile="clientStub.jar" basedir="${buildDir}">
         <exclude name="**/*.java"/>
       </jar>
     </target>
     <target name="clean" description="remove files created by target prepare">
       <delete dir="${buildDir}"/>
     </target>
   </target>
   </project>
   ```

   This XML file uses the WebLogic Server clientgen task to automatically generate a utility library that provides low-level SOAP communication (client stubs).

3. Run the following command, which creates the client stubs:

   `ant -file some_name_1.xml`

   This process generates the `clientstubs.jar` file, which contains stubs used by the client. The test client code (`source_home/TestClient.java`, where `source_home` is the directory where your source code files are stored) then creates an flist, converts it to XML, and calls the PCM_OP_TEST_LOOPBACK opcode.

   The following is a sample listing of `TestClient.java`:
import java.io.IOException;
import test_client.*; // corresponds to package name clientgen generated

public class TestClient {

    public static void main(String[] args) {
        try {
            String wsdlUrl = "http://198.51.100.1:7001/infranetwebsvc/services/Infranet?wsdl";
            InfranetWebServiceService service = new InfranetServiceService_Impl( wsdlUrl );
            InfranetWebService port = service.getInfranet();

            // convert flist to XML representation
            String XMLInput = "<flist xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">" + "<POID>0.0.0.1 /account 80408 8</POID>" + XMLInput + "</flist>";
            System.out.println("Input: " + XMLInput);
            // invoke web service 'opcode' method
            String result = port.opcode("TEST_LOOPBACK", XMLInput);
            System.out.println("result: " + result);
        } catch (Exception ex) {
            ex.printStackTrace();
        }
    }
}

4. Create another XML file (some_name_2.xml) using the following text:

    Note: Replace the paths for the JAR files as required.

```xml
<project name="test_client" default="all">
    <target name="all" depends="run"/>

    <path id="classpath">
        <pathelement path="clientstubs.jar"/>
        <pathelement path=".:\classes"/>
        <pathelement path="D:\WebServices\webman\lib\jaxrpc.jar"/>
        <pathelement path="D:\bea\wlserver_10.3\server\lib\wseeclient.jar"/>
        <pathelement path="D:\bea\wlserver_10.3\server\lib\weblogic.jar"/>
    </path>

    <target name="compile">
        <mkdir dir="classes"/>
        <javac srcdir="src" destdir="classes" classpathref="classpath"/>
    </target>

    <target name="run" depends="compile">
        <java classname="TestClient" fork="yes"/>
    </target>
</project>
```
Testing the Web Service

To test your Web services implementation, use a SOAP development application that supports importing WSDL files, for example SoapUI. SOAP development applications may have minor differences in project configuration. Consult your SOAP development environment documentation for configuration information.

To test your Web services implementation, write a client application that communicates with the Web service using the SOAP protocol.

Note: You can test Web services using SoapUI version 4.6.3.

The sample procedures use WebLogic Server, but you can apply the concepts to any other supported server.

For example, with WebLogic Server:

1. Run one of the following commands, which set up the WebLogic Server environment:
   - If WebLogic is installed on a UNIX host: `WebLogic_home/wls/server/server/bin/setWLSEnv.sh`
   - If WebLogic is installed on a Windows host: `WebLogic_home/server/bin/setenv.exe`
     where `WebLogic_home` is the directory in which you installed the WebLogic Server.

2. Create an XML file (`some_name_1.xml`) using the following text:

   ```xml
   <project name="buildWebservice" default="all">
   <property name="buildDir" value="./myapps" />
   <property name="jarFiles" value="jars" />
   <target name="all" depends="jar" description="builds everything">
   </target>
   <target name="generate-client">
    <clientgen
     wsdl="http://198.51.100.1:7001/infranetwebsvc/services/Infranet?wsdl"
     package="test_client"
     destDir="./myapps"/>
   </target>
   <target name="compile" depends="generate-client" description="compile source files">
    <echo> Compiling adapter files</echo>
    <javac destdir="${buildDir}"
      srcpath="${buildDir}"/>
   </javac>
   </target>
   <target name="jar" depends="compile" description="generate jar file(s)"/>
   </project>
   ```

5. Build and run the test with the `some_name_2.xml` file using regular Ant tasks:

   ```
   ant -file some_name_2.xml
   ```
This XML file uses the WebLogic Server clientgen task to automatically generate a utility library that provides low-level SOAP communication (client stubs).

3. Run the following command, which creates the client stubs:

```
ant -file some_name_1.xml
```

This process generates the `clientstubs.jar` file, which contains stubs used by the client. The test client code (`src\TestClient.java`) then creates an flist, converts it to XML, and calls the `PCM_OP_TEST_LOOPBACK` opcode.

The following is a sample listing of `TestClient.java`:

```java
import java.io.IOException;
import test_client.*; // corresponds to package name clientgen generated

public class TestClient {
    public static void main(String[] args) {
        try {
            String wsdlUrl = "http://198.51.100.1:7001/infranetwebsvc/services/Infranet?wsdl";
            InfranetWebServiceService service = new InfranetServiceService_Impl(wsdlUrl);
            InfranetWebService port = service.getInfranet();

            // convert flist to XML representation
            String XMLInput="<flist xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">"+POID+"0.0.1 /account 80408 8</POID>"+fists;
            System.out.println("Input: "+XMLInput);

            // invoke web service 'opcode' method
            String result = port.opcode("TEST_LOOPBACK", XMLInput);

            System.out.println("result: "+result);
        } catch (Exception ex) {
            ex.printStackTrace();
        }
    }
}
```

4. Create another XML file (`some_name_2.xml`) using the following text:

```
<project name="test_client" default="all">
    <jar jarfile="clientStub.jar" basedir="$(buildDir)"
         exclude name="**/*.java"/>
    <target name="clean" description="remove files created by target prepare">
        <delete dir="$(buildDir)"/>
    </target>
</project>
```

**Note:** Replace the paths for the JAR files as required.
Using Sample Web Services Programs

Web Services Manager includes sample programs that demonstrate how to write code for various tasks when customizing Web services. For example, the sample program InfranetBALTestClient.java creates an flist, converts it to XML, and calls the PCM_OP_BAL_GET_BALANCES opcode.

Example of Creating an Account in BRM Using Web Services

This section describes an example of creating a customer account using Web services. The example shows a sample SOAP request and a response message for creating a customer account in BRM by calling a Web service in Web Services Manager.

To create an account in BRM using Web services, you call the pcmOpCustCommitCustomer Web service API that maps to the PCM_OP_CUST_COMMIT_CUSTOMER opcode. The pcmOpCustCommitCustomer Web service API is included in the BRMCUSTServices_v2 Web service. The BRMCUSTServices_v2 Web service contains Web service APIs that are related to customer accounts. See "About WSDL Files and BRM Opcodes" for more information about the Web services included in the Web Services Manager package.

You use URLs to create SOAP clients for Web services. The URL to create a SOAP client is generated by the JAX-WS in Oracle WebLogic Server.

To generate the URL for a Web service:

1. Log in to the WebLogic Server Administration Console.

5. Build and run the test with the some_name_2.xml file using regular Ant tasks:

```xml
ant -file some_name_2.xml
```
2. Go to the deployments section.

3. Click BrmWebServices link.
   The Settings page for the deployment appears and all the web services are listed in the Modules and Components section.

4. Click the web service in the Modules and Components section.

5. Click the Testing tab.

6. The WebLogic Server Administration console displays an URL for the BRM web services.

A sample URL for the BRMCUSTServices_v2 Web service is as follows:
http://hostIPAddress:port/BrmWebServices/BRMCUSTServices_v2?wsdl

To call a Web service, users are required to authenticate using a valid username and a password. Users can call only those Web services that they are authorized to call.

**Sample SOAP Request Input XML File**

The following sample shows a SOAP request for the pcmOpCustCommitCustomer Web service API.

```xml
- <soapenv:Envelope
  xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:bus="http://xmlns.oracle.com/BRM/schemas/BusinessOpcodes">
  <soapenv:Header/>
  <soapenv:Body>
    <bus:pcmOpCustCommitCustomer>
      <PCM_OP_CUST_COMMIT_CUSTOMER_Request xsi:type="bus:PCM_OP_CUST_COMMIT_CUSTOMER_RequestType">
        <flags xsi:type="xsd:int">1</flags>
        <PCM_OP_CUST_COMMIT_CUSTOMER_inputFlist
          xmlns="http://xmlns.oracle.com/BRM/schemas/BusinessOpcodes">
          <ACCTINFO elem="0">
            <ACCOUNT_NO>a022020202011992</ACCOUNT_NO>
            <BAL_INFO />
            <BUSINESS_TYPE>1</BUSINESS_TYPE>
            <CURRENCY>840</CURRENCY>
            <POID>0.0.0.1 /account -1 0</POID>
          </ACCTINFO>
          <BAL_INFO elem="0">
            <BILLINFO />
            <LIMIT elem="840">
              <CREDIT_LIMIT>"0"</CREDIT_LIMIT>
            </LIMIT>
            <NAME>Account Level Balance Group</NAME>
            <POID>0.0.0.1 /balance_group -1 0</POID>
          </BAL_INFO>
          <BILLINFO elem="0">
            <BAL_INFO />
            <BILL_WHEN>1</BILL_WHEN>
            <BILLINFO_ID>88-CYZZ5</BILLINFO_ID>
            <CURRENCY>840</CURRENCY>
            <PAY_TYPE>10001</PAY_TYPE>
            <PAYINFO />
            <POID>0.0.0.1 /billinfo -1 0</POID>
          </BILLINFO>
        </PCM_OP_CUST_COMMIT_CUSTOMER_inputFlist>
      </PCM_OP_CUST_COMMIT_CUSTOMER_Request>
    </bus:pcmOpCustCommitCustomer>
  </soapenv:Body>
</soapenv:Envelope>
```
Sample SOAP Response Output XML File

The following sample shows a SOAP response message for the pcmOpCustCommitCustomer Web service API.

```xml
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"

<soapenv:Body>
  <brm:PCM_OP_CUST_COMMIT_CUSTOMER_outputFlist
    xmlns:brm="http://xmlns.oracle.com/BRM/schemas/BusinessOpcodes">
    <brm:ACCOUNT_OBJ>0.0.0.1 /account Obj 225757 0</brm:ACCOUNT_OBJ>
    <brm:ACCTINFO elem="0">
      <brm:ACCOUNT_NO>a022020202011992</brm:ACCOUNT_NO>
      <brm:BAL_INFO elem="0"/>
      <brm:BUSINESS_TYPE>1</brm:BUSINESS_TYPE>
      <brm:CURRENCY>840</brm:CURRENCY>
      <brm:POID>0.0.0.1 /account -1 0</brm:POID>
    </brm:ACCTINFO>
  </brm:PCM_OP_CUST_COMMIT_CUSTOMER_outputFlist>
</soapenv:Body>
</soapenv:Envelope>
```
Example of Creating an Account in BRM Using Web Services

- `<brm:BAL_INFO elem="0">
  `<brm:ACCOUNT_OBJ>0.0.0.1 /account 225757 0</brm:ACCOUNT_OBJ>
  `<brm:BILLINFO_OBJ>0.0.0.1 /billinfo 226269 0</brm:BILLINFO_OBJ>
  `- `<brm:LIMIT elem="840">
    `<brm:CREDIT_LIMIT />
  </brm:LIMIT>
  `<brm:NAME>Account Level Balance Group</brm:NAME>
  `<brm:POID>0.0.0.1 /balance_group 225341 0</brm:POID>
  `<brm:SERVICE_OBJ>0.0.0.0 0 0</brm:SERVICE_OBJ>
</brm:BAL_INFO>
- `<brm:BILLINFO elem="0">
  `<brm:BAL_GRP_OBJ>0.0.0.1 /balance_group 225341 0</brm:BAL_GRP_OBJ>
  `<brm:BILLINFO_ID>88-CYZZ5</brm:BILLINFO_ID>
  `<brm:BILL_WHEN>1</brm:BILL_WHEN>
  `<brm:CURRENCY>840</brm:CURRENCY>
  `<brm:CURRENCY_SECONDARY>0</brm:CURRENCY_SECONDARY>
  `<brm:EFFECTIVE_T>2010-02-17T22:37:49Z</brm:EFFECTIVE_T>
  `<brm:PAYINFO_OBJ>0.0.0.1 /payinfo/invoice 226781 0</brm:PAYINFO_OBJ>
  `<brm:PAY_TYPE>10001</brm:PAY_TYPE>
  `<brm:POID>0.0.0.1 /billinfo 226269 0</brm:POID>
</brm:BILLINFO>
  `<brm:END_T>2010-02-17T22:37:49Z</brm:END_T>
  `<brm:FLAGS>0</brm:FLAGS>
  `<brm:GROUP_INFO />
- `<brm:HOST elem="1">
  `<brm:HOSTNAME>XXX.XXX.XXX.XXX</brm:HOSTNAME>
  `<brm:TYPE>1</brm:TYPE>
</brm:HOST>
- `<brm:HOST elem="2">
  `<brm:HOSTNAME>XXX.XXX.XXX.XXX</brm:HOSTNAME>
  `<brm:TYPE>1</brm:TYPE>
</brm:HOST>
- `<brm:HOST elem="3">
  `<brm:HOSTNAME>XXXXXXXXX.XXX</brm:HOSTNAME>
  `<brm:PORT>0</brm:PORT>
  `<brm:TYPE>2</brm:TYPE>
</brm:HOST>
- `<brm:HOST elem="4">
  `<brm:HOSTNAME>XXXX.XXX</brm:HOSTNAME>
  `<brm:TYPE>3</brm:TYPE>
</brm:HOST>
- `<brm:HOST elem="5">
  `<brm:HOSTNAME>XXXX.XXX</brm:HOSTNAME>
  `<brm:TYPE>4</brm:TYPE>
</brm:HOST>
  `<brm:HTTP_URL>XXXXXXXXXXXXXXX</brm:HTTP_URL>
- `<brm:LOCALES elem="1">
  `<brm:LOCALSEN>en_US</brm:LOCALSEN>
</brm:LOCALES>
- `<brm:NAMEINFO elem="1">
  `<brm:ADDRESS>123 Hollywood Boulevard</brm:ADDRESS>
  `<brm:CANON_COUNTRY>US</brm:CANON_COUNTRY>
  `<brm:CITY>Los Angeles</brm:CITY>
  `<brm:COMPANY />
  `<brm:CONTACT_TYPE>Account holder</brm:CONTACT_TYPE>
  `<brm:COUNTRY>USA</brm:COUNTRY>
  `<brm:ELEMENT_ID>1</brm:ELEMENT_ID>
  `<brm:EMAIL_ADDR>test_001</brm:EMAIL_ADDR>
  `<brm:FIRST_NAME>Chetn3457</brm:FIRST_NAME>
  `<brm:LAST_NAME>Chet8905</brm:LAST_NAME>
About Data Masking in Web Services Responses

SOAP output response XML files may contain masked fields as configured by your BRM implementation. Subscriber fields, including payment information and user credentials, may be hidden in responses for securing sensitive subscriber data.

See "About Securing Sensitive Customer Data with Masking" in BRM Managing Customers for more information on configuring data masking.
Installing Web Services Manager

This document contains information on the requirements for installing and setting up Oracle Communications Billing and Revenue Management (BRM) Web Services Manager.

Installing Web Services Manager

**Note:** If you already installed the product, you must uninstall its features before reinstalling them.

Before you install and configure Web Services Manager, you must install the following:

- A supported, standards-compliant server. See "Supported Servers" for a list of the servers supported by Web Services Manager. See server documentation for more information.
- BRM. See *BRM Installation Guide* for more information.

You must also increase the heap size used by the Java Virtual Machine (JVM) before running the installation program to avoid "Out of Memory" error messages in the log file. For information, see "Increasing Heap Size to Avoid "Out of Memory" Error Messages" in *BRM Installation Guide*

**Note:** Oracle recommends that you install Web Services Manager on the system on which BRM is installed.

To install Web Services Manager, see "Installing Individual BRM Components" in *BRM Installation Guide*.

Supported Servers

Web Services Manager is supported on the following servers:

- Oracle WebLogic Server
- Apache Tomcat server

Uninstalling Web Services Manager

To uninstall Web Services Manager, see "Uninstalling Optional Components" in *BRM Installation Guide*.
Uninstalling Web Services Manager
Deploying Web Services Manager

This chapter describes how to deploy and launch the Oracle Communications Billing and Revenue Management (BRM) Web Services Manager.

Topics in this document:

- Deploying and Launching Web Services Manager on WebLogic Server
- Deploying and Launching Web Services Manager on Tomcat Server

See also:

- Using Web Services
- Configuring Web Services Manager
- Customizing Web Services

Deploying and Launching Web Services Manager on WebLogic Server

You can deploy Web Services Manager on WebLogic server through the WebLogic Server Administration Console. Depending on the type of payload supported by web services, deploy one of the following files:

- infranetwebsvc.war: Includes web services that support the payload as an XML string data type.
- BrmWebServices.war: Includes web services that support the payload as an XML element data type.

If you customize web services, regenerate infranetwebsvc.war or BrmWebServices.war and use the generated version. Otherwise, you should use the default infranetwebsvc.war or BrmWebServices.war file. For more information about customizing web services, see "Customizing Web Services."

To deploy Web Services Manager on WebLogic Server:

1. Create the WebLogic Server domain. See the discussion about creating a WebLogic domain in Fusion Middleware Creating Domains Using the Configuration Wizard for detailed instructions.

2. If you deploy the BrmWebServices.war file, set the heap size required to start WebLogic Server:
   a. Open the WebLogic_home/user_projects/domains/Domain_Name/setDomainEnv.sh file in a text editor.

   Where:

   WebLogic_home is the directory in which WebLogic Server is installed.
domain_name is the name of the domain you created in step 1.

b. Add the following entry:

```
USER_MEM_ARGS ="-Xms2048m -Xmx2048m"
```

c. Save and close the file.
d. Restart WebLogic Server.

3. Do one of the following:

- If you customized web services:
  - Extract the `BRM_home/deploy/web_services/infranetwebsvc.war` or the `BRM_home/deploy/web_services/BrmWebServices.war` file to `local_dir`.
    
    Where:
    
    `BRM_home` is the directory in which BRM is installed.
    
    `local_dir` is a directory on the machine on which you installed WebLogic Server.
    
    - Copy the `CustomFields.jar` files to the `local_dir/WEB-INF/lib` directory. See "Setting Up Web Services Manager to Support Custom Opcodes" for more information.

    **Note:** The JRE version that was used to generate `CustomFields.jar` must be the same or lower than the version of the WebLogic Server JRE.

    - Open the `BRM_home/deploy/web_services/Infranet.properties` file in a text editor.
    - Modify the following entry:
      
      ```
      infranet.custom.field.package = package
      ```
      
      where `package` is the name of the package that contains the CustomOp.java file; for example, `com.portal.classFiles`.
    - Add all the custom fields to the `Infranet.properties` file.
    - Save and close the file.
    - Copy the `BRM_home/deploy/web_services/Infranet.properties` file to the `local_dir/WEB-INF/classes` or in the home directory on the machine on which WebLogic Server is installed.
    - Regenerate the WAR file by running one of the following commands:
      To regenerate the `infranetwebsvc.war` file:
      ```
      jar -cvf infranetwebsvc.war *
      ```
      
      To regenerate the `BrmWebServices.war` file:
      ```
      jar -cvf BrmWebServices.war *
      ```

- If you did not customize web services:
  - Extract the `BRM_home/deploy/web_services/infranetwebsvc.war` or the `BRM_home/deploy/web_services/BrmWebServices.war` file to `local_dir`.
Deploying and Launching Web Services Manager on WebLogic Server

- Copy the `BRM_home/deploy/web_services/Infranet.properties` file to the `local_dir/WEB-INF/classes` directory or in the home directory on the machine on which WebLogic Server is installed.

- Regenerate the WAR file by running one of the following commands:

  To regenerate the `infranetwebsvc.war` file:

  ```bash
  jar -cvf infranetwebsvc.war *
  ```

  To regenerate the `BrmWebServices.war` file:

  ```bash
  jar -cvf BrmWebServices.war *
  ```

4. Log in to WebLogic Server Administration Console.
5. In the Domain Structure pane, click **Deployments**.
   The Summary of Deployments page appears.
6. Click **Install**.
   The Install Application Assistant page appears.
7. Enter the local directory path for `infranetwebsvc.war` or `BrmWebServices.war` in the Path field and press **Enter**.
8. Select **Install this deployment as an application** and click **Next**.
9. Select the server or servers on which you want to deploy Web Services Manager and click **Next**.
10. Click **Next**.

---

**Note:** When you deploy Web Services Manager on WebLogic Server, select the **Custom Roles and Policies** option in the Security section of WebLogic Server Administration Console.

---

11. Verify your deployment options and click **Finish**.
   WebLogic Server displays the deployed application on the Deployments page.

To launch Web Services Manager for web services, do one of the following:
- If you have deployed `infranetwebsvc.war`, launch web services from the WebLogic Server Administration Console:
  1. On the Home page, select **Deployments** in the Domain Structure pane.
     The Summary of Deployments page appears.
  2. Click the `infranetwebsvc` link.
     The Settings page for the deployment appears.
  3. Click the **Testing** tab.
  4. Click the default url link.
     The Apache Axis Web page appears.
  5. Click the link for the list of the supported BRM web services for this Web Services Manager deployment.
- If you have deployed `BrmWebServices.war`, launch web services from the WebLogic Server Administration Console:

2. Click the **BrmWebServices** link, a list of all the web services is displayed.

3. Click on any of the web service.

4. Click the **Testing** tab.

5. The WebLogic Server Administration console displays an URL for the BRM web services.

### Deploying and Launching Web Services Manager on Tomcat Server

You can deploy Web Services Manager on Tomcat Server through the Tomcat Web Application Manager. Depending on the type of payload supported by web services, deploy one of the following files:

- **infranetwebsvc.war**: Includes the web services that support the payload as an XML string data type. See "Deploying and Launching infranetwebsvc.war."
- **BrmWebServices.war**: Includes the web services that support the payload as an XML element data type. See "Deploying and Launching BrmWebServices.war."

If you customize web services, regenerate **infranetwebsvc.war** or **BrmWebServices.war** and use the generated version. Otherwise, you should use the default **infranetwebsvc.war** or **BrmWebServices.war** file. For more information about customizing web services, see "Customizing Web Services."

### Deploying and Launching infranetwebsvc.war

To deploy Web Services Manager for web services that support the payload as an XML string data type, on Tomcat server:

1. Create the Tomcat server domain. See Tomcat documentation for detailed instructions.


3. Extract the **jaxws-ri-2.2.10.zip** file and copy the following files to **Tomcat_home/lib**, where **Tomcat_home** is the directory in which the Tomcat server is installed:

   - gmbal-api-only.jar
   - jaxb-api.jar
   - jaxb-impl.jar
   - jaxws-rt.jar
   - management-api.jar
   - policy.jar
   - stax-ex.jar
   - jaxb-core.jar
   - ha-api.jar
   - jaxws-tools.jar
   - jaxb-xjc.jar
Deploying and Launching Web Services Manager on Tomcat Server

- jaxb-impl.jar
- jaxb-jxc.jar
- streambuffer.jar

4. In the **War file to deploy** section, click **Browse**.

5. Download `jaxrpc.jar` from the Download jaxrpc.jar web page

6. Click **Deploy**.

   Tomcat Web Application Manager displays the deployed application in the **Applications** list.

---

**Note:** (UNIX only) The GUI user must have execute permissions for `infranetwebsvc.war`. If the GUI user does not have the appropriate permissions, Tomcat will deploy the application, but will be unable to start it. Verify that the Running column of the Applications list has a value of true for `/infranetwebsvc` to ensure that Web Services Manager has been deployed successfully.

---

To launch Web Services Manager for web services that support the payload as an XML string data type, from the Tomcat Web Application Manager:

1. In the Applications list, click the `/infranetwebsvc` link.

   The Apache Axis Web page appears.

2. Click the **Services** link to view a list of the supported BRM web services for this Web Services Manager deployment.

---

**Deploying and Launching BrmWebServices.war**

To deploy Web Services Manager for web services that support the payload as an XML element data type, on Tomcat server:

1. Create the Tomcat server domain. See Tomcat documentation for detailed instructions.


3. Extract the `jaxws-ri-2.2.10.zip` file and copy the following files to `Tomcat_home/lib`, where `Tomcat_home` is the directory in which Tomcat server is installed:

   - gmbal-api-only.jar
   - jaxb-api.jar
   - jaxb-impl.jar
   - jaxws-rt.jar
   - management-api.jar
   - policy.jar
   - stax-ex.jar
   - jaxb-core.jar
Deploying and Launching Web Services Manager on Tomcat Server

- ha-api.jar
- jaxws-tools.jar
- jaxb-xjc.jar
- jaxb-impl.jar
- jaxb-jxc.jar
- streambuffer.jar

4. Download xalan-2.7.0.jar from the Download xalan-2.7.0.jar web page (http://www.java2s.com/Code/Jar/x/Downloadxalan270jar.htm) and copy the file to Tomcat_home/lib.

5. Download jaxrpc.jar from the Download jaxrpc.jar web page (http://www.java2s.com/Code/Jar/j/Downloadjaxrpcjar.htm) and copy the file to Tomcat_home/lib.

6. If you plan to deploy the BrmWebServices.war file, do the following:
   a. Open the BrmWebServices.war/WEB-INF/web.xml in a text editor.
   b. Uncomment the serverlet-to-URL mapping.
   c. Save and close the file.

7. Copy the BRM_home/deploy/web_services/BrmWebServices.war file to a local directory on the machine on which you installed Tomcat server, where BRM_home is the directory in which BRM is installed.

8. Log in to the Tomcat Web Application Manager.

9. In the War file to deploy section, click Browse....

10. Select the BrmWebServices.war file.

11. Click Deploy.

   Tomcat Web Application Manager displays the deployed application in the Applications list.

To launch Web Services Manager for web services that support the payload as an XML element data type, from the Tomcat Web Application Manager:

1. In the Applications list, click the /BrmWebServices link.

2. The Tomcat Web Application Manager displays an HTTP and an HTTPS URL for the BRM web services.

   A sample URL for the BRMCUSTServices_v2Web service is as follows:
   http://ipaddress:port/BrmWebServices/BRMCUSTServices_v2?wsdl

   where:
   - ipaddress is the domain IP address of the application server on which Web Services Manager is deployed.
   - port is the domain port number of the application server on which Web Services Manager is deployed.

   Web Services Manager displays the WSDL URLs for each available service.

   For more information on the BRM web services included in Web Services Manager that take the payload as an XML element data type, see "Customizing Web Services".
This chapter describes how to configure Oracle Communications Billing and Revenue Management (BRM) Web Services Manager. Configuring Web Services Manager requires connecting the deployed application to the BRM system and configuring security, authorization, and Java logging for the deployed application.

Topics in this document:

- About Connecting Web Services Manager to the BRM System
- Configuring Security for Web Services Manager
- Disabling the JarScanner Feature in Tomcat Server
- Configuring Java Logging for the Application Server

See also:

- Using Web Services
- Deploying Web Services Manager
- Customizing Web Services

About Connecting Web Services Manager to the BRM System

Web Services Manager connects to the BRM system through a BRM Connection Manager (CM). Figure 4–1 shows how BRM and the SOAP client communicate with the deployed application. Web Services Manager translates Portal Communication Module (PCM) communications sent from a CM in the BRM system into SOAP requests sent to the SOAP client over HTTP. Web Services Manager translates SOAP responses sent from the SOAP client over HTTP into PCM communications that are returned to the CM.
Connecting Web Services Manager to the BRM System

Before you connect Web Services Manager to the BRM system, verify that your instance of Web Services Manager is deployed to an application server.

If you customized Web services, use the custom `infranetwebsvc.war` or `BrmWebServices.war` file. Otherwise, you should use the default `infranetwebsvc.war` or `BrmWebServices.war` file. For more information about customizing Web services, see "Customizing Web Services."

To connect Web Services Manager to the BRM system, do the following:

1. On your application server, copy the $BRM_HOME/deploy/web_services/Infranet.properties$ file to one of the following:
   - `$local_dir/WEB-INF/classes` directory, where `$local_dir` is a directory on the machine on which you installed your application server.
   - The home directory on the machine on which you installed your application server.

   **Note:** If you copy the `Infranet.properties` file to the `$local_dir/WEB-INF/classes` directory, extract the `infranetwebsvc.war` file or `BrmWebServices.war` file to a local directory ($local_dir) on the system on which your application server is installed.

2. Open the `Infranet.properties` file in a text editor.

3. Locate the following lines:

   ```
   infranet.connection=pcp://root.0.0.0.1:password@ipAddress:port/0.0.0.1/service/admin_client 1
   infranet.login.type=1
   ```

4. Do the following:
a. Replace password with the password for the BRM server.
b. Replace ipAddress with the IP address of the system on which BRM is installed.
c. Replace port with the port number used by the application server on which BRM is installed.

5. If SSL is enabled in connection manager (CM), locate the following lines and update the parameters if necessary:

```
infranet.pcp.ssl.enabled=true
infranet.pcp.ssl.wallet.location=wallet_directory
```

where wallet_directory is the path to your client Oracle wallet. The client Oracle wallet contains the optional client SSL certificate and the private key, and it contains the Trusted CA certificate.

6. If you added custom opcodes or custom fields for Web services, add the enum values of the custom fields.

For example, if you created the custom fld usage id custom field and the enum value for the custom fld usage id field is 10001, add the following entry:

```
infranet.custom.field.10001=custom fld usage id
```

For information about mapping enum values, see "Creating Custom Fields" in BRM Developer’s Guide.

7. (Optional) To configure the connection pool parameters, modify the following entries:

```
infranet.connectionpool.minsize=min_connections
infranet.connectionpool.maxsize=max_connections
infranet.connectionpool.timeout=connection_timeout
```

where:

- min_connections is the minimum number of connections allowed in the pool. The default number is 1.
- max_connections is the maximum number of connections allowed in the pool.
- connection_timeout is the connection pool timeout in milliseconds.

8. (Optional) To configure logging for Web Services Manager, modify the following entry:

```
webservices.log.enabled=log_value
```

where log_value is one of the following:

- **true** enables logging. This option saves and displays the log files as standard output in the application server console.
- **false** disables logging. This option saves the log files in the /domain/logs/BRMWebSvcMgr.log file. Configure the BRM_home/deploy/web_services/lib/weblogic_ws_startup.jar file to use this option.

9. Save and close the file.

10. If you are working in the local dir/WEB-INF/classes directory, regenerate the WAR file by running one of the following commands:

To regenerate the infranetwebsvc.war file:

```
jar -cvf infranetwebsvc.war *
```
To regenerate the BrmWebServices.war file:

```
jar -cvf BrmWebServices.war *
```

11. Deploy the regenerated infranetwebsvc.war or BrmWebServices.war file on the server. See “Deploying Web Services Manager.”

### Changing the Instance of BRM to which Web Services Manager Connects

If you customized Web services, use the custom infranetwebsvc.war or BrmWebServices.war file. Otherwise, you should use the default infranetwebsvc.war or BrmWebServices.war file. For more information about customizing Web services, see “Customizing Web Services.”

To change the instance of BRM to which Web Services Manager connects, do the following:

1. On your application server, copy the BRM_home|deploy/web_services/Infranet.properties file to one of the following:
   - local_dir/WEB-INF/classes directory, where local_dir is a directory on the machine on which you installed your application server.
   - The home directory on the machine on which you installed your application server.

   __Note:__ If you copy the Infranet.properties file to the local_dir/WEB-INF/classes directory, extract the infranetwebsvc.war or BrmWebServices.war file to a local directory (local_dir) on the system on which your application server is installed.

2. Open the copied Infranet.properties file.

3. Locate the following lines:

   ```
infranet.connection=pcp://root.0.0.0.1:password@ipAddress:port/0.0.0.1/service/admin_client 1
   infranet.login.type=1
   ```

4. Do the following:
   a. Replace `password` with the password for the BRM server.
   b. Replace `ipAddress` with the IP address of the system on which BRM is installed.
   c. Replace `port` with the port number used by the application server on which BRM is installed.

5. If SSL is enabled in the Connection Manager (CM), locate the following lines and update the parameters if necessary:

   ```
infranet.pcp.ssl.enabled=true
   infranet.pcp.ssl.wallet.location=wallet_directory
   ```

   where `wallet_directory` is the path to your client Oracle wallet. The client Oracle wallet contains the optional client SSL certificate and the private key, and it contains the Trusted CA certificate.

6. If you added custom opcodes or custom fields for Web services, add the enum values of the custom fields.
For example, if you created the `custom_fld_usage_id` custom field and the enum value for the `custom_fld_usage_id` field is 10001, add the following entry:

```
infranet.custom.field.10001=custom_fld_usage_id
```

For information about mapping enum values, see "Creating Custom Fields" in the BRM Developer’s Guide.

7. (Optional) To configure the connection pool parameters, modify the following entries:

```
infranet.connectionpool.minsize=min_connections
infranet.connectionpool.maxsize=max_connections
infranet.connectionpool.timeout=connection_timeout
```

where:

- `min_connections` is the minimum number of connections allowed in the pool. The default number is 1.
- `max_connections` is the maximum number of connections allowed in the pool.
- `connection_timeout` is the connection pool timeout in milliseconds.

8. (Optional) To configure logging for Web Services Manager, modify the following entry:

```
webservices.log.enabled=log_value
```

where `log_value` is one of the following:

- `true` enables logging. This option saves and displays the log files as standard output in the application server console.
- `false` disables logging. This option saves the log files in the `/domain/logs/BRMWebSvcMgr.log` file. Configure the `BRM_home/deploy/web_services/lib/weblogic_ws_startup.jar` file to use this option.

9. (Optional) To configure searching in BRM using the PCM_OP_SEARCH opcode, restrict the PCM_OP_SEARCH opcode to pre-defined search templates by modifying the following entry:

```
allowed.search.template.ids=template_id
```

where `template_id` is the template ID of the search template that you want the PCM_OP_SEARCH opcode to use for searching. Use a comma (,) to separate multiple template IDs. If you do not want to restrict the PCM_OP_SEARCH opcode to any pre-defined search templates, set `template_id` to `None`.

For a list of template IDs, connect to the BRM database and check the list of POIDS and the respective templates in the SEARCH_T table in the BRM database. For more information, see "Searching for Objects in the BRM Database" in the BRM Developer’s Guide.

10. If you added custom opcodes or custom fields for Web services, add the enum values of the custom fields. For information about mapping enum values, see "Creating Custom Fields" in the BRM Developer’s Guide.

For example, if you created the `custom_fld_usage_id` custom field and the enum value for the `custom_fld_usage_id` field is 10001, add the following entry:

```
infranet.custom.field.10001=custom_fld_usage_id
```

11. Save and close the file.
12. If you are working in the local_dir/WEB-INF/classes directory, regenerate the WAR file by running one of the following commands:

To regenerate the infranetwebsvc.war file:

```
jar -cvf infranetwebsvc.war *
```

To regenerate the BrmWebServices.war file:

```
jar -cvf BrmWebServices.war *
```

13. Deploy the regenerated infranetwebsvc.war or BrmWebServices.war file on the server. See “Deploying Web Services Manager.”

---

## Configuring Security for Web Services Manager

By default, secure sockets layer (SSL) security for Web Services Manager is enabled. If you disabled SSL during the BRM server installation, you can enable SSL in Web Services Manager by configuring security parameters and enabling the SSL security feature in the application server on which Web Services Manager is deployed.

### Configuring Security for Web Services Manager in WebLogic Server

Before you configure security for Web Services Manager, ensure that WebLogic Server and Web Services Manager are installed and that Web Services Manager has been deployed on a WebLogic Server domain. See "Installing Web Services Manager" and "Deploying Web Services Manager" for more information.

To configure security for Web Services Manager in WebLogic Server, do the following:

- Configure authentication for Web Services Manager. See "Configuring Authentication for WebLogic Server."

- Configure authorization for Web Services Manager by doing one of the following:
  - If you have deployed infranetwebsvc.war, configure role-based authentication for Apache Axis. See "Configuring Role-Based Authentication for Apache Axis in WebLogic Server."
  - If you have deployed BrmWebServices.war, configure WebLogic security policy for JAX-WS. See "Configuring WebLogic Security Policy on BRM Web Services for JAX-WS in WebLogic Server."

### Configuring Authentication for WebLogic Server

Before you configure authentication for Web Services Manager, create a user, group, and security realm for Web Services Manager in WebLogic Server. For more information about creating users and groups, see the discussion about users, groups, and security roles in *Fusion Middleware Securing Resources Using Roles and Policies for Oracle WebLogic Server*. For more information about security realms, see the discussion about security realms in WebLogic Server in *Fusion Middleware Securing Oracle WebLogic Server*.

To configure authentication for Web Services Manager in WebLogic Server:

1. Open the local_dir/infranetwebsvc.war/WEB-INF/weblogic.xml file in a text editor, where local_dir is a directory on the WebLogic host where you copied the infranetwebsvc.war file.

2. Uncomment the following lines:

```
# <security-role-assignment>
```
3. Save and close the file.

4. Open the local_dir/infranetwebsvc.war/WEB-INF/web.xml file in a text editor.

5. Uncomment the following lines:

```
# <security-constraint>
# <web-resource-collection>
#  <web-resource-name>restricted web services</web-resource-name>
#  <url-pattern>/*</url-pattern>
#  <http-method>GET</http-method>
#  <http-method>POST</http-method>
# </web-resource-collection>
# <auth-constraint>
#  <role-name>brmws</role-name>
# </auth-constraint>
# <user-data-constraint>
#  <transport-guarantee>CONFIDENTIAL</transport-guarantee>
# </user-data-constraint>
# </security-constraint>
# <login-config>
#  <auth-method>BASIC</auth-method>
#  <realm-name>default</realm-name>
# </login-config>
# <security-role>
#  <role-name>brmws</role-name>
# </security-role>
```

6. Save and close the file.

7. Log in to WebLogic Server Administration Console.

8. Expand Environment and select Servers.

The Summary of Servers page appears.

9. Select the server for which you want to enable the SSL port.

10. Click the Configuration tab.

11. Click the General subtab.

12. Select the SSL Listen Port Enabled check box.

13. In the SSL Listen Port field, enter a free port number. The default is 7002.

14. Click Save, which configures Web Services Manager with the following default port numbers:

- HTTP connection: 7001
  
  Web services that take the payload as an XML element have the default connection set to HTTP.

- HTTPS connection: 7002

If you use a SOAP development application to generate a Web service client and use port numbers other than the default port numbers, the URLs for the Web services that take the payload as an XML element will show port numbers that do not match the port numbers you configured in WebLogic Server Administration
Configuring Security for Web Services Manager

Console. Populate the correct port numbers in the URLs for the WSDL files that are generated dynamically by your SOAP development application by doing either of the following:

- Change the port numbers manually in your SOAP development application request.
- Change the port numbers in the `infranetwebsvc.war/WEB-INF/conf/axis2.xml` file as follows:
  
  Open the `local_dir/infranetwebsvc.war/WEB-INF/conf/axis2.xml` file in a text editor.
  
  Locate the following line and change the HTTP port number:
  ```xml
  <parameter name="port">7001</parameter>
  ```
  
  Locate the following line and change the HTTPS port number:
  ```xml
  <parameter name="port">7002</parameter>
  ```
  
  Save and close the file.
  
  Regenerate the `infranetwebsvc.war` file by running the following command:
  ```shell
  jar -cvf infranetwebsvc.war *
  ```
  
  Deploy the regenerated `infranetwebsvc.war` file on WebLogic Server. See "Deploying and Launching Web Services Manager on WebLogic Server" for instructions on deploying Web Services Manager.

Configuring Role-Based Authentication for Apache Axis in WebLogic Server

You configure access restrictions for Web Services Manager by creating security roles and by adding conditions to the security roles.

To configure role-based authorization for Apache Axis in WebLogic Server:

1. Log in to WebLogic Server Administration Console.
2. In the Domain Structure pane, click Deployments and click the `infranetwebsvc` link.
3. Click the Security tab.
4. Click the Application Scope subtab.
5. Click the Roles subtab.
6. Click New.
   
   The Create Stand-Alone Web Application Scoped Roles page appears.
7. In the Name field, enter a name for the role. For example `brmws`.
8. In the Provider Name list, select XACMLRoleMapper.
9. Click OK.
10. Click the link with the role name you created, for example the `brmws` link.
11. Click Add Conditions.
12. In the Predicate List list, select Group.
13. Click Next.
14. In the Group Argument Name field, enter the name of the desired group.
15. Click Add.
16. Click Finish.
17. Click Save.

Configuring WebLogic Security Policy on BRM Web Services for JAX-WS in WebLogic Server

You define access restrictions for Web services in security policies in WebLogic Server. To configure WebLogic Security Policy on BRM Web Services for JAX-WS in WebLogic Server:

1. Log in to WebLogic Server Administration Console.
2. In the Domain Structure pane, click Deployments and click the BrmWebServices link. A list of all the web services is displayed.
3. Click any of the web service.
4. Click the Configuration tab.
5. Click the WS-Policy subtab.
6. Click the WS-Policy files associated with this web service.
7. Select WebLogic in the Configure the Policy Type for a Web Service section.
8. Add the policies from the Available Endpoint Policies for the selected service.

   If you want to use the policy for HTTPS with basic authentication, add policy:policy_name-Https-BasicAuth.xml, where policy_name is name of the policy for the selected service; for example, policy:Wssp1.2-2007-Https-BasicAuth.xml.

   If you want to use the policy for HTTPS without authentication, add policy:policy_name-Https.xml, where policy_name is name of the policy for the selected service; for example, policy:Wssp1.2-2007-Https.xml.

9. Click Finish.
10. Click OK in the Save Deployment Plan Assistant section.

   If you have multiple deployments, then the plan.xml, which is created when you assign a policy to the service, should be saved in its respective deployment directory.
11. Click the Security tab.
12. Click the Policies subtab.
13. In the Web Service Methods list, select the web method that you want to secure.
14. Click Add Conditions.
15. In the Predicate List list, select one of the following: Roles, Users, or Groups.
16. Click Next.
17. In the User Argument Name field, add the user/group.
18. Click Add.
19. Click Finish.
20. Click Save.

If you have enabled SSL, add the following entry in the BRMWebServices.war/WEB-INF/web.xml file to enable cookie security:
Configuring Security for Web Services Manager

Configuring Security for Web Services Manager in Tomcat Server

Before you configure security for Web Services Manager, ensure that Tomcat server and Web Services Manager are installed and that Web Services Manager has been deployed on a Tomcat server domain. See "Installing Web Services Manager" and "Deploying Web Services Manager" for more information.

To configure security for Web Services Manager in Tomcat server, do the following:

- Configure authentication for Web Services Manager for Apache Axis in Tomcat server. See "Configuring Authentication for Web Services Manager for Apache Axis in Tomcat Server."
- Enable SSL in Tomcat server. See "Enabling SSL in Tomcat Server."

Configuring Authentication for Web Services Manager for Apache Axis in Tomcat Server

To configure authentication for Web Services Manager for Apache Axis in Tomcat server:

1. Open the `local_dir/infranetwebsvc.war/WEB-INF/web.xml` file in a text editor.
2. Uncomment the following lines:

   # <security-constraint>
   # <web-resource-collection>
   #  <web-resource-name>restricted web services</web-resource-name>
   #  <url-pattern>/*</url-pattern>
   # </web-resource-collection>
   # <auth-constraint>
   #  <role-name>brmws</role-name>
   # </auth-constraint>
   # <user-data-constraint>
   #  <transport-guarantee>CONFIDENTIAL</transport-guarantee>
   # </user-data-constraint>
   # </security-constraint>

   # <login-config>
   #  <auth-method>BASIC</auth-method>
   #  <realm-name>default</realm-name>
   # </login-config>
   # <security-role>
   #  <role-name>brmws</role-name>
   # </security-role>

3. Locate the following line and specify the Web resource name as follows:

   <web-resource-name>Protected Resource</web-resource-name>
4. Locate the following lines and specify the realm name as follows:

```xml
<login-config>
  <auth-method>BASIC</auth-method>
  <realm-name>Apache version</realm-name>
</login-config>
```

where version is the Tomcat server version on which you deployed Web Services Manager.

5. Save and close the file.

6. Open the local_dir/apache-tomcat-version/conf/tomcat-users.xml file in a text editor.

7. Locate the following lines and specify the login details of the user:

```xml
<role rolename="brmws"/>
<user username="username" password="password" roles="brmws"/>
```

where:
- `username` is the user name for accessing Web services.
- `password` is the password for accessing Web services.

8. Save and close the file.

9. Open the config/server.xml file in a text editor.

10. In the `<Engine>` section, add the following class path:

```xml
<Realm className="org.apache.catalina.realm.MemoryRealm" />
```

11. Save and close the file.

12. Restart the Tomcat server.

### Configuring Authentication for Web Services Manager for JAX-WS in Tomcat Server

To configure authentication for Web Services Manager for JAX-WS in Tomcat server:

1. Open the local_dir/BrmWebServices.war/WEB-INF/web.xml file in a text editor.

2. Add the following lines:

```xml
# <security-constraint>
  # <web-resource-collection>
    # <web-resource-name>restricted web services</web-resource-name>
    # <url-pattern>/*</url-pattern>
    # <http-method>GET</http-method>
    # <http-method>/POST</http-method>
  # </web-resource-collection>
  # <auth-constraint>
    # <role-name>brmws</role-name>
  # </auth-constraint>
  # <user-data-constraint>
    # <transport-guarantee>CONFIDENTIAL</transport-guarantee>
  # </user-data-constraint>
# </security-constraint>

# <login-config>
  # <auth-method>BASIC</auth-method>
# </login-config>
# <security-role>
  # <role-name>brmws</role-name>
```
Configuring Security for Web Services Manager

# </security-role>

3. Save and close the file.

4. Open the `local_dir/apache-tomcat-version/conf/tomcat-users.xml` file in a text editor.

5. Locate the following lines and specify the login details of the user:

   ```xml
   <role rolename="brmws"/>
   <user username="username" password="password" roles="brmws"/>
   ```

   where:
   - `username` is the user name for accessing Web services.
   - `password` is the password for accessing Web services.

6. Save and close the file.

7. Open the `config/server.xml` file in a text editor.

8. In the `<Engine>` section, add the following class path:

    ```xml
    <Realm className="org.apache.catalina.realm.MemoryRealm" />
    ```

9. Save and close the file.

10. Restart the Tomcat server.

### Enabling SSL in Tomcat Server

To enable secure communication for Web Services Manager, enable secure sockets layer (SSL) in the Tomcat server domain on which you deploy Web Services Manager.

To enable SSL for Tomcat server:

1. Generate the keystore by running the following command:

   ```
   keytool -genkey -alias mykes -keyalg RSA -keystore mykeystore
   ```

   where:
   - `mykes` is the alias.
   - `mykeystore` is the name of the keystore.

2. Open the `conf/server.xml` file in a text editor.

3. Uncomment the following lines and specify the path for the keystore file:

   ```xml
   address="IPAddress"
   maxThreads="150" SSLEnabled="true" scheme="https" secure="true"
   clientAuth="false" sslProtocol="TLS"
   keystoreFile="filepath"
   keystorePass="password" />
   ```

   where:
   - `IPAddress` is the IP address of the machine on which you installed the Apache Tomcat server.
   - `filepath` is the keystore file path.
   - `password` is the password for the keystore file.

4. Save and close the file.
Disabling the JarScanner Feature in Tomcat Server

The JarScanner feature in the Tomcat server is used to scan the web application for JAR files. To avoid unnecessary warnings displayed for optional JAR files, disable the JarScanner feature in the Tomcat server.

To disable the JarScanner feature in the Tomcat server:

1. Open the `local_dir/apache-tomcat-version/conf/context.xml` in a text editor.
2. Search for the following entry:
   ```xml
   <JarScanner scanClassPath="true" scanAllFiles="false"
               scanAllDirectories="false"></JarScanner>
   ```
3. Set the `scanClassPath` entry to `false`:
   ```xml
   <JarScanner scanClassPath="false" scanAllFiles="false"
               scanAllDirectories="false"></JarScanner>
   ```
4. Save and close the file.

Configuring Java Logging for the Application Server

Depending on your configuration, you may wish to change the level of Java logging on the application server. To configure the Java logging level, do the following:

- For WebLogic Server, see "Configuring Java Logging for WebLogic Server" for Web Services Manager-specific configuration. For more information, see the discussion about application logging and WebLogic logging services in *Fusion Middleware Using Logging Services for Application Logging for Oracle WebLogic Server*.
- For Tomcat server, see the discussion about logging in Tomcat in *Tomcat User Guide*.

Configuring Java Logging for WebLogic Server

To configure Java logging in WebLogic Server:

1. Specify the Java Unified Logging (JUL) mechanism. See "Specifying the Java Unified Logging (JUL) Mechanism."
2. Create a startup class. See "Creating a Startup Class."

Specifying the Java Unified Logging (JUL) Mechanism

Specifying the JUL mechanism allows Web Services Manager to use JUL in addition to the WebLogic Server Administration Console logging.

To specify the JUL mechanism:

1. Open the `BRM_home/deploy/web_services/Infranet.properties` file in a text editor.
2. Uncomment the following entry:
   ```properties
   # webservices.log.enabled = true
   ```
3. Change the value to `false`:
   ```properties
   webservices.log.enabled = false
   ```
4. Save and close the file.
Creating a Startup Class
You define a startup class to enable JUL and create log files for the following Web service classes:

- `com.portal.webservices.BRMFlistToXML`
- `com.portal.webservices.BRMXMLToFlist`
- `com.portal.webservices.OpCodeCaller`
- `com.portal.webservices.WebServicesUtilities`

To create a startup class:

1. Copy the `BRM_home/deploy/web_services/weblogic_ws_startup.jar` file to the `server_domain/lib` directory, where `server_domain` is the WebLogic Server domain in which Web Services Manager is deployed.
2. Log in to WebLogic Server Administration Console.
3. Click Lock and Edit.
4. In the Domain Structure pane, expand Environment and then click Startup and Shutdown classes.
   The Startup and Shutdown Classes pane appears.
5. Click New.
   The Configure a New Startup or Shutdown Class: Class Type pane appears.
6. Select Startup Class and click Next.
   The Configure a New Startup or Shutdown Class: Startup Class Properties pane appears.
7. In the Name field, enter `BRMWSLoggerStartUpClass`.
8. In the Class Name field, enter `com.portal.webservices.BRMWSLoggerStartUp`.
9. In the Argument field, set the log level. This field sets the log level for all the classes in Web Services Manager:
   - To log problems that require attention from the system administrator, enter SEVERE. This is the default.
   - To log the most detailed trace and debug messages, enter FINEST.
   - To log highly detailed trace and debug messages, enter FINER.
   - To log trace and debug messages for performance monitoring, enter FINE.
10. Click Next.
    The Configure a New Startup or Shutdown Class: Select Targets pane appears.
11. From the Servers list, select the server on which to deploy the class.
    The Startup and Shutdown Classes pane appears.
12. Click Finish.
13. Click `BRMWSLoggerStartUpClass`.
    The Settings for `BRMWSLoggerStartUpClass` pane appears.
14. Select Run Before Application Deployments and Run Before Application Activations and click Save.
15. Click Activate Changes.
16. Restart the WebLogic server, which applies changes.

17. Redeploy any existing Web Services Manager deployments. See “Deploying Web Services Manager.”

By default, log files are created in the `WebLogic_home/user_projects/domains/domain_name/logs/BRMWebServicesMgrLogs/BRMWebServicesMgr.log` file. Where:

- `WebLogic_home` is the directory in which WebLogic Server is installed.
- `domain_Name` is the name of the domain you are configuring.
Customizing Web Services

This chapter contains information about customizing Oracle Communications Billing and Revenue Management (BRM) Web Services Manager.

Topics in this document:

- Setting Up Web Services Manager to Support Custom Opcodes
- Creating a Custom Web Service
- Generating the Schema Files for Your System
- Generating the Schema for Your Opcodes

See also:

- Using Web Services
- Deploying Web Services Manager
- Configuring Web Services Manager

Setting Up Web Services Manager to Support Custom Opcodes

To expose custom opcodes as web services, enable Web Services Manager to support custom opcodes. For more information on custom opcodes, see "Using Custom Opcodes" in BRM Developer’s Guide.

**Note:** Before you customize an opcode for a web service in Web Services Manager, implement the custom opcode in BRM.

To enable Web Services Manager to support custom opcodes:

1. Do one of the following:

   - Create the **CustomOp.java** file by entering the following command:
     ```
     parse_custom_ops_fields -L pcmjava -I input -O output -P java_package
     ```
     where:
     
     - **input** is the header file you create for your custom opcodes and fields.
     - **output** is the memory-mapped file or directory for the output of the script. **output** must be a directory having some correspondence with the Java package. For example, if **java_package** is in `com.portal.classFiles`, **output** must be `f:/mysource/com/portal/classFiles`.
     - **java_package** is the Java package in which to put the generated classes.
For more information, see the discussion about the `parse_custom_ops_fields` utility in BRM Developer’s Guide.


2. Verify that the `CustomOp.java` file contains the following:
   - The opcode-name-to-opcode-number mapping for all the custom opcodes in the file.
   - The `opToString` method, which converts opcode numbers to opcode names.
   - The `stringToOp` method, which converts opcode names to opcode numbers.

The following is a sample `CustomOp.java` file:

```java
public class CustomOp {
    public static final int CUSTOM_OP_ACT_INFO = 100000;
    public static final int CUSTOM_OP_READ_ACT_PRODUCT = 100001;

    public static String opToString( int op ) {
        try {
            java.lang.reflect.Field[] flds = CustomOp.class.getFields();
            for( int i = 0; i < flds.length; i++ ) {
                try {
                    int val = flds[i].getInt(null);
                    if( val == op ) {
                        return flds[i].getName();
                    }
                } catch( IllegalAccessException e ) { continue; }
                catch( IllegalArgumentException e ) { continue; }
            }
        } catch( SecurityException e ) {} 
        return null;
    }

    public static int stringToOp( String op ) {
        try {
            java.lang.reflect.Field[] flds = CustomOp.class.getFields();
            for( int i = 0; i < flds.length; i++ ) {
                try {
                    String name = flds[i].getName();
                    if( name.equals(op) ) {
                        return flds[i].getInt(null);
                    }
                } catch( IllegalAccessException e ) { continue; }
                catch( IllegalArgumentException e ) { continue; }
            }
        } catch( SecurityException e ) {} 
        return -1;
    }
}
```

_Important:_ Verify that the mapping includes the full name of each opcode. If any opcode name is truncated, replace the truncated name with the full name.
3. Compile the `CustomOp.java` file into the `CustomOp.class` file by entering the following command:

```
javac -d . path/CustomOp.java
```

For example:

```
javac -d . com/portal/classFiles/CustomOp.java
```

4. Package the `CustomOp.class` file into the `CustomFields.jar` file by entering the following command:

```
jar -cvf CustomFields.jar path/CustomOp.class
```

For example:

```
jar cvf CustomFields.jar com/portal/classFiles/CustomOp.class
```

5. Make the `CustomFields.jar` file available to Web Services Manager by doing one of the following:

   a. If you have not deployed Web Services Manager, do the following:
      
      a. Copy the `path/CustomFields.jar` file to the `local_dir/WEB-INF/lib` directory, where `path` is the path to the `CustomFields.jar` file (for example, `com/portal/classFiles`).
      
      b. Open the `BRM_home/deploy/web_services/Infranet.properties` file in a text editor.
      
      c. Add or modify the following entry:
      
      ```
      infranet.custom.field.package = package
      ```
      
      where `package` is the name of the package that contains the `CustomOp.java` file; for example, `com.portal.classFiles`.
      
      d. Add all the custom fields to the `Infranet.properties` file.
      
      e. Save and close the file.
      
      f. Copy the `BRM_home/deploy/web_services/Infranet.properties` file to the `local_dir/WEB-INF/classes` directory or the home directory on the machine on which you installed WebLogic Server.

   b. If you have deployed Web Services Manager, do the following:
      
      a. Copy the `path/CustomFields.jar` file to the `local_dir/WEB-INF/lib` directory.
      
      where `local_dir` is the directory in which you deployed Web Services Manager on your application server.
      
      b. Open the `Webservices_deployment_dir/WEB-INF/classes/Infranet.properties` file in a text editor.
      
      c. Add or modify the following entry:
infranet.custom.field.package = package

where package is the name of the package that contains the CustomOp.java file; for example, com.portal.classFiles.

d. Add all the custom fields to the Infranet.properties file.
e. Save and close the file.

Creating a Custom Web Service

You can extend Web Services Manager to support custom web services. Before you create a custom web service or customize an existing web service in Web Services Manager, implement your custom opcodes in the BRM system. For more information, see "Creating Custom Fields and Storable Classes" in BRM Developer’s Guide.

To create a custom web service:

1. If you created an opcode with custom fields for your custom web service, configure BRM to recognize the custom fields. See "Creating Custom Fields and Storable Classes” in BRM Developer’s Guide.

2. Create a WSDL file for the web service. See "Generating WSDL Files for Web Services" in BRM JCA Resource Adapter.

   To create a WSDL file manually, do the following:
   - For web services that support payload as XML string, see the deploy/web_services/wsdl sample file and create the WSDL file.
   - For web services that support payload as XML element, see the deploy/web_services/BrmWebServices.war/WEB-INF/wsdl sample file and create the WSDL file.

3. Create the XML specifications for your custom opcodes. See "Creating Opcode Specification Schema Files”.

4. Generate web service classes for your custom service by doing the following:

   a. Create the following directory structure in a local directory (local_dir) on the machine on which your application server is installed.

   ```
   /wsdl
   /src
   /classes
   /jar
   ```

   b. Copy your custom WSDL files and schema (XSD) files into the local_dir/wsdl directory.

   c. Copy the BrmWebServices.war/WEB-INF/wsdl/BRMWebServiceException.xsd file into the local_dir/wsdl directory.

   d. Create the custom_services.xml as an ant build file.

   The following is a sample custom_services.xml file:

   ```
   <?xml version="1.0"?>
   <project name="Custom BRM WebServices build file" default="all"
    basedir=".">
    <property name="buildDir" value="classes"/>
    <property name="srcDir" value="src"/>
    <property name="wsdlDir" value="wsdl"/>
   ```
Creating a Custom Web Service

```xml
<property name="pinwsgen" value="pin_wsgen"/>

<!-- define the classpath -->
<path id="classpath">
  <pathelement path="${buildDir}"/>
  <pathelement path="jar/web_services.jar"/>
  <pathelement path="jar/webServicesUtils.jar"/>
</path>

<!-- create Source files from WSDL and XSDs -->
target name="custom_service_gen" description="Create java source files from wsdl">
  <exec executable="BRM_home/deploy/web-services/pin_wsgen/pin_wsgen" failonerror="true">
    <arg value="-s"/>
    <arg value="src"/>
    <arg value="-d"/>
    <arg value="com.portal.jax."/>
    <arg value="yourpackagesubdirname"/>
    <arg value="${wsdlDir}/YourCustomServices_v2.wsdl"/>
  </exec>
</target>

<target name="all" depends="custom_service_gen, custom_jar" description="build everything"/>

<!-- compile task -->
target name="compile" depends="custom_service_gen" description="compile source files">
  <echo>" Compiling JAX-WS impl classes"</echo>
  <javac srcdir="${srcDir}" destdir="${buildDir}" classpathref="classpath" debug="on" source="1.5"/>
</target>

<!--Create custom service jar -->
target name="custom_jar" depends="custom_service_gen, compile" description="generate jar file">
  <jar jarfile="custom_services.jar" basedir="${buildDir}"/>
</target>

<!--ant clean task -->
target name="clean" description="remove derived objects">
  <delete dir="classes/com"/>
  <delete dir="custom_service.jar"/>
</target>

</project>

where:
Generating the Schema Files for Your System

Web Services Manager uses schema files to validate data it sends to or receives from BRM.

To generate the schema files for your system, do the following:

1. If you modified any opcodes, generate schemas for the opcodes in your BRM system. See "Generating the Schema for an Existing Opcode".

2. Generate schemas for the storable classes and subclasses in your BRM system. See "Generating the Schema for Your Storable Classes and Subclasses" in BRM JCA Resource Adapter.

3. In your opcode schema files, specify the location of your storable class schema files. See "Specifying the Location of the Storable Class Schema Files in the Opcode Schema Files" in BRM JCA Resource Adapter.

---

BRM_home is the directory in which BRM is installed.

YourCustomServices_v2 is the custom service WSDL file name.

yourpackagesubdirname is the package directory for your custom service.

5. Generate and build your custom web services by running the following command:

    ant -file custom_services.xml

6. Add all the custom field enum constants to the Infranet.properties file. See "Connecting Web Services Manager to the BRM System" for more information.

7. Package your custom web service with the BrmWebServices.war file by doing the following:
   a. Extract the BrmWebServices.war file to a local directory (local_dir) on the machine on which you installed your application server.
   b. Do one of the following:
      (For WebLogic server) Modify the local_dir/WEB-INF/Web.xml file to include your custom service URL mapping similar to existing URL mapping.
      (For Apache Tomcat server) Modify the local_dir/WEB-INF/sun.jaxws.xml file to add your custom service implementation class.
   c. Copy your custom WSDL files and schema (XSD) files into the local_dir/WEB-INF/wsdl/ directory.
   d. Copy your custom_services.jar into the local_dir/WEB-INF/classes directory.
   e. Copy your CustomFields.jar into the local_dir/WEB-INF/lib directory.
   f. Delete the existing BrmWebServices.war file.
   g. Create a new BrmWebServices.war file by running the following command:

        jar -cvf BrmWebServices.war *
Generating the Schema for Your Opcodes

The Web Services Manager package includes all the opcode schemas and flist specifications you need for a default integration.

If you customized any of the opcodes that are supported by Web Services Manager or if you added support for new opcodes, you must generate XSD schema files for the opcodes.

Note: After generating the opcode and storable class schema files, copy the schema files to a location that is accessible to the Web Services Manager. Make sure that this location is the same as the location that is specified in the include section of the opcode schema files and in the opcode schema InteractionSpec attribute in the WSDL files. See "Specifying the Location of the Storable Class Schema Files in the Opcode Schema Files" and "Generating the WSDL Files for Your System" in BRM JCA Resource Adapter.

Generating the Schema for an Existing Opcode

To generate schema files for an opcode that you customized and Web Services Manager already supports:

1. Modify the opcode’s XML specification file. By default, the opcode specification XML files are installed in the BRM_home/apps/brm_integrations/opspecs directory, where BRM_home is the directory in which you installed the BRM components.

2. Do one of the following:
   - For web services that take payload as XML string:
     - Run the pin_opspec_to_schema utility. See "Creating Opcode Specification Schema Files".
     - Copy the customized XSD files to the BRM_home/deploy/web_services/schemas directory.
   - For web services that take payload as XML element:
     - Run the pin_opspec_to_schema_v2 utility. See "Creating Opcode Specification Schema Files".
     - Copy the customized XSD files to the infranetwebsvc/WEB-INF/services/InfranetWebservices.aar/META-INF directory.

Creating Opcode Specification Schema Files

You must create opcode flist specification files for opcodes that you customize or add to the Web Services Manager. Create the specification XML files by following the BRM_home/apps/brm_integrations/stylesheets/opspec.xsd file.
You then convert the opcode flist specification XML files into XSD schema by using the `pin_opspec_to_schema` and `pin_opspec_to_schema_v2` utilities.

To convert opcode flist specification XML files into XSD schema, go to the `BRM_home/apps/brm_integrations` directory and do the following:

- For web services that take payload as XML string, run the following command:

  ```bash
  pin_opspec_to_schema -i input_file [-o output_file]
  ```

- For web services that take payload as XML element, run the following command:

  ```bash
  pin_opspec_to_schema_v2 -i input_file > output_file
  ```

  where:

  - `input_file` specifies the name and location of the opcode’s XML flist specification. By default, the utility looks for the file in the current directory.
  - `output_file` creates the XSD schema output file using the name you specify. By default, the utility creates a file named `opcodename.xsd` in the directory from which you run the utility.

You can also create XSD schema for web services that take payload as XML element by using the `pin_opspec_to_schema_v2` XSD generator utility that is located in the `BRM_home/bin` directory.

To create the XSD schema file by using the `pin_opspec_to_schema_v2` utility, run the following command using Groovy:

```bash
groovy pin_opspec_to_schema_v2 -i input.xml > output.xsd
```

where:

- `input.xml` specifies the name of the opcode’s XML flist specification
- `output.xsd` creates the XSD schema output file using the name you specify

### Specifying the XSL Rules to Create the Opcode Schema

The `pin_opspec_to_schema` utility uses the `BRM_home/brm_integrations/stylesheets/pin_opspec_to_schema.xsl` style sheet to generate the schema for BRM opcodes. If your opcode references custom fields, you must customize the `pin_opspec_to_schema.xsl` style sheet to handle your custom fields.

For a list of the supported BRM data types, see "Understanding the BRM Data Types" in `BRM Developer’s Guide`. 